UNIVERSIDAD POLITÉCNICA DE MADRID
ESCUELA TÉCNICA SUPERIOR DE INGENIEROS DE
TELECOMUNICACIÓN

**CONTRIBUTIONS TO SPEECH AND LANGUAGE
PROCESSING TOWARDS AUTOMATIC SPEECH
RECOGNIZERS WITH EVOLVING DICTIONARIES**

PH.D. THESIS

ALEJANDRO COUCHEIRO LIMERES

**Graduado en Ingeniería de Tecnologías y Servicios de
Telecomunicación**
por la Universidad Politécnica de Madrid

Máster Universitario en Ingeniería de Telecomunicación
por la Universidad Politécnica de Madrid

2019

DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA

**ESCUELA TÉCNICA SUPERIOR DE INGENIEROS DE
TELECOMUNICACIÓN**

**Contributions to Speech and Language
processing towards Automatic Speech
Recognizers with Evolving Dictionaries**

PH.D. THESIS

Alejandro Coucheiro Limeres

**Graduado en Ingeniería de Tecnologías y Servicios de
Telecomunicación**

por la Universidad Politécnica de Madrid

Máster Universitario en Ingeniería de Telecomunicación

por la Universidad Politécnica de Madrid

Director: Javier Ferreiros López

Doctor Ingeniero de Telecomunicación

por la Universidad Politécnica de Madrid

2019

UNIVERSIDAD POLITÉCNICA DE MADRID
E.T.S.I. DE TELECOMUNICACIÓN
DEPTO. DE INGENIERÍA ELECTRÓNICA

TESIS DOCTORAL

TÍTULO: Contributions to Speech and Language processing towards Automatic Speech Recognizers with Evolving Dictionaries

DOCTORANDO: D. Alejandro Coucheiro Limeres

DIRECTOR: D. Javier Ferreiros López

El día _____ de _____ de _____, el Magnífico y Excelentísimo Señor Rector de la Universidad Politécnica de Madrid, designó el siguiente Tribunal para evaluar la Tesis Doctoral arriba mencionada:

Presidente:

Secretario:

Vocal 1º:

Vocal 2º:

Vocal 3º:

Suplente 1º:

Suplente 2º:

Realizado el acto de Lectura y Defensa de dicha Tesis Doctoral en Madrid, el día _____ de _____ de _____, en la ETSI de Telecomunicación, el Tribunal acuerda otorgarle la siguiente calificación:

CALIFICACIÓN:

EL PRESIDENTE

EL SECRETARIO

LOS VOCALES

A mis padres, por tanto y por todo

*“You can recognize a small truth because its opposite is a falsehood.
The opposite of a great truth is another truth”*

Niels Bohr (1885-1962)

*“For last year’s words belong to last year’s language
And next year’s words await another voice”*

T.S. Eliot (1888-1965)

Agradecimientos

Muchas son las personas a quienes quiero agradecer su apoyo y colaboración que tanto me han ayudado en la realización de esta Tesis, así que espero no olvidarme de ninguna.

En primer lugar, quiero dar las gracias a mi director de Tesis, Profesor Javier Ferreiros, por su labor extraordinaria guiándome en este largo y enriquecedor camino de la investigación. Desde mis primeras incursiones en el campo, un par de años antes de comenzar la Tesis, hasta hoy, he disfrutado con las enseñanzas y la pasión por la materia que Javier me ha ido transmitiendo. Especialmente, valoro las disquisiciones sobre los distintos puntos del trabajo que teníamos entre manos, donde la creatividad hacía sus apariciones para resolver los diversos retos a los que nos íbamos enfrentando. Además, este mismo afán lo reutilizábamos en unas cuantas ocasiones para dialogar acerca de cuestiones de tan distintas índoles, algo que aprecio también muchísimo por poder contar de nuevo con su curiosidad, rigor y humildad.

Quiero agradecer a Julián Echeverry por ser mi primer mentor y nexo con el Grupo de Tecnología del Habla. Con su cercanía y buen humor, consiguió despertar en mí un grato cariño por esta ciencia. Mis mejores deseos para tu camino en Colombia junto a Bea, a quien también echamos de menos. No dejéis nunca de visitarnos (y, con suerte, alguna vez nos tocará a nosotros ir).

Gracias a todos los demás profesores del Grupo de Tecnología del Habla: Manolo, Fernando, Ricardo, Rubén, Juancho, Luisfer y también Manuel. Ha sido todo un placer colaborar con vosotros desde el 2014. Conocer vuestros diversos trabajos ha resultado una experiencia muy enriquecedora. Además, no faltaron los comentarios constructivos y demás ayudas para la realización de mi Tesis.

Mención especial a Fernando, Ricardo y Juancho por el recuerdo que guardaré de nuestras épicas batallas contra las máquinas del labo, que tantas veces nos desafiaron, mas siendo siempre nuestro el éxito.

De mucha ayuda fueron también las distintas colaboraciones con estudiantes de grado y máster que incluso tuve el gusto de cotutorizar: Daniel, Ismael y especialmente Ricardo Kleinlein, quien además se ha embarcado en su propia Tesis dentro del grupo. No dudo de las audaces hazañas que serás capaz de obtener.

Gracias a Fernando González por las largas y variopintas conversaciones que con frecuencia manteníamos. También por mostrarme tu talento para capturar escenas y lugares tanto cercanos como lejanos. Que tus viajes, los de avión y los de espíritu, prosigan siempre.

Agradezco a Silvia y a Mariano todas las gestiones con las que me han ayudado, pues no fueron pocas, y su disposición fue siempre pronta y eficaz.

Gracias a la Universidad Politécnica de Madrid y la Escuela Técnica Superior de Ingenieros de

Telecomunicación, especialmente sus profesores, por toda la formación que me otorgaron durante estos años, considerando un orgullo el haber podido cursar estudios en esta escuela y con unos compañeros tan brillantes.

A mi familia le debo la mayor gratitud de todas, pues su ayuda comenzó mucho más atrás del inicio de esta Tesis, siendo indispensable para convertirme en quien soy actualmente, y no faltando su apoyo durante todos estos años de realización de la Tesis. Mis padres, Mónica y Antonio, me ofrecieron una educación basada en el esfuerzo y la confianza, la honestidad y el amor. Mis hermanos, Paloma y Roi, aparte de hermanos, siempre han sido amigos míos, y hemos compartido tantas cosas que nuestro aprendizaje de la vida ha transcurrido en paralelo, por lo que alcanzamos unos niveles de comprensión y cariño que nos ayudaron a crecer siempre un poco más. Mis tías Tita y Nita, una para Vigo y otra para Madrid, con las que puedo contar para todo, sea bueno o malo. Y así podría seguir con otros tantos miembros de la familia, todos ellos estimados y queridos.

Un millón de gracias a mis grandes amigos Alberto y Miguel, pues nuestra amistad pronto llegará a la década, en la cual hemos vivido de todo y hemos aprendido muchísimo. Estoy seguro de que esta década es sólo la primera de tantas por venir.

No me olvido tampoco de vosotros: Natalia, Sastre, Luisro, Romero, Andreu, Mario Pereda y Martíns, Isidro, Celia, Winston, Ana Vic y también Sara. Que continúen nuestras quedadas, charlas, viajes, debates, disfraces y por supuesto fiestas.

Gracias a la panda salsera por descubrirme una nueva afición que tan bien funciona para hacer un reset mental, muy necesario para liberarse uno de obstinaciones atascadas que, después del baile, casi solas se resolvían. Gracias a Christian por ser el introductor, y gracias a los demás por continuar la rueda: Ali, Almu, Diego, Germán, Ale y Carmen.

Gracias a mis amigos en el extranjero por las aventuras vividas y el soporte recibido cuando convivimos en la misma ciudad. En Viena, mil gracias a Laura, Mònica, Kathleen, Miriam, Andy y Marco. En Baltimore, siempre agradecido a Laureano, Jesús, Paola, mis compis de labo y mis super-caseros Tina y Chuck. Con gusto me esforzaré para que sigamos coincidiendo a pesar de la distancia.

Muchas gracias al Center for Language and Speech Processing, especialmente a su director Hynek Hermansky, y a la Johns Hopkins University por resultar tan buen destino para mi estancia predoctoral y por su generosa acogida. He comprobado que el renombre que les acompaña es merecidísimo, sobre todo por la calidad de sus docentes, investigadores y estudiantes, cuyo trabajo he tenido la suerte de conocer de primera mano.

Agradecido estoy también al CMU Aquinas, por ser mi receptor en Madrid y lugar de encuentros tan heterogéneos y valiosos. No se me ocurre mejor modo de pasar mis primeros años en la capital que en semejante institución. Gracias Javier, Carlos y César.

Muchas gracias a los CMUs Vedruna y Berrospe, donde conocí a tantas amigas y donde los talleres de teatro fueron tan divertidos e inspiradores.

En fin, gracias a todos los que contribuyeron de alguna forma a que esta aventura llegara a buen puerto. Han sido unos años que recordaré siempre con felicidad por tanto que me han aportado en múltiples aspectos, valorando especialmente el humano y el académico. Gracias, de verdad.

Abstract

Automatic speech recognizers offer nowadays a notable performance in multiple and challenging tasks. This has propitiated their incorporation in user applications that are used very frequently. Such presence in the habitual human life has been reinforced due to the fact that speech is one of the most natural ways of human communication. And thus, our interaction with machines is evolving towards a speech-based communication, for accomplishing both simple tasks like dictation or web searches, and more complex tasks like human-machine dialogs for configuring e.g. domestic appliances.

The acoustic and language conditions of the audios that serve as input for these systems can vary enormously from source to source. Thus, a great work into improving their acoustic and language models has kept developing continuously since the last decades, where a recent impulse is being experimented thanks to the resurgence of neural networks.

On the acoustic part, the effort is focusing on dealing with any kind of noises that may be present on the audio, on being able to recognize the speech of any kind of speaker (including different accents and any other varying feature of the human voice) and on addressing any additional condition of the speech process, like far-field recognition.

On the language part, the effort is focusing on how to manage the large amount of different topics and domains that can be present in the speech. Beyond a brute-force approach, where a huge vocabulary and language model is used to try to face any of these scenarios, more interest is growing for systems that can restrict their modeling capabilities to recognize optimally certain topics or domains. This restriction allows the language models to be centered in the uses of language of interest, which otherwise would result in an inoperably big model (and likely less accurate).

Furthermore, a variable language restriction which keeps tuning the model to the language characteristics of the current speech would offer the recognition system an adaptation capability able to reach optimal performance regardless the topic of the input speech. This kind of adaptation is the main focus of this thesis work.

More specifically, we are interested in an automatic and unsupervised adaptation to the current speech which does not require an explicit identification of the current topic nor domain which would drive the adaptation process. Instead, we want the adaptation to be driven by the vocabulary that is employed in the speech, so as to tune accordingly the dictionaries of the recognition systems (with effects in both their vocabularies and language models).

Apart from the in-vocabulary (IV) words that we may be able to decode from the speech, we are more interested in the words that were not present in the current vocabulary of the systems, or out-of-vocabulary (OOV) words, as they can indicate implicitly changes of topic or domain. Thus,

we propose strategies for detecting OOV terms in the speech, finding the best candidate words for them and ultimately learning about its syntax and semantics so they can be incorporated properly into the recognition systems, causing modifications that result in an adaptation, considering together various OOVs appearances and resolutions.

The strategies that we propose in this thesis work can be divided in two levels of operations, one that works in a local, static level, and one that works in a dynamic, evolving level. They are called **static Term Discovery strategy** and **dynamic Vocabulary strategy**, respectively.

The processes involved in the **static Term Discovery strategy** can be enumerated as follows:

- Detection of OOV terms that might appear in the input speech.

We have contributed with a couple of OOV detection methods that can work in conjunction. The first method employs a OOV word model defined in both acoustic and language terms, while the second method is based on confidence analyses over the output word lattice that is delivered by the recognition systems.

- Search for candidate words for every OOV detection.

We have contributed with a search scheme that performs two different kind of searches, one that is acoustically driven, and another that is semantically driven. For this scheme, we take advantage of external knowledge sources where to find the best candidates. Furthermore, we also proposed a distributed representation scheme for the resources found in graph-organized corpora. These type of representations can benefit this search, and could be also employed in other semantic tasks like those found in the area of natural language processing and understanding.

- Correction of the output transcription with the best candidate found, if any.

We have contributed with a series of candidate scores that can improve the decision whether a candidate is suitable enough to substitute the original content of a detected OOV region.

And as regards the **dynamic Vocabulary strategy**, we proposed a series of processes to be executed iteratively when needed, as a reaction to the speech being decoded:

- Continuous collection of the terms retrieved by the static Term Discovery strategy, so as to assess whether some terms become interesting to be added to the system's vocabulary.

We have contributed with a scoring scheme that takes into account the scores that the static Term Discovery strategy gave to a word and also the time passed between the moments when that word was retrieved, so as to give more importance to the new terms used more recently.

- Selection of the most interesting new terms from the previous collection to add to the system's vocabulary, and also selection of the least interesting IV terms that will be removed from the vocabulary.

We have contributed with schema for both selecting terms to add to and delete from the vocabulary. For the words to add, we verify whether there are enough trainable material in external sources about the new terms, and we can also reconsider whether the transcription corrections made by a term were reliable enough, so as to refine our decision. And for the word to remove, we consider both which IV words are not being employed enough in the

input speech, and which words do not fit sufficiently the current state of the language model of the system.

- Update of the vocabulary and language model of the recognition systems considering the previous word selections.

We have contributed with an update scheme that considers both the previous language model of the system and a language model built with texts from external knowledge sources that contain the new terms, proposing as well an interpolation scheme of both models so as to produce a new language model with the designated vocabulary.

The proposed strategies have been evaluated under realistic experimental frameworks, where we employed state-of-the-art automatic speech recognizers, designed with different sizes of the vocabulary, and large external knowledge sources. The speech test corpora contains a great variety of speakers and natural, spontaneous speech, where multiple topics are discussed. Such features are in consonance with the conditions for which we want our strategies to offer benefits.

The evaluation results allow us to measure how noticeable are the improvements given to the recognition systems that are equipped with our strategies, in comparison with the systems which lack them. In fact, we were able to achieve significant improvements over the baseline systems for both strategies and in most of the experimental configurations.

Lastly, we were also able to study the behavior of the dynamic systems over time, in order to assess how fast and in which manner the desired adaptation is happening. We observed that, on average, the dynamic systems were capable of offering significant improvements over the baseline systems (and even over the systems equipped with the first, static strategy) just after few hours of operation and for very different speech corpora. Also, the type and number of words that the dynamic systems added/deleted over time were consistent with the expected behavior (mainly, adding words that actually appeared in the input speech and removing those that did not fit enough in the current state of the evolving language model), which accounts for the notable benefits that were measured.

Keywords

Automatic Speech Recognition, Out-Of-Vocabulary Words, Term Discovery, Automatic Lexicon Adaptation, Deep Neural Networks, Deep Learning, Machine Learning

Resumen

Los reconocedores automáticos de habla ofrecen a día de hoy un rendimiento muy notable en múltiples y complicadas tareas. Esto ha propiciado su incorporación en aplicaciones de usuario que ya son usadas con cierta asiduidad. Tal presencia en la cotidianidad de la vida humana se ha visto reforzada por el hecho de que el habla es una de las formas más naturales para comunicarnos. Así, nuestra interacción con máquinas está evolucionando hacia una comunicación basada en el habla, ya sea para realizar tareas tan simples como un dictado o búsquedas web, o más complejas como las que involucran diálogos para configurar, por ejemplo, aparatos domésticos programables.

Las condiciones acústicas y lingüísticas de los audios que sirven de entrada a estos sistemas pueden variar enormemente de una fuente a otra. Así, se sigue desempeñando de continuo, y desde las últimas décadas, un notable trabajo para mejorar los modelos acústicos y de lenguaje que estos sistemas emplean, habiéndose hallado un reciente impulso en tales investigaciones gracias al resurgimiento de las redes neuronales.

En la parte acústica, el esfuerzo se centra en lidiar con cualquier tipo de ruidos que pueden estar presentes en el audio, también en ser capaces de reconocer el habla de cualquier tipo de locutor (incluyendo distintos acentos y cualquier otra característica que pueda variar entre voces humanas) y, en fin, en abordar cualquier otra condición presente en el proceso de habla, como el reconocimiento en campo lejano.

En la parte lingüística, el esfuerzo se centra en cómo gestionar la gran cantidad de temáticas y dominios distintos que están presentes en el habla. Más allá de una solución de fuerza bruta, en la cual emplearíamos un vocabulario y un modelo de lenguaje gigantescos para tratar de enfrentarnos con cierto éxito a cualquiera de estos escenarios, resultan más interesantes los sistemas que pueden restringir sus capacidades de modelado para reconocer óptimamente ciertos tópicos o dominios. Dicha restricción permitiría que los modelos de lenguaje se centrasen en los usos lingüísticos bajo interés, que de otro modo desembocaría en un modelo tan grande que sería inoperable (y probablemente menos preciso).

Además, una restricción del lenguaje que fuera variable en el tiempo, manteniendo una sintonización del modelo con las características del habla de cada momento, ofrecería a los sistemas de reconocimiento una capacidad de adaptación que sería capaz de alcanzar un rendimiento óptimo, sin importar las temáticas que hayan podido aparecer en el habla. Este tipo de adaptación es el principal enfoque que hemos tomado en este trabajo de tesis.

Específicamente, estamos interesados en una adaptación automática y no supervisada al habla de cada momento, y que por tanto no requiera de una identificación explícita de la temática discutida en cada momento que pudiera dirigir el proceso de adaptación. En su lugar, queremos que la adaptación esté dirigida por el vocabulario que está siendo empleado en el habla, para de este modo tratar de sintonizar convenientemente los diccionarios de los sistemas de reconocimiento

(que tendría efectos tanto en sus vocabularios como en sus modelos de lenguaje).

Aparte de las palabras dentro-del-vocabulario (*In-Vocabulary*, IV) que pudiéramos transcribir del habla, nos interesan más las palabras que no estuvieran ya presentes en el vocabulario de los sistemas, conocidas como palabras fuera-del-vocabulario (*Out-Of-Vocabulary*, OOV), ya que estas pueden indicar implícitamente los cambios de tópico o dominio que se pudieran producir. Así, proponemos estrategias para detectar palabras OOV en el habla, hallando las mejores palabras candidato para ellas y, en último término, aprender su sintaxis y semántica para que puedan ser introducidas convenientemente en los sistemas de reconocimiento, provocando modificaciones que resultarían en una adaptación, sobre todo considerando varias apariciones de OOVs y sus respectivas resoluciones.

Las estrategias que proponemos en este trabajo de tesis pueden dividirse en dos niveles de operación, uno que funciona a un nivel estático y local, y otra que funciona a nivel dinámico y evolutivo. Las hemos llamado **estrategia de Descubrimiento estático de Términos** y **estrategia de Vocabulario dinámico**, respectivamente.

Los procesos involucrados en la **estrategia de Descubrimiento estático de Términos** se enumeran a continuación:

- Detección de palabras OOV que puedan aparecer en el habla de entrada.

Hemos contribuido con un par de métodos de detección de OOVs que pueden trabajar conjuntamente. El primer método emplea un modelo de palabra OOV definido en ambos términos acústicos y lingüísticos, mientras que el segundo método está basado en análisis de confianza aplicados a los grafos de palabra que los sistemas de reconocimiento entregan a su salida como resultado.

- Búsqueda de candidatos de palabra para cada OOV detectado.

Hemos contribuido con un esquema que efectúa dos tipos diferentes de búsquedas, una dirigida por las evidencias acústicas, y otra dirigida por la semántica contextual observada. Para este esquema, hemos aprovechado fuentes de conocimiento externas donde poder encontrar los mejores candidatos posibles. Adicionalmente, también hemos propuesto un esquema de representación vectorial para los recursos que se pueden encontrar en corpora organizados en grafos. Este tipo de representaciones beneficiaría esta búsqueda, y aparte podrían ser empleados en otras tareas semánticas como las que se pueden encontrar en el área de procesado y comprensión de lenguaje natural.

- Corrección de la transcripción de salida con el mejor candidato hallado, si lo hubiera.

Hemos contribuido con un conjunto de puntuaciones a los candidatos encaminadas a mejorar la decisión acerca de si finalmente un candidato es lo suficientemente adecuado para sustituir el contenido original de la región donde detectamos una OOV.

Y con respecto a la **estrategia de Vocabulario dinámico**, hemos propuesto una serie de procesos para ser ejecutados iterativamente según se precisen, y así reaccionar al habla que se esté procesando:

- Colección continua de los términos hallados por la estrategia de Descubrimiento estático de Términos, para de este modo valorar si un término adquiere el suficiente interés para ser incorporado al vocabulario del sistema.

Hemos contribuido con un esquema de puntuación que tiene en cuenta las puntuaciones que la estrategia de Descubrimiento estático de Términos haya dado a una palabra y asimismo el tiempo que haya transcurrido entre los momentos en que fue encontrada, para de este modo dar mayor importancia a los nuevos términos que se hayan usado más recientemente.

- Selección de los términos que sean más interesantes, de entre todos los capturados por el anterior proceso, para ser incorporados en el vocabulario del sistema. También, selección de los términos IV menos interesantes para que sean eliminados del vocabulario.

Hemos contribuido con sendos esquemas para la selección de términos para añadir y eliminar al/del vocabulario. En cuanto a las palabras a añadir, verificamos si hay suficiente material de entrenamiento en fuentes externas sobre los nuevos términos, y asimismo podemos reconsiderar si las correcciones realizadas a la transcripción por un término fueron lo suficientemente fiables o no, y así poder refinar nuestra decisión acerca de su incorporación al vocabulario. Y en cuanto a las palabras a eliminar, nuestra consideración se basa tanto en cuáles de las palabras IV no están siendo utilizadas lo suficiente en el habla de entrada, como qué palabras no encajan del todo bien en el estado actual del modelo de lenguaje del sistema, ya que este irá evolucionando.

- Actualización del vocabulario y el modelo de lenguaje de los sistemas de reconocimiento considerando las anteriores selecciones de palabras.

Hemos contribuido con un esquema de actualización que considera, en primer lugar, el último modelo de lenguaje utilizado por el sistema y, en segundo lugar, un modelo de lenguaje construido con textos provenientes de fuentes externas donde aparezcan los nuevos términos, proponiendo asimismo un esquema de interpolación de ambos modelos para producir finalmente el nuevo modelo de lenguaje que contará con el vocabulario recién designado.

Las estrategias propuestas han sido evaluadas en marcos de experimentación realistas, en los cuales hemos empleado reconocedores automáticos de habla acordes con el estado del arte, diseñados con diferentes tamaños de vocabulario, y consultado fuentes de conocimiento externas considerablemente grandes. En cuanto a los corpus de habla utilizados para la evaluación, hemos escogido unos que contienen una gran variedad de locutores y que recogen un habla natural y espontánea, donde múltiples temáticas son discutidas. Tales características entran en consonancia con las condiciones para las que queremos que nuestras estrategias ofrezcan beneficios.

Los resultados de dicha evaluación nos permiten medir cuán significativas son las mejoras otorgadas a los sistemas de reconocimiento que se equipen con nuestras estrategias, y poder compararlos con sistemas que carezcan de ellas. De hecho, hemos logrado mejoras significativas sobre los sistemas de referencia para ambas estrategias y en la mayoría de las configuraciones de experimentación.

Finalmente, también hemos podido estudiar el comportamiento de los sistemas dinámicos a lo largo del tiempo, y de este modo valorar cuán rápido y de qué manera se da la adaptación que procuramos. Se ha observado que, de media, los sistemas dinámicos fueron capaces de ofrecer mejoras significativas sobre los sistemas de referencia (e incluso sobre los sistemas equipados con la estrategia estática) tan sólo en unas horas de operación y para corpus de evaluación muy distintos. Asimismo, el tipo y número de palabras añadidas/borradas en el tiempo por los sistemas dinámicos son coherentes con el comportamiento esperado (principalmente, añadiendo palabras que realmente se emplearon en el habla de entrada y eliminando aquellas que no encajan suficientemente en el estado actual del modelo de lenguaje, que a su vez evoluciona en el tiempo), lo cual explica los beneficios tan notables que se han medido.

Palabras Clave

Reconocimiento Automático de Habla, Palabras Fuera-De-Vocabulario, Descubrimiento de Términos, Adaptación Automática de Diccionarios, Redes Neuronales Profundas, Deep Learning, Aprendizaje Máquina

Contents

1	Introduction	1
1.1	Human and machine speech abilities	1
1.2	Aim of the work	3
2	State of the art	7
2.1	Term Discovery strategies for ASR	8
2.1.1	Term Discovery with single OOV language modeling	10
2.1.2	Term Discovery with subword OOV language modeling	10
2.1.3	Orthographic transcription of OOV terms	11
2.2	Language Model Adaptation	12
2.2.1	Cache-based models	12
2.2.2	Mixture-based models	13
2.3	Distributed representations from neural networks	15
2.3.1	Purposes and insight	15
2.3.2	History and usages	16
3	Objectives	19
4	Thesis work on Static Term Discovery strategies	21
4.1	Vocabulary limitation in ASR systems	21
4.2	Linked Open Data resources	22
4.3	Proposed Static Term Discovery strategy	23
4.3.1	OOV detection in speech	25
4.3.1.1	The special word <UNK>	25
4.3.1.2	Path pairs in the output graph	27
4.3.1.3	Operation of both methods	28
4.3.2	Potential-OOV Speech Segments processing	29
4.3.2.1	Search for candidates by word building	30
4.3.2.2	Search for candidates by LOD search	31
4.3.2.3	Candidate Scoring	35

4.4	Resource2Vec: Linked Open Data embeddings	36
4.4.1	Implementation of the Resource2Vec scheme	37
4.4.2	Quantitative analyses	38
4.4.2.1	Centroid quality evaluation	40
4.4.2.2	KNN quality evaluation	42
4.5	Experimental framework for Static Term Discovery evaluation	44
4.5.1	LVCSR system	44
4.5.2	Phone ASR system	47
4.5.3	OOV detector	47
4.5.4	Search for candidates system	49
4.5.4.1	LOD corpus	49
4.5.4.2	Word embedding model	50
4.5.4.3	Resource embedding model	51
4.5.5	Evaluation scheme	54
4.6	Evaluation results	56
4.6.1	Overall WER results	56
4.6.2	OOV solving success	59
4.6.3	OOV coverage study	60
4.7	An extension to the Static Term Discovery Strategy: the Word Vector Predictor System	62
4.7.1	Scientific contextualization	63
4.7.2	Proposed Architectures	64
4.7.3	Experiments	66
4.7.3.1	Word embeddings	67
4.7.3.2	Training scheme	67
4.7.3.3	Evaluation scheme	68
4.7.3.4	Results	70
5	Thesis work on Dynamic Vocabulary Strategies	73
5.1	Proposal for a Dynamic Vocabulary strategy	73
5.1.1	Mathematical model	75
5.1.1.1	Naive word rate model	81
5.1.1.2	Circumstances to consider in the dynamic processes	81
5.1.2	Vocabulary selection	83
5.1.2.1	Term collection	84
5.1.2.2	Selection of words to add	86
5.1.2.3	Selection of words to delete	92
5.1.3	Language Model update	93
5.1.3.1	Interpolation scheme	94

5.1.3.2	Vocabulary arrangements	96
5.1.3.3	Incorporation to the LVCSR system	96
5.1.4	Reconsideration mechanism	98
5.1.4.1	Re-correction to the transcription	98
5.1.4.2	Additional filter for words to add selection	100
5.2	Experimental framework for Dynamic Vocabulary evaluation	100
5.2.1	Term Discovery strategy	101
5.2.1.1	LVCSR system modifications	101
5.2.1.2	Phone ASR modifications	103
5.2.1.3	Term Discovery strategy modifications	104
5.2.2	Vocabulary manager	105
5.2.2.1	Text Corpora	106
5.2.2.2	Dynamic variation of thresholds	106
5.2.3	Language Model retraining	108
5.2.4	Reconsideration mechanism	110
5.2.4.1	Creation of word classes	110
5.2.4.2	Determination of reconsideration thresholds	111
5.2.5	Evaluation scheme	113
5.2.5.1	Evaluation corpora	113
5.2.5.2	Evaluation metric and comparative systems	115
5.2.5.3	Dynamic behavior studies	116
5.3	Evaluation results	117
5.3.1	Overall WER results	117
5.3.1.1	Speech redecoding with the adapted systems	120
5.3.2	Progressive WER evolution study	121
5.3.3	Word addition evolution study	129
5.3.4	LM evolution study	134
5.3.4.1	LM size growth analysis	140
6	Conclusions	141
6.1	On Static Term Discovery Strategies	142
6.2	On Dynamic Vocabulary Strategies	145
7	Future Work	151
7.1	On Static Term Discovery Strategies	151
7.2	On Dynamic Vocabulary Strategies	155
8	Publications	159

List of figures

4.1	Architecture of the proposed static TD strategy	23
4.2	Topology of different kind of phones in the LVCSR system. Up: regular phones. Middle: silence phone. Down: “mumble” phone	26
4.3	Example of 2 best-paths obtained for different LM weights applied to the output graph, in Spanish language (up: linguistically emphasized; down: acoustically emphasized). A disagreement region is marked with a red rectangle, covering most of the temporal occurrence of a OOV word indicated with a double-ended arrow in the middle	28
4.4	Operation scheme of the search for candidates module	30
4.5	Search for candidates by word building	30
4.6	Search for candidates by LOD search	31
4.7	Generation of search triggers	32
4.8	Resource2Vec training scheme. Up: an example of a R_a resource with its true connections inside some LD corpora. Down: neural network architecture and training operation for R_a resource, to be carried out in a pairwise fashion, $\{R_a, R_{a,b}\}$	39
4.9	Schematic Resource2Vec representation for a resource R_a and neighboring resources that can be achieved by a 2-hop path from it in the original knowledge graph	41
4.10	Intersection ratios between resources regarding distance in Resource2Vec model	52
4.11	Percentage of retrieved neighbors, precision, recall and $F1$ score regarding distance threshold in Resource2Vec model for $K=100$	54
4.12	OOV coverage and representation in testing set of substitutes obtained from the Resource2Vec TD system. Also, the number of unexpected-OOVs selected as substitutes. The results are shown as a function of the number each substitute was obtained over all POOV SSs	61
4.13	Baseline architecture of WVPS	65
4.14	Stacked architecture of WVPS	65
4.15	Attention architecture of WVPS	66
4.16	Histogram of cosine similarities applied to the TCSTAR evaluation corpus, for the <i>attention architecture</i> of WVPS. In dashed line, Gaussian approximation of the central part	69
4.17	Evolution of the local loss and, in dashed line, the average cosine similarity (TCSTAR evaluation corpus), for the <i>attention architecture</i> of WVPS	71
5.1	Architecture of the proposed dynamic vocabulary strategy	74
5.2	Vocabulary size control scheme for the proposed Dynamic TD strategy	76

5.3	Evolution of the vocabulary size for some ground-truth speech transcriptions . . .	108
5.4	Naive word-addition-rate model as a function of the vocabulary size, in added words per minute units	108
5.5	ROC curve for a sweep of values for the μ coefficient in inequality 5.14	113
5.6	Progressive WER for the comparative systems without TD strategy. Up: TCSTAR test corpus. Down: TV test corpus	122
5.7	Progressive WER for the systems with $V=20K$. Up: TCSTAR test corpus. Down: TV test corpus	123
5.8	Windowed WER (last 4 hours) for the systems with $V=20K$. Up: TCSTAR test corpus. Down: TV test corpus	125
5.9	Progressive WER for the systems with $V=50K$. Up: TCSTAR test corpus. Down: TV test corpus	126
5.10	Progressive WER for the systems with $V=12.5K$. Up: TCSTAR test corpus. Down: TV test corpus	128
5.11	Presence of natural OOVs in vocabulary (without starting offsets) for all dynamic systems. Up: TCSTAR test corpus. Down: TV test corpus	130
5.12	Distribution evolution of the unigram log-likelihood of the systems' LM, $V=20K$. The darker the later in time the distribution was sampled. Up: TCSTAR test corpus. Down: TV test corpus	135
5.13	Distribution evolution of the unigram log-likelihood of the systems' LM, $V=50K$. The darker the later in time the distribution was sampled. Up: TCSTAR test corpus. Down: TV test corpus	136
5.14	Distribution evolution of the unigram log-likelihood of the systems' LM, $V=12.5K$. The darker the later in time the distribution was sampled. Up: TCSTAR test corpus. Down: TV test corpus	137
5.15	Up: distribution evolution of the unigram log-likelihood of the <i>Random</i> system's LM. Down: differences of the final unigram distribution between the TCSTAR dynamic system and the <i>Raw</i> baseline, and also between the <i>Random</i> system and the <i>Raw</i> baseline. $V=20K$	139
7.1	Example of a LM of mixed words and classes, in graph form. Sub-graphs are presented for each of the class tokens	153

List of tables

4.1	Real examples of search seeds and DBpedia search triggers for different POOV SSs detected, in Spanish language	35
4.2	Corpora used for LVCSR system implementation	45
4.3	Number of n-grams in the Full and Partial LVCSR LM	47
4.4	Number of n-grams in the Phone ASR LM	47
4.5	OOV detection results comparison with other works	48
4.6	Texts employed for Word2Vec training	50
4.7	Mean Euclidean distance measures in the Resource2Vec model	52
4.8	WER results for the different systems	57
4.9	Overall WER improvement comparison with other works	57
4.10	Texts employed for the training of word embeddings. Vocabulary: 248K one-words, 25K two-words, 60K three-words	67
4.11	Evaluation results of the WVPS architectures proposed, excluding the 250 most frequent words in the vocabulary	70
5.1	Coefficients configuration for the three thresholds $\gamma^{Subs}(t)$, $\gamma^{Coll}(t)$ and $\gamma^{Updt}(t)$, with $\epsilon_1 \leq \epsilon_2 \leq \epsilon_3$	85
5.2	Different cases for the terms <i>interesting for update</i> and expected behavior of our dynamic TD strategy, when no reconsideration mechanism is employed	89
5.3	Number of n-grams in the LVCSR LM for the three values of V	102
5.4	Number of n-grams in the Phone ASR LM for the case of $V=12.5K$	104
5.5	Examples of the classes used in the reconsideration mechanism, in Spanish language	111
5.6	Reconsideration cases used for computing FPR and TPR	112
5.7	Characteristics of the two evaluation corpora for the dynamic vocabulary strategy	115
5.8	Overall WER results for the systems with a vocabulary of 20K	119
5.9	Overall WER results for the systems with a vocabulary of 50K	119
5.10	Overall WER results for the systems with a vocabulary of 12.5K	119
5.11	Raw redecoding WER results comparison for the systems with a vocabulary of 20K	121
5.12	Time to significant improvements (TSI) for the dynamic systems, per test corpus .	127
5.13	Starting offsets and representativeness of natural OOVs present in the original vocabulary of 20K, 50K and 12.5K, per test corpus	131
5.14	Number of added words to the vocabulary by the dynamic systems, per type of OOV and test corpus	133

5.15	Number of deleted words from the vocabulary by the dynamic systems, per type of word and test corpus	133
5.16	LM size growth for the different dynamic systems after decoding entirely a test corpus	140

Chapter 1

Introduction

1.1 Human and machine speech abilities

The most natural and characteristic way of human communication is speech. As so, it can be employed in all fields of human intervention, although we find that nowadays it is suppressed by a written form in a rising number of them. However, there will be always several ones where speech is inescapable, like close human relationships, politics, important business agreements, education or various art forms (music, cinema or theater).

The technology plays part in both directions of favouring/substituting the use of speech. This way, on the one hand, we can think of phone calls, television and web videos (in portals like Youtube or Vimeo) as very extended channels where speech owns an important role of communication. As a notable example, the popularity of the TED talks indicates the suitability of speech to communicate appealing and encouraging ideas often accompanied with emotions.

On the other hand, we find the email, chats and web-based *apps* (meeting or buy/sell social portals, food ordering, etc.) as examples of how speech is substituted. Also, social networks like Facebook, Twitter or Instagram, together with opinion sinks like Reddit or Tripadvisor, facilitate the expression of individual appreciations in a written form.

All the same, we believe that speech will never loose a notorious part of our communication, since it is the user of these technologies who decides which way is preferred at any moment. Therefore, we can think that speech-suppressing-technologies simply make us able to multiply the number of conversations and improving the efficiency of some managements, while we keep the speech for the other mentioned occasions. Even more, the arising and improved human-machine communication products (e.g. Google Assistant ; Siri, from Apple; or Alexa, from Amazon), may propitiate in the near future the use of speech for some applications that occurred traditionally in a written form, like web queries or calendar appointments, to name just a few.

Such an important ability as speech has led to the engineering of methodologies and technologies that deal directly with it. They are known as machine speech recognition and machine speech synthesis.

Machine speech recognition (known as Automatic Speech Recognition, ASR) focuses on the transcription of speech into words. Thus, ASR is a crucial area of research for automatically obtaining a transcription from utterances present in television, phone calls, political debates or a user of a consumer electronic product. In the last example, ASR allows the user to find a human-machine interface based on speech (at least in the direction from the human to the machine).

Machine speech synthesis is the ability of generating an audio wave that imitates the human utterances, with optional extensions like the imitation of a particular person's voice or the addition of an emotion (surprise, anger, etc.) or a style (to sound like a politician, a football commentator, etc.). This spoken articulation allows a human-machine interface to be based on speech (now in the direction from the machine to the human), as the machine would be conferred the capability of uttering a desired text.

Some of the applications derived from human-machine communications are the dictation systems or the reception of spoken instructions from a computer, when only one of the two directions of the communication happens. When both are present, we find the dialog systems as their major exponent, like the virtual speech assistants we mentioned before.

Also, these machine speech abilities can be used for enriching human-human speech communications, like automatic translation and subtitling or, for people with problems to speak, uttering the typed text or gestures done by this kind of users.

It has to be pointed out that ASR and machine speech synthesis are focused on speech-to-text and text-to-speech processes, respectively. Then, it is not necessary for these systems to know anything about the meaning of what is been said. This way, a human-machine spoken dialog application has to be composed, at least, of an ASR module, a text-to-speech module (both front-end interfaces) and a specific semantic processor module (back-end) that decides the response of the system depending on what the user said and the purposes of both the system and the user.

All these technologies belong to the field of Computational Linguistics. This field can be circumscribed in the widespread field of Machine Learning and, in turn, Artificial Intelligence. However, it is difficult to set the barriers between these fields, since they share a considerable amount of knowledge and methods from areas like logic, mathematics, acoustics, linguistics or cognitive science, among many others.

We could divide the efforts of traditional ASR in two different but coupled lines, which would be the acoustical modeling and the linguistic modeling.

The acoustic modeling is focused on the study of how speech sounds are produced and how we can discriminate better between them, given the difficulties added by multiple speakers, noises, and other sound effects like reverberation. This study has benefited other important applications, apart from ASR, like speech enhancement, information indexing by speaker, speaker verification or language identification (this last one also needs other supra-acoustic information). Internal progresses in these areas might also influence the progress in acoustic modeling.

In turn, the linguistic modeling is focused on the study of how language occurs and how we can predict the next words given some context. In ASR, this is usually modeled in a statistical form. However, in other research domains like Natural Language Processing and Understanding we can find more sophisticated versions, with multiple mechanisms, that have led to other

applications like machine translation or natural language question and answering systems. Again, any improvement in these sub-areas can potentially impact on each other, and the same can be claimed for upper-level areas, which lead us to a more holistic vision of fields like Computational Linguistics or Machine Learning.

1.2 Aim of the work

In our thesis work, we will focus on the ASR arena. The relevance of contributing to this field can be already inferred from the importance of speech in the growing number of machine applications we exemplified above, which require of robust and efficient recognizers. More specifically, we are interested in ASR systems that are able to dynamically adapt the **dictionaries** they can manage, understood as a vocabulary of words for which we know their pronunciations as well their syntax and semantics (these last two features would be reflected on the language model). Thus, such systems would be able to adapt to the language of the speech being decoded at every moment, which can belong to very heterogeneous domains.

As we mentioned, there are two main models inside a traditional ASR system: an acoustic model and a language model. Our efforts are centered in the second, where the pursued adaptation is to occur. This would require, firstly, a general and robust acoustic model capable to work in different noise and speaker conditions. And second, a language model susceptible to be modified by a special strategy that identifies the most appropriate vocabulary at every moment and accordingly updates the language model. We want this mechanisms to work in a fully unsupervised way.

We find this new feature of ASR systems a natural evolution of their capabilities, specially if we attend to their history.

At the end of the past millennium, these systems had an error rate in their output transcription (known as Word Error Rate, WER) at least an order of magnitude higher than for human recognition across a wide variety of tasks (e.g. WER of 43% in machines versus 4% in humans), as shown in [Padmanabhan and Picheny \(2000\)](#).

This notable difference is understandable due to the hard problem of machine learning that involves the task. In a general view, we can think of the spontaneous nature of the speech, which is full of variations in pitch, duration or emotion for the same word, even within the same speaker. Also, some extra-lexical events are present, like hesitations, false starts, ungrammatical constructs, laughs or coughs. And then, there are infinite kinds of situations, with different language domains and ways of expressing ideas with language, multiple types of noise, audio qualities and a high variety of speakers and accents. Therefore, without the proper computation power at that time (although good models already existed), to construct a reliable general recognizer was hardly unimaginable. That is why the recognition tasks in this period aimed at controlled scenarios, mostly without natural speech.

During the first decade of our century, the size of the vocabularies managed by these systems experimented a growth from 10K-20K words to the order of 100K words. The better computational capabilities and the refinement of the models made this happen, and also promoted the interest in facing more realistic scenarios with spontaneous speech in an increasing number of situations.

Finally, with the reborn of neural networks in the past decade (mainly due to the parallel computation allowed by the Graphic Processing Units, GPUs), the performance of the recognizers has increased considerably, narrowing more and more the gap with human performance. Thus, it is possible to find ASR systems with vocabularies of some hundreds of thousands words or even in the order of 1M words. A language model managing such number of words can be useful for a general purpose scenario, with a coverage of multiple topics and speakers of different linguistic skills (usually, no more than 10K words are employed by a single person). However, if we intend to take in all the richness of different language domains and topics, such a language model would become inoperably big.

By limiting the vocabulary and language modeling to fit some specific domain or topic, we would be able to implement a recognizer with high performance thanks to the modern technology and the algorithms that are available. And if we keep adapting to some extent our language model to the type of language present in the speech being decoded, the performance of an ASR system may get closer to the best recognition possible, provided that the system is ready to deal with the audio conditions. Then, the main purpose of this thesis work is to study and propose strategies that make possible this language adaptation, without the need of explicitly identifying the topic or domain of the speech being decoded, but attending to the vocabulary being used and updating it accordingly through time (with consequences also for the language model, as we will see). For doing so, we have to attend not only to the frequency of use of the words in the system's vocabulary, but also to the words not present in the vocabulary (named out-of-vocabulary (OOV) words) that we can somehow suspect that were actually uttered.

Sometimes, OOV words are just rare words that hardly happen. However, it is usual that words not representative in some domain are indeed important in others (like proper names, technicalities or poetic figures with all kinds of words), and so they are likely to be discarded from the original design of the vocabulary of the system. Thus, a symptom that our recognizer may not be modeling properly the language of the input speech is the occurrence of such OOV words.

For this reason, our adaptation strategies will focus on the OOV detection and search mechanisms that try to **discover** the missing words. Then, as new words are found they could dynamically update the system's dictionary and language model. To update the language model, we need to search for text containing contexts (surrounding words) of the new terms, so we can properly learn the sequences of words that the new terms are part of. This means that we will need some knowledge source where to perform those searches.

Fortunately, the size of the resources employed by modern computing tasks involving language and semantics like Natural Language Processing (NLP) is increasing. We can find repositories with vast amounts of information in the form of documents or also Uniform Resource Identifiers (URIs) and the connections between them. The organization of this knowledge entails the creation of massive vocabularies, thesauri or complex ontologies and logics, to name a few. Therefore, we will make use of some of these resources, which seem to fulfill our requirements of dictionary extension and proper organization, so our searches can be done efficiently. Furthermore, some of these resources were designed for machine operability, as the case of Linked Data resources (see sections [4.2](#) and [4.5.4.1](#)), so the automaticity we seek for our systems can be easily achieved.

To manage such an amount of words and resources, a numeric representation of them is usually desirable, since computers perform better when working with numbers, vectors or indices rather than with characters, words, documents or URIs. In fact, most important machine learning algorithms such as Deep Neural Networks (DNNs) or Support Vector Machines (SVMs) require numeric values at their input (in contrast with, for example, decision trees). For these reasons, we

will study the vectorial representations (popularly known as embeddings) of words and resources.

Finally, in the experimental framework of this thesis work, we have developed ASR systems with the mentioned Term Discovery strategies that we propose, making use of external knowledge sources for the searches related to the new words. These systems will be evaluated over different speech corpora that appear interesting because of their characteristics: in acoustic terms, they have many speakers and recording conditions; in language terms, they belong to different domains where various topics arise. The experimental setup will be described in more detail in chapters 4.5 and 5.2, depending on whether we are evaluating the sole Term Discovery strategy or its dynamic use, respectively.

State of the art

In this chapter we present the fundamentals and most important advances of the different areas related to our thesis work. In section 2.1, we present the Term Discovery strategies for ASR that deal with the problem of OOV words, a major issue when we intend to perform automatic dictionary updates, as we do. In section 2.2, we point out the techniques related to adaptation of language models, which will be useful for dynamically adapting our language model to the speech being decoded, helped by the dictionary updates and when enough material for learning a new model exists. And in section 2.3, we introduce the distributed representations of entities (like words) that neural networks can compute, and which will take an important role in our ASR systems, specifically for the Term Discovery strategy we will propose.

It is important to mention that in this work we will use traditional-architecture ASR systems, i.e. with separate acoustic and language model. Recently, there are other works that substitute all or most of this architecture by a single deep neural network of some kind that implicitly comprises both models, like in [Hori et al. \(2017\)](#), known as End-To-End ASR systems ([Graves and Jaitly, 2014](#)). Then, the input to such a network can be directly the audio signal (usually its spectrogram) or the features extracted after processing the audio signal, and the output is the recognition result.

End-To-End systems can work at character, subword (e.g. syllables) or word level, so in the two first cases it could be possible that they can deal with OOV words to some extent. However, this would depend profoundly on the training material, where we arrive again to the problem of generalization versus specialization to some topic or topics. Apart from this, it remains unclear how to apply some external strategy that modifies these systems so they can deal properly with OOV terms. A straightforward solution in line with our thesis would imply to retrain the neural network with speech samples containing sequences of words with the new terms retrieved by our strategies, so an additional source of information in audio format has to be employed in order to find those speech samples (e.g. Youtube videos).

In any case, in traditional-architecture ASR systems, to keep the acoustic model separately allows us to modify the language model in a more controllable and understandable way, and that is why we chose them. Also, this kind of adaptation strategy could be also used without much modification in End-To-End systems whenever they use additionally a language model to refine their results. In the chapter reserved for future work, we indicate a convenient End2End architecture with such capabilities (section 7.2).

2.1 Term Discovery strategies for ASR

In this section we study the different ways described in the literature for dealing explicitly with OOV terms in ASR. These strategies are employed when a palliative workaround is not feasible or adequate. One of the most common workarounds is to employ a huge vocabulary, so it is likely that the threat of OOV terms get attenuated. However, the terms involved would cover so many topics that a proper language model (LM from now onwards) would be of an enormous size. In order to avoid this, another workaround would involve a LM adaptation to the topic being listened to, which requires to have a topic identification module and the subsequent LM for each of the topics we want to handle, something that we want to avoid in order to be able to adapt to any possible and unexpected topic (we will go back to this approach in the next section, 2.2, focused on LM adaptation).

We are interested in the strategies that face the OOV problem directly, since OOV terms can implicitly inform us about the topic and domain of the speech that we are listening at every moment. There is also another problem: some OOV words are created on a day by day basis, so it is difficult to create an adequate dictionary of terms that does not become obsolete (for example regarding the names of new companies). Therefore, some updates are needed to be performed and, if we are aware of those OOV occurrences, we can propitiate in time the updates. An alternative would be to perform periodic updates, making use of some external source of knowledge, though it would remain unclear which parts of those sources would be taken into account. We prefer to employ strategies that solve this problem in a more direct way.

The mentioned strategies are known as Term Discovery strategies, whose purpose, then, is both to identify OOVs and propose a suitable transcription for them. We would like to highlight that this process has to occur with no information relative to each OOV term other than the speech associated to it and its context. Thus, this challenge should not be confused with others that present some variations (sometimes very slight), so we will explain now what are the differences among them.

We can classify the challenges related to Term Discovery in two groups: one involving some kind of **discovery** or resolution of terms, and one just involving some kind of **detection** of terms. In the first group, we find Term Discovery (the one subject of this thesis work) and Spoken Term Discovery (Versteegh et al., 2016). In the second group, there are Spoken Term Detection (Tejedor et al., 2017), and Query-by-example Spoken Term Detection (Rodriguez-Fuentes et al., 2014).

Spoken Term Discovery is a challenge oriented to automatically identify words and phrases in speech by searching for acoustic patterns repeated consistently. As it occurs with Term Discovery, no prior information about which kind nor number of words are to be found in the speech. However, this challenge has more to do with settings in which few or even zero resources about a language are allowed or available, so the vocabulary might be required to be built from scratch, applying intense acoustic processing and pattern clustering, in contrast with Term Discovery, where most of the vocabulary is already set in the ASR systems.

In **Spoken Term Detection**, a list of terms is given with the purpose of searching for them in some test speech (such list might include both in-vocabulary (IV) and OOV terms). If such list is provided prior to the decoding of the test speech, we would be talking about a much easier detection task, **Keyword Spotting** (Smídl and Psutka, 2006), where we could explicitly prepare the system to identify such words (and therefore all words could become IVs).

Nonetheless, in Spoken Term Detection, the decoding of the speech is carried out without any prior knowledge of the words to find. This also occurs for Term Discovery. However, for Spoken Term Detection, the list of terms to find (in grapheme form) is provided immediately afterwards, so the search inside the processed speech (in the form of a lattice of phones, for example) could be directed by the phonetics inferred from the terms in the list, thanks to a grapheme-to-phone conversion, which is specially useful for searching OOV words. In the case of Term Discovery, the sole strategy has to find where in the speech an OOV might be uttered, if any, and then it has to obtain some information of the OOV in order to transcribe it correctly (for example, making use of the phones found in the region of the speech believed to contain an OOV).

The previous explanation can show insight about the different levels of difficulty between Term Discovery and Spoken Term Detection. Then, Term Discovery is a more challenging task that arises in realistic scenarios, in which we may have no clue about what particular terminology or which new terms of the language are going to be used.

Lastly, **Query-by-example Spoken Term Detection** is just a variant of Spoken Term Detection in which the list of terms to find is not in a text form, but in audio recordings where different speakers utter just the terms themselves, having typically one to three samples per term. The difficulty to perform a transcription of the terms lead to strategies focused on the acoustics, complicating considerably the plain Spoken Term Detection task (although not necessarily the systems behind, which can be in fact much simpler than those designed to deal with the plain Spoken Term Detection task).

From now on, we will refer to Term Discovery as **TD**, which should not be confused with Term Detection, specially since the differences are very notable, as we have just reviewed.

Then, let us focus on how TD strategies handle the OOVs they have to discover. For doing so, they usually make use of an OOV word modeling so the LVCSR system is able to distinguish between normal words and OOVs. As it occurs with normal words, OOVs have to be modeled in both acoustic and linguistic terms, in the manners that we will see later, where sometimes just a single OOV model is needed to host any kind of OOV word. Thus, OOV models may help in the direct detection of OOV occurrences and possibly give some hints on their orthographic transcription, although it is also true that, additionally, some OOVs can still be mistakenly substituted by IVs by the ASR and this fact will lead us to the necessity of more elaborated detection strategies.

A complementary detection strategy, apart from having some OOV model, would require to additionally find and manipulate low-confidence decoded speech segments, which may be produced by different causes apart from an OOV. These low-confident segments can be detected through some algorithm that handles the acoustic and linguistic scores that are produced in the decoding process of traditional ASR systems (in order to compute a final word (or speech segment) confidence score) or by employing additional techniques able to detect general ASR errors, as in Ghannay et al. (2015).

We study in the next subsections the two main approaches in the literature that actually involve some OOV model, which we can denominate as *TD with single OOV language modeling* and *TD with subword OOV language modeling* approaches. Then, in a third subsection, we take a look at how the orthographic transcription of OOVs is performed.

2.1.1 Term Discovery with single OOV language modeling

In this approach, an explicit OOV word model in the vocabulary is used by adding a new term like $\langle UNK \rangle$, an auxiliary term that is intended to enclose any kind of OOV word. As we said, this model has to be reflected both in acoustic model (AM) and LM terms, finding many variants in different works. Nonetheless, they have in common that, in LM terms, the tokens handled by the LM include the special word $\langle UNK \rangle$, so the occurrence of an OOV is quantified explicitly. This quantification can be obtained, for example, making use of standard smoothing techniques in n-grams LMs (Chen and Goodman, 1999), so the discounts we applied to different n-grams can be absorbed by the special word $\langle UNK \rangle$. Also, we could quantify it directly from the training texts, selecting an amount of words to be excluded from the vocabulary, so all their occurrences would be used to model $\langle UNK \rangle$.

Regarding the corresponding OOV acoustic modeling, we mention several particularities of various works. In Asadi et al. (1990), any sequence of two or more context-independent phones can represent an OOV in AM terms. This means that the acoustic realization of OOVs are modeled at the phone-level, from which we try to extract the corresponding orthographic transcription. Instead of these phone-level sequences, we can find more sophisticated implementations at the syllable-level (Kemp and Jusek, 1996), multi-phone-level (Bazzi, 2002), or semi-word-level (Schaaf, 2001), where an OOV is conceived as a head plus a tail model (the head would be made up of regular phones, modeling the prefix of an OOV, while the tail consists of a sequence of generalized “mumble” phones, in charge of modeling the remainder of the OOV).

A more recent example is the work of Réveil et al. (2014), which also involves larger vocabularies than those found in the previously referred works. Here, a subword-level modeling of OOVs composed of n-grams of phones (up to 9) is employed, appearing to be suitable for carrying out the correct subword transcriptions of OOVs. However, such long sequences may not work properly with new terms like neologisms, which could be constructed with unexpected sequences of phones.

2.1.2 Term Discovery with subword OOV language modeling

As regards the second approach, a LM is used to model words and subword units simultaneously. Thus, an OOV is not detected as a singular word class (no special $\langle UNK \rangle$ term is used), but by its constituent subword components, directly in one conventional search. This idea implies the definition of these subword units, as in the previous approach, whose modeling would require, apart from their pertinent acoustic model, how they are conceived in language model terms, so the system can decode sequences containing both normal words and subword units.

The type of subword units vary among works, finding different ones that cover: individual phones and sequences of few phones (Klakow et al., 1999); graphemes (sequences of phones that yields an orthographic representation) (Bisani and Ney, 2005); or even 5-gram phones (Rastrow et al., 2009). All these subword units can be acoustically modeled by the general AM of the system, which already handles phones. In LM terms, each subword unit is quantified separately, regarding its occurrences in a training text, composed of normal words and subword units (for example by substituting some normal words by their decomposition in the defined subword inventory).

In order to improve the detection of OOVs, some works like Rastrow et al. (2009) or Parada et al. (2010a) make use of additional classifiers on the output of the ASR systems, i.e. sequences of

words and subword units, possibly in a graph form to enclose multiple paths. The classifiers used in these works are based on Maximum Entropy (ME) and Conditional Random Fields (CRFs), respectively. These classifiers aim to label regions of the speech as IV or OOV, so it can be seen as a sequence labeling problem. CRFs were shown to be more suitable since they are able to find an optimal labeling for entire sequences, rather than the local decisions that the ME classifiers offer.

In our thesis work, we are more interested in the single OOV language modeling TD approach, mainly because its OOV modeling allows us to easily treat any OOV as one single auxiliary word. Thus, we do not depend on the frequency of the subword units during training, as it occurs with this second approach, which can be a danger when intending to face OOVs that may be composed of subword units distributed distinctly from what is observed during training, as we do.

2.1.3 Orthographic transcription of OOV terms

In all the previous works concerning both approaches, the orthographic transcription of the OOVs found is made simply by some pre-built correspondence between graphemes and some subword units like phones or syllables. Thus, no certainty about a correct OOV transcription is reached (i.e. there is no correct written form of an OOV available for comparison), which may be useful for deciding whether or not to incorporate it into the system's transcription, being this an appropriate feature we desire for the strategies proposed in this thesis work.

Some works try to address this issue by different means, like, for example, [Palmer and Ostendorf \(2005\)](#), [Parada et al. \(2010b\)](#) or [Sheikh et al. \(2017\)](#).

In the first one ([Palmer and Ostendorf, 2005](#)), which focuses on the OOV resolution of proper names (PNs), they generate a large PN list by filtering news broadcasts. When a PN error is detected in the transcription, the phone sequence of the pertinent ASR hypothesis is used to prune the PN list, by computing a distance that measures the string matching between this sequence and the pronunciation of each PN on the list. The resolution, then, can be completed by taking the first candidate of the ranked survivors.

They also propose to carry out a LM adaptation with the survivors and their surrounding sentences of the news from which they were collected, computing with this LM the perplexity of the transcribed sentence for each of the survivors, so this additional LM score might allow a finer-grained decision about the proper candidate.

In the second work ([Parada et al., 2010b](#)), web harvesting techniques are used for OOV resolution. Once an OOV region is detected in an utterance, a single query is built with some relevant words in that utterance, based on tf-idf scores. Then this query is sent to the Google search engine, and the top 20 documents are retrieved, scanning them to create a list of potential substitutes for the OOV region detected (those new terms different to already-known IVs), looking within a fixed window around where the query terms appeared, so it is more probable to find the real OOV uttered. Finally, this list serves as a source of query terms for a standard Spoken Term Detection system, transforming the Term Discovery scenario into an Spoken Term Detection one.

The third work ([Sheikh et al., 2017](#)) returns back to the focus on PNs. Similar to the first one, they collected a list of OOV PNs through text news from the Internet belonging to different topics. In this case, they associate a semantic context to each of the OOV PNs collected, making use of the news where they were found, in the following manner:

First, words in the corresponding news of an OOV PN are collected; and second, with the use of Skip-gram word embeddings (to be described in section 2.3), the average vector of the embeddings of the collected words is computed (equivalent to a centroid operation). This average vector becomes the semantic context associated to an OOV PN.

Then, a two-stage strategy is employed. In the first one, a certain portion of the utterance is delivered to the LVCSR system, from which they obtain the 1-best hypothesis. After this, they represent the semantic context of this hypothesis. Since it is composed of words, they can proceed the same way as when they computed the semantic context for each OOV PN.

Thus, they can obtain a ranking of OOV PNs regarding the proximity between the vector representing the context of the spoken chunk and the vector associated to each OOV PN, by computing cosine distances in the embedding space. A list of the closest OOV PNs is then used to update the vocabulary and LM of the LVCSR.

Finally, in the last stage, a second-pass speech recognition is carried out to hopefully recognise the missed PNs.

2.2 Language Model Adaptation

In this section we review the most effective techniques found in the literature for performing language adaptation. Since the introduction of this document, we put special emphasis on the difficulty of a language model to cover properly a variety of domains, topics or styles (we will refer indistinctly to any of them). On the one hand, to employ a model that is exhaustive in the modeling of multiple topics is likely to reach such a size that prevents its normal use in applications like in the ASR area, apart from the fact that it is likely to experiment deteriorations in the performance due to the high confusion arisen. And on the other hand, too much specificity for modeling a certain topic would make the model unusable in other domains.

Therefore, multiple techniques have been developed in order to perform a contextualised adaptation of LMs, so we can dynamically react to a given input source (which can be text or, in the case of ASR systems, speech), in many applications such as dialogue systems (Lucas Cuesta, 2013) or statistical machine translation (Lu et al., 2012), apart from the general ASR arena (e.g. for improving the recognition by performing dynamic topic adaptation (Echeverry-Correa et al., 2015)). In the next subsections, we describe some of these techniques, mostly developed for statistical LMs, although more effort is being put recently into the adaptation of LMs based in DNNs ((Chen et al., 2015), (Moriokal et al., 2018)), not discussed here for scope reasons.

2.2.1 Cache-based models

These models are based on just one statistical LM (of considerable size if we intend to cover multiple domains) which employs a special strategy for dynamically modifying the probabilities of its words and sequences of words (Kuhn and De Mori, 1990). This strategy consists of a cache constantly filled by the processed words, so any word or sequence in the cache would increase their probability for the next future. The assumption behind it is that when a certain word or sequence is used, it might be used again soon. Also, it makes sense to apply a decaying history approach by

considering that the more recent a word is, the higher its chance to re-occur (Clarkson, 1999).

This technique is also associated to the one called **triggering**, in which if certain words appear, then some other words or sequences are triggered to have more probability (usually just word pairs, i.e. one trigger word and one triggered word) (Tillmann and Ney, 1996). This technique could be able to capture longer-span relations between words than the cache technique, for example if the triggered words are chosen to be topic-related to the words already uttered, it is probable that they will be employed in the near future. Then, a combination of both techniques might capture both short- and long-span relations between words.

As regards our thesis work, these kind of models present two crucial limitations for our purposes: first, they operate over a fixed vocabulary; and second, the knowledge about the relations between the words in the vocabulary (e.g. for the triggering processes) is required, something unknown when trying to handle OOV words.

2.2.2 Mixture-based models

In this type of models, we employ separate LMs for different subsets of the training data. The training data is usually composed of a set of documents, so the subsets we want to define would be subsets of those documents. Thus, we make sure that the language information of each subset is not lost in a global model whose statistics might harm the desired modeling specificities. Then, we have to define these subsets, taking into account that small ones might result in models that over-estimate the characteristics of that subset.

If the data is already labelled, e.g. by topic, then the subsets can be defined by topic or by some grouping criteria (based, for example, in some similarity metric). In case those labels are not available or we prefer an automatic grouping, we can opt for an unsupervised clustering (Clarkson, 1999), for which various algorithms exist, based, for example, in Probabilistic Latent Semantic Analysis (Hofmann, 1999) or, more simply, in a cluster merging criterion that attends to the number of words shared by pairs of clusters, in an iterative fashion (Iyer and Ostendorf, 1999).

It is usual that the specific LMs of each subset are not used individually, but interpolated with a general, background LM trained on a heterogeneous corpus with various topics, so when a specific LM is too small, we are still able to not lose the modeling of the general use of the language. This interpolation with the background LM can occur with just one specific LM (*hard* approach), or with some of them (*soft* approach), with the weights for each model properly distributed so as to adapt optimally to the speech to decode. This weighting is already a problem by itself, for which different solutions were proposed, as we will see. Then, to compute the probability of a word w given its past history h with such an interpolation, we can make use of next equation:

$$P(w|h) = \lambda P_B(w|h) + \sum_i^K \alpha_i P_{S_i}(w|h) \quad (2.1)$$

where $P_B(w|h)$ is the probability computed by the background model, $P_{S_i}(w|h)$ is the probability computed by the i th specific model out of K and λ and α_i are the interpolation weights given to the background model and the i th specific model, respectively. Of course, $(\lambda + \sum_i^K \alpha_i) = 1$ to keep

the probabilities consistent. When $K=1$, we are in the *hard* approach, and otherwise in the *soft* approach. For the second one, K is often much smaller than the total number of specific LMs at our disposal, so we only consider the K ones that may fit best the specific speech to be decoded.

In order to compute those interpolation weights, we can opt for setting them experimentally (in some development set); estimating them under some criterion, such as Maximum A Posteriori (MAP) (Wang and Stolcke, 2007); or dynamically modifying them depending on the current situation of the dialog (Lucas Cuesta, 2013) or on the topic detection of the speech (Echeverry-Correa et al., 2015).

When we intend to perform an adaptation to a specific input of unknown topic, a **topic identification** of the input has to be done in order to select the topic or topics to adapt to. Although this identification can occur in simple terms such as finding some keywords from a given set, it is usual to employ a more sophisticated scheme, like the vector representation approach.

In the last approach, every document that composes the training data is represented in some vectorial space, based for example in Latent Semantic Analysis (Deerwester et al., 1990) or in Latent Dirichlet Allocation (Blei et al., 2003). Indeed, this kind of representation can guide the automatic clustering of topics, as we mentioned in the work of Hofmann (1999). Then, the same representation used for the topics would be applied to the current input we are handling, so some similarity measurements can be computed in this space in order to find the topic or topics more suitable for adaptation. Furthermore, these measurements could also play a part in the computation of the *soft* interpolation weights, as in Echeverry-Correa et al. (2015), described next. In their work, they perform topic adaptation employing a two-stage scheme. In the first stage, a segment of the speech is transcribed by an ASR system with a background LM. Then, a topic identification algorithm is employed on this transcription, resulting in an adaptation of the LM by interpolating the background LM with one or more specific LMs. Then, in the second stage, the same speech segment is decoded again by the ASR system now with the adapted LM. This way, by repeating this process over all the segments of the speech, we can dynamically adapt the LM accordingly to the topic that is spoken at every moment.

For our thesis work, we are not interested in an adaptation that requires a topic identification module, but in an adaptation driven by the OOVs that appear in the speech. However, the mixture-based approach is still interesting for us in the sense that we could employ a background LM that is interpolated with small and specific LMs implicitly related to new and unknown topics from the input speech (although in our case the background LM should be already tuned to the input speech to some extent), obtaining thus an updated version of the background LM. These specific LMs would be created from the word contexts associated to the OOVs that we want to incorporate to the system's dictionary over time, so the background LM is continuously transformed towards the proper direction. Therefore, we will invest effort in the design of a special mixture-based LM working in such conditions.

Finally, we mention a couple of works that consider OOV words for modifying the LM of ASR systems, although in a more limited manner compared with the scheme we propose, as we will argue. In Naptali et al. (2012), a class-based n-gram LM is employed to host special OOV terms that can be enclosed in a word class, which however limits the types of OOV words that the system is potentially able to deal with (mainly proper names). And in Currey et al. (2016), an n-gram LM is extended with new n-grams containing OOVs that are going to be included in the vocabulary. Those new n-grams are built by attending to similarities with existing IV terms. Nonetheless, this means that the underlying grammar is not properly transformed and evolved as in the previous mixture-based scheme that we propose.

2.3 Distributed representations from neural networks

In the last section of this chapter, we take a closer look to the distributed representations (i.e. embeddings) of some kind of entity that can be obtained from neural networks. The motivation was already stated in the introduction of this document, where we pointed out that computers perform better when working with numbers, vectors or indices rather than with characters, words, documents or URIs.

If, for example, we want to manage words through some algorithm (like a DNN), as a first solution we could opt to transform each word into an index vector: a vector with a dimension equal to the size of the vocabulary where only the index of the word takes the unit value (and the rest are zeros). This is known as one-hot codification. However, the size of current vocabularies can get close to the million of words, leading to impractical models. This fact brought about the creation of embeddings, which started to become popular after the presentation of very simple unsupervised methods for their training, known as Word2Vec (Mikolov et al., 2013a).

An embedding, then, is a distributed representation of an entity in a vectorial space, with real-valued vectors (not just 1's and 0's), and with a dimension typically not superior to 300 (in comparison with the $\sim 1M$ of the previous example). Thus, a set of embeddings will represent the different entities distributed over a D -dimensional vectorial space, maintaining some meaning in the position of an entity with respect to the others, as we will see. This kind of representation appears in contrast to discrete representations (such as one-hot), that are only able to indicate membership(s) to some set, for example through organizations such as trees of clusters of entities (provided that their memberships are worked out in advance), and where a vectorial neighborhood does not necessarily imply any significant relationship.

In the next subsection, we give more details about the purposes and the insight behind embeddings. After that, we explore the different usages that can be found in the literature, including embeddings of different entities (emphasizing those types that will be relevant to this thesis work) and applications in various arenas (including ASR). For scope reasons, we have only focused on embeddings obtained from neural networks, leaving aside other approaches based, for example, on distributional similarity (Sahlgren, 2006).

2.3.1 Purposes and insight

Two objectives are pursued when looking for an embedding. The first, already mentioned, is a reduction in the dimension of the representation of the entity. If we return to the vocabulary example, the reduction from the one-hot codification can experience up to four orders of magnitude, since typical word embeddings use 50 to 300 as dimension. The second is a semantic compactness in the space of representation conformed by all the embeddings, so some topological properties arise, which allow a semantic discrimination between entities. To understand this, we have to think of the way they are obtained. We exemplify next the case for word embeddings, although similar schemes can be used for other type of entities.

In its simplest form, word embeddings are taken from the weight matrix W from the single hidden layer of a trained feedforward neural network (which correspondingly has another weight matrix S_w and its corresponding biases S_b for the softmax output layer). At its input I , we can introduce words employing 1-hot codifications (I would be a matrix of dimension $1 \times V$, being V

the vocabulary size), so the i th row of W would become the embedding for the entity with index i . In theory, the output O of the network is trained to activate with 1's the positions of those words in the vocabulary that hold some kind of **relation** with the selected input I , and 0's in the other positions, as if we employed several 1-hot codifications simultaneously (O would be a matrix of dimension $1 \times V$). In practice, several efficiency arrangements are made, like for example limiting the number of negative samples (0's at O) to a certain number, as we will see. So, O can be obtained with the next equation:

$$O_{(1 \times V)} = \sigma^*(I_{(1 \times V)} \cdot W_{(V \times D)} \cdot S_w_{(D \times V)} + S_b_{(1 \times V)}) \quad (2.2)$$

where D is the desired dimension for the embeddings. σ^* refers to some softmax function, designed with regards to an efficient training loss, limiting the number of negative samples (a technique found for example in the Skip-gram version of Word2Vec (Mikolov et al., 2013b)).

The semantics that they should assimilate depends on the relation between I and O . For example, in the Skip-gram version of Word2Vec, each word in a text is used as I , and its context words as O (details about the training operation are left to the referenced paper, (Mikolov et al., 2013a)). We can expect that a word represented in this model would be close to other words with similar contexts, which agrees with the distributional hypothesis of Harris (1954). The intuition behind this idea lies in the following interpretation of the network weights: while W would be in charge of representing words, S_w and S_b would be in charge of representing the contexts of each word, so two words represented closely with W would experiment a similar forward propagation through S_w and S_b that will lead to similar O 's (i.e. similar contexts).

Also, as the embeddings start to get closer during the training, they are getting further away from the non-similar ones at the same time, because of the discriminative power of neural networks whenever we use contrastive samples. With this reasoning, we believe that these semantic relationships will extend consecutively among the words, keeping a proper distribution of semantics throughout the overall model. So, once the model is trained, some topological properties may arise, allowing the semantic discrimination we pursued at the beginning. Thus, apart from finding the vector $\vec{king}$ close to the vector $\vec{emperor}$, we would be able to do some arithmetic in this vector space and finding that $\{\vec{king} - \vec{man} + \vec{woman}\}$ is close to the vector $\vec{queen}$.

2.3.2 History and usages

The study of obtaining a distributed representation of entities dates back to Hinton (1984). However, we have to wait for the resurgence of neural networks to find functional word embeddings, like the early example in Bengio et al. (2003). The initial purpose was to be able to use words in machine learning algorithms in which a vector representation was required, and avoiding a one-hot codification that made them unfeasible.

The previously cited work and others such as Turian et al. (2010), Collobert et al. (2011) or Huang et al. (2012) obtain their word embeddings inside a neural network with a few hidden layers trained for language modelling and other NLP tasks. This means that by the end of the training, apart from being able to acquire the embeddings, an operative neural language model is available.

The popularity of Word2Vec is due precisely to the unnecessary to end up with a fully-functional

neural network, which is translated in a very fast and simple training scheme. Here, the network is still being trained as a language model, in the sense that given some context words, it is trained to predict the central word of this context (or vice versa). Nonetheless, this training occurs in an incomplete way, since an incomplete 1-hot is used in the output layer, so that only the neurons that correspond to the pertinent words (plus a random selection for negative sampling (Mikolov et al., 2013b)) are subject to the backpropagation training algorithm. In any case, this has proven sufficient for a proper embedding training. Thus, state-of-the-art embeddings can be achieved with just one hidden layer and much shorter training times.

As we saw, the kind of **relation** between the input and output instances during training determines the semantics present in the embedding space. As we mentioned, Word2Vec is trained to make use of the context of each word. This context is simply the C words on both sides in the text used for training. As stated in Bansal et al. (2014), C plays an important role as regards semantics: when large (e.g. 10), the words are automatically grouped together in a topic-based relationship; when small (e.g. 1), more importance is given to sharing the same POS tag so as to be grouped together.

There are works that choose the context of each word more accurately (and then not just any surrounding ones), based for example on the syntactic relationship that each word has with other words in a sentence, obtained using parsing technology (Levy and Goldberg, 2014). For example, in the sentence “*Then, a dog white as snow barked*”, the context words for *dog* would be better $\{a, white, barked\}$ than $\{then, a, white, as\}$, for a $C=2$.

Also, in Bian et al. (2014) we find an enrichment of the context of each word, which is extended with additional specific text-related tags for each word that makes up the context, like its root, affixes, POS tag or synonyms.

In all these examples, the context used is always the **local** one that we find for every occurrence of a word inside a text. With the aim of leveraging statistical information throughout the whole text, we find GloVe (Pennington et al., 2014), which makes use of **global** word-word co-occurrence count matrix from all the sentences of the text (details can be found in the referenced paper), rather than attending to the words inside the local contexts windows that consider separately every sentence of the text (as in Word2Vec). However, this approach would not allow sentence-dependent information to be incorporated into the external layers of the network, as we saw in Bian et al. (2014), since this local information has to be extracted as we train the neural network with each of the specific sentences in a text (in GloVe, the particular context of each word occurrence is abstracted with the construction of a global word-word co-occurrence count matrix before the neural network training). So, the choice of the approach should depend on the purpose and the information available.

Further applications of these training schemes for word embeddings (especially Word2Vec), have led to the production of embeddings for **other types of entities**. The most immediate one appears when passing from word-level to phrase-level (Mikolov et al., 2013b), and then to sentence-, paragraph- or even document-level (Le and Mikolov, 2014), Doc2Vec, where a distributed representation of an arbitrarily large set of words can be achieved.

In different veins, we can find embeddings for entities such as tags (e.g. *#sport* or *#summer*) assigned to videos (Singh et al., 2016), Tag2Vec, where a video can also be mapped in the same tag space and then find relevant tags for it (the nearest ones).

Similarly, tags for documents such as blog posts or news articles are embedded in DocTag2Vec

(Chen et al., 2017b), learning the representation of words, documents and tags in a joint vector space simultaneously, so a new document can be represented and related to close tags. A similar work also finds user representations through the use of document and tag embeddings, with the purpose of personalized tag recommendation (Kataria and Agarwal, 2015).

The **ASR arena** has also benefited from the use of embeddings, for example Audhkhasi et al. (2016), Ghannay et al. (2015) or Salamea et al. (2016), apart from the work of Sheikh et al. (2017) that we already saw in section 2.1.

In Audhkhasi et al. (2016), LMs based on neural networks are used for ASR. In order to represent words in these networks, they use GloVe word embeddings, as opposed to embeddings learned with other neural network language models that are optimized to model local n-gram dependencies. It is shown that the ability of GloVe representations to capture semantic information demonstrates improvements in both perplexity and word error rate.

In the work of Ghannay et al. (2015), a neural network is used to detect ASR errors. The network is fed with the 1-best output from an ASR system, together with the ASR confidence score and other features of lexical, syntactic and prosodic nature. Word embeddings are used to represent each of the words in the 1-best path. Although this type of error detection can be used to detect OOVs in a TD strategy, different methods are normally employed for this purpose, which do not require such an elaborate training, as we saw in section 2.1.

Finally, in Salamea et al. (2016), we can find a new type of embedded entity: phone-grams (i.e. sequences of phones). These phone-gram representations are obtained in a similar way to word representations, after converting each word of a training text into its corresponding sequence of phone-grams. These embeddings are used in a Language Identification task, where they are also able to carry out an inventory reduction of phone-grams by studying their dispersion in the new space, especially between high-order ones that are likely to suffer from an unreliable estimation.

Following this trend of extending the types of entities being embedded, we find an entity that appears useful for the strategies we intend to develop in this thesis work, apart from words. These are the knowledge base resources, where the searches for OOV candidates are to be performed. Therefore, we will study an appropriate representation of resources that facilitate our searches. Since these resources typically take the form of nodes in a graph, it is interesting to mention works that deal with the problem of embedding nodes in general types of graphs, like Deepwalk (Perozzi et al., 2014), LINE (Tang et al., 2015) and Node2Vec (Grover and Leskovec, 2016).

In a general overview, the neural network in charge of obtaining the embeddings is fed with the indexes of the nodes in a graph, using some mechanism to define the input-output pairings to train the network. Usually, a node is set to be related to the nodes that are connected to it, directly or after a certain amount of hops of connections. Apart from this, additional refinements can quantify more specifically those connections, specially if the arcs of the graphs are weighted in some manner.

We can also find works, like Mittal et al. (2017), which handle embeddings of entities from these knowledge graphs but without dispensing the use of the specific information gathered in the graphs. Thus, they are able of benefiting from the compact semantic representation of concepts from the embeddings and, at the same time, from the complex dependency relations that can be inferred by analyzing the graphs. However, this requires an ad hoc structure that can manage and unify both channels of operation, and which would depend on each use case.

Objectives

In this chapter we present the objectives pursued in this thesis work. The most general and fundamental objective is to propose strategies to improve the current techniques of speech and language processing oriented to the achievement of ASR systems that are able to dynamically adapt to the speech being decoded at each moment, regardless of the underlying domain or topic.

The adaptation we are interested in is concentrated on the modification of the dictionaries and language models of ASR systems. When decoding a certain speech, we want to pay special attention to the appearances of OOV words, since they might carry valuable information about the topic of the speech, and also indicate implicitly a change of topic. The resolution of these OOVs, in terms of discovering the missing words, would guide the update of the dictionaries and subsequently facilitate the adaptation and retraining of the language models, following a reactive mechanism that is repeated appropriately through time.

We believe that these mechanisms will lead to an overall improvement in ASR performance through dynamic long-term adaptation of both vocabulary and language model in order to cope with natural occurring evolution of the language caused by change of speaker, topic, new adopted terms, etc.

Our proposals will be distributed in **two parts**:

In the **first part**, we will focus on the problem of OOV words and how an ASR system could deal with them so as to improve the transcription performance at a static level. Therefore, we will design a static Term Discovery strategy motivated by the next objectives:

- To detect OOV occurrences in speech, with a precise temporal delimitation of them.
- To find the most appropriate candidates for those speech segments believed to contain some OOV term, putting special effort towards the correctness of their orthographic representation by making use of external knowledge sources.
- To explore the use of distributed representations (i.e. embeddings) of different kinds of entities (like words or knowledge base resources) that may enrich the search for the previous candidates, given the semantic power that embeddings are able to hold.

- To perform corrections in the output transcription of an ASR system, for those segments where a found candidate is disposed to substitute the original content.

In the **second part**, we aim to endow the expected dynamicity to the ASR systems. For doing so, we set the following objectives:

- To iteratively collect the terms discovered in the previous part, accompanied by a scoring scheme that give signs of their usefulness for the language used in the current speech.
- To dynamically update the dictionaries of the ASR systems, making use of a selection scheme over the previously collected terms through time. Also, any update might involve the deletion of a set of terms believed not to contribute anymore to the performance of the systems.
- To dynamically adapt the language models of the ASR systems, driven by the updated dictionaries, accessing to external knowledge sources where to learn the proper use, in actual language, of the newly incorporated words.

Finally, we are also interested in the implementation of a **experimental framework** for this thesis work, where we aim to achieve the next objectives:

- To implement experimentally the mentioned strategies of both parts.
- To perform evaluation experiments of those strategies.

The last objective will serve to obtain evidences of the improvements that our strategies can actually achieve, and to which extent. In this sense, the systems that we develop are intended to be evaluated over different speech corpora that contain the variety we pretend to face, emulating a realistic scenario of application.

Thesis work on Static Term Discovery strategies

In this chapter we cover the contributions to the Term Discovery (TD) field in ASR achieved by this thesis work. First, we will introduce in more detail the need for TD strategies in ASR systems. Then, we will present the TD strategy that we propose, pointing out the advantages that it holds in comparison with the TD strategies found in the literature (reviewed in section 2.1). After that, we describe the experimental framework designed for the evaluation of such strategy, followed by the presentation of the results obtained. Finally, we end the chapter with a section for exploring an ulterior complement for our TD strategy.

The application of these strategies in a special **dynamic** framework would allow an ASR system to experiment an unsupervised lexicon and language model adaptation. The study of such possibility and the contributions performed in this direction will be presented in the next chapter, and this is the reason why we employ the **static** qualifier to this first type of strategy.

4.1 Vocabulary limitation in ASR systems

As we claimed in the introduction of this work (chapter 1.2), a designed vocabulary limitation of an ASR system may be beneficial so as to achieve the optimal performance for speech with specific linguistic characteristics. This would be possible because the LM of the system could be focused on the topics or domains which are most relevant to the speech to decode, and so the LM can be developed to such extent that is able to capture sufficiently the linguistic richness of those topics. Pretending to do so with a large vocabulary and a vast variety of topics in the LM would result in an inoperable model or at least in a model where the confusability between topics will probably degrade the overall performance.

Of course, since the topics found in speech are likely to change over time, this vocabulary limitation should occur in a dynamic fashion, reacting to those changes. In a realistic scenario, there would be no information available about which topics are to be talked about. Therefore, we would like to react in an unsupervised manner to the mentioned changes, which we believe can be noticed by the appearance of OOV words.

In order to equip an ASR system with such abilities, the next three steps are required to be performed iteratively over time. First, we need to detect speech segments where there is a high probability of a OOV occurrence. Second, we need to find the most appropriate candidates for them and to decide whether to perform corrections in the output transcription. And third, we need to update the dictionary and LM of the ASR system with the discovered terms that appear to be useful for the current speech. The first two steps will be handled by the TD strategy that we are presenting in the next section, while the third step will be the subject of chapter 5.

It is important to remark that our main goal is to propose strategies that, once introduced in ASR systems, an improvement in overall performance should be experimented with respect to the same systems lacking the abilities endowed by our strategies. This means that the experimental framework of this chapter (section 4.5) and the next one (section 5.2) will be focused on ASR systems where a certain vocabulary limitation is set on their design (i.e. vocabularies not higher than 100K), leaving out of the scope of this thesis work other ASR systems types like, for example, general-purpose ASR systems (i.e. with vocabularies that can be much higher than 100K).

4.2 Linked Open Data resources

Before proceeding to the description of the proposed TD strategy, we consider appropriate to give a brief insight about a kind of knowledge source of increasing relevance nowadays, as we mentioned in the introduction of this document, called Linked Data (LD), and which we will employ in our TD strategy.

An LD corpus is made up of resources that can refer to persons, places, events or just to Wikipedia articles, for example. If a corpus belongs to the Linked Open Data (**LOD**) project ([LinkedOpenData](#), [nd](#)), subscribed to W3C, then all of its content is released under an open license, which makes it even more interesting for the academic community. Each resource is identified by a single, unique URI, and we can see every connection it has to other resources or constant values (e.g. an abstract or a date of birth). Thus, all of the data is made up of triplets consisting of $\langle \textit{subject}, \textit{property}, \textit{object} \rangle$, which makes it easy for computers to navigate through their resources. Furthermore, the LOD design was conceived from the start to make the connections between resources belonging to different LOD corpora as transparent as if the two resources belonged to the same corpus (in other words, we can say that all the LOD corpora can be conceived as a big single corpus).

To give an example, let us consider the case for the resource *Madrid*, which is a resource of Wikipedia in multiple languages. Firstly, since a considerable number of language versions of Wikipedia have their own LOD form called DBpedia ([DBpedia](#), [nd](#)) (English DBpedia, Spanish DBpedia, etc.), we will find in each of these LOD datasets a resource named *Madrid*. Also, we will find a connection between the different *Madrid* resources of each DBpedia, indicating equivalence (e.g. with a triplet $\langle \textit{es.dbpedia.org/resource/Madrid}, \textit{sameAs}, \textit{fr.dbpedia.org/resource/Madrid} \rangle$). Besides them, there are other datasets where we will also find a resource named *Madrid*, like Babelnet (which captures lexical definitions and their translation to a large amount of languages) or GeoNames (which collects geospatial semantic information of millions of places on the planet), and with the proper references to the previous datasets (they can be distinguished by the prefix of their URIs). Therefore, when we access the content of *Madrid* in any of the indicated datasets, we are able to extend it seamlessly with the content from the other datasets, so information from multiple knowledge areas can be gathered together.

The interconnectivity among corpora does not limit to the *subject* and *object* items of the triplets, but also to the *property* item. Thus, we do not have to redefine a property like *isCapitalOf* or *homepage* in every dataset, and we can just point to the definition found in a standardized ontology.

This interconnectivity, together with the openness, notorious size, and machine-orientation of the corpora, make LOD a great source of knowledge for tasks like the ones we are considering in this thesis work. As a matter of fact, the Wikipedia corpus has already proven to be a useful source of knowledge by itself (e.g. as a text training material for obtaining word embeddings (de Melo, 2017)). So, we can imagine that its LOD version, DBpedia, which is properly organized and connected to a vast amount of other datasets, is potentially more useful for accurately finding information about almost any topic and therefore the specific vocabulary associated to them, which is why we chose to employ LOD corpora in our TD strategy.

Since the number of resources in LOD corpora can be in the order of millions, we believe that finding an appropriate vector representation of them may lead to a more efficient use of the semantics they enclose individually and as a set. In such a vectorial form, they could be employed as easily as embedded words, enriching the possibilities of the tasks we are facing. For this reason, we propose in this thesis work a scheme for obtaining these desired representations, a scheme that we call **Resource2Vec**, and which will be employed in our TD strategy described in the next section.

The Resource2Vec scheme will be fully explained in a separated section of this chapter, 4.4, together with some evaluation metrics that will allow us to assess its modeling capabilities. The proposed scheme will resemble those already presented at the end of section 2.3, dealing with embeddings for nodes of knowledge graphs (as LOD corpora can be thought as interconnected knowledge graphs), although in our approach we simplify the way we extract data from the graphs, as we will see.

4.3 Proposed Static Term Discovery strategy

The general architecture of the proposed static TD strategy is depicted in figure 4.1. It has to be taken into consideration that a plain ASR process (i.e. without a TD strategy) would involve just a LVCSR ASR module whose input is the speech and its output is already the final transcription. Therefore, the rest of the modules in the scheme were added so as to make possible the special mechanisms involved in the TD strategy (furthermore, our LVCSR ASR module was also modified from the traditional vanilla design to facilitate those mechanisms, as we will see).

Figure 4.1: Architecture of the proposed static TD strategy

The operation of such strategy can be described as follows.

First, the speech to decode is delivered to a LVCSR ASR module. The output of this module is in a graph form (also known as lattice), where the nodes correspond to timestamps and the arcs to recognized words. We are interested in processing this output graph to pursue two independent objectives. The first objective is to process the output graph for detecting speech segments where an OOV may have been uttered. We call these segments potential-OOV speech segments (**POOV SSs**). The second objective is to find the best path resulting from the graph. Depending on the AM-LM tradeoff selected, the best path may vary. Experimentally we can find an optimum for the best path, and this solution would become the baseline transcription we attempt to correct with our TD strategy.

Second, the detection of OOVs is performed by the OOV detector module. This module receives the previous graph, and it selects the POOV SSs that will trigger the search for candidates, as it will be explained in section 4.3.1.

Third, a search for candidates is performed for each POOV SS. This search makes use of the phonetic transcription of the corresponding segment, given by the phone ASR module. Also, a vectorial representation of two types of entities (words and LOD resources) is needed, and a knowledge base in the form of LOD corpora, such as DBpedia, where the search is mainly carried out (this process will be fully explained in section 4.3.2).

And **fourth**, after selecting the best candidates for each POOV SS, we decide finally whether or not to make a correction to the baseline transcription. In a positive case, the substitute chosen will replace the original content of the POOV SS in the LVCSR transcription. When all these replacements are made by all substitutes obtained from the whole set of POOV SSs, we get the final transcription.

This TD strategy that we are proposing can be differentiated from existing strategies in that it holds the following attributes at the same time, some of them not even found in any of the other strategies, as we saw in section 2.1, and which will be explained in detail in the subsequent subsections:

- It can be inscribed in the *TD with single OOV language modeling* approach (section 2.1.1), since the LVCSR system has been modified from standard implementations to host an explicit OOV model (an additional term $\langle UNK \rangle$ in the vocabulary and LM, and a corresponding modeling in the AM) with a special treatment. Additionally, the detection of OOVs is helped by a special confidence and output graph analysis. This OOV detection and the modifications introduced to the LVCSR system will be explained in section 4.3.1.
- It does not require a second-pass audio decoding for obtaining the OOV words and correcting the transcription.
- It obtains almost always the correct orthographic representation of the candidates, as they are looked for in external knowledge sources (where however some typos might exist), instead of constructing them by just some correspondence between subword units and graphemes. Of course, a substitute word can be the true OOV word that was uttered or a different, incorrect one (and in this case not resulting in an improvement for the final transcription), but anyway the substitute words will be in a proper orthographic representation most of the times.
- It makes use of external knowledge sources for where to find those substitutes. Moreover, it can work with open corpora such as LOD. This allows a wider search in multiple domains,

instead of depending on, e.g. news broadcasts, and also allows free access. In any case, it could also take advantage of the vast amount of information that is reachable by current search engines (like Google), provided that their crawling restrictions are suppressed.

- It carries out a search based on two types of embeddings: words and LOD resources. The first ones are obtained with Skip-gram Word2Vec and the second ones with a scheme that we also propose in this thesis work, **Resource2Vec** (see section 4.4). For the understanding of our TD strategy, it is enough for now to conceive Resource2Vec as a provider of embeddings for a list of LOD resources (like those found the DBpedia corpus), in the same manner that Word2Vec provides embeddings for a list of words in a vocabulary.

We now describe the most challenging modules involved in our TD strategy: the *OOV detector* and the *POOV SSs processing* modules that are found in figure 4.1.

4.3.1 OOV detection in speech

The OOV detection occurs in the *OOV detector* module of figure 4.1, in charge of selecting the POOV SSs in which the search for candidates is to be made. This OOV detection is based on two methods, taking advantage of the combination of both, as the benefits of a similar combined use (with different methods than here) was already experimented in works such as Hazen and Bazzi (2001). The first method is based on a single OOV word modeling (a special term referring to OOVs, $\langle UNK \rangle$, is added to the vocabulary and LM), like those discussed in section 2.1.1, while the second one analyses the output graph, with a new proposal that takes into account AM-LM tradeoffs and low-confident regions.

4.3.1.1 The special word $\langle UNK \rangle$

The first of the methods used for OOV detection uses a single OOV word model by introducing a new word into the vocabulary of the LVCSR system, $\langle UNK \rangle$. This special word is intended to appear in the output transcription whenever an OOV term is uttered. For example, a transcription in the following form, where we can expect $\langle UNK \rangle$ to host the name of a country that is not included in the vocabulary of the LVCSR system:

The president of $\langle UNK \rangle$ will give an official statement on that matter.

In order the LVCSR system to handle this special token, it is required to model $\langle UNK \rangle$ in both AM and LM terms. Our design is described as follows.

In **AM terms**, $\langle UNK \rangle$ consists of one special “mumble” phone that refers to imprecise speech sounds. In turn, IV units are made up of certain sequences of phone models like, for example, the phone model s , which would model a sequence of frames (typically one every 10 ms) where the s sound is present. In the case of the “mumble” phone model, it would model a sequence of frames where vaguely any set of phones could be present. Thus, the “mumble” phone model could fit any OOV term acoustically. $\langle UNK \rangle$ would become then the best solution whenever the LVCSR is not able to decode some sequence of speech sounds with a likely IV term, given the context the sequence appeared in. This acoustic modeling for the $\langle UNK \rangle$ token based on just one phone contrasts with others approaches mentioned in section 2.1.1, where it is usual to find a modeling

based on a loop of different acoustic realizations, aimed to account for each of the different parts that compose an OOV, either at the phone-, syllable- or even semi-word-level. In our case, we opted for this simpler approach as the phonetic transcription of detected OOV regions will be given by a Phone ASR that can be properly trained to automatically learn phonotactic rules.

Now, for modeling the acoustic states of our “mumble” phone, instead of employing three states in the HMM as usual, we use eight. And instead of a forward connection flow, we allow backward connections to reinitiate some parts of the sound. The proposed topology of this phone model is detailed in figure 4.2, where it can be compared to those of regular and silence phone models (these last two models were taken as in Kaldi speech recognition toolkit (Povey et al., 2011)). It is important to highlight the backward connections of the silence and “mumble” phone models. This can be seen as an attempt to model a complex sequence of acoustic events with no predefined order.

Logically, trying to model a whole word (in our case an OOV) with just one phone is more challenging than to model a silence. For this reason, the “mumble” phone model was designed to have a wider and more complex topology than the silence phone model (including also backward connections as found in the silence phone). Thus, we expect to model reasonably the long and unpredictable sequence of states that an OOV word is likely to be composed of.

In order to train these eight states, we have to modify the training transcription by substituting a set of words (normally unusual ones) by the special word $\langle UNK \rangle$, so the ASR system is able to automatically train the “mumble” phone model that will be associated to it.

Figure 4.2: Topology of different kind of phones in the LVCSR system. Up: regular phones. Middle: silence phone. Down: “mumble” phone

In **LM terms**, $\langle UNK \rangle$ refers to any OOV word that we might expect to appear among the IVs, so it has to be taken into consideration inside our LM in some manner.

In the case of n-gram models, we are able to assign probabilities to $\langle UNK \rangle$ through different n-grams easily. For example, if we limit the vocabulary to the V most frequent terms for a training text of W different words, being $W > V$, all occurrences of the $W - V$ least frequent words will be treated as occurrences of $\langle UNK \rangle$, and then we can collect statistics in different n-gram orders.

A simpler approach would be to build an n-gram model with the standard “open vocabulary” approach, applying some discounting algorithm to the n-grams of the model so as to quantify the likelihood of the $\langle UNK \rangle$ token, an approach which is usually applied in conjunction with smoothing techniques for n-grams (Chen and Goodman, 1996).

4.3.1.2 Path pairs in the output graph

The second OOV detection method is used with the aim of obtaining more POOV SSs. In fact, we seek to overgenerate them, for two reasons. First, there are many OOVs that are expected to be recognized as one or more similar IV terms, so without this approach the recall would be considerably low. And second, the false alarms (in OOV detection terms) that this overgeneration would imply are expected to be discarded properly thanks to the robust candidate scoring we developed (section 4.3.2).

This proposed method works directly on pairs of paths from the LVCSR output graph. More precisely, the selection of POOV SSs is based on synchronized comparisons between two 1-best paths obtained with different AM-LM tradeoffs (i.e. different LM weights ρ in the traditional formula of ASR which is indicated in equation 4.1), looking for some discrepancy or low-confidence regions, as we will see. We could also have worked instead on only one 1-best path and detect low-confidence regions there, but we discarded this option since we observed it retrieving fewer POOV SSs than this proposed approach.

$$\hat{w} = \arg \max_w [\log P(\mathbf{o}|w) + \rho \cdot \log P(w)] \quad (4.1)$$

where $\hat{w}$ is the transcribed sequence of words given the acoustic observations $\mathbf{o}$. $P(\mathbf{o}|w)$ refers to the probabilities given by the AM for a sequence of words w and the mentioned $\mathbf{o}$. $P(w)$ refers to the probabilities given by the LM for a sequence of words w . So, ρ can be understood as a LM weight to directly control the influence of the LM during the hypothesis search.

In this OOV detection method, the LM weights for the two 1-best paths should be chosen at both sides of an optimal weight that produces the best transcription before our TD system refinement, assessed through some performance metric (usually WER). For example, if the best baseline transcription (again, without a TD strategy applied) is obtained for a LM weight of 12, the two 1-best paths could be obtained with LM weights of 10 and 14 (or similarly 8 and 16, or 12 and 14, or 10 and 12, etc.). Thus, we are able to compare 1-best paths that are one more acoustically, and the other more linguistically emphasized.

The functionality of this approach is intuited as follows: when an OOV appears, a 1-best path that is acoustically emphasized is forced to fill it with IV terms that are phonetically more similar, while a 1-best path that is linguistically emphasized is forced to fill it with more likely IV terms from the LM statistics. Then, when both paths disagree, we can assume that at least one of the models was not completely prepared for the input speech (hopefully the LM), and one plausible reason is the presence of an OOV. This functionality partially recalls the approach followed in [Burget et al. \(2008\)](#), where the comparison takes place between a LVCSR and a Phone ASR (in our case, preliminary attempts with this approach resulted less effective than the proposed one).

In order to select POOV SSs, we use a path comparator that looks for regions in which a pair of these paths disagrees, and also where, having the same word solution, the confidence scores across a region (as given by Kaldi toolkit in its simplest form, based on the posterior probabilities in the output graph) are considerably different, since that means that one model had more doubts about its solution than the other (e.g. though one path presents a term as likely, the other may not, since that term may not fit a certain acoustical/linguistic emphasis, and no better candidate is available).

We can use multiple 1-best pairs for a majority voting selection (e.g. determining a POOV SS when at least some number of pairs evidenced it), so we can use a wider variety of LM weights.

In figure 4.3, we illustrate this operation with an artificial example. It can be seen that the disagreement between both best-paths covers most of the temporal occurrence of the OOV word. Nonetheless, if we consider as well the score difference between both best-paths for the word connecting nodes 2 and 3 (i.e. 0.9 versus 0.6), we would be able to extend the previous OOV region (marked with a red rectangle in the figure) to the whole temporal occurrence of the OOV. In any case, it is advisable for this OOV detection method to increase the span of the analysis at both ends of the initial OOV region detected.

Figure 4.3: Example of 2 best-paths obtained for different LM weights applied to the output graph, in Spanish language (up: linguistically emphasized; down: acoustically emphasized). A disagreement region is marked with a red rectangle, covering most of the temporal occurrence of a OOV word indicated with a double-ended arrow in the middle

4.3.1.3 Operation of both methods

Taking into account both methods, a speech segment is then marked as potential-OOV whenever one or more of the following three requisites is fulfilled for at least a minimum duration (we set it to 0.3 seconds) involving as well a minimum of e.g. four different phones (as decoded by the phone ASR system):

- Appearance of the special word $\langle UNK \rangle$.
- Different word solutions between a 1-best pair, including substitution, insertion and deletion events.
- Same word solution between a 1-best pair, but considerably different confidence scores.

The first requisite belongs to the first detection method, while the others belong to the second one (for which, as we said, we could employ multiple 1-best pairs for a majority voting selection).

When the special word $\langle UNK \rangle$ appears, it is likely that, in fact, an OOV was uttered. However, as we have already mentioned, the fulfillment of the last two requisites is just a special way of detecting speech segments where the recognizer is not confident at its output. Thus, the underlying reason could be that an OOV term appeared, but also (and quite often) a defect in the AM, LM or both arose, causing a common ASR error.

For example, the AM might not have been trained for some noise conditions or a certain kind of speaker. Similarly, the LM might be not rich enough for some IV expression. In cases like these, the recognizer may be choosing the most likely (though poor) solution for a given AM-LM tradeoff, and it would probably change as soon as this tradeoff is modified. Consequently, as we said, the POOV SSs obtained this way may contain a large number of false alarms in terms of OOV detection, so we need to pay special attention to trigger or not a search for candidates for a segment (e.g. by requiring a minimum duration and number of phones for every POOV SS) as well as for finally deciding to incorporate a candidate into the final transcription (thanks to a candidate scoring scheme described later).

4.3.2 Potencial-OOV Speech Segments processing

The processing of each POOV SS determined by the previous OOV detection takes place in the *POOV SSs processing* module of figure 4.1. The purpose of this module is to analyze, for each POOV SS, whether it is sensible to propose a certain word candidate for substitution. In a positive case, we call this candidate a substitute, since it will replace the original content of the corresponding POOV SS in the baseline transcription. A candidate is every word (for some alphabet) found during the search, to which we apply a filter in order to keep only those terms that do not belong to the original vocabulary of the LVCSR system. Thus, only OOVs are allowed to be taken into consideration, since IV terms were already considered during decoding. Also, each candidate is accompanied by a score we define later, which also serves for finally deciding whether or not to make a correction in the baseline transcription.

The search for candidates works separately for each of the speech regions under evaluation, so the solving of all of them can be done in a highly parallelized computing fashion. Thus, we can explain its operation for one speech segment, following the operation scheme detailed in figure 4.4. For an easier interpretation of the figure, two submodules were simplified in individual boxes, and are expanded in other figures which we will indicate and describe later. We will follow a top-down description for a better understanding.

The two inputs for the overall module of figure 4.4 are the POOV SS phone transcription and the context words around the POOV SS. This context is obtained by looking for the baseline LVCSR transcription around the timestamps where the POOV SS occurs (about half a minute on both sides, in order to gain proper insight of its semantic context), ignoring function words (also known as stopwords in the NLP arena) and any part of the transcription marked as POOV SS. With this information we carry out two different kinds of search for candidates, as we will see. And with both searches we obtain two lists of candidates for the POOV SS that we merge. We filter these candidates using the LVCSR vocabulary to be sure we do not consider an IV term. Then, we award a scoring for all the surviving candidates (detailed in section 4.3.2.3). Finally, a score threshold helps us to decide whether to promote the best candidate to become a substitute and actually incorporate it into the transcription, replacing the original content associated to the POOV SS.

We now describe the two types of search for candidates that appear in figure 4.4.

Figure 4.4: Operation scheme of the search for candidates module

4.3.2.1 Search for candidates by word building

Figure 4.5 details the mechanism inside the *search for candidates by word building* module from figure 4.4. This module processes the phone transcription of a POOV SS. Since this information is very valuable for identifying an OOV, we construct proto-words directly from the corresponding sequence of phones, taking multiple segments of different sizes (e.g. skipping the first or the last phone of the sequence), and then applying some phone-to-grapheme transformation (quite straightforward in the case of the Spanish language). This occurs in the word builder process, after which we use a lemmatizer that enlarges the number of words built (e.g. if we were able to construct the word *leading* from the phones, we would additionally obtain the word *lead*).

Figure 4.5: Search for candidates by word building

All of these words are searched for in the open dictionary from the Wikimedia Foundation, Wiktionary (Wiktionary, nd), in order to validate their existence as written words and, in a positive case, obtain variants of them like singular/plural, masculine/feminine, verb conjugations, etc. If they cannot be validated by Wiktionary as candidates, we discard them. This last step provides the list of candidates for a POOV SS that this type of search is able to find. However, there are usually mistaken phones in the transcription that prevent a direct matching with the real OOV word. For this reason, we make use of the second type of search (the LOD search), which also

benefits from the other output of this module, called search seeds from phones. These search seeds are just the words we were able to build plus their extension by the lemmatizer (so as to extend their semantics), and they will be used in the way we explain later on (they do not need to be validated by Wiktionary, since other source will do so).

4.3.2.2 Search for candidates by LOD search

The *search for candidates by LOD search* module of figure 4.4 is detailed in figure 4.6. Now this search will be oriented by the semantics that can be assimilated from the context inside the LVCSR transcription around the POOV SS, plus the search seeds from phones we have just obtained. Thus, the candidates will be searched for throughout LOD resources that are related to this information, so it is probable that they will contain the uttered OOV we are searching for. In order to handle these specific LOD resources, we make use of what we call **search triggers**, which are references to the URIs of the LOD resources we are interested in for a certain POOV SS. The generation of these search triggers will be studied in our next and last step in the ongoing top-down description. For the moment, we establish that they are available at the output of the *generation of search triggers* module in figure 4.6. With these search triggers, the search through LOD corpora is carried out in the following way.

Figure 4.6: Search for candidates by LOD search

Firstly, we access the content of the LOD resources that the search triggers represent. Then, we collect all non-stopwords terms present there, especially in their abstract (if any) and the names of the resources they are connected to. This could be extended by mining their reference documents, like the corresponding Wikipedia article when handling DBpedia resources. The candidates collected are subject to a Wiktionary extension, in the same way we did for words constructed with the phones of the current POOV SS, with the difference that now they were already validated, since they were found in written form in LOD corpora. For better efficiency, we could score the candidates before the Wiktionary extension, and proceed with this extra search only for the best-scored candidates (this would require the use of the *candidate scoring* process that we found in figure 4.4).

Generation of search triggers

We describe now the inner part of the *generation of search triggers* module of figure 4.6, which can be found in figure 4.7, in charge of providing the search triggers needed for the LOD search.

First, the POOV SS context words, extended again by a lemmatizer, are gathered together with the search seeds from phones, so we obtain what we could simply call search seeds.

Figure 4.7: Generation of search triggers

The next step is to assimilate the information gathered in these search seeds semantically. Thus, we represent them in an embedding space that can be word- or resource-based, thanks to Word2Vec or Resource2Vec (again, to be explained in section 4.4), respectively. This can be seen as a validation of existence for the search seeds from phones: if we have an embedding for a search seed, it is because it has appeared in some knowledge source, since we are training the embeddings with them (additionally, these sources should be refreshed periodically so as to keep them up to date, as well as the embeddings, which would be pertinently retrained). In the case of a word-based space, the representations can be obtained immediately, since the entities involved are already words. For a resource-based space, we have to connect the search seeds to some resource of our corpora, if any. Normally, it is enough to apply a word lemmatizer and uppercase the first letter (e.g. if the word *political* is a search seed, it is more probable that in a LOD corpus like DBpedia there is a resource for *Politics* than for the original word). As an additional improvement, both word and resource embeddings could share the same vectorial space, so we would take advantage of the semantics offered by both kinds of entities; nonetheless, this is left for future lines of work.

In the next step, we try to compact all these representations into just one vector. To do so, we could make use of a centroid operation, obtaining the average vector of the input set of embedded search seeds, in a similar way as we saw in section 2.1 for Sheikh et al. (2017). In our case, this operation will be done in a weighted manner, since not all of the input terms are expected to be equally important in defining the semantics of the segment. In order to do so, we make use of a weight for each search seed, as we see below. Thus, we expect that the surrounding of a computed centroid contains enough semantics shared with the OOV to better retrieve it during the

search for candidates by LOD search.

Weighted centroid computation

The weighting of each search seed is produced in the *term weight assignation* process in figure 4.7, which requires two additional processes (a POS tagger used for each search seed and a POS tag predictor used for the POOV SS), detailed in the following description.

First, each search seed is assigned an initial α score. This score is assigned in the following way: if the search seed belongs to those obtained from the POOV SS context or its lemmatisations, then α will be directly proportional to time closeness to the speech segment; and if it belongs to the search seeds from phones, then α will be a constant high score.

Second, a POS tag predictor is used to find out which POS tag would be expected for an OOV inside the POOV SS. This information will be used to obtain a new β score for each search seed, as explained later. In order to obtain this tag, several words around the POOV SS are given to the POS tag predictor, together with their POS tags (given by some POS tagger, as the one we mention below). Our predictor consists of a small feedforward network trained with the same texts as for the LM training of the ASR (see section 4.5.1). We have restricted the output tags to only three word categories: *Noun*, *Adjective* and *Verb*, since any OOV term is supposed to belong to one of these categories (in any modern LVCSR system a wide coverage of other POS categories is expected). This is in line with a multi-class OOV modeling approach, as in Bazzi and Glass (2002), where some special $\langle \text{UNK-CLASS} \rangle$ terms were added to the vocabulary, taking into account its OOV behavior as a member of a certain class *CLASS*. In our case, we do not incorporate a multi-class OOV modeling in the LM, but we do obtain this class information to improve the search for candidates, as we will see.

Third, we determine the word categories of each search seed. This can be done with a POS tagger (obtaining normally only one category) or just with a dictionary that collects all possible categories for any word (take into account that we chose this last option for our experiments, making use of Wiktionary). We only consider search seeds belonging to the following word categories: *Noun*, *Adjective* and *Verb*, the same as those that the POS tag predictor handles. This means that all words with no category belonging to this set are discarded.

Fourth, we assign the second score for each search seed, β , following this scheme: firstly we take the results from the POS tag predictor applied to the corresponding POOV SS, so as to ascertain which POS tag is expected for the missing OOV. The results of this POS tag predictor are in the form of POS-tag-likelihoods values for the three categories under consideration: *Noun*, *Adjective* and *Verb*. Then, we make β equal to the POS-tag-likelihood value that corresponds to the word category to which the search seed belongs, so the search seed acquires a score that indicates its adequateness in POS terms for the POOV SS. In the case that the search seed belongs to more than one of these three categories (as it is our case due to the use of Wiktionary as a direct POS tagger), we choose the highest tag likelihood between those that correspond to the categories of which the search seed is member.

The β score can be understood under the hypothesis that, for the same text, semantic relationships between words of the same category are stronger than the semantic relationships between any other kind of words, so we emphasize the category of the expected OOV. As an example, consider the next sentence:

When people from conflicted countries begin a dangerous journey through many frontiers, it is often too late to look after the situation that motivated such a desperate decision.

Imagine that the adjective *conflicted* is an OOV, so that an ASR system is able to detect it and to decode the other words correctly. It seems reasonable to think that this adjective has a stronger relationship with other adjectives in the sentence (*dangerous, desperate*), than with other words in different classes (apart from, of course, the individual word that it modifies, the noun *countries*). Then, taking the semantics of these adjectives into account would be beneficial for a proper search for candidates. We could argue the same if the OOV were another adjective or a noun of the sentence, focusing now on the importance of the proximity between words (e.g. the noun *people* seems more related to the closest noun *countries* than to the furthest one, *decision*). Remember that proximity is already taken into consideration with α scores, while β scores serve to quantify the semantic relationships between an OOV and other words that belong to the same or distinct category. In this sentence, verbs do not show the same strong relationship between them, but it would be easy to think of other examples in which they do. Thus, we expect that, for the general case, this type of weighting terms may improve the search for candidates.

Fifth and finally, the weight for each search seed can be calculated as the product of α and β . And with these weights we can finally carry out a weighted centroid $\overline{WC}$ computation:

$$\overline{WC} = \frac{\sum_i^{N_{ss}} \alpha_i \beta_i \overline{X}_i}{\sum_i^{N_{ss}} \alpha_i \beta_i} \quad (4.2)$$

where N_{ss} is the number of search seeds of a POOV SS and α_i , β_i and $\overline{X}_i$ are the two scores and the vector, respectively, of the i th search seed. The vector $\overline{X}_i$ can be in the Word2Vec or Resource2Vec space, depending on whether we are computing a word- or resource-based centroid, respectively.

If we would like to obtain a finer centroid, we could proceed in the following manner. Prior to computing a centroid with all search seeds with weights, we could prune the number of search seeds that belong to each word category separately (*Noun, Adjective* and *Verb*). For example, the search seeds of a certain word category could be pruned by computing an auxiliary centroid just with them, and then setting some radius that would become the frontier out of which any representation is considered an outlier, and thus can be dismissed.

At this point, both word- and resource-based centroids were computed in the weighted manner we have just described. The word-based centroid will be used in any case for candidate scoring (as we will describe in section 4.3.2.3). Apart from this, we can choose the word- or the resource-based centroid (or even both) to generate the search triggers, a set of references to LOD resources whose content will be used in the way we have explained before. They can be obtained simply by computing the K-nearest-neighbors (KNNs) of the selected centroid, for a specific KNN search configuration (for example, we could allow only the first K neighbors within a radius from the centroid to be accepted as search triggers). Now, the search triggers would already be these KNNs in the case we choose a resource-based centroid. However, a string transformation like that mentioned before (lemmatisation plus uppercasing the first letter) has to be applied to them in the case we prefer a word-based centroid, in order to be able to access the corresponding resource,

since we pretend to use LOD resources to search for candidates. Additionally, we could consider the neighboring words in the word-embedding space as candidates directly.

We provide in table 4.1 some real examples of search triggers generated with this weighted centroid computation, employing Resource2Vec embeddings from the Spanish DBpedia (more details in the experimental framework, section 4.5). We selected examples for which we were not able to build the missing OOV directly from the sequence of phones decoded by the Phone ASR, relying then on semantics for finding the right candidate. We also indicate the twelve best-scored search seeds from the context of the POOV SS. The twelve search triggers listed were manually selected to show an idea of the types of resources that we consulted in each case (trying to preserve the original variety), indicating in bold those that led to the discovery of the missing OOV word.

Table 4.1: Real examples of search seeds and DBpedia search triggers for different POOV SSs detected, in Spanish language

OOV word	Best-scored search seeds	Some DBpedia search triggers
<i>relativismo</i>	<i>principios, juicio, sillón, intelectual, crítico, gente, definición, valores, punto, vista, difícil, considera</i>	<i>Crítica, Valor, Intelectual, Juicio, Alienación, Sociedad_de_consumo, Moral, Paradigma, Fanatismo, Emancipación, Relativismo, Castigo</i>
<i>ecologista</i>	<i>máxima, sensibilidad, contaminación, tirado, realidad, motor, arreglar, consumidores, obligación, persona, coche, recuperar</i>	<i>Contaminación, Seguridad, Reciclaje, Combustible, Motor, Humo, Medio_ambiente, Asbesto, Envase, Gasolina, Consumidor, Teoría_de_Olduvai</i>
<i>tributaria</i>	<i>declarado, comunitario, factura, valor, fraude, caso, comisario, país, importaciones, precios, pantalones, reducir</i>	<i>Productividad, Precio, Empleo, Fraude, Consumo, Falsificación, Capital_financiero, Tarifa, Tributo, Interés, Contrabando, Keynesianismo</i>

4.3.2.3 Candidate Scoring

In order to evaluate each candidate that could fit in a POOV SS we compute several scores for it:

1. **Levenshtein distance:** we calculate the Levenshtein distance between the speech segment recognized phones (from the phone ASR system) and the phonetic transcription of a candidate (from a grapheme-to-phoneme transformation). This distance is normalized by dividing by the number of the speech segment recognized phones.
2. **Candidate to centroid distance in word-embedding space:** we use the word-based centroid for measuring the distance between it and the word embedding of a candidate. A default value is assigned if we do not have a word embedding for the candidate. As we specify below, this default value is chosen so that we do not discard the candidate by now, because although we are not able to know whether the candidate is actually related to the OOV we are searching for, the candidate could be a recently created term interesting to consider (nonetheless, in a posterior retraining of the word embeddings, triggered by the refreshment of the pertinent knowledge sources, we might have finally an embedding for

such kind of candidates).

3. **Likelihood of the POS tag of the POOV SS:** we want to assign a score in this case dependent on the match between the POS tags of both the candidate and the POOV SS. Note that a POS tag predictor was already used to assign to the POOV SS three POS-tag-likelihoods for the three categories we are considering (*Noun*, *Adjective* and *Verb*). Proceeding in the same manner as we did for obtaining the β score for the search seeds, the candidate will be firstly tagged with some of these three POS tags making use of Wiktionary. Then, to generate this score, we just select the highest of the three POS-tag-likelihoods that match the corresponding POS tags of the candidate. In the event that the candidate does not participate in any of the three POS tags we are interested in, the candidate is directly rejected. And if we are not able to find a tag for the candidate, we assign a default value that will not discard the candidate because of this fact, as we reasoned for the previous score.

It is important to highlight that not all scores are computed for every candidate. A candidate can be discarded early if the first score computed (Levenshtein distance) is less than an acceptable minimum. We believe this score is the most important one for a coarse delimitation of suitable candidates, since it evaluates how acoustically consistent a candidate is compared to what the phone ASR system could actually understand (in contrast to the other two, which seek for a semantic or syntactic suitability). This filter allows us to save computation time and focus on those candidates that can really have a chance for matching a POOV SS.

A final score FS , the greater positive the better, is computed by a weighted sum of each of the scores S_s (equation 4.3), in which we give more importance to some scores over the rest (λ_s factors). These λ_s factors (to be set experimentally) will be negative for the two first scores, since in those cases smaller values mean better candidates. In order to allow an equalized sum that gives more human sense to the values of λ_s , we standardize each score, for which the different means (μ_s) and standard deviations (σ_s) can be obtained by a brief statistical analysis of experimental values of each score. Finally, the assignation of a default value for the second and third scores can be made by making those λ_s equal to the corresponding μ_s , so the final score would only depend on those scores where this assignation did not take place.

$$FS = \sum_{s=1}^3 \lambda_s \frac{S_s - \mu_s}{\sigma_s} \quad (4.3)$$

4.4 Resource2Vec: Linked Open Data embeddings

The purpose of Resource2Vec is to obtain a vectorial representation of every resource inside some LOD corpora, because of their interest for our TD strategy in the manner we have just shown.

As we described in section 4.2, a resource is made up of an identifier (its URI), and a list of resources to which it is connected by some predicate or property, following a triplet fashion, $\langle \text{subject}, \text{property}, \text{object} \rangle$. For the implementation of Resource2Vec, we are ignoring the specificity of these properties (e.g. *birthPlace*) and the connections of a resource to objects that are not a resource, like constant values (e.g. the runtime of a film in minutes). Thus, we prepare a

simplified version of the data involved, so that, for each resource, we have an enumeration of the resources it is connected to throughout any property. This allows us to carry out a training that is similar to the Skip-gram found in Word2Vec, where now the *context words* of a resource R_a would be the resources it is directly connected to, the so-called 1-hop nodes. A more detailed description can be found in section 4.4.1.

In Word2Vec, the semantics are learned by successive presentations of context words for each word to the network, so relevant relationships are eventually absorbed after large quantities of texts are mined. Thus, words with similar relevant contexts will appear close together in the vector space of the model. In turn, in Resource2Vec we know every connection that a resource has beforehand, similarly as in GloVe (Pennington et al., 2014) we know beforehand all the context words of every word, thanks to the global word-word co-occurrence matrix, so the learning of semantics will be produced more similarly to GloVe than to Word2Vec. Besides, the Resource2Vec model is very compact and does not need to quantify every connection that a resource has (as in GloVe, where we take into account the closeness of the context words to the word under study), since all of the connections will be considered equally important. In any case, the effect we are pursuing is that, for two resources, as the number of connected resources they share increases, they will obtain closer vectors.

This compact learning allows different ways of quantitative analyses of the model to be designed, apart from the qualitative one typically reported for unsupervised embeddings. The most immediate one is to ask our model which are the closest neighbors for a specific resource. If our model is well trained, the closest resources should have an intersection ratio of connections greater than farther ones. Studying how this ratio decreases with distance, we can find a distance limit above which resources with low intersection ratios are more likely to appear (in this document, the term distance is used for Euclidean distance unless otherwise specified). In section 4.4.2, we will study this and other quantitative measures, which were designed paying attention to how Resource2Vec is taking part in the proposed TD strategy.

4.4.1 Implementation of the Resource2Vec scheme

The embeddings we want to learn with Resource2Vec refer to the resources found in LOD corpora. The preparation of the data that will eventually be used in the training of a neural network involves the following steps:

First, we keep only those triplets where both the subject and object consist of a resource (i.e. a URI). And second, we ignore the specificity of the properties in order to keep only the information on connections between resources. Therefore, by collecting all these modified triplets where a resource R_a is the subject, we can obtain the N_a 1-hop connections of R_a (i.e. $\{R_{a,1} \dots R_{a,b} \dots R_{a,N_a}\}$, since in this notation each comma in the subindex of R accounts for one hop), and so on for each of the resources in the corpora. The enumeration of the 1-hop connections of each resource is precisely the data used for the training process (in contrast to other approaches that aim to preserve more explicitly the topology of the graph, by considering also n -hop connections for $n > 1$ (Grover and Leskovec, 2016)). Also, resources from different corpora can be represented in the same vectorial space, which is especially useful when these resources are connected through more than one corpus, which is inherent in the nature of LOD creation.

Then, we proceed with the training in a similar way to the Skip-gram scheme found in Word2Vec, with a similar neural network architecture. This arrangement can be understood if we make the

following considerations. First, we have to consider the LOD resources and their connections as nodes and edges of a graph, respectively. Then, we can handle this LOD graph as if it was a text (a virtual text), in the following manner: nodes represent words, and edge paths represent how far in the virtual text a word appears with respect to the other words (for example, 1-hop nodes would refer to the set of words that appear within the closest context of a word w_i in a text, like one or two words to the left/right, i.e. $\{w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}\}$; then, 2-hop nodes would refer to words that appear within a bit farther context of w_i , i.e. $\{w_{i-4}, w_{i-3}, w_{i+3}, w_{i+4}\}$; and so on). In any case, and as we explained before, we only take into account the 1-hop nodes for each node.

The previous training scheme is illustrated in figure 4.8, and explained with algorithm 1. In the presented algorithm, take into account that a negative sampling (as in the Word2Vec scheme) is produced inside the *optimize* method, and also that the batches are built with 1-hop pairs, $\{R_a, R_{a,b}\}$, inside the *append* method (besides, a batch might contain 1-hop pairs from different R_a resources).

Algorithm 1 Resource2Vec training procedure

```

1: procedure RESOURCE2VEC TRAIN( $D, M, E, B, N$ )      ▷  $D$ : embedding dimension;  $M$ :
   number of resources;  $E$ : number of epochs;  $B$ : batch size;  $N$ : negative sampling size
2:   initialize( $[W \ S_w \ S_b], D, M$ )              ▷  $W$ : network embedding weights ( $M \times D$ );  $[S_w \ S_b]$ :
   network softmax weights ( $D \times M$ ) and biases ( $1 \times M$ )
3:   BATCH  $\leftarrow []$ 
4:   for  $e=1:E$  do
5:     for  $a=1:M$  do
6:        $b \leftarrow 0$ 
7:       while  $b < N_a$  do
8:         for  $\text{batch}=\text{size}(\text{BATCH}):B$  do
9:            $b \leftarrow b + 1$ 
10:          append(BATCH,  $[R_a \ R_{a,b}]$ )
11:          if  $b==N_a$  then
12:            break
13:          end if
14:        end for
15:        if  $\text{size}(\text{BATCH})==B$  then
16:          optimize( $[W \ S_w \ S_b], \text{BATCH}, N$ )
17:          clear(BATCH)
18:        end if
19:      end while
20:    end for
21:  end for
22:  return  $W$ 
23: end procedure

```

4.4.2 Quantitative analyses

Because of the compact learning of Resource2Vec, it becomes quite straightforward to design a first quantitative analysis that illustrates the correctness of the training and, with little effort, to design other analyses that are able to show the expected usefulness for our TD strategy. For these analyses, both Euclidean and cosine distances might be used, since in our experiments very similar results were obtained with both of them.

Figure 4.8: Resource2Vec training scheme. Up: an example of a R_a resource with its true connections inside some LD corpora. Down: neural network architecture and training operation for R_a resource, to be carried out in a pairwise fashion, $\{R_a, R_{a,b}\}$

First of all, we look for a correspondence between distances in our vectorial model and intersection ratios between any pair of resources, considered as the proportion of shared resources in their 1-hop connections. Since this intersection ratio is implicitly one of the learning objectives in Resource2Vec (as we reasoned at the beginning of this section 4.4), we expect to find a decreasing behavior as the distance grows. Therefore, the correctness of the training can be confirmed whenever this relationship, which we can call *distance-intersection relationship*, has a decreasing nature. The equation for computing the intersection ratio between two resources R_x and R_y can be seen below.

$$IR(R_x, R_y) = \frac{len(T_x \cap T_y)}{N_x} \quad (4.4)$$

where T_x and T_y refer to the set of resources that are 1-hop connected to resource R_x and R_y , respectively. The $len(A)$ function is used to obtain the number of elements in set A . N_x is $len(T_x)$. Note that the intersection ratio is calculated with respect to the first resource, R_x , so N_x common resources would make an intersection ratio of 100% for the pair made up of R_x and R_y .

Furthermore, it will be useful for our TD strategy to find a threshold in distance from where we cannot expect a considerable intersection ratio between resources, nor a proper semantic relationship.

As we already presented in section 4.3, the TD strategy proposed in this thesis work carries out searches through LOD resources in which to find a proper word candidate for a detected speech segment that might contain an OOV. A search starts at a certain point in the vectorial space of Resource2Vec (or Word2Vec, although our interest now is focused on Resource2Vec). We called this starting point a resource-based centroid, which is computed as a center of gravity of different

resources with different weights in the Resource2Vec space, depending on the speech segment (see section 4.3.2.2).

From this centroid, we collect neighboring resources (through a KNN operation) in order to search for candidates in their content. Therefore, with the previous *distance-intersection relationship* we could set a distance threshold that limits the resource collection from the centroid, given a minimum intersection ratio that we consider acceptable. Although in section 4.4.2.2 we will provide a procedure to reflect this minimum intersection ratio automatically, we could also make an arbitrary selection of it, depending on the purpose.

We present now two finer-grained evaluations settled in the direction of how Resource2Vec embeddings will be used in our TD strategy, specifically during the LOD search in the Resource2Vec space. As we have just anticipated, in this LOD search we make use of two operations that are the centroid and KNN calculations. We have designed two additional measures that might help to evaluate the quality of both.

4.4.2.1 Centroid quality evaluation

To evaluate the quality of the centroids and assess whether the vector space holds some semantic geometry, we will test how well the centroid operation keeps the semantic relationships we anticipate.

Our first idea is that if we select the set of resources that are 1-hop connected to a certain resource R_a , they should each determine part of the semantics of R_a . If this is the case, the semantics of the centroid, calculated as the center of gravity of the resources $\{R_{a,1} \dots R_{a,b} \dots R_{a,N_a}\}$, should be determined jointly by all these semantic restrictions and consequently this centroid should not be far from the original resource R_a .

For better understanding of the previous idea, we have depicted in figure 4.9 a schematic Resource2Vec representation (in two dimensions) of a resource R_a and its neighboring resources, where the arrows refer to the real resource connections found in the original knowledge graph (which are not anymore present in the vectorial space, but are indicated for illustrative purposes). As can be seen, the five 1-hop resources of R_a were placed in the surroundings of R_a , as they are semantically related. Therefore, the centroid C_a computed with all the five 1-hop resources (i.e. $R_{a,1}$, $R_{a,2}$, $R_{a,3}$, $R_{a,4}$ and $R_{a,5}$) is situated very closely to R_a , as we could expect. The distance between C_a and R_a is called *centroid-resource distance*. Thus, a **first sign of centroid quality** is that the *centroid-resource distance* is small on average, after computing this distance for each of the resources present in the Resource2Vec model. In order to assess how small is the average *centroid-resource distance*, we can perform a couple of comparisons with other two average distances that we explain below, for which we will make use again of figure 4.9:

- The first of the other average distances is related to the distances between R_a and any of its 1-hop connections $R_{a,b}$. From figure 4.9 we can see how these distances are greater than the *centroid-resource distance* for R_a , as we could expect to happen, at least on average. We expect so since any pair $\{R_a, R_{a,b}\}$ can be conceived as two clearly separated entities, whereas any pair $\{R_a, C_a\}$ can be conceived as two manners of denoting the same entity. Thus, a **second sign of centroid quality** is that the *centroid-resource distance* is smaller than this *1-hop distance*, on average.

Figure 4.9: Schematic Resource2Vec representation for a resource R_a and neighboring resources that can be achieved by a 2-hop path from it in the original knowledge graph

- The last average distance is related to the distances between R_a and any of its 2-hop connections that comply with some restrictions. These restrictions have to do with some 2-hop paths from R_a that can occur and we want to avoid, for the reasons we give below. All the 2-hop paths possibilities can be found in figure 4.9, starting from R_a and following the arrows. The three dashed arrows correspond to the types of 2-hop paths from R_a that we do not want to consider:
 - The first type are 2-hop paths that ends in some 1-hop resource of R_a (in the figure, the path $R_a \rightarrow R_{a,1} \rightarrow R_{a,5}$), since the 1-hop resources were already considered in the previous *1-hop distance*.
 - The second type are 2-hop paths that end in R_a (in the figure, the path $R_a \rightarrow R_{a,3} \rightarrow R_a$), since we want to consider separate entities from R_a .
 - And the third type are 2-hop paths that end in resources already considered (in the figure, the path $R_a \rightarrow R_{a,4} \rightarrow R_{a,3,1}$), since we do not want to consider the same 2-hop resource more than once (in the figure, $R_{a,3,1}$ was already found through the path $R_a \rightarrow R_{a,3} \rightarrow R_{a,3,1}$).

The valid 2-hop resources that were found by 2-hop paths different from the previous three are denoted as $R_{a,b,c}^*$. In the figure, we can see how this $R_{a,b,c}^*$ resources are situated the furthest from R_a , as a resource of this kind would share even less semantics with R_a than a 1-hop resource would do, on average (since we might find exceptions like the resource $R_{a,4,1}$ in the figure, which seems to be quite semantically related to R_a). Thus, a **third sign of centroid quality** is that the *centroid-resource distance* is smaller than this *2-hop distance*, on average.

Then, when we study the previous three distances, we should verify, on average, the following inequation:

$$\text{dist}(R_a, C_a) < \{\text{dist}(R_a, R_{a,b}), \text{dist}(R_a, R_{a,b,c}^*)\} \quad (4.5)$$

where $\text{dist}(X, Y)$ refers to the Euclidean distance between points X and Y . Thus, the distance on the left is the *centroid-resource distance*, while the distances on the right refer, respectively, to the *1-hop distance* and the *2-hop distance*. Depending on the corpus, we might also observe that $\text{dist}(R_a, R_{a,b}) < \text{dist}(R_a, R_{a,b,c}^*)$, since it is reasonable to think that the resources $R_{a,b,c}^*$ are not so related to R_a as the resources $R_{a,b}$, as it was observed in figure 4.9.

Additionally, we would like to measure, from the original knowledge graph, the average intersection ratio between a resource R_a and all of its 1-hop connections $R_{a,b}$. By knowing such average ratio, we can ascertain which magnitude of intersection is expected between any 1-hop pair of resources, which can give us a notion of how close or far a resource R_a should be from $R_{a,b}$ in our Resource2Vec space (the higher intersection, the closer). Then, with the use of the *distance-intersection relationship* discussed at the beginning of section 4.4.2, we can have in turn an idea of the magnitude of the distances that correspond to the previous intersection ratio. Thus, we can also compare this magnitude of distances to the three distances we have just discussed before, obtaining more insight about the meaning of remoteness and closeness among resources as held by a particular Resource2Vec space (an empirical analysis will be offered in section 4.5.4.3).

In order to obtain such measure, we have designed a *general auto-sharing ratio (GASR)*, which computes the mentioned intersection ratio for each R_a resource out of the M resources of a corpus, so as to calculate an average. The equation for this *GASR* is presented below.

$$\text{GASR} = \frac{1}{M} \sum_a^M \frac{1}{N_a} \sum_{a_b}^{N_a} \text{IR}(R_a, R_{a,b}) \quad (4.6)$$

where it can be seen that we are averaging, first, by the total number of 1-hop resources that a resource R_a has (N_a) and, second, by the total number of resources of the corpus (M). Finally, to calculate the intersection ratio between R_a and each $R_{a,b}$, we make use of the expression $\text{IR}(R_a, R_{a,b})$, which can be expanded with equation 4.4.

4.4.2.2 KNN quality evaluation

Now, we present the last quantitative measure that refers to the second operation mentioned: the KNN calculation. We would like an evaluation of how suitable the KNNs of a vector are in the Resource2Vec space. We would especially like to study the **maximum radius** we should limit the KNN searches to. As a first intuition, we could return to the first analysis where we showed how to obtain the *distance-intersection relationship* so, once we have computed this relationship, we can arbitrarily set an intersection ratio limit which would correspond to a distance, which in turn we would use as the maximum radius, as we suggested then. Nonetheless, we designed a finer-grained evaluation as follows.

Attending to the learning objective of Resource2Vec, the KNNs of a resource R_a should be those resources with the highest intersection ratios with R_a (in the terms defined by equation 4.4). However, it is also reasonable to think that these KNNs might be the 1-hop resources of R_a , following a similar reasoning as we did for the previous evaluation. Therefore, we make the following soft statement: «the *expected-true neighbors* of a resource R_a , denoted again as T_a , are all of its 1-hop connections, $\{R_{a,1} \dots R_{a,b} \dots R_{a,N_a}\}$ ». Under these conditions, it will be easier to

design a KNN evaluation, without needing to collect those resources with the highest intersection ratios for every resource of the model (these would be more accurate *expected-true neighbors*).

Then, for each resource, we obtain its KNNs (for some K , which should be chosen by taking into account the typical K that will be used in the LOD search: ≤ 100 LOD resources in whose content we will search for candidates) whenever they are at a distance of less than some threshold d (the maximum radius that we mentioned before), making a sweep of different values for d . Finally, we can calculate the precision and recall of the computed neighbors, for each distance threshold d , so as to know their quality in the following manner.

As regards **precision**, we would like to know in which proportion, out of the retrieved neighbors for a resource R_a , we were able to find the previously established expected-true neighbors (i.e. T_a). These retrieved neighbors depend on some K for the KNN search and a distance threshold d . The equation for precision is, then:

$$Pre(K, d) = \frac{1}{M} \sum_a^M \frac{len(I_a \cap T_a)}{len(I_a)} \quad | \quad I_a = KNN(K, R_a) < d \quad (4.7)$$

where $KNN(K, R_a)$ refers to the KNN search from resource R_a . Therefore, I_a is the set of resources obtained with this KNN search which are at a distance of less than d from R_a . The $len(A)$ function and T_a mean the same as in equation 4.4. As can be seen, the overall precision of a corpus is computed as the average precision for each of its M resources.

And as regards **recall**, we measure now the proportion of true neighbors that we were able to find, out of the total number of those expected-true neighbors, for a resource R_a . This total number of true neighbors is computed as the minimum between the size of T_a and K , since with the KNN operation we are not retrieving more than K resources. The equation for recall is, then:

$$Rec(K, d) = \frac{1}{M} \sum_a^M \frac{len(I_a \cap T_a)}{\min(K, len(T_a))} \quad | \quad I_a = KNN(K, R_a) < d \quad (4.8)$$

where $KNN(K, R_a)$, I_a , $len(A)$ and T_a mean the same as in the previous equation. The $\min(x, y)$ function takes the minimum value of its arguments. Again, the overall recall of a corpus is computed as the average recall for each of its M resources.

It could be the case that, for some K employed in our experiments (e.g. $K=100$), the denominator of the recall equation will almost always be K , since most of the resources of a corpus like DBpedia have larger number of 1-hop connections. This means that there may be some $d=d'$, for all the M resources, where we will have already found all of the K nearest neighbors, so K would equal $len(I_a)$, and both precision and recall would have equal or similar values (similar if there are some R_a resources in the corpus whose $len(T_a)$ is inferior to K , which are expected to be very few for such small K and a corpus like DBpedia). Also, the same equal or similar values will be maintained in this case for $d > d'$, since no new neighbors are to be retrieved.

Finally, with both metrics we can obtain the $F1$ score:

$$F1(K, d) = \frac{2 \cdot Pre(K, d) \cdot Rec(K, d)}{Pre(K, d) + Rec(K, d)} \quad (4.9)$$

With this metric, we would be able to find a distance threshold for which the $F1$ score is maximum. And this threshold should be used for the configuration of the KNN calculation in the LOD search. However, it might be possible that slight modifications to this threshold would achieve a better retrieval of candidates, so further experimentation is advisable.

4.5 Experimental framework for Static Term Discovery evaluation

This section presents the design of the experiments carried out to show the usefulness of our static TD strategy (and consequently also of Resource2Vec). The setup of every module in figure 4.1 will be detailed separately, they are: LVCSR, phone ASR, OOV detector and the module in charge of the search for candidates. In the latter, we specify both the LOD corpus chosen, in which the candidate search takes place, and the two types of embeddings used (words and resources), carrying out a specific quantitative analysis on the embeddings obtained with Resource2Vec (as introduced in 4.4.2). The section ends with the determination of the evaluation scheme for the overall result after TD application that we will use in our experiments, and whose numeric results are collected in the next section 4.6.

This experimental framework, though similar, will differ from the more elaborated and realistic experimental framework designed for the dynamic use of the TD strategy (the scope of next chapter 5), since now we are just focused on assessing the effectiveness of our TD strategy (mostly as a proof of concept), and not in the evolution of the overall system through time (the intended use case for the proposed TD strategy), as we will face in the next chapter.

4.5.1 LVCSR system

This ASR system was constructed following the DNN-HMM approach. We used the Kaldi toolkit with Dan’s implementation of DNN AMs (Povey et al., 2011).

The two corpora used for training this system are both in Spanish. This means that our experimentation is carried out solely in this language. The first corpus is the Spanish partition of the EPPS and the CONG-PARL databases of the TCSTAR European Project (audio+text) (Mostefa et al., 2007), while the second one is EUROPARL (text) (Koehn, 2005). Both corpora belong to the political domain, and they are further detailed in table 4.2. As can be seen, only TCSTAR has audio, so we have used it for both AM and LM training, as well as testing our systems, while EUROPARL is used only for LM training. It is important to remark that the audio quality of the EPPS database is considerable better than of the CONG-PARL database. This will be useful for a better interpretation of the results presented in section 4.6.

The **AM** of this system consists of a feedforward DNN with four hidden layers. Each of the hidden layers has a dimension of 3,000 neurons, and at their output the neurons are grouped in sets of 10,

Table 4.2: Corpora used for LVCSR system implementation

Corpus	Lang.	No. Words	Vocab.	Duration
TCSTAR	Spa	1M	29,165	111.9 h
EUROPARL	Spa	44M	134,068	(text)

to which a special maxout function (called p-norm) is applied. The input layer has 360 nodes, that correspond to 9 frames (the actual frame plus four at both sides) of 25 ms analysis width window with a 10 ms shift, each one of dimension 40, corresponding to MFCC (Mel-Frequency Cepstral Coefficients) with LDA (Linear Discriminant Analysis) + MLLT (Maximum Likelihood Linear Transform) + fMLLR (feature-space Maximum Likelihood Linear Regression) features. The output layer has 2,176 units, which in turn correspond to states in the HMM using tied triphone states for modeling phones. We trained this DNN for 20 epochs using two thirds from TCSTAR corpus. The remaining third is used for testing in our experiments.

In order to train the HMM states of the “mumble” phone model (corresponding to the special word $\langle UNK \rangle$) described in section 4.3.1.1, we selected the 45K words less frequent in the whole training text set (i.e. TCSTAR + EUROPARL, with an overall vocabulary of more than 130K) to appear as OOVs during training. They meant only around 650 occurrences in the audio training set, as many of them were not present in the 24,845 vocabulary of this audio training set. We considered these 650 occurrences as enough to train the “mumble” phone model. During testing, those 45K words are present in the LM, so they could be recognized in the case they appear in the testing set, since the phones they are made up of were properly trained along the remaining words.

Regarding the **LM**, we implemented n-gram models of order 3 with Good-Turing discount and the special word $\langle UNK \rangle$ for unseen words (trained with the set of words marked as OOV tokens that we will explain below), using the SRILM toolkit (Stolcke, 2002). First, we built the 3-gram for the training partition of TCSTAR and another one for all EUROPARL. Then, we interpolated them with a weight of 0.7 for TCSTAR, since the test data belongs to this corpus.

A vocabulary issue should be addressed here. If we join the two vocabularies of the training corpora (i.e., training partition of TCSTAR (24,845) and all EUROPARL (134,068)), then we obtain a vocabulary of 135,185 different tokens, which means a 96% coverage of the vocabulary of the testing set. This leaves around 600 words in the testing set that would be natural OOV terms for our LVCSR system. Furthermore, the number of occurrences of these words during the testing set is less than 900, which means that they are in fact very rare words and very likely just misspellings in the human reference transcription that mostly appear just once. Therefore, it is hard to evaluate a TD strategy under such poor conditions. For this reason, we have prepared a controlled vocabulary reduction in order to increase the OOV appearances in the testing set, generating an artificial OOV list (so we do not evaluate our TD strategy only over the poor 600 natural OOVs).

Thus, we will be able to evaluate our TD strategy against a much greater number of OOV terms, and so the confidence intervals of our results (as shown in the next section) will be small enough to observe significant tendencies. As we will see, most of the OOV terms in this list are not low frequency, as opposed to typical OOV terms. However, apart from the previous reason, we find useful the study of a system which expects a considerable amount of OOV terms but is also provided with a way to deal with them, so we might apprehend already some insight for the dynamic use of the strategies inside such a system (section 4.6.3), since that system should perform

a dynamic update of vocabulary and LM retraining to better deal with such speech conditions, as we will experiment thoroughly in the next chapter.

We detail now how the selection of words for this **artificial OOV list** was made, indicating also how we produced additionally a **garbage word list** of terms that will be as well removed from the original vocabulary of the ASR, for the reasons we give below:

1. We grouped together all of the utterances that belong to the same topic. A topic classification of utterances is already provided in TCSTAR corpus. We then constructed a *tf-idf* matrix with the largest 67 topics and selected the 20 terms with the highest *tf-idf* score for each of the topics, so as to expect a large number of occurrences during the test, allowing a proper evaluation of our TD strategy. Some of these terms appeared in the 20 best *tf-idf* list for more than one topic. This led to an **OOV list** of 1,127 different words. This **OOV list** is reduced to 1,012 after manually removing essential ones like days of the week, months or very common verbs that any LVCSR system should have always considered in its original vocabulary.
2. We collected into the mentioned **garbage list** any misspelt form of these 1,012 OOV terms that might trigger during decoding and thus conceal the original OOV with such an undesirable IV. We did not add these useless terms to the **OOV list** since we do not want them to be considered as OOVs. This can be done by searching for words in the original vocabulary of 135,185 whose Levenshtein distance is very small to any term in the 1,012 **OOV list**. In order to avoid the removal of phonetic competitors that would ease the task improperly, we make use of a huge vocabulary (almost 4M) that is involved in our word embedding implementation (section 4.5.4.2). When we sort the terms in the 4M vocabulary by their frequency in the word embedding corpus, any term below the first million always has fewer than 80 occurrences. In a corpus with 2.2G running words, it is a fair threshold for determining whether or not a word is a very rare term or, hopefully, just a misspelling. Thus, any of those similar words in Levenshtein terms that do not belong to this first million will be removed from the vocabulary as we will do with the terms from the **OOV list**. These undesirable words had 297 different forms in the ASR vocabulary, so this is the size of the **garbage list**.
3. Finally, we enlarged the 1,012 **OOV list** by adding any morphological variation in number and gender for nouns and adjectives, whenever a variation is present in the 135,185 vocabulary. We believe that a real OOV in modern LVCSR systems occurs beyond such a simple variation. In most of the cases, the OOV is a totally new word. Thus, though this step might ease the detection of some OOVs because of the deletion of some phonetic competitors, it is still a realistic scenario for a TD evaluation. In any case, these variations are not very important for the training corpora (otherwise they would have appeared during the *tf-idf* collection made in the first step), and therefore they are not expected to occur often during the testing set. The length of our **OOV list** is, finally, 1,927.

Then, we removed from the system's vocabulary the 1,927 terms of the OOV list and the 297 misspelt forms of the garbage list. Of the 1,927 terms, there are 901 terms that never appear in the testing set. As we anticipated before, these 901 terms are mostly the morphological variations that we added to the OOV list in the third step. Then the 1,026 words that do appear make a total number of occurrences of almost 18K, way larger than the original 900 for natural OOVs when no vocabulary reduction was made. If we study in how many utterances these 18K occurrences take place, we find that from the 13,294 utterances that belong to the testing set, they appear in 8,598 utterances, so there are 4,696 utterances without any OOV term of the OOV list.

To end this point, we show in table 4.3 the number of different n-grams that the LM holds when there is no vocabulary removal and when the vocabulary is reduced to 132,961 (135,185-1,927-297), where for the LM we also have to count the start and end of sentence and the special *<UNK>* word. Comparing the number of 2-grams and 3-grams of both LMs we observe a reduction to the 86% and 85% of the original size, respectively, when the vocabulary removal is made. This occurs because the 1,927 OOV list contains important terms for the training text (and hence also the testing set), which were responsible for the production of many 2-grams and 3-grams that now no longer exist. This is also reflected in perplexity terms, when testing both LMs against the testing set, where we observe an increase of a relative 8.9% from the full to the partial LM (a perplexity of 124 and 135, respectively). Thus, we can say that the partial LM becomes weaker, increasing the difficulty of our task considerably.

Table 4.3: Number of n-grams in the Full and Partial LVCSR LM

LM	1-grams	2-grams	3-grams
Full	135,188	3,361,273	13,144,168
Partial	132,964	2,893,520	11,152,150

4.5.2 Phone ASR system

This system is implemented in a similar way to the previous one. The training set is exactly the same as before. As regards the AM, we can take directly the same phone models as for the LVCSR. Since the AM is modeling the same HMM states to be used in both LVCSR and phone ASR systems, no further modifications are required. However, the LM has to handle phones instead of words, modeling phonotactic rules. Thanks to a lexicon, we can break down all the training set words (including EUROPARL) into phones and create a LM for the phones that represents the phonotactic probabilistic rules. We decided again for an n-gram model, in this case of order 6, so we can model general word and multi-word phone construction. The numbers of different n-grams that this system uses are shown in table 4.4, where n-grams of $n > 2$ were required to appear at least twice. This growth is clearly not exponential because many arbitrary combinations of phones are actually absurd (not only for the Spanish language) and hence not seen in the training set.

Table 4.4: Number of n-grams in the Phone ASR LM

1-grams	2-grams	3-grams	4-grams	5-grams	6-grams
51	1.6K	27K	211K	865K	2.3M

4.5.3 OOV detector

The selection of POOV SSs that are fed into the search for candidates was done in the way explained in 4.3.1. After finding the optimal LM weight for the WER performance in the LVCSR system (it took a value of 12 in Kaldi), we made the selection with three pairs of 1-best paths with LM weights chosen at different values around it (4, 5 and 6 against 18, 19 and 20, respectively).

This resulted in 25K POOV SSs with a minimum duration of 0.3 seconds and a phonetic transcription of at least four or more phones, otherwise they would likely be false alarms in terms of OOV detection. Remember that there are actually 18K OOV occurrences, so we can anticipate

that there will be a considerable number of false alarms (at least 7K). More precisely, there are around 3K OOV occurrences that were not detected, which means that there are 10K false alarms in the 25K proposed POOV SSs. Then, there are 15K detection hits, which represents more than 83% of the total 18K OOV occurrences.

Also, although all of these 25K POOV SSs activated a candidate search, a large number of them will not carry out a substitution for different reasons (such as unsuccessful searches), explained in section 4.6. This will be useful for preventing false alarms from reaching the final transcription.

We compare this OOV detection results (recall, false alarm rate and F1 score) with other works in table 4.5, indicating as well the most important characteristics of each system (in an attempt to evidence the great difficulty of carrying out such comparison). These characteristics include, in this order: the language of operation, the size of the vocabulary of the ASR systems, the number of running words of the texts employed for LM training, the domain correspondence between the train and test corpora and the OOV rate in the test corpus. Take into account that we have marked our test domain correspondence as $\sim$ In-domain because although the test corpus belongs to TCSTAR, as the train corpus does, we have removed important words for the topics present in the overall TCSTAR so that our ASR systems lack a proper modeling of the those topics, complicating the task. Also, for the other works listed, we indicate in the table the OOV results for the highest recall possible, in case multiple results are provided in the original paper.

Despite the large feature differences between the listed systems, we can notice that our OOV detection results (e.g. F1 score) are in line with those reported in the recent literature. In terms of **recall**, our system performs in most of the cases better than the other ones, being able to achieve a high recall that will offer more opportunities to improve the baseline transcription (whenever we are able to find the proper candidate). Only the system of Parada et al. (2010a) reaches a higher recall, which can be due to two reasons. First, their LM is trained with much more text material (a difference of an order of magnitude, indeed), which propitiates that IV words are more properly modeled so their system can trigger OOV detections with more confidence, without harming the rate of False Alarms (which is the same as ours). Second, they employ a complex OOV detector module based on CRFs which is also fed with a considerable amount of context information from e.g. additional LMs. In our case, the OOV detector module is preserved much simpler.

In terms of **false alarm rate**, we can observe that our system behaves similarly to the best-performing ones, although the system of Rastrow et al. (2009) appears to be considerably better. In any case, the feature differences are again too large to be able to compare them appropriately, specially since this last system has a very small vocabulary of 20K with large LM training material, which facilitates a more precise OOV detection (there are not many acoustically- nor linguistically-similar IVs with which to fill in their occurrences).

Table 4.5: OOV detection results comparison with other works

System	Language	Vocab	LM	Test domain	OOV rate	Results		
						Recall	FA	F1
Ours	Spanish	133K	45M	$\sim$ In	7.4%	83%	40%	70%
Thomas et al. (2019)	English	27K	68M	In	4.2%	75%	60%	52%
Réveil et al. (2014)	Dutch	100K	1.2G	In	1.6%	69%	44%	62%
Réveil et al. (2014)	Dutch	100K	1.2G	Out	4.6%	61%	30%	65%
Rastrow et al. (2009)	English	20K	335M	In	2.5%	80%	27%	76%
Parada et al. (2010a)	English	83K	400M	In	2.0%	93%	40%	73%

4.5.4 Search for candidates system

This system was implemented with modules that perform the different processes seen in figures 4.4, 4.5, 4.6 and 4.7. It depends on multiple parameters that will have to be calibrated empirically. Here we highlight the crucial ones:

- **Levenshtein threshold for candidate acceptance:** the first score that is computed for any OOV candidate is the Levenshtein distance. We discard any candidate with a Levenshtein distance greater than this threshold.
- **K for centroid KNNs:** K would be the maximum number of search triggers in which to carry out the LOD search for each POOV SS (apart from other search extensions that we can decide to use, as we suggest in section 4.7, not employed in this experimental framework). Furthermore, we can accept only those neighbors close enough to the selected centroid (word- or resource-based), with an additional parameter we shall now call **distance threshold for search trigger acceptance**. When the LOD search makes use of a resource-based centroid instead of a word-based centroid, we can get an idea of the value of this last parameter thanks to the pertinent quantitative analysis introduced in 4.4.2.2, as we will see in section 4.5.4.3. However, for the case of a word-based centroid, we have to make a greater effort in experimentation so as to find a proper value.
- **Final candidate score integration:** the final candidate score is computed with equation 4.3, so λ_s parameters should be adjusted accordingly. Although they are set experimentally, we can expect that the weight for the Levenshtein distance will be relatively larger, as we have already emphasized the importance of this score.
- **Candidate score threshold for substitution:** as the experimental results in section 4.6 show, it is not a good idea to decide just for the highest-scored candidate within a POOV SS and consummate a substitution in the baseline transcription. It is better to determine a minimum score that the best candidate has to reach, avoiding substitutions with poor candidates.

Finally, we detail below the LOD corpus and the word and resource embedding models that this system uses in our experiments.

4.5.4.1 LOD corpus

Although this strategy could make use of multiple LOD corpora for the LOD search, for the experiments we decided to use only a wide one that could show the performance of our TD strategy with Resource2Vec. There are few large, diverse, open and continuously growing corpora like Wikipedia. This is the reason why we chose its LOD form, DBpedia; specifically, the version for the language our ASR systems work in: Spanish. We believe its vast variety of articles make it suitable for finding a considerable set of resources on any topic or domain. This can also be seen as a challenge for a candidate search that should have the ability to skip a great number of useless resources to become efficient. Furthermore, the continuous update of Wikipedia means that new important terms such as company names are likely to have their own articles as soon as they become minimally popular.

We downloaded the DBpedia version 2016-04, Spanish partition, and queried it with local Virtuoso servers. This version holds more than 1.5M resources including articles, templates, etc., with more than 103M different connections between them.

We also make use of Wiktionary (another project from Wikimedia Foundation) for validating and expanding terms during the search for candidates, as explained in section 4.3.2. It is especially useful when we only found the base form of a word in DBpedia, as is usually the case for a verb. With Wiktionary, though, we are able to find any variant, like verb conjugations.

4.5.4.2 Word embedding model

The word embeddings used in our experiments were implemented with the Skip-gram Word2Vec scheme. The corpus used consists of more than 2.2G words with an original vocabulary of more than 4M words that was reduced to the **1M** most frequent words.

The types of texts that make up this corpus, together with their sizes, can be seen in table 4.6. First, the literature set, which covers different narrative forms (mostly novels) and also essays, is included with the intention of making proper use of the general language, rich in vocabulary and domains. These works were collected from different public Internet sites.

Second, the Wikipedia articles are needed in order to have a wide coverage of finer-grained areas of knowledge, which however are collected in a more succinct language. We used the dump from the Spanish Wikipedia on date 2017-04-01.

And third, the political records set was selected in order to emphasize the specific domain of the corpora involved in our experiments, but without the risk of losing generalization (it represents only the 17% of the data). It is composed of both the texts used for LM training in section 4.5.1 and the Spanish portion of the United Nation documents that belong to the OPUS Project Corpora (Tiedemann, 2012).

Thus, we expect to obtain word embeddings that can abstract a universal use of the Spanish language, suitable for our TD strategy evaluation as well as for any kind of general-purpose task that requires word embeddings, as NLP does.

We chose a dimension of 100 for the embeddings, which were trained in an unsupervised fashion with contexts of two words on each side, in batches of 480. This means that an epoch consists of more than 18M batches ($2.2G \cdot 4/480$). We trained this model for just one epoch, for computing cost reasons.

Table 4.6: Texts employed for Word2Vec training

Literature	Wikipedia (Spanish)	Political records
1.2G words	621M words	382M words

4.5.4.3 Resource embedding model

As detailed in section 4.4, resource embeddings were obtained with Resource2Vec, in a similar scheme to the Skip-gram Word2Vec. As can be expected, they were trained on exactly the same corpus as the one in which the LOD search is to be made, so a resource-based centroid is able to guide the search properly. Thus, the corpus used was the DBpedia in Spanish.

As stated in section 4.5.4.1, we manage a total number of 1.5M resources, with 103M different connections. The embedding dimension we chose is again 100, and the batch size 512. Then, an epoch consists of around 200K batches ($103M/512$), a considerably inferior number compared to the previous Word2Vec training, so to train over more epochs is computationally cheaper. We detected no more notable improvements after 20 epochs, stopping the training at this moment.

We can proceed now with the quantitative analyses that we introduced in section 4.4.2, for which we used Euclidean distances.

Quantitative analysis - distance-intersection relationship

Firstly, we look for a correspondence between the Euclidean distances in our model and the intersection ratio between any pair of resources (with the help of equation 4.4), what we called the *distance-intersection relationship*, so it can give insight into the other quantitative evaluations.

This is reflected in figure 4.10, where the relationship was computed as the average 1-hop intersection ratio of resource pairs that were placed at the same distance in the trained space (within intervals of 0.01). We can infer two facts from this figure. First, the correctness of the training is confirmed because of the decreasing nature of this relationship. And second, we can establish an initial *distance threshold for search trigger acceptance* (around 1.0), following the arbitrary selection of a minimum intersection ratio of 10%, which can help us to configure the generation of KNNs involved in the TD strategy. The objective of selecting such a low intersection ratio (compared to a ratio of 50%, for example) was to extend the generation of KNNs in order to also take into account resources that are not so semantically related to the centroid, so the LOD search is made more exhaustively, hoping as well that the larger number of bad candidates will not eventually create bad substitutes for the transcription because of low scores when evaluating them.

It is interesting to highlight the magnitude of the distances plotted in figure 4.10. The embeddings are taken from the surface of a high-dimensional sphere of unit radius. This means that the maximum Euclidean distance between two embeddings would be 2. However, even if we perform a uniform random placement of a large number of vectors (say, for example, 1M), the maximum distance that we would find would be around 1.7. In fact, the average separation between any vector would be around 1.4. Therefore, to find distances much below 1.4 in our Resource2Vec model, as figure 4.10 states, indicates that the model has made a great effort in the direction of learning the desired semantic relationships between resources.

Now, we show the results for the other metrics, presented in sections 4.4.2.1 and 4.4.2.2, that attempt to evaluate the quality of both the centroid and KNN calculations that occur in our TD strategy, respectively.

Figure 4.10: Intersection ratios between resources regarding distance in Resource2Vec model

Quantitative analysis - centroid calculation

We proceed first with the evaluation of the centroid calculation, which measures the quality of a computed centroid in terms of how its vicinity is arranged in the Resource2Vec space. Remember that a centroid C_a should appear in the expected vicinity of a resource R_a , when the centroid was computed with all 1-hop connections $R_{a,b}$ of R_a .

The evaluation consists of comparing the mean values of the following distance measures: distances between a resource R_a and the centroid C_a obtained from all its 1-hop resources $R_{a,b}$, what we called the *centroid-resource distances* (R_a to C_a); distances between a resource and any of its 1-hop resources (R_a to $R_{a,b}$); and distances between a resource and any of its restricted 2-hop resources, as detailed in section 4.4.2.1 (R_a to $R_{a,b,c}^*$).

The mean values of these three measurements throughout all resources are shown in table 4.7. At first glance, we confirm what we suspected in section 4.4.2.1 (equation 4.5), that $\{R_a, R_{a,b,c}^*\}$ pairs will be farther than $\{R_a, R_{a,b}\}$ pairs (in fact, we can say that $\{R_a, R_{a,b,c}^*\}$ pairs are near to the expected distance value for randomly-distributed vectors that we mentioned before, around 1.4). Moreover, $\{R_a, C_a\}$ pairs are closer than $\{R_a, R_{a,b}\}$ pairs, so the centroid hypothesis agrees with the metrics.

Table 4.7: Mean Euclidean distance measures in the Resource2Vec model

$\{R_a, C_a\}$	$\{R_a, R_{a,b}\}$	$\{R_a, R_{a,b,c}^*\}$
0.96	1.26	1.4

If we return to figure 4.10, we can see that the intersection ratios at these three distances (about $\sim 0\%$ for the last two) emphasize the different organization between $\{R_a, C_a\}$ pairs and both $\{R_a, R_{a,b}\}$ and $\{R_a, R_{a,b,c}^*\}$ pairs. Due to the steep slope of the intersection ratios between 0.6 and 1, we can see that the difference between the first two intersection values is more notable than between the last two (as we said, both near 0%).

In conclusion, this means that the centroid operation should work properly for our intentions, since with this operation we can obtain, in general, a centroid that is close to some related resource at a

distance (0.96) that is relatively much smaller than for an arbitrary selection of a $\{R_a, R_{a,b}\}$ pair (1.26), attending to the non-linearities reported in figure 4.10.

As mentioned in section 4.4.2.1, we can gain more insight about the training corpus and also the semantics of the distances inside the Resource2Vec space by means of the proposed *general auto-sharing ratio*, *GASR* (equation 4.6). For our DBpedia corpus, the *GASR* takes the value of 20.03%. If we return to figure 4.10, the distance that corresponds to this intersection ratio is around 0.93. This value is similar to the mean *centroid-resource distance* we indicated before, 0.96. Then, we can verify that the training of the embeddings was done in the proper semantic direction, as vectors that are semantically related are situated at distances of an adequate range (in contrast with what would happen if, instead of assessing 0.93 (from the *GASR* distance) vs 0.96 (from the average $\{R_a, C_a\}$ distance), we had 0.6 vs 0.96 (same origins as before), which would indicate an insufficient training).

Quantitative analysis - KNN calculation

Finally, we evaluate the KNN calculation, which measures how appropriate the KNNs of a vector in the Resource2Vec space are. Then, we will be able to find a more precise *distance threshold for search trigger acceptance* value for configuring the KNN search, compared to the value (1.0) obtained by observation of figure 4.10. Remember that this distance limit will act as a delimiting radius for the area around a centroid where we carry out the KNN search, so as to discard any neighbor farther away from this radius.

Then, we can make use of equations 4.7 – 4.9 for computing the precision, recall and *F1* score, respectively, of the neighbors obtained from all resources of the corpus. In order to do so, we established in section 4.4.2.2, with the aim of simplifying this evaluation, that we would consider as the expected-true neighbors of a resource R_a all its 1-hop resources $R_{a,b}$.

In figure 4.11 we plotted the precision, recall and *F1* score as a function of the distance threshold, and also the proportion between the retrieved neighbors and K , for $K=100$ (very similar results were obtained for other K in the interval of 5 to 500, and this is a typical value used in our TD experiments). As might have already been inferred from figure 4.10, when reaching distances of around 1 there is a marked degradation in precision, because we start to collect resources with low intersection ratios.

There is a distance threshold, $d'=1.12$, at which already 100 neighbors were accepted for any resource in our corpus. Consequently, the precision and recall take similar values at this d' (and hence the *F1* score too), and also for the subsequent distance thresholds ($d>1.12$), evidencing the behavior we anticipated in section 4.4.2.2.

We believe that the low values of both precision and recall are mainly due to the soft statement we made about which are the expected-true neighbors of a resource. As we said in section 4.4.2.2, the real true neighbors of a resource R_a should be those resources with the highest intersection ratios with R_a , which, for example, might be a subset of the 1-hop resources of R_a . This observation gains relevance if we compare the value of the *general auto-sharing ratio*, 20% (which involves intersection ratios between a resource and its 1-hop resources), with the highest intersection ratios reflected in the *distance-intersection relationship* (figure 4.10), more than 90% (which involves intersection ratios between any pair of resources). In any case, the objective of this analysis is just to find an operational starting point for configuring KNN searches in our TD strategy (by setting the *distance threshold for search trigger acceptance* parameter), where the relevance of the

Figure 4.11: Percentage of retrieved neighbors, precision, recall and $F1$ score regarding distance threshold in Resource2Vec model for $K=100$

retrieved neighbors is to be ultimately acknowledged in WER terms.

As can be seen in figure 4.11, we find an optimum for the $F1$ score at a distance threshold of 0.94. This would become the aforementioned operational starting point for configuring KNN searches in our TD strategy. Though this evaluation is carried out on KNNs computed for resources instead of for centroids (the true object of the KNN calculation operation in our TD strategy), we believe it should work for centroids too, since this result was obtained across all the M resources, achieving some generalization for any KNN search in the Resource2Vec model. This can be supported with the fact we observed for the previous centroid calculation operation, where distances between $\{R_a, C_a\}$ pairs are relatively much smaller than distances between $\{R_a, R_{a,b}\}$ pairs, as we saw in table 4.7. Thus, starting a KNN search from a resource or its 1-hop centroid will lead to very similar results. Nevertheless, this study points to the necessity of further experimental exploration to find the best value for our purposes regarding the overall TD strategy in ASR systems.

4.5.5 Evaluation scheme

The evaluation of the overall TD strategy will be made in WER terms, as is usual in ASR system evaluations. We will provide baseline WER results for two configurations of the LVCSR system described in this section. The first one, called a *full-vocabulary* system, consists of the LVCSR on which we did not carry out any vocabulary reduction, and thus it experienced the whole LM training (also, the OOV rate to be experimented by this system can be considered negligible, for the reasons explained before). In the second system, called the *partial-vocabulary* system, the vocabulary and LM of the LVCSR were limited by vocabulary reduction, making use of the 1,927 OOV list. These two results should become the relative lower and upper limits of the WER with respect to the final systems with a TD strategy, respectively, as reasoned below.

On the one hand, although the final systems are potentially capable of recovering OOVs by correcting the transcription, they lack a proper LM trained with those OOVs that could work

together with the AM during decoding, as the first system does. Thus, the first system would obtain the relative lower limit of the WER for this experimental configuration. And on the other hand, we hope that the parameters for candidate scoring should be able to properly treat a number of real OOV events and to restrict substitutions from OOV-detection false alarms with the consequence of improving the baseline transcription of the second system. If the WER of the final system is greater than this relative upper WER limit, then it does not support our initial guess, although it could be used for other purposes such as a vocabulary update or LM retraining, since modifying the baseline transcription would worsen it more than correct it if this was the case.

Then, these first two systems serve as a reference for understanding the performance of the final TD systems, which are partial-vocabulary systems with some configuration of our TD strategy.

There will be three TD systems that differ only in the type of embedding used for the LOD search, if any: either none (called the phone-TD system), Word2Vec (called the Word2Vec-TD system), or Resource2Vec (called the Resource2Vec-TD system), respectively. We will not show results for any system making use of both Word2Vec and Resource2Vec, since in preliminary experimentation the performance did not differ significantly from the systems employing just one type of embedding.

The phone-TD system, which does not make use of any type of embeddings, only works in a phone-based approach, explained as follows. If we return to figure 4.4, this system would not make use of the *search for candidates by LOD search* module. In this phone-TD system we do use a POS tag predictor on the immediate words around each POOV SS transcription, with the aim of awarding a proper candidate scoring.

In order to allow a more precise WER evaluation, every WER result will be threefold: overall WER, WER for utterances with some OOV from the 1,927 OOV list and WER for utterances with none. Remember that from the 13,294 utterances that belong to the testing set, 8,598 utterances have some of these OOVs while 4,696 ones do not. Thus, a WER over the partition without OOVs (4,696 utterances) lets us measure how much the OOV-detection false alarms that led to some substitutions might worsen a baseline transcription without OOVs. And a WER over the partition with OOVs (8,598 utterances) informs us how well we might be able to correct a baseline transcription with some OOV, at the same time canceling the effects of, first, OOV-detection false alarms substitution errors that we can still commit and, second, incorrect substitutions in correctly detected OOVs.

Therefore, large WER differences between these two test partitions are to be expected for any partial-vocabulary system (with or without a TD strategy). Nonetheless, this will also happen with the full-vocabulary system, for the two following reasons. First, in the test partition with OOVs, 67% of the utterances belong to the EPPS corpus and the rest to the CONG-PARL corpus (recall that the audio quality of EPPS is better), while in the test partition without OOVs the proportion of utterances belonging to EPPS drops to 49%. This means that an ASR system with full vocabulary will perform better for the partition with OOVs than for the other, as we will see. The second reason refers to the length of the utterances in each of these two partitions. The utterances of the partition with OOVs contain more words than the utterances of the other partition (these last ones contain 43% less words). When having a bunch of utterances of less than one or two minutes, as it is the case, short utterances would be normally more difficult to decode than longer ones (resulting in worse WER), since the dependencies learned by the LM cannot be employed so exhaustively as with large utterances (keep in mind that if the utterances were longer than a number of minutes, then the difficulties of decoding would rise again, due to the fast growing of the decoding search space).

We preferred this kind of test partitioning in utterances with/without OOVs from the 1,927 OOV list than to separate the TCSTAR test corpus in each of the two parts it is composed of (EPPS and CONG-PARL), because it is more informative to evaluate which effect the TD strategy is propitiating on each type of utterance, as explained before. In any case, we provide now an example of WER results separated by EPPS and CONG-PARL, for the full-vocabulary system: while the overall WER for the TCSTAR test corpus is $11.33 \pm 0.13\%$, the WER for the EPPS part is $8.32 \pm 0.15\%$, and the WER for the CONG-PARL part is $15.21 \pm 0.22\%$. These results can already inform us about the different levels of difficulty of both parts, mainly due to their different acoustic conditions (less favorable for CONG-PARL). We observed a similar performance behavior for the other systems that we will study.

All WER results, like the previous examples, will be presented with confidence intervals of 95%, approximated by equation 4.10 for large sample sizes (N), adapted from Weiss (2017). In our case, N takes the values 243,312 words for the whole testing set, 185,602 words for its OOV partition and 57,710 words for the non-OOV partition (in the previous WER examples, we used an N of 136,927 for the EPPS partition and 106,385 for the CONG-PARL partition).

$$\text{WER (95\%)} \approx \text{WER} \pm 1.96 \cdot \sqrt{\frac{\text{WER} \cdot (1 - \text{WER})}{N}} \quad (4.10)$$

Apart from these WER assessments, we will study other relevant specific results found in the last two TD systems (Word2Vec-TD and Resource2Vec-TD), in order to gain insight about the use of each type of embedding.

4.6 Evaluation results

In this section we present different experimental results obtained by the five systems mentioned in section 4.5.5, and which we called full-vocabulary, partial-vocabulary, phone-TD, Word2Vec-TD and Resource2Vec-TD systems.

4.6.1 Overall WER results

The overall WER results can be seen in table 4.8. The first positive observation is that the performance of TD systems is within the limits marked by the first two. Thus, instead of perceiving an overall WER increase of a relative 85.6% from the full-vocabulary system to the partial-vocabulary one, TD systems can limit this loss to 82.9%, 72.0% and 72.9%, for the phone-TD, Word2Vec-TD and Resource2Vec-TD systems, respectively. If we compare the partial-vocabulary system with those of TD, there is a significant decrease in overall WER, of a relative 1.5%, 7.3% and 6.8%, in the same order of systems as before.

Studying the behavior of performance by test partitions both with and without OOVs, we can see the effects of this strategy more clearly. The non-OOV partition is not affected significantly by the editions from OOV-detection false alarms that the TD systems made in the corresponding baseline transcription. Furthermore, the OOV partition had a significant WER improvement from

Table 4.8: WER results for the different systems

System	Total WER	w/ OOV WER	w/o OOV WER
Full Vocabulary	$11.33 \pm 0.13\%$	$10.58 \pm 0.14\%$	$13.74 \pm 0.28\%$
Partial Vocabulary	$21.03 \pm 0.16\%$	$23.40 \pm 0.19\%$	$13.40 \pm 0.28\%$
Partial Vocabulary + phone TD	$20.72 \pm 0.16\%$	$22.98 \pm 0.19\%$	$13.47 \pm 0.28\%$
Partial Vocabulary + W2V TD	$19.49 \pm 0.16\%$	$21.32 \pm 0.19\%$	$13.59 \pm 0.28\%$
Partial Vocabulary + R2V TD	$19.59 \pm 0.16\%$	$21.45 \pm 0.19\%$	$13.59 \pm 0.28\%$

the second system of a relative 1.8%, 8.9% and also 8.3% for the phone-TD, Word2Vec-TD and Resource2Vec-TD systems, respectively. The large WER differences between test partitions for all systems occur as we anticipated in section 4.5.5.

Although there is no significant difference between WER from TD systems making use of some type of embedding (Word2Vec-TD or Resource2Vec-TD), there is in fact a very significant improvement with respect to the TD system with only a phone-based approach (phone-TD), achieving 5.9% and 5.5% relative WER improvement in the overall testing set, respectively.

In all these three cases, our TD strategy has appeared to be useful for OOV solving in ASR tasks without significantly worsening the transcription where only IVs were uttered, especially with the use of embeddings.

We compare now our WER improvement results with other works dealing as well with the TD challenge, listed in table 4.9. We indicate the same experimental setup characteristics of the systems as when we compared the OOV detection results (table 4.5). Again, we want to emphasize the difficulty of carrying out a sensible comparison of the results, due to the large differences between the system characteristics. The results that we are showing in the table are, in this order: overall WER results in test for the baseline system and the TD system, the average confidence interval of the previous two WERs (CI), as given by equation 4.10, and the relative WER improvement from the baseline to the TD system (RI). We selected external works which provided the appropriate WER results so that our comparisons can be as informative as possible.

Table 4.9: Overall WER improvement comparison with other works

System	Lang	Voc	LM	Test	OOV	Base	Results		
							TD	CI	RI
Ours	ES	133K	45M	~In	7.4%	21.03%	19.49%	0.16%	7.3%
Thomas et al. (2019)	EN	27K	68M	In	4.2%	14.80%	13.00%	0.37%	12.2%
Inaguma et al. (2018)	EN	11K	~20M	In	2.4%	27.20%	26.91%	0.42%	1.1%
Inaguma et al. (2018)	JA	13K	~7M	Mix	2.0%	12.85%	12.21%	0.21%	5.0%
Réveil et al. (2014)	NL	100K	1.2G	In	1.6%	14.10%	12.20%	0.66%	13.5%
Réveil et al. (2014)	NL	100K	1.2G	Out	4.6%	26.60%	23.40%	0.76%	12.0%
Rastrow et al. (2009)	EN	20K	335M	In	2.5%	16.00%	14.90%	0.33%	6.9%

Attending to the RI, we can notice that our best TD system outperforms most of the other works listed, as occurred when comparing the OOV detection performance. The systems from Thomas et al. (2019) and Réveil et al. (2014) experimented a greater RI, which nonetheless requires some discussion.

First of all, their systems either make use of a very small vocabulary or they employ LM training material orders of magnitude bigger than ours, which may have an important effect on WER performance as we also argued when comparing the OOV detection results.

Second, our CI is much smaller, accounting mainly for a longer test corpus, which makes our results statistically more accurate. Thus, we can study how significant is the separation between the baseline and the TD system, by counting how many significant slots exist between both performances (understanding this slot as twice the CI, since a minimum separation of about twice the CI is required to consider the improvement as significant). For our system, there are 4.8 significant slots, while for [Thomas et al. \(2019\)](#) and the two systems of [Réveil et al. \(2014\)](#) there are 2.4, 1.4 and 2.1, respectively, which makes our system to be 2.0, 3.4 and 2.3 times better in this sense. Nonetheless, we should also consider against us our larger OOV rate, which in principle leaves more room for WER improvement (as long as OOV occurrences can be appropriately handled by the system, which is not trivial and could lead to the introduction of too much noise to the ASR system so that it stops working adequately). If we consider larger OOV rates as a linear advantage, we would see that our system has an advantage factor of 1.8, 4.6 and 1.6, respectively (obtained by dividing our OOV rate with the other ones). Then, we should divide the previous values $\{2.0, 3.4, 2.3\}$ with these factors $\{1.8, 4.6, 1.6\}$ (acting as a gain attenuator), to obtain a more realistic number of times that our system can be considered better than the others. This operation results in the values of $\{1.1, 0.7, 1.4\}$, which are still favorable to us (>1) in two out of three cases.

Third and last, we should point out that the transcription corrections that are allowed in our TD systems requires that the appropriate word candidate is found in some external knowledge source, which is more strict than just to employ a phone-to-grapheme conversion from the phones present in a OOV region, as the other works do.

After this review, we would like to highlight the suitability of Resource2Vec for this ASR task, making a comparison with Word2Vec. As we saw, in the TD experiments designed in this work, both Word2Vec-TD and Resource2Vec-TD systems achieved a not significantly different WER performance, which makes it impossible to choose one over another in these terms. However, we believe that the learning of LOD resources produced in Resource2Vec has an advantage over Word2Vec for the general use of our TD strategy, for the following two reasons:

- First, in an LOD search led by a word-based centroid, we will be able to retrieve only those search triggers (i.e. LOD resources) that have a correspondence with an embedded entity of the Word2Vec model (words or sequences of words), for example by uppercasing the first letter of the KNNs of the word-based centroid. Therefore, it is very likely that we will not be able to search for candidates in many LOD resources such as, for example, “*Mars 2020*” or “*33rd Goya Awards*”, since it is very difficult to properly train in the Word2Vec space an embedding of these sequences of words (we would have to treat them as a single word by explicitly giving them new word indexes). However, Resource2Vec learns the semantics of every resource attending to its respective resource connections already defined in a LOD corpus, so any resource (whose name is composed of one or multiple words) is able to appear in the vicinity of a resource-based centroid that leads an LOD search and, therefore, to be gathered as a search trigger.
- The second reason is regarding the OOV list generated in this work, which is made up of important words (in *tf-idf* terms) for different topics in a political domain. The purpose of this decision was to be able to assess the performance of our TD strategy with a proper resolution that might lead to significant measurements. Nonetheless, this is just a part of the

OOV taxonomy that an ASR system can confront with. For example, terms of contemporary creation as well as unusual terms from diverse domains may also appear in spontaneous speech. Thus, it seems likely that, in many cases, the OOVs to be retrieved might be found in very specific resources whose semantics only Resource2Vec was able to learn. This means that these more appropriate resources may be visited during a LOD search led by a resource-based centroid, so any kind of OOV could finally be retrieved.

An alternative to Resource2Vec would be to train a Doc2Vec model with all Wikipedia articles. However, with Resource2Vec we can learn about resources belonging to different LOD corpora, taking into account only their resource connections, instead of all the words they are made up of, which is computationally more expensive.

So, for all these reasons, Resource2Vec is a promising model to explore in TD tasks like that designed and experimented in this work, due to its compact modeling capacities of LOD resources. Nevertheless, further experimentation is required to fully verify the better behavior of Resource2Vec over Word2Vec for its use in general TD scenarios.

4.6.2 OOV solving success

We focus now on the performance obtained with the POOV SSs selected for the search for candidates in the Word2Vec-TD and Resource2Vec-TD systems. As mentioned in section 4.5.3, the OOV detector found 25K POOV SSs. After the search for candidates that was activated for each of them, both systems discarded almost 20K (in which most of the OOV-detection false alarms were included), either because no candidate was found, or every retrieved candidate had a score of less than the *candidate score threshold for substitution* parameter.

In the remaining $\sim 6K$ (i.e. POOV SSs that resulted in a modification of the transcription), the substitution could be made by two different kinds of terms. The first refers to terms belonging to the 1,927 OOV list or the natural OOVs found in the testing set. And the second refers to terms that belong neither to the LVCSR vocabulary (as we only retrieve OOVs) nor to the 1,927 OOV list nor the natural OOVs. Let us call this second type of substitute unexpected-OOV terms.

Regarding substitutions with unexpected-OOV terms, the Word2Vec-TD system made around 650 of these substitution errors, while the Resource2Vec-TD did it around 710 times. This occurs because POOV SS (especially those with a small number of phones, like just 4) may be filled with small and strange unexpected-OOV terms that we find in the LOD search, which are difficult to dismiss.

Then, of the $\sim 5K$ remaining POOV SSs, around 82% correctly completed an OOV in both systems (the rest resulted in substitution errors with terms belonging to the 1,927 OOV list or the natural OOVs). So, they were able to reach 4.1K and 3.8K correct OOV completions (for Word2Vec-TD and Resource2Vec-TD systems, respectively), out of almost 18K OOV instances during the test.

In this sense, both Word2Vec-TD and Resource2Vec-TD systems behaved similarly. The results also show that more than three quarters ($\sim 20K$ of 25K) of the detected POOV SSs were finally ignored when correcting the transcription. The elevated number of OOV-detection false alarms that the OOV detector produces is thus addressed by the candidate search module, where the *candidate score threshold for substitution* parameter (see section 4.5.4) appears to be critical, also limiting

the substitution errors. This parameter also reduces the number of correct OOV completions, since not all the correct potential substitutes were scored above this threshold. Thus, there is a tradeoff between correct OOV completions, on the one hand, and the removal of OOV-detection false alarms plus substitution errors, on the other, with regards to optimal WER performance.

Lastly, we would like to compare our OOV solving success rate with the one experimented in other works like those indicated in tables 4.5 and 4.9. However, only the work of Réveil et al. (2014) provide these numbers for their systems, which are 51.5% and 30.0%, respectively for the in- and out-domain evaluations. In our case (not a pure in-domain evaluation), this rate takes the maximum value of 22.8% (4.1K out of 18K). Although it might be considered a low value, it should be taken into account the very different characteristics of our system and the ones in Réveil et al. (2014), as already emphasized before. Also, consider that in our TD strategy a successful word candidate must be found in some external knowledge source, while in the other mentioned systems just a phone-to-grapheme conversion is employed. Besides, they use a 9-gram phone LM for the transcription of POOV SSs, which may memorize the sequences of phones of the OOVs seen during training, most of them appearing as well in the test speech.

4.6.3 OOV coverage study

In another perspective, we would like to study whether we can gain some insight from this experimentation as to how we should design a vocabulary update of an ASR system, since such vocabulary update might lead to some LM retraining and other dynamic arrangements, which are among the objectives pursued in the next chapter. With that purpose, we study here the coverage of the OOV terms after the TD strategy is used.

A **coverage** of the OOV terms can be computed with equation 4.11, as the proportion of the distinct substitutes obtained after the TD strategy, $subs$, that belongs to the list of OOV terms that actually occurred in the testing set, $List_{Occ}$, of size M (in our case 1,026). This computation is dependent on c , the number of times that is required for a substitute to be retrieved at least, so, for example, to compute the coverage at $c=5$, we only take into account those substitutes that were retrieved at least 5 times through our TD strategy.

$$\text{Vocabulary Coverage } (c) = \frac{\text{distinct}(subs(c) \cap List_{Occ})}{M} \quad (4.11)$$

We make this study over the testing set as its size is big enough such that most of the 1,026 terms appeared a considerable amount of times. Nonetheless, a more precise view is obtained when we study the total **representativeness** of the retrieved OOVs along the testing set (equation 4.12), where the representativeness of the n th OOV in $List_{Occ}$ is the proportion of all its true appearances along the testing set, t_n , out of the N total true OOV occurrences of all the terms in $List_{Occ}$ (18K in our case). Again, this metric is dependent on c .

$$\text{(Total) Representativeness } (c) = \sum_{i \in S(c)} \frac{t_i}{N} \quad (4.12)$$

where $S(c)$ is equal to $\text{distinct}(subs(c) \cap List_{Occ})$.

Please note that in both metrics we are requiring to retrieve an OOV term a number c of times for counting it, independently of its number of true appearances, t_n . We are aware of the limitations and imprecisions of this, so we insist that our purpose here is just to gain an initial insight on the vocabulary update problem, since in a real scenario the decisions we can make might depend on a general c but not in each specific t_n , which would be unknown.

Apart from these two metrics, it is important to quantify the number of **unexpected-OOVs** retrieved, since they are undesirable words to introduce in an updated vocabulary.

To make this study, we took the substitutes obtained with the Resource2Vec-TD system. The three previous metrics are illustrated in figure 4.12: the coverage of the 1,026 OOV list, the representativeness of the covered OOVs and the number of unexpected-OOVs substitutions. We plotted these results as a function of the number of times each substitute was retrieved by our TD strategy.

Figure 4.12: OOV coverage and representation in testing set of substitutes obtained from the Resource2Vec TD system. Also, the number of unexpected-OOVs selected as substitutes. The results are shown as a function of the number each substitute was obtained over all POOV SSs

We can see that the coverage of the OOV list descends from 42% to 11% very quickly as we retrieve from 1 to 10 substitute occurrences. However, those OOVs covered involve from 77% to 51% of representativeness in the same interval. This means that the retrieved OOVs are used frequently during the test, and then it would be interesting to take them into consideration for further steps such as the mentioned vocabulary update.

As regards unexpected-OOV substitutes, the most interesting observation is their much sharper decrease, revealing that almost all took part in just one substitution, which could also be taken into account for the experimentation of the next chapter. For example, we could think about filtering the substitutes by requiring some minimum number of retrievals for each one given a certain lapse of time, so as to have an idea about how important are the retrieved terms for the most recent decoded speech. This way, we would prevent most of the noisy substitutes from damaging the baseline transcription as well as for a better selection of substitutes for a vocabulary update.

These thoughts, together with the ability of the proposed TD strategy to recover OOV terms and correct the transcription of ASR systems, showing significant improvements in WER terms, point the way to follow in the dynamic framework of ASR systems that we will discuss in the next chapter.

4.7 An extension to the Static Term Discovery Strategy: the Word Vector Predictor System

We present here an additional module for the static TD strategy already presented, which may result in improvements over the original scheme. Since we have already shown the usefulness of the TD strategy, we reserve the experimentation of the extended strategy to the next chapter, where the strategies will be employed in a dynamic framework which will be evaluated under more realistic conditions, as we commented before. Therefore, this section acts as a presentation of this new module that we propose in this thesis work. We also make in this section a brief evaluation of the module (without employing ASR systems) to give some hints about its usefulness, and to distinguish between the three different architectures we propose for it.

This new module is called word vector predictor system (WVPS), and its intended use in our TD strategy is to complement the search triggers generation, of which we give a quick review here (the original description can be found in section 4.3.2). When a speech segment was selected by the OOV detector to become a POOV SS, we performed two types of search for candidates. The first one was based on the construction of words from the phones found in the speech segment. And the second one, the one of interest now, searched for OOV candidates in the content of some LOD resources. The specific LOD resources that were interesting for a certain POOV SS were referenced by the search triggers that we generated explicitly for each POOV SS. For this generation of search triggers, we computed a weighted centroid in the Word2Vec or Resource2Vec space, thanks to the context words that surrounded the POOV SS. Then, with a KNN operation applied to that centroid, we could finally obtain those search triggers.

The purpose of WVPS is precisely to obtain a vector in the Word2Vec space that could act as an alternative to the previous weighted centroid. With this vector, we could proceed transparently as when employing a word-based centroid in the original TD strategy, computing its KNNs and looking for the LOD resources associated to them. The advantage of WVPS is that it would be fed just with the context words of the POOV SS, without the need of the other submodules that the weighted centroid computation requires (e.g. a POS tagger).

In practice, we do not intend to substitute the weighted centroid by the output of this WVPS, but to employ both so as to enrich the search for candidates, since they can provide different semantics for the search, as we will see.

The WVPS, then, can be seen as a LM which makes use of both past and future contexts (the surrounding words of a POOV SS) of a word w in order to predict it, but instead of predicting the word w itself, we are predicting a vector in a word embedded space (Word2Vec) which is semantically related to w , so there is no need for having w in the vocabulary of the system, which is specially useful for our TD purposes.

It is important to note that, although our interest for such WVPS resides in the previously described approach for OOV resolution in TD strategies, it can be employed in any task where a two-side

(past and future contexts) word vector prediction might appear useful.

The architecture for WVPS that we are proposing in this thesis work are based on LSTMs and, one of them, also on attention mechanisms. We give a short contextualization of similar systems found in the literature, starting from neural LMs, because of their similarity with WVPS.

4.7.1 Scientific contextualization

Traditional LMs are based on n -gram statistics, typically with an n of order 3. The enduring use of these models has lead to numerous improvements through smoothing techniques that are able to leverage the counts of the tokens and also to take into consideration unseen words by setting apart some probability (Chen and Goodman, 1999).

In fact, n -gram LMs are still used in ASR, specially for the first speech decoding leading to a lattice generation. It is usual that this first LM is constrained to an operative reasonable size, and compiled into a Weighted Finite State Transducer (WFST) together with the lexicon and a HMM containing the acoustic states related to the acoustic model, as in the Kaldi toolkit (Povey et al., 2011). Then, a second, bigger LM is employed for the rescoring of the lattice, which can be based also on n -grams (perhaps of a higher order) or, more recently, on neural networks (NN)s. (Please note that in our TD scenario we are not employing a second LM since our purpose is to discover new terms, and therefore they will not appear in the output lattice, which is the input for the second LM).

LMs based on NNs have proven to achieve a good performance in perplexity terms (Bengio et al., 2003), and also their suitability for different tasks including ASR (Mikolov et al., 2010). In recent years, multiple variants have been proposed, where various regularization and optimization techniques have also been explored, as shown in Merity et al. (2018). When employing RNNs like LSTMs, the context that these models can regard is arbitrarily large for the same amount of parameters (although usually it is limited to some span), in contrast with feed forward networks, whose number of parameters increases as longer spans are required. For the input to this kind of LMs, we can employ pretrained word embeddings of the context words instead of an index-codification (like 1-hot) (Audhkhasi et al., 2016).

A recent improvement in the area of NNs is the **attention** mechanism (Bahdanau et al., 2014), employed typically in seq2seq tasks, like machine translation (Vaswani et al., 2017) or abstractive summarization (Celikyilmaz et al., 2018), but also for improving DNN classifiers like those found in more general NLP tasks (e.g. sentiment analysis (Chen et al., 2017a)). Attention allows to focus on different parts of the input sequence, which is useful both for single decision problems (because usually not all the input elements are equally important) and for seq2seq problems (because each of the output elements can be decoded more precisely focusing on different input elements). Machine translation is a clear example where the resulting alignment of both sequences via attention is inherently advantageous.

We find some works that attempt to introduce some attention mechanism into a NN LM (as we said, the most similar system to a WVPS). In Grave et al. (2016), a cache is added to the network so recent words acquire a higher probability of reappearance, while in Merity et al. (2016) a pointer sentinel is appended in order to add the possibility of selecting one of the previous words as the output instead of a prediction.

In the best architecture found for our WVPS, we will employ an attention mechanism that is applied to both sides of the handled word, in contrast to Grave et al. (2016) and Merity et al. (2016). Also, the motivation for our framework is to perform a robust estimation of a difficult word vector modulating how the surrounding terms contribute in different proportions to its prediction, rather than considering whether some recent word is to be repeated at some moment (more oriented to a long-range dependency modeling in LMs).

4.7.2 Proposed Architectures

We propose three different architectures of WVPSs that employ the context at both sides of the word to predict (or POOV SS for the TD case). The first one would act as a baseline architecture, due to its simplicity. Then, the second one will present some refinements, while the third one expands the refinements by employing an attention mechanism. Despite their different levels of complexity, they will be designed with a similar number of parameters so as to perform a comparable evaluation (shown in section 4.7.3).

For all the upcoming descriptions, we use the following notation. The letters L and R refers to the left (past) and right (future) context of the word to predict, respectively. We take the same span $C/2$ at both sides, for a total context of C . The sub-indices of each context word are numbered as their order of entrance to the front LSTMs, so the string of words that could be found in a text (or transcription) when predicting a word w would be $\{L_1, L_2, \dots, L_{\frac{C}{2}}, w, R_{\frac{C}{2}}, \dots, R_2, R_1\}$.

In the depictions, an LSTM block would receive its inputs from the bottom, in the direction of the arrow drawn inside; the outputs would leave the block from the top; and its inner state is initialized through an input at one of the sides. The same applies to the Feed-Forward (FF) block, except for the inner state, since they are memory-less.

The inputs to every architecture are the word embeddings associated to the context words L_i and R_j , of dimension d_{embed} . Their output, $\overline{predicted}$, is a vector in the same embedded space, so it can be interpreted as a point in that space with which the surrounding words have some semantic relation, and consequently this point should be close to the embedding of the actual w , $\overline{w}$, in the case it exists.

This embedding, $\overline{w}$, would be the golden standard solution during training, since we would have an embedding for any w in the training text (and similarly when evaluating the systems). However, when making use of the WVPS, we do not need to have any clue about w (it can even be an OOV word). The output $\overline{predicted}$ can already provide valuable semantic information about w , thanks to the semantics embedded in the Word2Vec space. So, this is why we could employ WVPS inside our TD strategy for solving the POOV SSs, in the manner we explained at the beginning of this section 4.7.

The *baseline architecture* is depicted in figure 4.13. It consists of one LSTM per context side and a FF block of two hidden layers. The FF is fed with the concatenation of the last outputs of the two LSTMs, so each LSTM can contribute with the information extracted from all of the inputs from its pertinent context (either left or right), and in the proper feeding direction towards the word to predict.

The second architecture, that we call *stacked architecture*, can be seen in figure 4.14. With the aim of providing the network with more complex modeling capabilities, we introduced a second layer

Figure 4.13: Baseline architecture of WVPS

of LSTMs, one per side of the context, each one receiving the pertinent outputs from the preceding LSTMs. Also, in order to take advantage of the information gathered by both LSTMs of the first layer, we add their last outputs (i.e. $\overline{L1_{oc/2}} + \overline{R1_{oc/2}}$) for initializing the states of both LSTMs of the second layer, as this can predispose the second layer of LSTMs to perform a more accurate prediction. Then, the FF is fed with the concatenation of the last outputs of the top LSTMs, as we did in the first architecture.

Figure 4.14: Stacked architecture of WVPS

Finally, the third architecture, or *attention architecture*, is drawn in figure 4.15. Now, the previous second architecture is extended with an attention mechanism, applied to the outputs of the LSTMs of the second layer. In order to generate a query vector that is able to assess the importance of each of these outputs, we make use of an additional FF block, FF_1 , fed with the concatenation of

the last outputs of the first layer of LSTMs. This way, the query should carry information of both sides of the context, so it can be employed to compute the attention weights of the outputs of both LSTMs of the second layer. These weights are computed with a dot product between the query and each of those outputs, followed by a global softmax transformation of all the dot products, so all the attention weights together sum up one. Following a similar reasoning as in the previous architecture, the query is also employed to initialize the states of the top LSTMs. When all the attention weights are computed, we perform a weighted sum of the attended vectors to feed the last FF block, FF_2 , in charge of producing the output of the network.

Figure 4.15: Attention architecture of WVPS

We changed the feeding direction of the top LSTMs in this last architecture, because, apart from that now the attention mechanism should select the adequate output combination, we are thus more sure that the information placed in their initial states (i.e. $\overline{query}$) does not vanish when reaching the most immediate context words to the word to predict, which are often more important than the other words.

4.7.3 Experiments

We conducted a series of experiments to assess the performance of the three architectures proposed. We start by describing some specificities about the word embeddings that all the architectures employed and which did not experiment any change during the training of the distinct WVPSs. Then, we present the training and evaluation materials and schemes followed. Both training and evaluation designs are common among architectures, including a similar number of

parameters (about 46M), so their results are comparable. Also, we made the context C in all our experiments to be 20 (i.e. 10 past words plus 10 future words of the word to predict).

4.7.3.1 Word embeddings

The word embeddings employed were trained following the Skip-gram version of Word2Vec Mikolov et al. (2013b), choosing a dimension $d_{embed}=100$, and contexts of four words at each side. This is the same scheme as for the word embeddings trained for the TD experiments (they can be found in section 4.5.4.2). However, the entities managed now, as an evolution, can be composed of one, two or even three words (e.g. ‘Vigo’, ‘San Francisco’, ‘Puerta del Sol’, respectively), totaling 248K, 25K and 60K tokens, respectively. We reduced the one-word tokens from the past 1M in 4.5.4.2 to 248K, in order to have more frequent entities. These entities were selected as the most frequent tokens (until some threshold) found in the training texts. For the case of multi-word tokens, we also required that they had their own Wikipedia article, paying attention to discard those entities that are not specially interesting to a general WVPS (e.g. scientific names of plants).

The corpora used for their training belong to different sources in Spanish language, which are the same ones as in section 4.5.4.2. For convenience, we indicate again their sizes in table 4.10, where we also indicate the number of lines (sentences) they are composed of, since we will also employ these sources for the training of the WVPS, which is line-based, as we will explain later. With such a variety of texts and length (62M of lines of text in total), and also shuffling their lines, we aimed to train quite universal word embeddings for the Spanish language.

Table 4.10: Texts employed for the training of word embeddings. Vocabulary: 248K one-words, 25K two-words, 60K three-words

Source	# of lines	# of words
Literature	32M	1.2G
Wikipedia (Spanish)	19M	621M
Political records	11M	382M

4.7.3.2 Training scheme

The training data for the proposed WVPSs comes as well from the training material of the previous word embeddings (table 4.10). Now, the data presentation to the systems was done in a line-based **curricular-learning** fashion, for what we performed the next preprocessing:

First, we trained an *attention architecture* WVPS with the shuffled corpora, for a quarter of an epoch, when a reasonable level of convergence was obtained. We chose this architecture as it believed to reach the best modeling and prediction capabilities so as to optimally perform the subsequent preprocessing, although not considerably differences are expected if employed one of the others for this purpose.

Second, we computed with the previous system the average cosine similarity of the predicted vectors to the K words of every line l in the corpora in vector form. This metric should give values close to 1.0 when the predictions of the WVPS over the words that compose a line are very

accurate, while values close to 0 or even negative means that the WVPS was not able to predict anything meaningful.

$$sim_l = \frac{1}{K} \sum_{j=1}^K (\overline{w_j} \cdot \overline{predicted}) \quad (4.13)$$

And **third**, we sorted all the lines by their similarity and we removed the lines with a cosine similarity inferior to 0.2 (mostly very short sentences or with unusual syntax and/or rare terms). We also removed the best 2M lines (similarities close to 1.0), since they were mostly template sentences used in Wikipedia or in political proceedings that do not appear valuable for our purposes.

With this preprocessed text, we firstly presented to our systems the M lines with the best cosine similarity, restarting the process to present again the M best lines plus the next N best ones, in an iterative manner so after each restart we present an additional bunch of N unseen lines, as in the following progression: $\{M, M+N, M+2N, \dots\}$. We made M and N to be both equal to 8M lines.

We decided for a curricular-learning fashion in training as it appears natural for the difficulty of the task. If we presented the different types of sentences to the network in a random manner, the possibility that the network is not even able to properly solve the easiest sentences would be higher than in the curricular-learning scenario, where we iteratively teach the network in a ordered way. The effects of this type of training will be studied in the results section (4.7.3.4).

Finally, the objective to minimize by the network, also called **loss**, is double. First, it takes into account, at a batch level, the cosine similarities between the output vectors, $\overline{predicted}$, and the actual words to be predicted, which we would like to minimize. And second, it performs a negative sampling from the rest of the words that should not be predicted (the cosine similarities between the output vectors and these words should be as large as possible), so as the system achieves a discriminative power (Mikolov et al., 2013b).

4.7.3.3 Evaluation scheme

We evaluated the systems after a certain amount of training batches were completed for all of them. The material employed for this testing is composed of speech transcriptions (ground truth) of two different corpora.

The first one is the Spanish partition of the TCSTAR corpus, the same employed for the TD evaluation in section 4.5, taking as well the same third that was chosen then for testing our TD systems (about 27 hours of speech), although now we are only interested in the corresponding human transcriptions.

The second corpus, denoted as TV corpus, collects audio recordings (and their human transcriptions) of political programs of the National Spanish TV, between 2015 and 2016, which were used in the Search on Speech Challenge of IberSPEECH 2018 (Tejedor and Toledano, 2018). We also took the transcription of 27 hours of speech.

The reason of evaluating over speech transcriptions is to assess the usefulness of the proposed architectures to be used as an OOV resolver in our TD systems, specially since these two corpora have very different acoustic and language characteristics, so we can obtain some insight about how useful WVPS would be for each type of corpus.

The evaluation metric employed, sim_C , is the average cosine similarity of the predicted vectors to every word in an evaluation corpus C in vector form. It can be computed with equation 4.13 by considering a corpus as a single long line with K words. Since most of the individual cosine similarities will be comprised between 0 and 1.0, the metric could be interpreted as the average probability that a predicted vector for a word w is in the semantic area of the embedded space that is most related to w .

Taking into account this idea and in order to provide the results with approximated confidence intervals of 95%, we employ the same equation we used for the WER evaluation in TD (equation 4.10), which now takes the form:

$$sim_C (95\%) \approx sim_C \pm 1.96 \cdot \sqrt{\frac{sim_C \cdot (1 - sim_C)}{N}} \quad (4.14)$$

where N is the total number of words in corpus C .

However, to employ this equation, a Gaussian distribution is assumed over all the items under evaluation (cosine similarities in our case). If we study the cosine similarities of the words in an evaluation corpus with any of the systems, we would observe the distribution showed in figure 4.16, which appears to be Gaussian except for the peak on the right, as the dashed line shows (a fitted Gaussian to the central part). The peak on the right is due to the function words, much easily to predict than normal words. Thus, if we ignore for our evaluation metric the 250 most frequent words in the WVPSs vocabulary, we can get rid of most of this peak and apply equation 4.14, with an updated value of N (smaller, so the confidence intervals would get wider).

Figure 4.16: Histogram of cosine similarities applied to the TCSTAR evaluation corpus, for the *attention architecture* of WVPS. In dashed line, Gaussian approximation of the central part

4.7.3.4 Results

The evaluation results of the three architectures are presented in table 4.8, averaging the results of two random initializations per system (though their differences were non-significant). All the systems were evaluated after the same number of training batches (with also the same training data). We stopped when the curricular-learning processed $M+4N$ ($\sim 40M$) different lines of text (120M lines in total) and significant differences between the systems were already achieved and maintained.

Table 4.11: Evaluation results of the WVPS architectures proposed, excluding the 250 most frequent words in the vocabulary

Architecture	sim_{TCSTAR}	sim_{TV}
<i>Baseline</i>	0.4112 ± 0.0031	0.3585 ± 0.0028
<i>Stacked</i>	0.5074 ± 0.0031	0.4254 ± 0.0029
<i>Attention</i>	0.5139 ± 0.0031	0.4330 ± 0.0029

As can be seen, all results show significant differences among architectures, preserving this for both evaluation corpora. The *attention architecture* achieves the best performance, with a relative improvement in cosine similarity over the *baseline architecture* of 25% and 20.8% for the TCSTAR and TV corpora, respectively; and with a relative improvement in cosine similarity over the *stacked architecture* of 1.3% and 1.8% for the TCSTAR and TV corpora, respectively. These results confirms our initial intuition about the usefulness of the proposed attention mechanism applied to this scenario.

The difference in performance between TCSTAR and TV can be understood due to the characteristics of each corpus. While TCSTAR has a clear political domain with speakers maintaining a strict formality (and therefore easier to predict), in the TV we find more spontaneous speech and more variety of topics, complicating the task.

We study now the effect of the **curricular-learning** fashion in the training of a WVPS. For doing so, we have plotted in figure 4.17 the evolution of both the local loss (i.e. not the accumulated loss over time, but the average loss of just the last 5K batches at every moment) and the average cosine similarity for the TCSTAR evaluation corpus (sim_{TCSTAR}), for the *attention architecture* of WVPS, so as to analyze the performance effects of the training.

We observe three interesting facts. **First**, there is an initial fast learning (the first $\sim 1M$ batches), which is due to the learning of basic syntactic notions and function words.

Second, every time we restart the curricular learning, delivering first the easiest sentences of the sorted data, there is a marked fall in loss, an abrupt **valley**, as expected, after which the loss starts to grow as the difficulty increases. In terms of sim_{TCSTAR} , however, there is a deterioration in each of the valleys, which can be understood since the easiest sentences in the corpora are not actually really useful for the language found in spontaneous speech (with which we perform actually the evaluation). That is why the similarity improves as the difficulty of the sentences increases, since the network is learning sentences more similar to those found in the evaluation corpora.

And **third**, there is a point in which the magnitude of the loss values grows (in the figure, this occurs after the first 15M of batches). We believe this fact does have a beneficial effect in the performance of the model (sim_{TCSTAR} in the figure), allowing it to continue with the observed

Figure 4.17: Evolution of the local loss and, in dashed line, the average cosine similarity (TCSTAR evaluation corpus), for the *attention architecture* of WVPS

progress, for the reasons we argue below.

These bigger values of the loss, together with the better cosine similarity values that are observed in parallel, may be showing that the network is outputting vectors that are close not only to the desired, ground-truth word, but also to other words that are taken by the negative sampling process. Thus, although the closeness to the correct word is pushing the loss to a low value, the proximity to the other negative words is pushing the loss to a high value. This suggests that the network is having more confusability between the word vectors that have to be predicted. Far from being a bad symptom, this is a natural phenomenon in language, where usually we can fit more than one type of word in some sentence (e.g. the words that can be between *there is a big yellow* and *in front of my house* might be *car*, *building* or even *kangaroo*, which probably have very different word vectors in a Word2Vec space).

Therefore, we can find cases where the blind negative sampling takes one of the words that are distinct to the actual word to be predicted in some position of a specific sentence, but also that this chosen word may properly fit in that position. Then, when the network is trained sufficiently, it can become aware of the suitability of both words for that position of the sentence, so it will propose in its output vector e.g. the centroid of both vectors. This means that the computed loss will be high, as observed, but in any case this is actually advantageous and the sim_{TCSTAR} can get improved, as far as the complexity of the model permits (e.g. a network with few parameters would reach a point when it is unable to capture more language features) and also as no instability of the learning is produced, which may harm the modeling capabilities of the network (if it starts, for example, to forget things it already learned before).

We also observed that this increase in magnitude of the loss values is produced in all the three architectures, at different moments of the training, even among different instances of the same architecture (keep in mind that each instance is initialized randomly). In all the cases, when this is produced, the performance of the model, in cosine similarity terms, experiments a slight boost in improvement, caused by the described phenomenon.

Final note on WVPS

The three proposed WVPS architectures have been compared in a realistic scenario involving speech transcriptions of two distinct corpora. The refinements applied to a baseline architecture have demonstrated notable improvements. Specifically, the attention mechanism introduced in the third architecture outperforms the other two with significant differences.

Therefore, their interest for a word vector prediction task is evidenced, where we can consider both the past and future contexts of a word, obtaining a vector in an embedded space semantically related to this word. This is specially useful when aiming to predict vectors for difficult words like OOV terms present in the speech decoded by an ASR system, for which the system has to previously detect the pertinent OOV regions, so its inclusion in our TD strategy is suitable, as we will do for the dynamic TD systems of the upcoming chapter.

Thesis work on Dynamic Vocabulary Strategies

In this chapter, we will deal with the problem of applying the static TD strategy presented previously to a dynamic scenario, where the evolution of ASR systems will be able to happen, in terms of dictionary and LM updates.

In such scenario, our intention is that the ASR systems react to the speech being decoded, which usually experiments changes in vocabulary or topic over time, so an automatic and unsupervised adaptation should take place to better transcribe what is said.

In the previous chapter, we showed the ability of our TD strategy to recover OOV terms present in the speech, specially when the ASR systems were limited in vocabulary so the speech contained a considerable number of OOVs. This enters in consonance with our intention to manage a medium-sized vocabulary (i.e. $<100K$) so the LM can properly model those words without increasing prohibitively in size. Now, we wish a strategy that can select some of those recovered terms for their introduction in the system's vocabulary, as well as to perform a LM update so the new words are also modeled, among other things that might result useful for the overall adaptation. This way, we will be able to have a vocabulary and LM that cover properly the linguistics at every moment of the speech, no matter if it changed topic or domain, so a reduced vocabulary size becomes significantly less harmful.

Thus, we propose a **dynamic vocabulary strategy** (or, interchangeably, **dynamic TD strategy**) that can achieve such goals, as it is detailed in section 5.1, where we also point out the contributions made in this thesis work to the field. In order to assess the performance of such strategy, we designed an experimental framework with two very different evaluation corpora, as explained in section 5.2. Finally, the results of the experimentation will be shown and discussed in section 5.3.

5.1 Proposal for a Dynamic Vocabulary strategy

The dynamic vocabulary strategy that we propose is depicted in figure 5.1. As can be noticed, it is an evolved scheme of the static TD strategy presented previously in figure 4.1 (the new abilities

of this strategy are performed by the modules of the gray area in figure 5.1).

Figure 5.1: Architecture of the proposed dynamic vocabulary strategy

At a first sight, we can appreciate in the figure the recurrent nature of the activity flow, as it should have been expected. This recurrence will allow that the LVCSR system could be modified through time, from the resulting actions performed against the speech already processed. Therefore, the input speech will be decoded in portions, so each subsequent portion might be decoded with an updated system, in case an update was triggered.

In short, we could describe the operation of this strategy as follows. Given the i th speech portion, we decode it with the last update available of the LVCSR system, and we proceed with the TD strategy of the previous chapter (with a slight modification as we mention below, due to the use of a **quarantine** of terms). This will result in the i th set of substitutes associated to the current speech portion. Then, we perform two different actions.

First, we proceed immediately with the correction of the transcription given by the LVCSR system, obtaining the *transcription portion i*. This is the same as we did for the static TD strategy.

And second, we proceed with the dynamic arrangements of this strategy (to be explained in sections 5.1.2 to 5.1.4). Firstly, we introduce the i th set of substitutes (or another set of candidates, as explained in 5.1.2.1) into a **term catalog** handled by the *vocabulary manager*, together with their scores. If a substitute was already present in the catalog, we accumulate its score. When this accumulated score exceeds some threshold γ , we say that the corresponding term is *interesting for update*. Then, when some number of terms are *interesting for update*, we activate a search in some Text Corpora in order to obtain chunks of text where these terms appear. If a term did not retrieve enough text material, we introduce it into the quarantine of terms, so during the next TD searches, we will prevent any word in the quarantine from being considered a candidate for a POOV SS (there will be some manners for removing a term from the quarantine, as we will explain). Otherwise, the term becomes *evaluable for update*.

At this point, if we decide not to use the *reconsideration mechanism*, any term *evaluable for update* would become immediately a term *ready for update*, so it will be introduced in an updated vocabulary and LM, as we will see. If we are employing the *reconsideration mechanism*, we would reconsider the previous appearances of that term in the past transcription portions, in the manner we will explain in section 5.1.4. This means we might undo the corrections we made to some *transcription portion* i^* , with $i^* \leq i$, in the most recent temporal window. And in case we undo too many times (proportionally to the number of substitutions originally made by the term), we can decide to introduce the term into the quarantine. Otherwise, the term becomes, finally, *ready for update*.

When there are terms *ready for update*, we can activate an $update(t)$ signal which would trigger the update processes to occur at time t (different ways of activating the $update(t)$ signal will be commented when presenting the mathematical model of this strategy, section 5.1.1). These update processes refer to the vocabulary selection and the LM update processes (which inherently implies a vocabulary update to the LVCSR system). As for the **vocabulary selection**, we expand the previous vocabulary with the terms *ready for update*. Also, we can remove a certain number of terms (e.g. so as to maintain constant the size of the vocabulary), by making use of the words statistics computed over the past LVCSR transcriptions and the previous background LM (this will be explained in section 5.1.2.3). This newly designated vocabulary will imply a corresponding phonetic dictionary update. And as for the **LM update**, we produce a new LM interpolating the previous background LM with a LM generated from chunks of text found in some Text Corpora that contain the *ready for update* terms, as explained in section 5.1.3. This new LM would become the new background LM. With this updated phonetic dictionary and LM, both handling the new vocabulary, we can upgrade the LVCSR system, and be ready for decoding the next speech portion, which in turn can trigger or not a new update after it is processed.

In the next subsections, we describe, first, the mathematical model behind the dynamic arrangements, and then, each of the three most important processes that are occurring in this dynamic vocabulary strategy: the **vocabulary selection** (mainly associated with the *vocabulary manager* module), the **LM update** (associated with the *LM retraining* module) and the **reconsideration mechanism** (associated with the module of the same name).

It is important to note that throughout this section 5.1 we will make an effort towards conveying all the ideas we want to propose in this thesis work for the achievement of an efficient dynamic TD strategy. This means that, at some aspects of the dynamic TD strategy, our ideas will remain in the theoretical level, and therefore not everyone will be studied experimentally. This happens due to the difficulty of implementing and exploring all the possibilities of configuration that such a complex strategy is susceptible of. Nonetheless, in the presentation of the experimental framework for this strategy (section 5.2) we will see that most of the proposed modules will be actually implemented and tested, while some other mechanisms will be simplified reasonably so as to make more affordable the experimentation.

5.1.1 Mathematical model

The dynamic processes that we have just introduced briefly (and which will be described in the following sections 5.1.2 to 5.1.4) allows an ASR system to introduce new words into its vocabulary, as well as the removal of IV terms that are considered not useful anymore. Therefore, depending on the rate of these word additions and deletions, the vocabulary size of the system may vary. In our proposal, the vocabulary growth of the ASR system will be controlled

by means of different actuators that can have an effect on the mentioned **word-add-rate** and **word-deletion-rate**.

In order to achieve such control, we designed a PID (proportional-integral-derivative) vocabulary size controller, which is depicted in figure 5.2 and which would be part of the *vocabulary manager* module of figure 5.1. This controller is in charge of ensuring that the **vocabulary size** of the ASR system at each time, denoted as $V_{Real}(t)$, is as similar as possible to a vocabulary size model, denoted as $V_{Model}(t)$, which allows us to define a PID error:

$$e(t) = V_{Model}(t) - V_{Real}(t)$$

$$Error(t) = K_P \cdot e(t) + K_I \int_{t-W}^t e(\tau) d\tau + K_D \frac{de}{dt} \quad (5.1)$$

where the coefficients K_P , K_I and K_D , all non-negative, should be empirically tuned to control the proportional, integral and derivative influence, respectively, on the final error to be addressed, $Error(t)$. W is the temporal window employed for the integral consideration (e.g. some tens of minutes).

Figure 5.2: Vocabulary size control scheme for the proposed Dynamic TD strategy

As can be seen in figure 5.2, $V_{Real}(t)$ is easily obtained by counting how many words are added, $A(t)$, and removed, $D(t)$, at each time t . Please note that the addition and deletion of words that actually affect $V_{Real}(t)$ is only produced when the $update(t)$ signal is active, and otherwise both $A(t)$ and $D(t)$ are zero, ignoring what the control mechanisms indicate about $A(t)$ and $D(t)$ during the inactivity of $update(t)$.

This $update(t)$ signal can be configured to activate, for example, whenever a certain number of words *ready for update* become available. A more efficient configuration would take into account the time-span since the last vocabulary update was performed, so as to avoid bursts of updates that may harm the efficiency of the ASR system. Additionally, the $Error(t)$ signal might also be considered in order to allow quicker updates when the error is considerable, despite the harm to the efficiency that may be caused.

We can employ the next equation to calculate $V_{Real}(t)$:

$$V_{Real}(t + \delta) = V_{Real}(t) + A(t) - D(t) \quad (5.2)$$

where δ refers to a tiny amount of time (with $\delta \rightarrow 0$) used just to indicate that the vocabulary update is produced right after time t . $A(t)$ is the number of selected terms for addition at time t (obtained by the *selection of words to add* (SWA) process, section 5.1.2.2); $D(t)$ is the number of selected terms for deletion at time t (obtained by the *selection of words to delete* (SWD) process, section 5.1.2.3). The vocabulary at time $t=0$, $V_{Real}(0)$, would be the starting vocabulary size with which we designed originally the ASR system.

$V_{Model}(t)$ is computed in the square-shaped module of figure 5.2 named V_{Model} . In its simplest version, we could set $V_{Model}(t)$ to a constant value, indicating that we want a constant vocabulary size that we would like to maintain indefinitely. Nonetheless, we propose a modeling which takes into account some circumstances Θ that can be observed during the temporal operation of the dynamic ASR system, as they can indicate that the performance of the system might be increased if the vocabulary size changed. These circumstances, and their associated consequences, will be studied later in section 5.1.1.2. The equation for $V_{Model}(t)$ is presented as follows:

$$V_{Model}(t) = V_0 + \sum_c^C k_c \cdot \left(\int_{t-W_c}^t \frac{\Theta_c(\tau)}{W_c} d\tau - \Theta_{c_0} \right) \quad (5.3)$$

where V_0 is the base vocabulary size to which we want to return whenever the C considered circumstances Θ_c are in the normal state or they cancel each other (in any case, the term to the right of V_0 would become zero). The normal state for the c th circumstance occurs when, for a pertinent temporal window W_c , the mean value of Θ_c (computed with the integral and the division by W_c) is equal to Θ_{c_0} . We can see that whenever a circumstance Θ_c moves away from Θ_{c_0} , the circumstance is indicating that the vocabulary size should be increased or decreased (depending on the sign of this difference and also the sign of k_c). Lastly, k_c is a factor that controls the influence on $V_{Model}(t)$ to be caused by the c th circumstance. The functional values for the different k_c terms may vary depending on the dynamic margin that the values of Θ_c may take, while the sign of k_c would depend on the desired direction of vocabulary increase/decrease associated to the c th circumstance (if some circumstance may propitiate a decrease to the model vocabulary size when its average value is higher than its normal state, then its associated factor should be negative, and vice versa).

An alternative form for $V_{Model}(t)$ would limit the lower and upper vocabulary sizes that the model can acquire, so we prevent the vocabulary (and indirectly the LM) from becoming inefficiently big or small for a successful recognition performance. We propose that the lower limit is indeed V_0 ,

as we consider that if a system can work efficiently with a certain vocabulary size, we should not decrease it so as not to unnecessarily loose modeling capabilities (we are assuming a $V_0 < 100K$, so there should not be the language confusability issues that may appear with greater vocabularies). The upper limit should be some cautious proportion of V_0 (e.g. 120%), so the model sizes of the ASR system are kept in a magnitude that does not harm the efficiency. The new equation for $V_{Model}(t)$ would be:

$$V_{Model}(t) = \min \left(k_V \cdot V_0, \max \left(V_0, V_0 + \sum_c^C k_c \cdot \left(\int_{t-W_c}^t \frac{\Theta_c(\tau)}{W_c} d\tau - \Theta_{c_0} \right) \right) \right) \quad (5.4)$$

where k_V is the proportion of V_0 , >1 , which sets the maximum vocabulary size that we will allow the ASR system to reach.

By making use of the previous equation, the vocabulary size will be able to grow in some controlled proportion. This can occur when the circumstances Θ enter in an abnormal state as a set (i.e. not canceling each other), indicating that it could be advantageous, at least for the duration of that abnormality, to increase the language modeling capabilities of the ASR system (since every time that a new word is added to the vocabulary, also the LM is updated accordingly, propitiating and adaptation to the input speech).

Now, we explain how we can influence the $A(t)$ and $D(t)$ terms of equation 5.2, so $V_{Real}(t)$ can be modified accordingly so as to minimize iteratively the error of equation 5.1. Firstly, $A(t)$ is tightly related to the threshold γ that we mentioned before, as it is in charge of deciding which words retrieved by the TD strategy are considered for addition to the vocabulary. Then, by modifying this threshold we would influence $A(t)$. In turn, $D(t)$ can be set more easily as we just have to select a number of IV words to remove (attending to some criteria we will explain).

At this point, we could design the modules in charge of influencing $A(t)$ and $D(t)$ (the square-shaped modules in figure 5.2 named F_γ , F_A and F_D) to depend only on $Error(t)$. However, we propose that they also depend on the word-add-rate (for $A(t)$) and the word-deletion-rate (for $D(t)$), because this will allow us to achieve an additional level of control of the dynamic processes, as the following example illustrates:

Imagine that $V_{Model}(t)$ is always set to a constant value, and that $V_{Real}(t)$ is already on the same value. This would mean that the error is zero, and therefore we do not allow to add nor remove any word to/from the vocabulary. Nonetheless, even in this case, it would be favorable that we keep adding, at some rate, the new words that are still likely to appear in the input speech, so we can adapt the vocabulary and LM to the current topics. If we also remove words at such rate, we can make more accurate the vocabulary of our ASR system. This way, the vocabulary size of the system is kept constant, and therefore the error is kept null, but we are allowing the system to renovate its vocabulary and adapt its LM, as it seems adequate.

Thus, we propose both word-add-rate and word-deletion-rate models ($WAR_{Model}(t)$ and $WDR_{Model}(t)$, respectively), explained below, so that we can compare them to the real rates ($WAR_{Real}(t)$ and $WDR_{Real}(t)$, respectively) and finally influence accordingly $A(t)$ (or, more precisely, γ) and $D(t)$, which would allow us to make the real addition/deletion rates equal to the model addition/deletion rates.

Firstly, we will present the computation of the real and model word-add-rate and

word-deletion-rate that are needed for such kind of control. Then, we will detail how the influence to both $A(t)$ and $D(t)$ can be carried out taking into account those rates.

For the real word-add-rate and word-deletion-rate, we just have to count the number of added and removed words in the most recent window of time W , respectively:

$$\begin{aligned} WAR_{Real}(t) &= \int_{t-W}^t \frac{A(\tau)}{W} d\tau \\ WDR_{Real}(t) &= \int_{t-W}^t \frac{D(\tau)}{W} d\tau \end{aligned} \quad (5.5)$$

As regards the models of the previous rates, we propose they have also a dynamic behavior, similar to the dynamism of $V_{Model}(t)$. In fact, the circumstances $\Theta(t)$ employed to influence the dynamism of the word-add-rate and word-deletion-rate models can be the same as for $V_{Model}(t)$, although we can give different relevance factors k'_c (instead of k_c) to each of the C circumstances. Thus, we will define a **generic word-rate**, denoted as $WR_{Model}(t)$, which refers to the rate of words that both the addition and deletion processes should be experimenting (so as to preserve a constant vocabulary size), and which would depend on those $\Theta(t)$. Of course, the word-add-rate and word-deletion rate models should also depend on $Error(t)$ (equation 5.1), so as to incorporate this signal to the control system. In our design, the previous $WR_{Model}(t)$ will be incorporated to the word addition/deletion models but modulated accordingly by $Error(t)$, in such a manner that these models will be equal to $WR_{Model}(t)$ only when $Error(t)$ is zero. In the next equations, we define the models, which would be computed in the module of figure 5.2 named WR_{Models} :

$$\begin{aligned} WR_{Model}(t) &= WR_{Naive}(V_{Real}(t)) + \sum_c^C k'_c \cdot \left(\int_{t-W_c}^t \frac{\Theta_c(\tau)}{W_c} d\tau - \Theta_{c_0} \right) \\ WAR_{Model}(t) &= \max(0, WR_{Model}(t) + k_E \cdot Error(t)) \\ WDR_{Model}(t) &= \max(0, WR_{Model}(t) - k_E \cdot Error(t)) \end{aligned} \quad (5.6)$$

where it can be seen that $WR_{Model}(t)$ depends on $\Theta(t)$, as we mentioned, and also on $WR_{Naive}(V_{Real}(t))$, a **naive model** of word rate that depends only on a vocabulary size, serving as an offset to $WR_{Model}(t)$ (we will give more details about this naive model in section 5.1.1.1). Then, the word-add-rate and word-deletion-rate models, restricted to be non-negative, are configured to follow $WR_{Model}(t)$ except for the modulations imposed by $Error(t)$ (these modulations are applied in different directions to each of the models, so as to propitiate an imbalance of addition and deletion rates that will allow the vocabulary to grow or decrease as required). Lastly, the influence of $Error(t)$ can be controlled with the positive coefficient k_E .

Finally, we can present the equations in charge of determining γ (occurring inside the F_γ module of figure 5.2), which in turn will influence $A(t)$ (whose determination occurs inside the F_A module), and also the equations in charge of determining $D(t)$ (occurring inside the F_D module). Firstly, the equations related to γ and $A(t)$ would be:

$$\gamma(t) = \max(\gamma_{min}, \gamma_0 - k_A \cdot (WAR_{Model}(t) - WAR_{Real}(t))) \quad (5.7)$$

$$A(t) = \begin{cases} f(t, \gamma(t)) & \text{if } update(t) = 1 \\ 0 & \text{otherwise} \end{cases} \quad (5.8)$$

where γ_{min} is the minimum value for the threshold γ that is reasonable to tolerate, since a very low value may imply that we introduce into the vocabulary terms of questionable usefulness for the ASR system (therefore, only intended for moments when the model indicates that we should add a big amount of words to the vocabulary). γ_0 is the value for the threshold when both real and model rates are equal, where we should still keep adding at least some new words that are likely to appear, as previously discussed. Then, γ_0 (bigger than γ_{min} , as we reason below) can be seen as a **regulator term**, specially for the case in which both real and model rates are equal, when otherwise γ would take the γ_{min} value, a so low value, as we said, that might compromise the quality of the added words. k_A is a positive coefficient to control the speed of correction of the word-add-rate. The minus sign before k_A is needed because the lower the threshold γ , the more permissive we want to be for introducing new words into the vocabulary. And in the equation for $A(t)$, $f(t, \gamma(t))$ is a function tightly related to $\gamma(t)$, although other temporal arrangements are also taken into account, as we will see in section 5.1.2.2. However, this function $f(t, \gamma(t))$ will be evaluated only when the $update(t)$ signal is active, as we explained at the beginning of this section, so $A(t)$ would be zero otherwise.

With the knowledge learned in the previous chapter, we can already have a notion about the values that both γ_{min} and γ_0 would take. Regarding γ_{min} , it should be close to the score threshold we imposed for a retrieved candidate to become a substitute of the transcription content of a POOV SS. Since this substitution threshold might be quite restrictive (as we wanted to reduce the risk of damaging the transcription with wrong modifications), we can afford a bit smaller value for γ_{min} , since the introduction of a bad term into the vocabulary does not seem as dangerous (a newly introduced term has to fit the current LM of the ASR system so as to the term is transcribed in the future, and otherwise it is likely to be removed in a forthcoming update, as we will discuss in section 5.1.2.2). In any case, the usefulness of the words entering the vocabulary through a very low value of γ_{min} are likely to be questionable, so γ_{min} should be close to the substitution threshold.

As for γ_0 , we should go back to figure 4.12, where we plotted the OOV coverage and representativeness along with the number of unexpected-OOVs selected as substitutes during the experiments of the previous chapter. Please recall that unexpected-OOVs are OOV words that never happened in the testing speech, so they are, in principle, undesirable words (as the modifications they cause to the transcription would be certainly wrong), although there is a chance that an unexpected-OOV is actually semantically related to the current speech and so it could benefit the LM if we learn from it. In any case, the main insight obtained from that analysis (across 27 hours of testing speech) was that by requiring a term to have been selected as a substitute at least twice, we could get rid of most of unexpected-OOVs without being penalized in terms of OOV coverage and representativeness. Therefore, by making γ_0 **twice** the substitution threshold, we can be confident that the terms that enter the vocabulary through γ_0 would be quite reliable, as it means that, on average, they are words that were retrieved at least twice and that performed modifications to the transcription that are likely to be correct. This already suggests that the scores that we keep accumulating in the term catalog should experiment temporal penalties so we only consider words of recent discovery, permitting that the previous observation still prevails (we will explain this disposition in more detail in section 5.1.2.1).

Finally, we proceed with the determination of $D(t)$, which can be achieved with the next equation:

$$D(t) = \begin{cases} \max(0, k_D \cdot (WDR_{Model}(t) - WDR_{Real}(t))) & \text{if } update(t) = 1 \\ 0 & \text{otherwise} \end{cases} \quad (5.9)$$

where k_D is a positive coefficient to control the speed of correction of the word-deletion-rate. The limitation imposed to $D(t)$ to be non-negative is obvious, as we cannot remove a negative number of IV words. Note that now it is not necessary to introduce a regulator term as before, since, in principle, there will be always a notable amount of words suitable to be removed, so the quality of the removal process should not be compromised (we will explain this process in section 5.1.2.3). Lastly, and as occurred with $A(t)$, we only evaluate the value of $D(t)$ indicated by the models when the signal $update(t)$ is active, and otherwise is zero.

5.1.1.1 Naive word rate model

With this model, denoted as $WR_{Naive}(V)$ and which appeared in equation 5.6, we intend to learn in advance how often we might find new words in any input speech given a vocabulary of some size V . For example, with a small vocabulary, we can expect that the rate of OOV words is higher than for a bigger vocabulary, since a smaller number of words are covered by the first vocabulary. Thus, such a model would inform us about which rate of word addition we should be experimenting in our dynamic TD strategy, for the vocabulary size of an ASR system (the same rate would be assumed for the word deletion, as we argued before). Then, if no circumstance Θ was to be considered, this naive model would constitute already our $WR_{Model}(t)$.

This model can be obtained by studying, in some training speech, how often distinct OOV words appear through time, assuming a starting vocabulary. This starting vocabulary should be updated as new words appear in the speech, so as to simulate that we are able to learn them immediately, growing in size. The more diverse are the different training sources, the better generality we may reach for the model, which will allow us to apply it to any other input speech. Then, we can easily find a relationship between the rate of new words found in the speech and the vocabulary size. In the experimental framework presented in section 5.2, we show in more detail how to construct this model (specifically on section 5.2.2.2).

5.1.1.2 Circumstances to consider in the dynamic processes

We will consider some circumstances Θ that can lead the $V_{Model}(t)$ and also the $WR_{Model}(t)$ models (so this last one is not only depending on the previous naive model). The influence of each circumstance on these models will be controlled by the k_c and k'_c coefficients of equations 5.4 and 5.6, respectively, which permits that a circumstance is more or less critical for a specific model.

In principle, we can think of any circumstance that we reasonably believe that it should make an impact in the mentioned models. For this thesis work, we propose two circumstances to consider:

- The first one refers to the **rate of POOV SSs** detected in the input speech during the most recent window of time. A low rate would most probably mean that the ASR system is properly tuned to the input speech, both in language and acoustic terms. Therefore, we can afford to maintain constant $V_{Model}(t)$, as no new words seems to be needed, and to keep low $WR_{Model}(t)$, for the same reason. Conversely, if the POOV SS rate is high, then it might be useful that we allow $V_{Model}(t)$ to grow, so the ASR system has more vocabulary options to succeed against the hostile speech. Similarly, $WR_{Model}(t)$ should increase, so the vocabulary (and therefore the LM) can adapt faster to the input speech. In this last case, if this circumstance governs notably the two models (i.e. high associated k_c and k'_c), the overall reaction should be to increase both $WAR_{Model}(t)$ and $WDR_{Model}(t)$, although $WAR_{Model}(t)$ would be increased more than $WDR_{Model}(t)$, so the vocabulary is still able to grow.

Nonetheless, the rate of POOV SSs does not allow us to distinguish between acoustic speech-to-system decoupling and language speech-to-system decoupling (as we argued when presenting the OOV detection operation in section 4.3.1.3). Thus, it would be advisable to introduce into the ASR systems some mechanism that allows us to discern between both types of decoupling, so we could reserve the actions of the proposed strategies for the language type of decoupling. Such additional mechanism would be an interesting line of ulterior research, out of the scope of this thesis work.

The rate of POOV SSs detected can be easily measured by counting the POOV SSs detected in the last temporal window, and dividing by the temporal span of that window, so we obtain a rate whose units are POOV SSs per second (this is similar to the way we calculate the average values of $\Theta_c(t)$ in equations 5.4 and 5.6). We can denote this rate by $\Theta_1(t)$. Then, Θ_{1_0} would be the rate of POOV SSs detected for speech that is coupled with the training conditions, so we can have a notion about which rate is normal for speech that the system is quite comfortable with. It can be obtained, for example, by measuring the POOV SS rate over a portion of the training speech from which the AM was trained, but employing a LM a bit more generalist inside the LVCSR system so it is not too related to the language used on the training speech, and so we can expect some OOVs to appear. Then, the associated k_1 and k'_1 coefficients should be positive so as to preserve the desired behavior (the greater, the more increase is expected for both $V_{Model}(t)$ and $WR_{Model}(t)$). Lastly, the temporal window for calculating average values, W_1 , could be configured to be equal to the window W used for the PID error (equation 5.1), at least until further experimentation suggests other values.

- The second circumstance we propose is related with the **rate of non-substituted POOV SSs** detected in the input speech during the most recent window of time. This circumstance is conceived as a complement of the previous one, where now we only count those POOV SSs that did not retrieve a proper candidate and therefore they did not propitiate a correction to the system's transcription.

By making this distinction, we can know whether we are being able to find substitutes (which in turn might be correct or incorrect) for the POOV SSs that we detect, which anyway can inform us about the difficulty of the input speech, as reasoned next. If this rate is notably high, it might mean that the OOVs uttered belong to a different topic or domain for which the LM is not prepared, since the reason why the missing terms are not discovered can be that the context words around the OOV regions are incorrect. In this case, we can force an extra increase to both $V_{Model}(t)$ and $WR_{Model}(t)$, so more new words are added to the vocabulary, hopefully some of them belonging to the new topic, which would propitiate at least a slight adaptation of the LM. This initial adaptation would allow that some context words for the most difficult OOVs to retrieve are actually in consonance with the topic that is present in the current speech, improving thus the term discoveries, so a proper adaptation

of the LM is reached eventually. The opposite case, however, in which this rate is notably low, is not as informative as the other, since it might just mean that the term discoveries are being produced adequately.

For this Θ_2 circumstance, we can proceed similarly as with the previous one so as to obtain Θ_{2_0} and to set k_2 , k'_2 and W_2 . Nonetheless, the influence of this circumstance over the models should be reserved only for propitiating extra increases (and not decreases) to the vocabulary and the word rate, as we reasoned before, so we would have to limit its associated modulation to be non-negative.

5.1.2 Vocabulary selection

With this process, we intend to **designate** a vocabulary for our ASR systems that would hopefully match the vocabulary being used in the input speech. This requires, therefore, the selection of a set of new words that should be added to the vocabulary at a certain moment, $A(t)$, and also the selection of a set of existing words in the vocabulary that should be removed, $D(t)$, in a magnitude determined by the vocabulary size controller we have just presented.

The module in figure 5.1 that is in charge of this process is the *vocabulary manager*. It can work in conjunction with the *reconsideration mechanism*, as we explained before, but since this mechanism is an optional feature of this strategy, we postponed its description to section 5.1.4.

We insist that not every speech portion which served as input to the strategy has to trigger a vocabulary selection (and hence neither a LM update), as it is a matter of this module to *decide* whether this update is possible or desirable at each moment (considering the different factors mentioned in section 5.1.1), activating in the positive case the $update(t)$ signal of figure 5.2.

The operation of this module can be divided in three subprocesses: **term collection** (5.1.2.1), in charge of introducing to the term catalog the set of substitutes from the TD strategy at some time t_i (t_i is the absolute end time of the i th speech portion with respect to some origin), being also in charge of updating the scores of all the terms in there; **selection of words to add (SWA)** (5.1.2.2), with which we select, if possible and needed, some of the terms in the catalog to be added to the vocabulary; and **selection of words to delete (SWD)** (5.1.2.3), with which we select, if needed, some of the existing terms in the vocabulary to be dropped from it. So, finally, the size of the updated vocabulary at some time t_i , $V_{Real}(t_i)$, can be described by rewriting the equation 5.2:

$$V_{Real}(t_i + \delta) = V_{Real}(t_i) + A(t_i) - D(t_i) \quad (5.10)$$

where δ refers, again, to a tiny amount of time (with $\delta \rightarrow 0$), indicating that the vocabulary update is produced right after time t_i . $A(t_i)$ is the number of selected terms for addition from the term catalog at time t_i , obtained by the SWA process; and $D(t_i)$ is the number of selected terms for deletion at time t_i , obtained by the SWD process.

5.1.2.1 Term collection

The purpose of the *term collection* process is to update the **term catalog** at each time t_i . The content of this catalog is composed of words accompanied by their accumulated score and the number of times they were collected into the term catalog. This content will be manipulated in the manner we explain below.

First, we have to recall that the TD strategy can find a set of candidates (possibly empty) for each POOV SS. From this set, we examine the candidate with the best score (as explained in section 4.3.2.3). In case this score was equal or higher than the **candidate score threshold for substitution**, introduced in section 4.5.4, it becomes a **substitute** to replace the content of the corresponding POOV SS.

Then, we could also select these substitutes to be introduced into the text catalog. However, it may be adequate that we apply a different score threshold to the best-scored candidates for the catalog introduction than the **candidate score threshold for substitution**, preferring a lower threshold that we can call **candidate score threshold for term collection**, reasoning as follows (similarly as we did in section 5.1.1 for configuring γ_{min} in equation 5.7).

When deciding whether or not to make a correction to the baseline transcription from the LVCSR system, we should be sufficiently sure about the correctness of a substitute, otherwise there would be an inadequate risk of worsening the transcription. Nonetheless, since the term catalog is intended to register any term that might be interesting to be considered for a future word addition to the vocabulary, we can be more permissive in order not to miss terms that would be actually useful for ulterior decoding, even if we did not score them enough for performing a correction to the transcription. This can occur because the SWA process will be in charge of applying additional and more strict requirements to the terms in the catalog so as to be taken into consideration for that addition to the vocabulary (thanks to the use, among other arrangements, of $\gamma(t)$ from equation 5.7), as we will see in the next subsection.

Therefore, we will collect, at each time t_i , all the best-scored candidates whose score is equal or higher to this **candidate score threshold for term collection**. We call such a score as **collection score**, and it can be denoted as $Score_{w,t_{coll_w}}$, since it is associated to a specific candidate w and a specific time t_{coll_w} when it was retrieved. Then, for each of these candidates, we proceed in the next manner. If the candidate was not in the term catalog, we create a new entry for it, starting its accumulative score with its current collection score, while its number of entrances to the term catalog is set to one (1). If the candidate was already in the catalog, we add its current collection score to its accumulative score, and we add one (1) to its number of entrances to the term catalog. These accumulations will be used by the SWA process.

As we said, we can afford to make the **candidate score threshold for term collection** (which we can denote now as γ^{Coll} , distinguishing it from the mentioned γ that will appear again in the next subsection) smaller than the **candidate score threshold for substitution** (which we can denote now as γ^{Subs}). Furthermore, we could also allow γ^{Coll} , and perhaps γ^{Subs} as well, to experiment a dynamic behavior similar to $\gamma(t)$ (which we can denote now as $\gamma^{Updt}(t)$, as it is employed for considering whether a term is interesting for a vocabulary update). Nonetheless, for the case of $\gamma^{Subs}(t)$, the allowed variations should be very limited, due to the immediate danger of deteriorating the speech transcription. Thus, we could configure these thresholds with new equations that would be identical to equation 5.7, changing pertinently the γ_{min} and γ_0 coefficients. A tentative configuration of all these coefficients is showed in table 5.1, where we took into

consideration the discussions held in this section and in section 5.1.1, and also the substitution threshold that we may have learned in advance (e.g. from the previous chapter), which is denoted as x . Now, the γ_0 value for $\gamma^{Updt}(t)$ is presented as $k_u \cdot x$, with $k_u > 1$, instead of $2x$, as we mentioned before, so there is more freedom for configuring this parameter (although $2x$ is already reasonable).

Table 5.1: Coefficients configuration for the three thresholds $\gamma^{Subs}(t)$, $\gamma^{Coll}(t)$ and $\gamma^{Updt}(t)$, with $\epsilon_1 \leq \epsilon_2 \leq \epsilon_3$

Threshold	γ_{min}	γ_0
$\gamma^{Subs}(t)$	$x - \epsilon_1$	x
$\gamma^{Coll}(t)$	$x - \epsilon_3$	$x - \epsilon_2$
$\gamma^{Updt}(t)$	$x - \epsilon_2$	$k_u \cdot x$

Temporal penalties in the term catalog

Since the importance of a term is likely to vary over time (e.g. a term might be employed just in a certain segment of the input speech), the accumulative score managed by the term catalog should not be kept independently of time. Then, this score should experiment a **decrease over time**, so when a term is not retrieved again for a considerable amount of time, its accumulative score should have been gradually decreased to zero (0), a moment when we should also reset its count of entrances to the term catalog. This way, we ensure that the accumulative score at any time is a reflection of the current importance of a term in the input speech, and therefore is a reliable measure for considering its addition to the vocabulary, as explained in the next subsection.

We can achieve this score decrease making use of the proper equations that describe the specific behavior we are looking for. Any of such equations should work independently for each term w in the catalog. For example, if we want a linear decreasing fashion, we could employ the next couple of assignments, presented in equation 5.11, to be carried out in **sequential** order for each of the words w to be managed by the catalog. For better understanding, we would like to emphasize that the accumulative scores will be computed only in two kinds of moments: first, after decoding a speech portion i , a time denoted as t_i ; second, after a candidate w was retrieved by the static TD strategy and also collected into the term catalog, a time denoted as t_{coll_w} . The first type of moments are needed because it is when we decide whether to carry out a vocabulary and LM updates, so we have to evaluate the pertinent accumulative scores of the words in the term catalog. And the second type of moments are needed in order to update accurately the new accumulative scores for the lastly collected terms.

$$\begin{aligned}
(1) \quad AS_w &\leftarrow \begin{cases} 0 & \text{if } t = 0 \\ \max(0, AS_w - a \cdot (t_i - t_{proc_w})) & \text{if } t = t_i \\ Score_{w, t_{coll_w}} + \max(0, AS_w - a \cdot (t_{coll_w} - t_{proc_w})) & \text{if } t = t_{coll_w} \end{cases} \\
(2) \quad t_{proc_w} &\leftarrow \begin{cases} 0 & \text{if } t = 0 \\ t_i & \text{if } t = t_i \\ t_{coll_w} & \text{if } t = t_{coll_w} \end{cases} \quad (5.11)
\end{aligned}$$

Firstly, we update the accumulative score of a word w , denoted by AS_w (and initialized to zero at

start), whenever a speech portion i is decoded (i.e. at time t_i) or the term w was last collected into the term catalog (i.e. at time t_{coll_w}). t_{proc_w} refers to the last moment for which we have already *processed* the temporal penalties for a word w , so it must be updated at t_i and t_{coll_w} , concurrently with the AS_w updates (this will be done in the second assignation). a is the slope of the linear decreasing, which would consider the time passed between a score-update moment (either t_i or t_{coll_w}) and the timestamp of the last temporal-penalty processing (t_{proc_w}). The temporal penalties are restricted not to be larger than the last accumulated score for a word w , which permits that the accumulated scores are non-negative. When the word w is collected, we accumulate its collection score $Score_{w,t_{coll_w}}$, which is obviously excluded from temporal penalties at that moment t_{coll_w} . **Second** and last, we update t_{proc_w} every time we update AS_w (at both t_i and t_{coll_w}). t_{proc_w} is also initialized to zero at start.

In case we prefer an exponential decrease over time, we would replace the **first** assignation of 5.11 (where the accumulated scores are updated, as the updates to t_{proc_w} would remain the same). Thus, the new assignations would now be:

$$\begin{aligned}
 (1) \quad AS_w &\leftarrow \begin{cases} 0 & \text{if } t = 0 \\ AS_w \cdot e^{-b(t_i - t_{proc_w})} & \text{if } t = t_i \\ Score_{w,t_{coll_w}} + AS_w \cdot e^{-b(t_{coll_w} - t_{proc_w})} & \text{if } t = t_{coll_w} \end{cases} \\
 (2) \quad t_{proc_w} &\leftarrow \begin{cases} 0 & \text{if } t = 0 \\ t_i & \text{if } t = t_i \\ t_{coll_w} & \text{if } t = t_{coll_w} \end{cases} \quad (5.12)
 \end{aligned}$$

where b is a coefficient that controls the decay rate of the scores. The rest of the terms have the same meanings as for the previous assignations.

5.1.2.2 Selection of words to add

The SWA process selects words from the term catalog that may be interesting for extending the vocabulary of the ASR systems, performing additional analyses so as to decide whether their addition is finally adequate or not. These actions are done in **three steps**, occurring after decoding each speech portion i .

First step

In the first step of the process, we select any word from the term catalog whose accumulative score is equal or higher than the threshold $\gamma(t)$ (configured dynamically with equation 5.7 already discussed), which we can call **accumulative-score threshold for update interest**. If there are some, these terms become *interesting for update*. As it is obvious, this threshold should be higher than the **candidate score threshold for term collection** and, in normal conditions (as explained in section 5.1.1), it should be higher than the **candidate score threshold for substitution**, so we only include terms into the vocabulary whose usefulness for the current speech is sufficiently evidenced.

Making this new threshold relatively high would enable that the terms that we might have permissively introduced into the term catalog tend to have a null accumulative score (due to the decreasing behavior of this score, as explained before) if they just happened once and not anymore for some amount of time. Also, those terms that were collected multiple times but with a small score, will be able to be considered *interesting for update* as long as they were retrieved at close instants of time, so they trigger our attention.

The specific values that are needed for configuring this threshold (we are referring to the γ_{min} and γ_0 coefficients of equation 5.7) were already discussed in section 5.1.1.

This scheme of initial selection of terms from the term catalog could be refined if we also make use of the other data present in the term catalog, which is the number of entrances to the term catalog that a term w performed (which should be subject to a temporal penalty, as we explained before, so our information reflects the characteristics of the current speech).

Thus, instead of relying on $AS_w(t)$ for deciding whether to proceed examining this word (whenever $AS_w(t) \geq \gamma(t)$), we could also consider the number of entrances to the term catalog (denoted as $NE_w(t)$) that w achieved. For example, we could be more permissive with terms that appeared very often but with small score, so we can benefit earlier from it by delivering it faster to the next step of this process (we will also be able to discard it faster, if necessary, so it stops from introducing noise into our systems, as we will see). Then, for this new criterion, we would design a function $g(NE_w(t))$ that would act as a scaling factor for the accumulated score $AS_w(t)$ (e.g. it could be 1 for a $NE_w(t)=1$, and it could be 1.1 for a $NE_w(t)=3$), so finally we would evaluate whether $g(NE_w(t)) \cdot AS_w(t) \geq \gamma(t)$ so as to make the selection of this first step.

Second step

Now, we want to make sure that the terms that became *interesting for update* in the previous step are actually trainable in LM terms and are not just very rare terms (and possibly mistaken forms of other words), otherwise their inclusion in the system's vocabulary would not make much sense (either the LM could not evolve properly so as to predict their occurrences, either they are likely rubbish terms that would introduce noise to the decoding).

For ascertaining so, we make use of some **Text Corpora** that should be sufficiently large so as to cover extensively the usage of a huge number of words, and also sufficiently updated so as the vocabulary covered includes the words of most recent usage and creation (e.g. neologisms), as well as how the words are used at the present time, since e.g. the syntax of some type of sentences might evolve distinctly among different domains. These two requirements suggest that the Text Corpora should be updated periodically and composed of multiple and diverse sources, as they would conform the knowledge pool that the dynamic vocabulary strategy is actually going to consult to learn about the language of the input speech. Thus, it would be appropriate that these sources of information are freshly retrieved from the Internet.

With this Text Corpora, we can search for occurrences of the terms *interesting for update*, and if we do not retrieve a **minimum number of lines** for a term, we will discard it, for the reasons we gave before. Furthermore, those terms without enough textual instances would be included in a **quarantine** of terms.

It is likely that terms that end in such quarantine are actually garbage or very rare terms which somehow fitted certain POOV SSs (most probably only in acoustic terms). Therefore, they are

potentially dangerous for introducing noise into the systems, so we could use the quarantine to prevent those terms from being retrieved again during the operation of the TD strategy in subsequent time steps.

In this line, one advantage of using the quarantine is the possibility of finding better candidates for POOV SSs, specially when the best candidate retrieved for a POOV SS was a rubbish or rare term which, thanks to this mechanism, was then added to the quarantine and therefore discarded in the future. In these cases, it is possible that this rubbish or rare term became the best-scoring candidate because it fitted some of the phones of the speech segment, while the true OOV that appeared in the POOV SS is in the second place of scoring, due to, for example, a mismatch between its phones and the phones that we were able to decode (perhaps because some part of the pronunciation of the word was never seen during the training of the phonotactic LM of the Phone ASR system, and so it is difficult that we could decode that precise sequence of phones). Therefore, after the introduction of the wrong term in the quarantine, the next time the OOV is uttered, it is likely that the correct candidate acquire now the best score, so we would be able to properly correct the transcription and to introduce the correct word into the term catalog (and then to the vocabulary of the system, if appropriate).

Nonetheless, it would be possible that some term in the quarantine was actually useful but of so recent creation/usage that the Text Corpora did not have enough occurrences of it. Then, this quarantine should also indicate the date of entry of every term, so we can verify whether in the next update of the Text Corpora there are enough occurrences of the **lastly** quarantined terms (for efficiency reasons). In a positive case, we would liberate the pertinent term from the quarantine, so it will be able to be retrieved again by the TD strategy. In a negative case, the term could still leave the quarantine if already enough time has passed since it entered the quarantine, as we may have already found the proper word candidates that the quarantined term was preventing us from retrieving (as well as for those cases in which the quarantined term finally becomes useful for the current speech).

All in all, if in this second step we were able to retrieve a sufficient number of lines of text for a term, that term becomes *evaluable for update*.

Third step

In the last step, we can decide to perform or not a final evaluation of the surviving terms from the previous step, the terms *evaluable for update*. This evaluation occurs in the *reconsideration mechanism* module, and since it is optional, we leave its description to section 5.1.4.

In case we decide to not perform this evaluation, all the terms *evaluable for update* become terms *ready for update*. In the other case, only the terms that passed the evaluation would become so.

Then, these terms *ready for update* will be the ones that we will introduce into the system's vocabulary at time t_i , although they can be subject of optional extensions that we will discuss later. In any case, the final **number** of terms that we add at time t_i would constitute the value of $A(t)$ for $t=t_i$. Also, we would like to mention that this process of selecting words to add was referenced with the function $f(t, \gamma(t))$ inside equation 5.8.

Reliability of the *selection of words to add* process

After explaining the three steps for deciding whether a term is to be introduced into the vocabulary, we discuss about how reliable is this *SWA* process. For doing so, we focus on the **second step** of this process, as it depends on external knowledge sources that can be arbitrarily trustworthy and clean, or to some extent polluted with garbage and misspelled words.

In that second step, we decide on a term regarding whether there are enough lines of text in some Text Corpora, which sometimes could lead to the acceptance of undesirable terms or the rejection of valuable ones. Then, as these two unwanted events do not seem to be easily avoidable (as we do not know in advance about the true usefulness of a term for the input speech), we will show how the overall dynamic TD strategy is able to face any possibility.

Thus, we sum up in table 5.2 the different cases that can occur in the second step of the *SWA* process, for which we indicate which future behavior is expected from the dynamic TD strategy so as to deal with them, even without making use of the special reconsideration mechanism. The use of *expectation* comes due to the fact that the information of the first column in table 5.2 is unknown for the strategy, but for this analysis we assume that this information is known by us as evaluators.

Table 5.2: Different cases for the terms *interesting for update* and expected behavior of our dynamic TD strategy, when no reconsideration mechanism is employed

Term is useful for input speech	Retrieved minimum number of lines	Expected behavior of the dynamic TD strategy
YES	YES	Term is added to the vocabulary and starts to be recognized
YES	NO	Term is quarantined, but hopefully there are more text instances in next Text Corpora update
NO	YES	Term is added to the vocabulary, but hopefully it will be purged in a forthcoming word removal
NO	NO	Term is quarantined and does not introduce noise into the ASR systems

The **first** and **fourth** cases of table 5.2 correspond to the optimal conditions of operation.

The **second** case serves to reinforce the importance of maintaining up to date the Text Corpora that the strategy is employing, since these cases are probably associated with terms of recent creation or words that acquired relevance due to some news or event (e.g. the surname of a the newly elected president). Also, we need the arrangements made to the quarantine that we explained before, in order to be able to quit interesting terms from the quarantine and possibly put them into the system's vocabulary when there are enough text instances for a term.

Finally, the **third** case is related to the purging ability of the dynamic TD strategy, which is associated with the *SWD* process of section 5.1.2.3. As an advance, we mention how a dynamic system would be able to automatically **purge** terms that are not useful for the current input speech. This purge of terms comes from the need of preserving a desired vocabulary size (as explained in section 5.1.1), which will also help us to focus the modeling capabilities of the LM around the topics that may be most useful for decoding the input speech. Thus, we employ a couple of criteria for deciding which IV terms would be eligible for removal. First, we can study how frequently used are the different IV terms (since a low or null frequency in the most recent window of time

would indicate little usefulness of that term for the current speech). And second we can study how well a term fit in the current state of the LM, which is assumed to be considerably adapted to the input speech thanks to the dynamic TD strategy (this LM-fitness of a term would be directly proportional to the whole n-gram log-likelihoods involving that term). Then, two types of purges can occur.

The **first purge type** is for terms that were actually uttered in the input speech, so it is possible that they were useful for a certain amount of time. If the terms are not employed anymore, and they do not fit enough the current state of the LM, we can hope that they are purged from the vocabulary in future word removals.

The **second purge type** is for the garbage or very rare terms mentioned before (terms which somehow fitted certain POOV SSs), and for which there were enough lines of text in the Text Corpora (hopefully the misspelled forms do not have enough text instances). This means that they became terms *evaluable for update* and immediately after *ready for update*, as no reconsideration mechanism is assumed to be employed. In fact, it would be useful to actually employ the reconsideration mechanism so it could hopefully prevent them from being introduced into the vocabulary. In any case, after such terms are introduced into the vocabulary, the dynamic TD strategy will still be able to purge them, as explained next.

This second purge type can occur because we would have learned how these terms are used in the language (thanks to the lines of texts containing them), so it is very likely that they do not fit the background LM, as this LM should have adapted to the input speech while these terms may belong to very diverse and singular topics. Then, such terms would have a low log-likelihood in the LM, making difficult that they appear in the LVCSR transcription (so they become also low-frequency IV terms). Under such attributes, these terms would be the most suitable candidates for a forthcoming word removal, probably sooner than the terms from the first type of purge.

Then, we have clarified all the possible cases in which the different conditions of the external knowledge sources could damage the performance of the dynamic systems, cases for which the proposed strategy has the proper tools that permit to resolve them successfully and in an automatic way.

After this discussion, and with the aim of complementing the description of the SWA process, we propose an additional refinement that might have a positive impact in the overall dynamic vocabulary strategy.

Optional extensions to the words to add

The described dispositions carried out by the SWA process would inform us about which new words should be added to the vocabulary. Nonetheless, this set of words could be expanded with related words that may enrich properly the knowledge behind them. This would be useful when we detect that the word-add-rate model ($WAR_{Model}(t_i)$) of equation 5.6 is increasing recently, so we can enlarge this way $A(t_i)$, propitiating that the real word-add-rate ($WAR_{Real}(t_i)$) of equation 5.5 reaches faster the associated model, which in turn would relieve the vocabulary controller of having to decrease imprudently the acceptance thresholds (specially the **candidate score threshold for term collection** and the **accumulative-score threshold for update interest**), which might harm the quality of the introduced words into the vocabulary, as we discussed before. We leave the specific mechanisms that would allow this kind of reaction for future lines of work.

Apart from this, the most important purpose of making such word extension is to **anticipate** the vocabulary that most probably is going to be used in the subsequent speech, so we are better prepared for performing a more satisfactory decoding, reducing the efforts of the TD strategy.

For example, every word in the initial set of words to add at some t_i might lead to an extension of the set through its morphological variants, in terms of **gender**, **number** or the most common **verbal conjugations**, whenever it proceeds (gender and number for most nouns and adjectives, verbal conjugations for verbs). Thus, for the case of the word *rojo* (masculine singular form of *red* in Spanish), we could also add the words *roja*, *rojos* and *rojas* (employing all combinations of gender and number).

Of course, we would apply some filters to these additional words. First of all, they must not be already contained in the current vocabulary or in the quarantine. Second, in the case we derive automatically the new forms of the words (e.g. by some morphological rules), we should verify their existence as real words using some external knowledge source like Wiktionary (as we did it in the TD strategy, section 4.3.2.1). And third, they should also comply with the requisite of finding them in the Text Corpora for at least the same minimum number of lines that we required to the terms *interesting for update*, so they can be trained properly in LM terms (section 5.1.3).

Apart from the mentioned benefits (specially the anticipation of the vocabulary of future speech), we find two additional benefits when we introduce the morphological variants of the words to add at time t_i . First, we would make more robust the language modeling of the underlying semantics of each word, since this semantics is shared by all the variants of a word.

And second, we would reduce the potential harm of the acoustic confusability that very phonetically-similar words may cause, in case they do appear in the input speech. As a matter of fact, it might occur that when we include a word w to the vocabulary without its variants, the TD strategy becomes unable to detect the occurrence of some variant w' that starts to be uttered, because the LVCSR might show enough confidence decoding w' as w so that the occurrence of w' is hardly ever proposed as a POOV SS by the OOV detector, and therefore obstructing its retrieval and addition to the vocabulary (at least while the LM does not evolve enough or w remains in the vocabulary). Please recall that a similar extension was carried out in the previous chapter when we enlarged the manual list of OOVs for the corresponding experimental framework (see section 4.5.1), although this did not affect the LM of the ASR systems of then, as it will now.

Finally, we can think of a different and more sophisticated word extension which would involve some **analysis** that inform us which other words not considered initially would be potentially valuable for the vocabulary, so we could also add them at time t_i . For example, we could analyze all the words added in the last interval of time, i.e. $(t_i - W)$ to t_i , being W half an hour, for example. Such analysis might involve the use of word embeddings for easily performing some algebraic operations like the centroid of a set of words. Likewise, we could also employ the Resource2Vec representations of the resources associated to the handled words, enriching thus the semantics. Then, we could think of clustering all these words and resources (so as to identify different subtopics) and then to obtain the centroid of such clusters. Afterwards, we could simply search for the KNNs of each centroid, so we will be provided with a list of related terms and resources (in turn, we could also consider the most important terms inside the content of these resources). This list of terms would be finally used to extend the initial set of terms that will be introduced in the vocabulary at time t_i . Also, we should apply similar filters as before to the new words obtained by this analysis (e.g. by requiring a minimum number of text instances in the Text Corpora).

5.1.2.3 Selection of words to delete

The *SWD* process is in charge of selecting words from the vocabulary to be removed. The temporal scheme would work as follows. At each time t_i , it selects a certain number of words from the vocabulary of the system at the previous time step, t_{i-1} , driven by the $D(t_i)$ term computed with equation 5.9. Then, by the end of time t_i , the new vocabulary would not contain those selected words (probably, it would contain other new words selected by the *SWA* process, as we have just seen and indicated by equation 5.10). Since the **number** of words that should be removed at each moment is already indicated by $D(t_i)$, it remains to clarify how we establish which words are the more **suitable** to be removed at each moment.

We can think of a couple of metrics for measuring the eligibility of each word to be removed. First and most obvious, is to employ the **word statistics** obtained from the transcription of the LVCSR system (i.e. counting the number of occurrences of each word in a transcription). When the statistics are accumulated over enough speech portions, we might achieve an insight about which words are not being used at all or very rarely. Of course, it is not enough to accumulate statistics through time, but we should also apply some temporal penalties to the words that did not get new occurrences (in the same manner as we did for the accumulated scores in the term catalog, section 5.1.2.1, equations 5.11 and 5.12). This way, the statistics would reflect, on the one hand, which words were used recently, which would also be likely to appear in subsequent speech and, on the other hand, which words are not being used at all or very rarely, which would become then suitable candidates for removal.

The other metric requires to make use of the **LM log-likelihoods** that are present in statistical LMs like n-grams. If we focus on the log-likelihoods associated to each of the words (i.e. 1-grams), we would know which words are more or less important in the overall LM. Therefore, the words with the worst log-likelihood (i.e. the most negative ones) would be the most suitable candidates for removal.

In order to employ both metrics (and possibly others we may think of) for this selection, we can apply some simple scheme as the next one, which gives more importance to the first metric (the words statistics) since it is based on the evidence of what was actually said recently and what not (as long as we have already integrated statistics for a sufficient amount of time, as we said).

First, we gather a **list** of terms with the worst word statistics (we will usually find a bunch of words whose counts are zero, either because they never happened so far or they appeared so back in time that their counts gradually became zero, due to the temporal penalties). Now, we have to choose which terms from this list are to be removed from the vocabulary.

In case we want to use just one additional metric (e.g. the LM log-likelihoods for 1-grams), we can easily sort the list with the scores provided by this metric, and we would choose for removal the $D(t_i)$ terms with the worst score.

And in case we want to employ more than one additional metric, we would sort the previous list in a different manner. We could set weighting factors to each of these additional metrics that will be employed to sort the list, depending on the importance given to each metric. Each of the metric scores should be normalized so the values they take are comparable among metrics. Then, for each word in the list, we can carry out an addition of the different scores it obtained. This addition will be done in a weighted manner, making use of the previous weighting factors. Finally, with an overall score for each term, we can sort the list and choose for removal the $D(t_i)$ terms with the

worst overall score.

5.1.3 Language Model update

With the LM update process, we intend to produce a new LM for the LVCSR system that includes the new words added to the vocabulary and excludes the removed ones, both lists of words obtained with the vocabulary selection process explained before, and therefore implying also an automatic lexicon update for the LVCSR system.

As a reminder, we indicate again that the LMs considered in this thesis work are based on statistics, specifically on n-grams, rather than in neural networks. This type of LMs would allow an easier manipulation of the considered vocabulary as well as to perform very useful operations for our goals (e.g. the interpolation between two LMs), as we will see.

When we decide to perform a LM update, at some time t_i , we will produce a new LM, denoted as LM_i , which can be considered an evolution of the previous LM of the LVCSR system, LM_{i-1} (*LM background $i-1$* in figure 5.1), since it will be computed with an interpolation operation involving LM_{i-1} and a small LM denoted as LM_{A_i} (explained below), where we give much more importance to LM_{i-1} . This way, we make sure that the LM of the LVCSR system experiments a smooth adaptation to the new language information which is enclosed in LM_{A_i} , but without preventing that, after enough time steps, the resulting LM has changed notably so as to be properly adapted to the speech of each moment. The interpolation scheme will be presented in section 5.1.3.1.

As any other n-gram LM, LM_{A_i} can be built from texts by employing standard n-gram counting algorithms, of some order (typically 3). The texts that are employed for building LM_{A_i} include chunks of text of **four different types**:

- Chunks of text from the Text Corpora that include the new terms to add at time t_i , whose number is equal to $A(t_i)$, and which we can denote from now onwards as $A_{words}(t_i)$. This text material was already retrieved by the *vocabulary manager* when assessing whether a term was LM-trainable (section 5.1.2.2). Also, they constitute a very important material for the LM adaptation, as they contain the new terms that were used most recently in the input speech, so it is likely that they indicate the correct semantic direction that the language is evolving towards.
- New text material, after a Text Corpora update, that might become available for new terms added recently, i.e. terms in $A_{words}(t_{i*})$, with $i^* < i$, in the most recent temporal window. This is specially useful for those added terms for which we barely found enough text at the moment of their addition. When we already found sufficient text for an added term, it might be preferable not to retrieve more, not increasing uncontrollably the size of the LM.
- Chunks of text from the Text Corpora containing useful IV words for the input speech but for which the LM lacks enough modeling. In this case, we would be interested in IV words that are actually being employed recently and that appeared next to some POOV SSs detected that did not retrieve any adequate candidate, as this can be a symptom of improper language modeling related to those terms.
- Transcription texts from the LVCSR system for recent input speech, in case we are confident enough about its decoding. Thus, we can learn directly from the input speech about what

kind of language is being employed. Nonetheless, we should be very careful about which parts of the transcription we select so as to not introduce noise into the systems, coming from transcription errors. Specially, a part containing a POOV SS should be chosen only when the substitute obtained for it was highly scored. Anyway, the texts of this type are to be small in comparison with the large text content available in the external sources consulted.

With this variety of texts (where the first type is likely to represent the great majority), we claim that the language knowledge gathered in LM_{A_i} can easily cover sufficiently the new uses of language and vocabulary of the most recent input speech, towards which we want the system's LM to adapt.

In case we want to give greater importance to the fourth type of texts, as their number is likely to be comparatively small, we could opt for replicating r times the texts of this type, which would prevent us from having to compute separate LMs for the different types of texts and then interpolating them to finally obtain LM_{A_i} .

Also, it is important to note that the actual texts from the Text Corpora (related to the first, second and third types) that is going to be modeled by LM_{A_i} will be limited to small **sequences of words** around the interested terms (mostly those in A_i), so we can get rid of arbitrarily long pieces of text that probably contain much information not strictly ligated to the new terms, as well as not to produce a large LM_{A_i} that will ultimately affect the overall LM_i , increasing excessively its size (we will go back to this issue in section 5.1.3.3). Then, we should choose a context window of c words to be considered at each side of the text appearances of the new terms (e.g. $c=4$). Following the same motivation for controlling the size of LM_{A_i} , we should also limit the number of such sequences per new term that will be finally considered for this LM (e.g. a maximum number of 10K sequences of $2c+1$ words per term in $A_{words}(t_i)$). Alternatively, we could filter the word sequences found to keep only the first e.g. 10K whose vocabulary intersects properly with the current vocabulary of the ASR system, as we suggest in more detail in section 7.2.

Lastly, we should also pay attention to the vocabulary handled by both LM_{i-1} and LM_{A_i} , specially for discarding n-grams containing terms that we recently decided not to be part of the vocabulary anymore. These arrangements will be explained in section 5.1.3.2.

5.1.3.1 Interpolation scheme

To carry out the interpolation between LM_{i-1} and LM_{A_i} we need to set an interpolation weight that preserves our intended behavior over time (i.e. a smooth adaptation of the system's LM towards the language in the input speech). The scheme followed in this work aims to predict the best interpolation weight λ_i given to LM_{A_i} , which we can already presume it will be considerably small.

At each time t_i that we perform a LM update, the amount of text employed to build LM_{A_i} is likely to be different. For example, the number of total sequences (related to the first, second and third type of texts) may vary depending on a number of factors, of which the most important might be the number of the specific terms in $A_{words}(t_i)$. Thus, we designed a simple formula for the computation of $\lambda(t_i)$ that takes into account that number of sequences, S_i , together with the context c employed (since the sequences will be longer or shorter depending on this parameter) and also the total word length T_i of the text taken from the output transcription (related to the

fourth type of texts):

$$\lambda(t_i) = k \cdot \frac{(2c + 1) \cdot S(t_i) + r \cdot T(t_i)}{NW(t_i)} \quad (5.13)$$

where $NW(t_i)$ refers to the estimated number of words composing the LM training texts that were used until time t_i . Thus, $NW(0)$ would be the number of words in the original training texts employed to build the first background model of the LVCSR system. As a simple scheme, $NW(t_i)$ should be increased/decreased proportionally to the current size of the system's LM, since it is probable that the LM does not keep its original size, as we will discuss later. r is the number of times we might have replicated the T_i words from the output transcription (intended to give them more relevance inside LM_{A_i}). Lastly, k is a multiplier coefficient that should be set depending on the scenario of use, as we reason below.

It can be seen that the fraction of the previous equation reflects the proportion of words in the texts used for producing LM_{A_i} (numerator) and the words in the original training texts (denominator). Therefore, this proportion is intended to give us insight about the order of magnitude that $\lambda(t_i)$ should have, while the k multiplier, which should be then ~ 1 , is conceived for optimization purposes that depend on the operation point of the dynamic strategy. We can think of two cases.

In the first case, the ASR system is dealing with speech whose language features are **unknown** (as usual), and therefore we do not know whether the input speech will be topic-related to the past speech (to which we are supposed to have already adapted the LM). In this first case we also include the scenario where we received some metadata informing us that the next input speech will be somehow **related** to the last LM of the ASR system (e.g. we are going to decode a new program of the TV show we were already decoding in the past). Therefore, a smooth evolution of the LM is advisable, either because we prefer to prudently adapt slowly to what is being said, or because we are already close to the domain of the speech, respectively. Then, k should take a value of close to 1 or slightly bigger so as some adaptation can be perceived.

In the second case, we received some metadata informing us that the ASR system will be employed in quite **different** domain than the last one (e.g. we are about to decode a TV program of political debates after decoding a TV program of science), without being able to change the background LM. Now, k could take greater values (e.g. 7), so we escape more rapidly from the specificity of the previous speech. Of course, after some time we might arrive to the first case, and we could decrease accordingly the value of k . This suggests that the k multiplier can also acquire a dynamic behavior, although this is not reflected in equation 5.13.

As we have just seen, the computation of $\lambda(t_i)$ is based on the proportion of sizes among the original training texts and, mostly, the texts with the new terms in $A_{words}(t_i)$, which is directly related to the proportion of the sizes among the respective LMs. So, it is interesting to compare this scheme with the ones we can find in works about explicit topic-based LM adaptation (where explicit LMs per topic are required), as in [Echeverry-Correa et al. \(2015\)](#).

In that work, instead of attending to the size proportion between the background LM and the context-dependent LM, they focus on a distance that measure how different the two LMs are (the greater, the more different), for example considering the average difference in the unigram probabilities of both models. Then, the greater the distance between both models, the greater the contribution of the context-dependent model should be to the final one.

In our case, we cannot compute a context-dependent LM since we are not defining a set of topics in advance, in favor of the pursued ability of adapting to any possible and unexpected topic. Instead, we are able to produce a LM_{A_i} that is likely to characterize **partially** the recent input speech, and this is the reason why we aim to partially modify the background LM with LM_{A_i} , in similar proportion to their sizes (as such proportion gives us an idea of how much we would be able to redefine the background LM with the new knowledge in LM_{A_i}), with the hope that after some iterations a notable adaptation to the speech finally happens.

5.1.3.2 Vocabulary arrangements

We would like to address now how to set the vocabulary configuration when producing LM_{A_i} and the new LM for the system, LM_i , at time t_i (since LM_{i-1} would have been computed before).

As we explained, LM_{A_i} is computed from different types of retrieved sequences of words, many of them containing the new terms in $A_{words}(t_i)$. However, the words that compose these sequences can contain other OOV words, different from the terms in $A_{words}(t_i)$, that might not be useful at all for the input speech. Also, terms we have just decided to remove from the vocabulary might appear as well, which are denoted as $D_{words}(t_i)$ and whose number is equal to $D(t_i)$. Therefore, it would be reasonable if we consider both types of terms as $\langle UNK \rangle$ words for the construction of LM_{A_i} . Thus, the vocabulary inside LM_{A_i} would be the same vocabulary we want for our ASR system to manage after time t_i (i.e. including the terms in $A_{words}(t_i)$ and discarding the terms in $D_{words}(t_i)$), whose size is defined by equation 5.1.2 and which can be denoted as $V_{words}(t_i)$.

With this vocabulary restriction made to LM_{A_i} , we save modeling effort for words we are not currently interested in, and at the same time we are able to model more properly (in LM terms) the special word $\langle UNK \rangle$ which contributes to the OOV detection in the input speech (back in section 4.3.1.1). This occurs because the appearances of words not in $V_{words}(t_i)$ are considered as occurrences of $\langle UNK \rangle$, so the model is able to take into account sequences of words containing $\langle UNK \rangle$ (until some order), apart from the typical discounting and smoothing algorithms employed in this kind of LM that also help in the modeling of $\langle UNK \rangle$.

Regarding LM_i , which is to be computed as the interpolation of LM_{i-1} and LM_{A_i} , we also want it to be limited to vocabulary $V_{words}(t_i)$. As we have just seen, LM_{A_i} is already limited to this vocabulary, so we have to pay attention to LM_{i-1} . This LM does not have modeling for the new terms in $A_{words}(t_i)$, but this is not a problem since this will be contributed by LM_{A_i} . However, there exists in LM_{i-1} some modeling effort for the now excluded terms in $D_{words}(t_i)$, which should be removed before the interpolation. This can be easily achieved in n-gram LMs by removing any n-gram (including unigrams) that contains some of the words in $D_{words}(t_i)$. We could then re-normalize the log-likelihoods in this shorter version of LM_{i-1} for consistency, although the removed material is probably so little that not to perform this re-normalization should not affect the model. Finally we are all set up to perform the interpolation that would lead to LM_i .

5.1.3.3 Incorporation to the LVCSR system

To end the description of the LM update process, we indicate how the new LM produced at time t_i can be incorporated into the LVCSR system (provided that some LM update is produced at t_i). This would depend on the architecture of our LVCSR system.

In case the LVCSR is composed of separate modules for the AM, LM, HMM, etc., the incorporation of the new LM is transparent as it only requires to substitute the old LM by the new one. However, a more efficient decoding is achieved when all these modules that conform a LVCSR system are compiled together into a WFST graph, as it is common to find in popular speech recognition toolkits like Kaldi (Povey et al., 2011). In this case, it is not as evident as before how to modify this graph to update its inner LM, so it turns out that to re-compile from scratch the graph is the most reasonable solution.

This **re-compilation** can be computationally expensive, specially when the LM involved is of a big size. Therefore, we find another argument for controlling the size of the LM, which can be achieved in two ways that should operate together.

The **first way**, already mentioned, is to keep constant the size of the vocabulary, since every new word we want to introduce into it will be accompanied with a new set of n-grams containing the new word. This was already addressed by the use of a $V_{Model}(t)$ in the vocabulary size controller that tends to stabilize into a constant V_0 size (as described by equation 5.4). Furthermore, the smaller the vocabulary, the faster the re-compilation can be, although a too small vocabulary may harm the overall recognition performance in comparison with using bigger vocabularies. Nonetheless, small vocabularies appear handy for demonstrating efficiently the capabilities of the dynamic TD strategy in ASR systems, as we will see in the experimental framework later in this chapter, whose results could be extended to bigger vocabularies, but requiring longer spans of input speech processing so as to perceive similar improvements, as we will see.

The **second way** is to purge some n-grams of order 2 or more, as we reason as follows. Every time we update the LM, new n-grams are introduced to the LM containing the new terms in $A_{words}(t_i)$, thanks to the interpolation with LM_{A_i} . Nonetheless, in this last LM it is likely that we find n-grams of order 2 or more containing only IV terms, since the number of possible combinations among IV terms is arbitrarily large, the greater the higher order of n-gram.

This means that even if we counteract the introduction of n-grams containing the new terms in $A_{words}(t_i)$ with the removal of the n-grams containing the terms in $D_{words}(t_i)$, there will be extra introductions (the newly observed n-grams of IVs) that would propitiate an **increasing-size fashion** in the evolution of the LM. Therefore, the removal in LM_i of an according number of n-grams of order 2 or more would lead to the size control we are seeking for. For example, we could remove those n-grams with the worst log-likelihood, selecting a higher number of n-grams as we increase the order of them, since the higher is this order, the less useful is a specific n-gram for generalizing arbitrary sequences of IV words. However, this second way of size control might harm the LM specially if it is properly tuned to the input speech, so we should apply this method much less often than the first one, as we comment below, or even reserve it for when we receive some metadata informing us that the next speech to decode is of a quite different domain than the past speech.

Besides, it is likely that with just the first way we would be able to control the size growth of the LM in the short and medium term, so we could reserve the second way for longer periods of time, saving computational effort. For example, if the LM update might be produced in a period of time in the order of minutes (where we would be applying the first way of size control), we could apply the second way in a period of time in the order of hours.

In any case, we want these arrangements to happen in an automatic fashion (even without the need of receiving external metadata about the input speech). So, if the LM update is already controlled by the $update(t)$ signal (which considers, for example, whether there are available a minimum

number of words to add to the vocabulary), we could automatically trigger the second way of size control whenever the LM has increased some percentage from the original or a desired size (e.g. a 50%), which should take notably more time than the span between two normal LM updates, as we will observe in the experiments of sections 5.2 and 5.3.

5.1.4 Reconsideration mechanism

This last process is conceived as an optional feature for the proposed dynamic vocabulary strategy. We say it is optional in the sense that the overall strategy is designed to work properly without this mechanism, so the interest of using it resides in the extra improvement in performance that might be gained.

This mechanism is able to perform two different operations: to undo or maintain part of the corrections made to the LVCSR transcription triggered by certain substitute terms; and to filter the terms that were about to be introduced in the vocabulary at some time t_i . We explain these two operations in separate subsections.

5.1.4.1 Re-correction to the transcription

With this operation, we intend to reconsider part of the corrections we made to the LVCSR transcription, undoing a correction if necessary so the corresponding transcription segment goes back to the original content provided by the LVCSR system. For ascertaining so, we make use of ad-hoc LMs that serve as external knowledge sources more capable of this assessment than the TD strategy which made the correction decision in the first place.

Since the construction of this ad-hoc LMs can be costly (as we will see), we will not reconsider all the corrections that the TD strategy makes to the transcription, limiting them to the corrections that were propitiated by a certain type of substitutes. These substitutes must be terms that became *evaluable for update* under the criteria of the *vocabulary manager* presented before, so they are substitutes that appeared so often and/or with large scores that they firstly became terms *interesting for update*, and for which we were able to retrieve a minimum number of lines of texts containing them (this last filter was the reason why they became then terms *evaluable for update*). We could also add here another filter that would be in charge, if we want so, of deciding whether each of such terms actually becomes a term *ready for update* (this is the subject of the next subsection 5.1.4.2).

Then, the reconsideration would work as follows.

First, for each term that become *evaluable for update* at some time t_i , denoted as w_i , we employ the lines of text containing it which were already retrieved by the *vocabulary manager* module (section 5.1.2.2) to construct an ad-hoc LM, denoted as LM_{w_i} . These lines of text will be also limited to word sequences centered in the appearance of w_i , in the same manner we did for the LM update process (section 5.1.3).

Since the number of these sequences is likely to be too small so as to produce a reliable LM, we will make use of exclusive word classes (in the sense that a word can only belong to one class) in order to substitute any word in the sequences belonging to one of these classes by the

corresponding class identifier. This way, the resulting LM (which would manage tokens of words and class identifiers) becomes more compact.

These exclusive word classes can be created manually (e.g. by employing some knowledge source that separate words into categories) or automatically. In the automatic approaches, we find optimization algorithms that iteratively distribute the words into classes, following an objective to minimize based e.g. in the popular perplexity metric.

Also, we can make an automatic clustering making use of word embeddings, so we group together words whose embeddings are close in terms of cosine similitude or euclidean distance, for example. In this last case, we have to set the number of clusters in advance. There are other clustering algorithms, like DB-SCAN (Ester et al., 1996), that can find automatically the proper number of clusters, by studying the density of embeddings along the vectorial space, grouping together regions of that space whose density is considerable (it requires, then, to set the maximum acceptable distance between elements to be considered in the same dense region, and also the minimum number of elements per cluster).

In case w_i already belongs to one of the classes we defined, we can skip the computation of LM_{w_i} by employing the pertinent LM for that class that we could have produced in advance, saving thus computation effort during the dynamic strategy. Nonetheless, these LMs associated to each of the classes should be recomputed periodically with the updated material from the Text Corpora.

Second, we focus on each of the j corrections that the w_i term made to the transcription (within the most recent temporal window) to ascertain separately whether the correction was adequate. We want to measure how comfortable is the LM_{w_i} with the j th corrected transcription sequence (i.e. a sequence containing w_i and the c adjacent words at both sides).

For doing so, we calculate the perplexity of the j th sequence given by LM_{w_i} , and denoted as $PPL_{w_i,j}$. Then, we compute the perplexity, given also by LM_{w_i} , of the same j th sequence but now substituting w_i by the special word $\langle UNK \rangle$, denoted as $PPL_{UNK_{i,j}}$. This way, if we find that the $PPL_{UNK_{i,j}}$ is quite smaller than $PPL_{w_i,j}$, it probably means that the correction was not adequate, since the special word $\langle UNK \rangle$ seems to fit much better in the transcription, and vice versa.

And **third**, we can make the reconsideration decision based on some inequality that compares the magnitude of both perplexity measurements. Also, we can take into account the substitution score (as given by the candidate scoring of the TD strategy, section 4.3.2.3), denoted as $Score_{w_i,j}$, since it can give a notion about how well w_i fitted in the j th transcription segment to correct. All these considerations can be gathered in the next inequality:

$$PPL_{w_i,j} \geq \mu \cdot PPL_{UNK_{i,j}} \cdot \frac{Score_{w_i,j}}{Score_{\max}} \quad (5.14)$$

where μ is a coefficient to be set experimentally, as we will see in section 5.2. $Score_{\max}$ is an estimated maximum score that a substitute can reach, which is present just for the purpose of limiting the variations that $Score_{w_i,j}$ might impose, making it easy to set μ . Whenever this inequality is true, we would **undo** the correction that w_i made to the transcription segment under evaluation. As can be seen, the substitution score $Score_{w_i,j}$ acts as a modulator of this decision, so the higher the score, the more difficult we reject the correction.

5.1.4.2 Additional filter for words to add selection

The second operation of the reconsideration mechanism is to decide whether a term that is *evaluable for update* is finally able to become *ready for update*, at some time t_i . In the positive case (and provided that the $update(t_i)$ signal is to be activated), it will trigger a vocabulary and LM update, so it will be part of the terms to add to the vocabulary at time t_i , which we denoted as $A_{words}(t_i)$. And in the negative case, the term will be added to the quarantine, as the corrections it made to the transcription would have been considered erroneous, and therefore it would not seem useful for the vocabulary of the ASR systems (nevertheless, the term would leave the quarantine after some amount of time has passed, for the same reasons discussed in section 5.1.2.2).

The filter, then, works independently for each of the terms *evaluable for update*. It will make use of the perplexity considerations performed by the previous operation, summarized in the inequality 5.14. Since it is likely that a term *evaluable for update* performed several corrections to the transcription (otherwise it is more difficult that we are considering it for addition to the vocabulary), we can measure adequately the proportion of the corrections that were rejected by the inequality 5.14. Thus, when this proportion is higher than a threshold ν (e.g. 0.5), we can decide that the corresponding term is finally not desirable for its addition to the vocabulary, and vice versa, as reflected in the next inequality:

$$\frac{1}{C_{w_i}} \sum_j^{C_{w_i}} R_{w_i,j} > \nu \quad (5.15)$$

where C_{w_i} is the total number of corrections made by the term w_i (a term *evaluable for update* at time t_i). $R_{w_i,j}$ is the result of inequality 5.14 applied to the j th substitution of the term w_i (1 when rejecting the correction and 0 when accepting it). As can be seen, the term w_i will be filtered away and introduced temporarily in the quarantine of the dynamic TD strategy whenever the proportion of rejected corrections exceeds the threshold ν .

5.2 Experimental framework for Dynamic Vocabulary evaluation

We describe in this section the experimental framework designed in this thesis work for the evaluation of the proposed dynamic vocabulary strategy. We will start by describing the different implementations of the modules of figure 5.1, indicating first the ones involved in the static TD processes, as they were taken from the experimental framework of the previous chapter (with some crucial differences we will point out, especially regarding the vocabulary and LM of the LVCSR system). Then, we detail the implementation arrangements made to the dynamic processes for vocabulary and LM updates, as well as for the reconsideration mechanism. Finally, we present the evaluation scheme of the developed systems, where we will emphasize the use of two test corpora of very different characteristics, which will have a notable impact in the evaluation results to be presented in section 5.3.

There are two main features for distinguishing this experimental framework with the one we found in the previous chapter for the static TD strategy. First, we will employ two test corpora instead of one. And second, the OOVs that will be present in the test speech will be more natural and

realistic, without making use of any manual list of OOVs. Besides, the length of the test speech portions delivered to the ASR systems becomes crucial since they will mark when the vocabulary and LM updates are able to occur. All these features will be detailed as well in this section.

5.2.1 Term Discovery strategy

The modules directly related with the static TD strategy (i.e. LVCSR and phone ASR systems and the TD strategy itself) were implemented in the same manner we did for the experimental framework of the previous chapter (this also includes the use of the same word and resource embeddings, as well as the LOD corpora, that are employed inside the TD strategy), so we refer the reader to section 4.5. Nonetheless, there exists some differences in the precise implementation of these modules, which affects the LVCSR system in a notorious form, and to a lesser extent the phone ASR system and some internal mechanisms inside the TD strategy.

5.2.1.1 LVCSR system modifications

The LVCSR system of this dynamic framework shares the same architecture (DNN-HMM) than the LVCSR of the previous chapter. Therefore, the differences are found in the AM and specially in the vocabulary and LM.

Regarding the AM, we extended its speech training material for making this system more robust, as this experimental framework will present more challenging conditions. Thus, apart from the previous train partition of **TCSTAR** (which added up to 85 hours, out of the total 112 hours, as detailed in section 4.5.1), we are employing another speech corpus which was divided also in a train and test partition. This second corpus is a **compilation of talk shows from the Spanish TV**, which were used in the Search on Speech Challenge of IberSPEECH 2018 (Tejedor and Toledano, 2018), and which was already used partially by us for the evaluation of the WVPS (mentioned in section 4.7.3.3). The programs were recorded between 2015 and 2016, with a total duration of 76 hours. The train partition for this second corpus adds up to 49 hours, leaving the same test partition of 27 hours that we employed for the evaluation of the WVPS. This way, we have two test partitions from different corpora for evaluating our dynamic TD strategy, both of a duration of 27 hours. More details about these test partitions will be given in section 5.2.5.1.

Regarding the linguistic features of this system, we are not interested now in having a vocabulary and LM that are related to some test corpus from the start (as it was the case in the previous chapter). Instead of that, we prefer to design a scenario that is more versatile, so we can evaluate our systems in any kind of test corpora (specifically, we will choose two different ones, detailed in section 5.2.5). Therefore, the vocabulary and LM of this system will be related to a more universal use of the language of operation (which is again Spanish).

For doing so, the **starting vocabulary** of this system is taken from the V most frequent words of a text corpus that includes very diverse contents. This corpus was already employed for the generation of the word embeddings in the previous chapter (section 4.5.4.2), which was composed of texts belonging to the Spanish Wikipedia, literature and political records. The vocabulary of such corpus is bigger than 4M, so we can believe that the most frequent 20K or 50K tokens belong to multiple areas of knowledge but without describing too exhaustively any of them, that is why we talk about an universal set of words for the language.

Apart from this universality, we also want a considerable number of natural OOV occurrences in any of the test corpora we will choose, so as to measure properly the effects of our dynamic strategy, but without the need of forcing the deletion of important words to the test corpora. This can be achieved just by selecting a low number for V , since the topic-specific terms that may appear in a test corpus are likely not included in such universal set of V words. At the same time, we will be able to demonstrate whether a limited size of the vocabulary is not that harmful for the recognition performance, provided that we have the ability to adapt it to the input speech.

Then, the sizes chosen for V are the next ones: **12.5K**, **20K** and **50K**. The larger value, 50K, is chosen as an upper limit of what should be considered a limited vocabulary, although it is already quite large. Then, 20K appears as a more reasonable size for demonstrating the abilities of our dynamic TD strategy. Finally, 12.5K is chosen as the lower limit of such vocabulary size, since lower values may lack sufficiently richness of language so that the recognition process becomes completely unreliable, as well as the detection of POOV SSs and the ability to recover something sensible for them. Also, it is worthwhile to advance now that the two test corpora selected for evaluation, as described in section 5.2.5, have a vocabulary of 15K and 17K, so the value of 12.5K for V can be seen as relatively small and unfavorable. Thus, it will be interesting to study whether the dynamic strategy is able to bring any improvement at all under this circumstance and compare its results among the three scenarios.

As regards the **LM** employed in this system, we are also using an n -gram LM of order 3, except that now the training text is selected to reflect a more universal use of the language, as we did for the selection of the vocabulary. For this purpose, we randomly chose a certain number of lines from the literature texts that partially compose the corpus employed for the generation of word embeddings, mentioned before. This way, we ensure that the training text is not circumscribed to a sole domain, as it would be if we selected lines from e.g. the EUROPARL corpus. Also, the reason for not having chosen texts from Wikipedia is because we preferred texts with not so uniform style so as to fit better with the variety and spontaneity of the spoken language. In any case, a LM based on written language instead of spoken language would be an additional obstacle that the dynamic strategy has to deal with.

The number of lines randomly chosen from the literature texts is 2M, totaling 80M words, which is bigger than the 45M words from the training texts for the LVCSR LM of the previous chapter (see table 4.2). However, since we want to keep in the LM only those n -grams of order 2 and 3 that are minimally frequent (so as to preserve the universality we are seeking for), we required the 2-grams and 3-grams to appear at least 2 and 3 times, respectively. Apart from this, we limited the vocabulary to the three different values of V (12.5K, 20K and 50K), so the special $\langle UNK \rangle$ was modeled accordingly. The different number of n -grams for the three produced LMs is shown in table 5.3.

Table 5.3: Number of n -grams in the LVCSR LM for the three values of V

V	1-grams	2-grams	3-grams
12.5K	12,520	1,725,161	3,281,882
20K	20,299	2,157,953	3,489,810
50K	50,246	2,934,790	3,567,336

We can compare these n -gram numbers to the ones found in the partial LM of the previous chapter (i.e. after the removal of important words to the test corpus), which were shown in the last row of table 4.3. As can be seen, the number of 2-grams remains similar, but the number of 3-grams

gets reduced more than three times, due to the imposed restrictions. This results in LMs that are more compact and therefore more universal, although with a high damage in performance (again, due to the restrictions), as we will see, which is another obstacle for the dynamic strategy. In any case, a reduced size of the LMs would allow the strategy to work more efficiently, making the experimentation of this part computationally more affordable.

5.2.1.2 Phone ASR modifications

This system is mostly the same as in the previous chapter, except for the AM (which, again, was trained additionally with the same material as for the AM of the LVCSR we have just described), plus a slight modification that affects its LM (which describes phonotactic probabilistic rules). This LM modification affects only the configuration of the dynamic TD strategy for which we handle a vocabulary V of just 12.5K, as we reason below.

The LM of this system is based on n -grams of phones, with an order of 6, as shown in the previous chapter (see table 4.4). This allows us to model long sequences of phones, which is very helpful for learning the phonotactic rules of the Spanish language.

Since this LM is to be obtained from a word-to-phone transformation of some training texts, there is a risk that the LM is actually modeling the OOV words that later we would like to find when decoding some test speech, in case that the OOV occurrences represent a considerable proportion of the training text, which is rare. Thus, in the previous chapter (working with the train/test partitions of TCSTAR), this risk was negligible as occurrences of the manual list of 2K OOV words did not represent a considerable proportion of the training texts. In this chapter, where we can advance that the test partition of TCSTAR is one of the two test corpora employed here, as detailed in section 5.2.5, a similar reasoning can be applied for the cases when V is 20K or 50K in size. As both 20K and 50K vocabularies are so universal and of considerable size, the natural OOVs that may appear in some test speech (e.g. test partition of TCSTAR) are not to be expected often in the training texts for this LM, even if they are domain-related (e.g. train partition of TCSTAR), so it is likely that the natural OOVs of the test speech are very specific and not related to the words not covered by V in the training texts. This is why the LM for a V of 20K and 50K was taken the same as in the previous chapter.

However, for the case when V is 12.5K, we may lack a sufficient universal coverage of word so that, when considering the train/test partitions of TCSTAR, we could find that a considerable number of OOV words in the test partition appear as well in the train partition. Then, in order to address this risk, we changed for this V the training text of the phone LM so, instead of taking a training text that is related to test TCSTAR, we employ a text that is topic-indeterminate. We already employed such a text for training the LM of the LVCSR system, so we will use it again for this LM (after a word-to-phone transformation). This way, the OOVs that the phone LM might model, if any, would pertain to indeterminate domains and therefore the mentioned danger disappear. Thus, we trained this phone LM with a similar configuration as for the one in the previous chapter. The different number of phone n -grams can be seen in table 5.4, which are very similar to those in the phone LM of the previous chapter (shown in table 4.4).

Table 5.4: Number of n-grams in the Phone ASR LM for the case of $V=12.5K$

1-grams	2-grams	3-grams	4-grams	5-grams	6-grams
51	1.6K	29K	247K	990K	2.3M

5.2.1.3 Term Discovery strategy modifications

The processes involved in the TD strategy did not experiment notorious modifications, so the implementation configurations detailed in sections 4.5.3 (OOV detector) and 4.5.4 (search for candidates system) still prevails. We present the only two modifications made to this strategy, which are related with the quarantine (introduced in section 5.1.2.2), and the WVPS (described in section 4.7). Apart from them, we also permitted a very slight dynamic variation of the **candidate score threshold for substitution** (as suggested in table 5.1), whose behavior can be described with equation 5.7 (more details about the involved word-add-rate model for this experimental framework will be given later).

Quarantine use in the Term Discovery strategy

One modification that can be noticed from figure 5.1 is the use of the quarantine handled by the *vocabulary manager*, and which affects the filtering of candidates that the TD strategy can propose for the POOV SSs detected. In the previous chapter, we saw that this filtering only allowed candidates that were composed of OOV words (i.e. not in the vocabulary of the LVCSR system, which in the dynamic strategy has to be understood as not in the current vocabulary of the LVCSR, at some time t_i). Now we will extend this limitation to the terms in the quarantine.

As we explained before, a term that is in the quarantine is an OOV term that was proposed by the TD strategy to fill some POOV SS, but after its collection in the term catalog, we decided either it was not trainable in LM terms (i.e. we did not find enough text containing it), either the reconsideration mechanism concluded it did not fit properly in the parts of the transcription where it was supposed to occur. Therefore, we do not want the TD strategy to propose it in the next future, at least until we verify that we have enough text associated to it after a Text Corpora update. However, as no updates to the Text Corpora are to be made in this experimental framework and the test corpora are not excessively long, we will not contemplate the case in which a word in the quarantine is eventually removed from it.

This modification will benefit the efficiency of the dynamic strategy, since it is likely that a term present in the quarantine is indeed a rubbish term (e.g. a misspelled form of a word) or a very short and rare term that can fit part of the phones of some OOV that actually occurs in the input speech, and which, without the use of the quarantine, may be wrongly proposed as the best candidate very often (as it would obtain a very good Levenshtein score that is the most relevant one).

Word Vector Predictor System use in the Term Discovery strategy

At the end of the previous chapter, in section 4.7, we proposed an extension to the TD strategy described in section 4.3. This extension consisted in the use of a WVPS as a complement way of performing the search in LOD corpora (the details can be found at the beginning of section 4.7).

Although we did not use the WVPS in the experimental framework of the previous chapter, as it was not needed for demonstrating the ability of the TD strategy to increase the performance of ASR systems, we intended now to incorporate it to the experimental framework of the dynamic TD strategy. Since the experimental scenarios for evaluating the dynamic TD strategy are more realistic, the WVPS might offer extra improvements in performance that can give us better insight about the abilities of this strategy.

Nonetheless, in our experimentation, we observed that the impact on overall performance when introducing such system into the TD strategy did not reach significant improvements, although for the case of the TCSTAR test corpus a slight positive tendency was measured. This can be understood due to the more favorable conditions of this test corpus, in contrast with the TV test corpus, as we will emphasize in section 5.2.5.1. Therefore, for clarity reasons in the presentation of the results, we will show only the results of the dynamic systems that do make use of WVPS when decoding the TCSTAR test corpora, and only the results of the dynamic systems that do not make use of WVPS when decoding the TV test corpora. In any case, we leave for lines of further work the training of a more robust WVPS for which more noticeable impact on performance is expected to be experienced in either of the test corpora.

5.2.2 Vocabulary manager

We implemented this module following the functionality described in section 5.1.2, with some distinctions we mention below. We would like to remind that the purpose of this module was to decide whether to trigger a vocabulary and LM updates to the system at some time t_i (with a set $A_{words}(t_i)$ of words to add and a set $D_{words}(t_i)$ of words to remove from the vocabulary), depending on the models managed by the controller of the vocabulary size and the word add/deletion rates presented in section 5.1.1. In this experimental framework, the vocabulary size model, $V_{Model}(t)$, will remain constant and equal to one of the three vocabulary sizes managed (either 12.5K, 20K or 50K). Then, only the word add/deletion rate models would experiment a dynamic behavior depending on the different circumstances we will mention later.

This module executes iteratively three processes: *term collection*, *SWA* and *SWD*. Inside the *SWA* process, we can opt to employ the reconsideration mechanism, whose experimental arrangements are left to a separate section (5.2.4).

We discuss now the most important parameters involved in this module, indicating to which of the three processes is associated to:

- **Candidate score threshold for term collection:** this threshold is associated to the *term collection* process, governing which candidates for the POOV SSs are introduced into the term catalog. We should configure the associated parameters (γ_{min}^{Coll} and γ_0^{Coll}), as suggested in table 5.1, to be slightly lower than the **candidate score threshold for substitution** of the static TD strategy, so we are able to consider a wider spectrum of words to be introduced into the vocabulary of the system. The dynamic modulation of this threshold is governed by equation 5.7.
- **Accumulative-score threshold for update interest:** this threshold is associated to the *SWA* process, in charge of triggering the text searches in some Text Corpora for ascertaining whether a term is trainable in LM terms. So, terms that equal or exceed this threshold become *interesting for update*. It is also object of a dynamic variation as the previous

threshold (described as well with equation 5.7), and we should configure its associated parameters (γ_{min}^{Updt} and γ_0^{Updt}) following the indications of table 5.1, making γ_0^{Updt} , for example, to be twice the **candidate score threshold for substitution**.

- **Minimum number of lines in Text Corpora:** this is the minimum number of lines of text that each term *interesting for update* must retrieve in some Text Corpora to become *evaluable for update* (and also *ready for update* if the reconsideration mechanism is not employed). In case of failure, the term is put in the quarantine of terms.

As can be noticed, the *SWD* process is a complete automatically process that does not require any parameter, as we explained in section 5.1.2.3.

As regards the optional extensions to the terms to add at time t_i , denoted as $A_{words}(t_i)$, we only implemented the extensions related to morphological derivations of the selected terms to add, so more sophisticated approaches based on semantic analyses are left for future work.

Now, we describe separately how we constructed the Text Corpora from which this module retrieves the texts associated to the terms in $A_{words}(t_i)$ in order to ascertain whether there are enough training material for them (indeed, these texts will be also used for LM retraining, as we will see in section 5.2.3), and also how the dynamic variations of the previous thresholds are implemented.

5.2.2.1 Text Corpora

The characteristics that we required in section 5.1.2.2 for these corpora (when considered together) were a sufficiently large size and diversity of topics, so as to find occurrences of terms belonging to very different areas of knowledge, and about its updated state, so as to cover newly created terms and modern use of the language (which should be addressed by periodic updates of the corpora).

For the experimental framework of this chapter, we chose a set of texts that fulfill those requirements, although there will not be any ulterior update of them, since the speech test corpora (section 5.2.5) are circumscribed to a temporal window that do not require such evolution of the texts.

These texts are the ones we employed for the training of the word embeddings of the previous chapter, described in section 4.5.4.2, and which included a recent dump of the Wikipedia database, apart from the literature and political texts. Thus, the universality of such collected corpora for the Spanish language results again useful.

5.2.2.2 Dynamic variation of thresholds

As we already mentioned, the three most important thresholds of this strategy (the **candidate score threshold for substitution**, the **candidate score threshold for term collection** and the **accumulative-score threshold for update interest**) will be subject to a dynamic variation described by equation 5.7, depending on the associated parameters we set for each one.

Then, we need to establish the word add/deletion rate models that are required for this dynamism

to occur, and which are described by equation 5.6. Therefore, we have to define which naive word rate model $WR_{Naive}(V)$ is to be used, as well as which circumstances Θ are to be considered. We detail below how we obtained the naive word rate model. Regarding Θ , we will consider only the first circumstance that we explained in section 5.1.1.2, referring to the **rate of POOV SSs**, which is easy to compute and very informative. The set up for this circumstance was made following the indications of the referred section 5.1.1.2.

Naive word rate model

In order to obtain this model $WR_{Naive}(V)$, we studied the vocabulary usage in several training speech transcriptions. At the beginning of a transcription, we assume that the vocabulary size is zero. Then, we count the appearances of new terms not included in the vocabulary, and we add the new terms to it, so the vocabulary is able to grow but not to decrease. Thanks to the timestamps of the transcription, we can observe how the vocabulary size varies over time.

We should find that during the first hours of the associated speech, the vocabulary growth is very fast as the function words are employed by their first time, together with other words. Then, this speed should slow down over time as the common words are likely to be already used in the past, so they do not enlarge the vocabulary anymore.

We can ascertain the previous intuition with figure 5.3. We plotted the evolution of the vocabulary size (until 50K) for the speech transcriptions from both train and test partitions of the TCSTAR corpus (the second partition just for illustrative purposes) and also from the EUROPARL corpus, as a function of time. Since there are not timestamps for the EUROPARL corpus, we assumed a similar word-uttering rate as for the TCSTAR corpus. First, we observe the high overlapping among the three evolutions, which suggests some kind of universality to be exploited by the model $WR_{Naive}(V)$ that we intend to apprehend. And second, the decreasing speed of the growth is clearly seen (this also happens, and more sharply, for vocabulary sizes $>50K$, not showed in the figure because we are not interested in such sizes for this experimental framework), confirming our expectations. Both train and test partitions of the TV corpus follow closely the shapes of the curves in the figure, and were not included there so as not to hinder visibility. It is also worth noting that for less-inflected languages than Spanish (e.g. English), the values of the vertical axis are expected to be quite smaller, although the shapes of the curves should be similar.

Then, the word-addition-rate model can be easily computed from the previous observations. We may employ as many speech transcriptions as possible to get an averaged evolution of the vocabulary size, so the model might generalize better. In our case, we averaged the evolution from the train partitions of TCSTAR and TV and from the whole EUROPARL. Thus, we can calculate the derivative in time of such evolution so as to obtain the naive word-addition-rate model $WR_{Naive}(V)$.

In figure 5.4, we plotted this model, until a V of 34K. Here we can observe how sharp is the decrease of the word addition rate until a vocabulary size of 10K, which reaffirms our reluctance to employ vocabularies of such a small size, being 12.5K a reasonable minimum of operation for our ASR systems. The plot is limited to 34K as we wanted to emphasize the two different slopes before and after the 10K vocabulary size. Please recall that our implemented systems will have a vocabulary of 12.5K, 20K and 50K, of which we are specially interested in the two first sizes. As for the case of 50K, the value that this model obtains then, $WR_{Naive}(50K)$, is around 2 added words per minute.

Figure 5.3: Evolution of the vocabulary size for some ground-truth speech transcriptions

Figure 5.4: Naive word-addition-rate model as a function of the vocabulary size, in added words per minute units

5.2.3 Language Model retraining

This module performs the operation of updating the LM (and therefore also the vocabulary) of the LVCSR system at some time t_i , as the interpolation between LM_{A_i} and LM_{i-1} , producing LM_i , in the manner described in section 5.1.3.

As we indicated in section 5.1.3, the texts that can be used to build LM_{A_i} can be of four types, so as to enrich the language believed to be related to the input speech. Nonetheless, for this experimental framework we restricted these types to only the first one, which refers to sequences of words containing the terms in $A_{words}(t_i)$, so as to simplify the implemented systems (and also for efficiency reasons, since these text sequences were already retrieved before from the Text Corpora when selecting which words from the term catalog were adequate to be introduced into the vocabulary). In any case, when building LM_{A_i} with texts of all the four types, the amount of text of the first type should be larger than the amount of text of the other types, as the text of the first type contains the new terms that are used most recently in the current speech being decoded, and therefore it should be given more relevance.

Then, such implementation requires to set three important parameters, which are the maximum number of sequences allowed per new term in $A_{words}(t_i)$, the parameter c , which controls the length of those sequences, and the k multiplier for the interpolation weight (equation 5.13).

The first two parameters refer to the quantity of text that is going to be employed (if available) for linguistically modeling the terms in $A_{words}(t_i)$, producing, as we said, LM_{A_i} . Ideally, we would like an infinite number of sequences for each of these terms, so as to totally ascertain how they are used (even better, we would like to restrict the sequences to be topic-related to the topics present in the most recent input speech).

However, the size of LM_{A_i} will affect the size of the LM_i which is the updated LM to be employed by the LVCSR system, whose size is very likely to grow over time, as we reasoned in section 5.1.3. In order to control this growth, we should limit the quantity of text employed for LM_{A_i} . An adequate number of sequences per term in $A_{words}(t_i)$, then, will depend on the limitation of the growth rate that we want to impose, specially attending to the efficiency of the dynamic strategy. In our experimental scenario, we found a maximum number of sequences of 10K to be functionally acceptable.

As for c , it is preferable to limit it to a short value, e.g. <5 , so as to only model the immediate context of the interesting words, in case we do not have information about the topics of the texts where they were extracted from; or to not limit it at all, in case we can select the texts to be topic-related to the input speech, learning as well other word constructions of the topic that may be helpful, although this would also have repercussion in the size of LM_{A_i} . Nonetheless, since we did not consider the second option for our experiments, we prudently selected a c of 4.

Lastly, the k multiplier for the interpolation weight will govern how fast we adapt the LM of the LVCSR system to the language associated with the terms in $A_{words}(t_i)$, which suggests it can experiment variations over time. Thus, if we somehow detected an abrupt change of topic, we should increase the value of k to rapidly adapt to the language related to the new topic. If the changes are to be produced in a smoother manner, then small values of k would be desirable.

In our case, we consider that smooth adaptations are more advantageous, provided the characteristics of our experimental scenario. As explained in section 5.2.1.1, the starting LM of the ASR systems participates of an universality of language so that no abrupt change of topic is to be expected at the beginning. Also, we preferred to avoid explicit detection of topic changes in the input speech, so we can evaluate whether a scheme of smooth and completely unsupervised adaptation of LM is a proper configuration for our proposed dynamic vocabulary strategy. For these reasons, we set k to have a value of 3.

5.2.4 Reconsideration mechanism

This last module was implemented to perform the operations described in section 5.1.4. It is capable of reconsidering whether a correction to the transcription made by some term in $A_{words}(t_i)$ was appropriate or not, restoring it to the original form in negative case. Also, if many of the corrections triggered by a term are undone, the mechanism prevents the term from being introduced into the vocabulary.

In order to perform these operations, a LM that manages words and classes has to be produced for each term in $A_{words}(t_i)$, so it can represent in a more compact form the limited knowledge acquired from the texts containing the term, retrieved in the Text Corpora. This requires the definition of a set of word classes. Then, as detailed in section 5.1.4, this LM will be employed to evaluate each of the corrections made by the term, which in turn requires to set the coefficient μ of the inequality 5.14. Finally, we decide whether to prevent a term from its addition to the vocabulary regarding the inequality 5.15, which depends on ν .

5.2.4.1 Creation of word classes

The creation of word classes was made by clustering word embeddings. These embeddings are the same as the ones employed for the WVPS evaluation, and they were described in section 4.7.3.1 (so the handled entities can be composed of one, two or three words). We chose k-means as the clustering algorithm, performing the following arrangements.

First, we need to think of the desired number of word classes. We are interested in grouping together words that are almost interchangeable (e.g. months, German female names, Portuguese surnames, etc.), so we can compact a text as much as possible without introducing too much noise. Then, we should not aim to have a large number of classes. For example, we can easily think of at least 50 classes whose entities are very interchangeable in any sentence. Then, a number between 100 or 300 should be also fine, but larger values are likely to loose this property.

Of course, it remains open the question of how much specificity we want for the classes (e.g. are all the animals in the same class or at which level of zoology we will split them?). Nonetheless, this can be addressed by applying some quality criteria to the clusters, becoming a matter of to which extent the word embeddings were able to learn these specificities from the training texts, as we mention below.

Second, we employ the k-means algorithm to divide the word embedding space in a large number of clusters, being reasonable to employ a number which is one order of magnitude higher than the desired number of classes (e.g. 4K). This parameter should be explored empirically, since depending on the word embeddings, it can affect distinctly the specificity of the clusters.

Third, we need to design a quality metric so we can assess which centroids found by the k-means algorithm are useful and which not. This metric should take into account the distance from a centroid to its closest word entities, since short distances between a word and a centroid suggest that both are tightly related in a semantic sense. The more words that are very close to a centroid, the better represents the centroid a specific semantic area. Thus, the centroid quality metric Q_C that we designed accumulates the distances between a centroid C and its N (e.g. 10) nearest

neighbors:

$$Q_C = \sum_n^N \text{dist}(\bar{C}, \overline{NN_n(\bar{C})}) \quad (5.16)$$

where $\text{dist}(\bar{X}, \bar{Y})$ computes the Euclidean distance between two vectors $\bar{X}$ and $\bar{Y}$. And $\overline{NN_n(\bar{C})}$ refers to the n th nearest neighbor of centroid $\bar{C}$.

Fourth, we sort all the centroids following the Q_C criterion, selecting a number of the best ones, which should coincide with the number of desired classes (i.e. 100-300). We can additionally purge some of these centroids whenever they do not fulfill the next requirement: a centroid must have at least a minimum number of members (e.g. 5); otherwise the resulting classes would not compact enough the language, as we intend. In order to consider a word as a member of a centroid, we can use the knowledge learned in the previous chapter regarding the importance of setting distance barriers: given a vector in the embedded space, the semantic relationship that it may have with other vector is lost after a certain distance of separation among both is reached. Thus, a word will be a member of a centroid only if it is closer than a specific distance barrier (e.g. 1.0 in Euclidean distance), so we assure that they are semantically related (if by this criterion a word is able to be a member of more than one centroid, we would choose the closest centroid to the word).

And **fifth**, the surviving centroids can be used to populate the word classes, computing the nearest neighbors of each centroid until some distance barrier (for example the same employed for purging the centroids). In our case, we obtained 131 classes that gather, in total, 7,090 entities, some of them composed of two or three words. In table 5.5, we give some examples of the classes created.

Table 5.5: Examples of the classes used in the reconsideration mechanism, in Spanish language

Class index	Members
1	<i>marzo, febrero, abril, octubre, junio, noviembre, mayo, septiembre, agosto...</i>
2	<i>kate, helen, sarah, susan, rachel, jenny, rebecca, emily, molly, amanda, lily...</i>
5	<i>mike, alex, bobby, danny, jason, larry, jake, rick, tim, josh, jamie, tom, sam...</i>
13	<i>perplejo, atónito, estupefacto, asombrado, desconcertado, boquiabierto...</i>
31	<i>ensalada, pollo, patatas, mantequilla, sopa, queso, pasteles, verduras, galletas...</i>
50	<i>cogía, agitaba, movía, levantaba, acariciaba, agarraba, tiraba, arrastraba...</i>
120	<i>precisa, apropiada, satisfactoria, equilibrada, oportuna, concreta, coherente...</i>

5.2.4.2 Determination of reconsideration thresholds

We need to establish an operational value for the coefficients μ and ν that are found in the inequalities 5.14 and 5.15, respectively.

As mentioned in section 5.1.4.2, we can already set the coefficient ν to a reasonable value, like for example 0.5. This means that we will require, for each term *evaluable for update* to be introduced into the vocabulary, that at least half of the corrections it made to the transcriptions are kept and not undone.

Table 5.6: Reconsideration cases used for computing FPR and TPR

		Correction was right	
		YES	NO
Correction is kept	YES	<i>TP</i>	<i>FP</i>
	NO	<i>FN</i>	<i>TN</i>

As regards μ , we decided to set it empirically. Due to the difficulty of designing a validation set (since large quantities of speech material are already needed for both training and testing), we applied the dynamic TD strategy to a portion of the test partition of TCSTAR, so as to obtain enough reconsideration instances. Thus, we were able to obtain a ROC curve along different values for μ in the inequality 5.14. In order to plot such curve, we need to define first what we understand for False Positive Rate (FPR) and True Positive Rate (TPR) in this scenario, since they constitute the X and Y axis of the ROC curve, respectively.

For this purpose, we defined in table 5.6 what is a *TP* (true positive), a *FN* (false negative), a *FP* (false positive) and a *TN* (true negative) action performed by the reconsideration mechanism. They are needed for computing the FPR (equation 5.17) and the TPR (equation 5.18), which in our case depends on the coefficient μ . We can know whether a correction made to the LVCSR transcription was true by looking to the ground-truth transcription, verifying if the term that substituted part of the LVCSR transcription is actually in the ground-truth transcription or not. And we will keep or reject such correction regarding the inequality 5.14, for some value of μ .

We would like to point out that in this analysis we only take into consideration the actions performed by the reconsideration mechanism, which means that for the *FP* and *TN* cases we do not care whether the original content of the LVCSR transcription (i.e. before a correction was made) was already correct or not, as all we want now is just to find the best configuration possible for this mechanism (the other issues might have more to do, for example, with improving the OOV detection).

$$FPR(\mu) = \frac{FP(\mu)}{FP(\mu) + TN(\mu)} \quad (5.17)$$

$$TPR(\mu) = \frac{TP(\mu)}{TP(\mu) + FN(\mu)} \quad (5.18)$$

We plotted the ROC curve in figure 5.5, sweeping all the values of μ that alter the FPR or the TPR. Finally, we just have to choose an operation point offered by the ROC curve that we find optimal for our scenario. In such scenario, we would prefer to reduce to the minimum the number of true corrections that we reject with the reconsideration mechanism (i.e. a TPR close to the 100%), since otherwise a lower TPR might lead to the rejection of interesting terms so as not to be introduced into the vocabulary (and thus preventing an optimal LM adaptation from happening). Besides, we can afford to experiment a high FPR that leads to the introduction into the vocabulary of undesirable terms, provided the mentioned characteristic of the dynamic vocabulary strategy, which is able to purge bad terms.

For all these reasons, we selected a μ which is linked to the operation point in the ROC curve for which the TPR is about 98% while the FPR is about 82%, so we barely reject true corrections while reducing to some extent the false corrections that we keep.

Figure 5.5: ROC curve for a sweep of values for the μ coefficient in inequality 5.14

5.2.5 Evaluation scheme

We present now the scheme we designed for the evaluation of the proposed dynamic TD strategy. For such evaluation, we will employ two test corpora that have very different characteristics, so as to study the performance of the dynamic strategy on each of them. They are described in section 5.2.5.1. Then, we indicate in section 5.2.5.2 an evaluation metric that can assess the performance of the dynamic strategy, presenting as well other systems that we will employ to compare performances. Finally, in section 5.2.5.3, we introduce some extra analyses that are oriented to obtain a more precise insight about the behavior over time of the dynamic strategy.

The experimental results will be shown and discussed in a separate section (5.3). In contrast to the previous chapter, we will not be able to carry out a comparison with other works, since, to the best of our knowledge, there are not information available about similar dynamic ASR systems. Nonetheless, we believe a proper understanding of the proposed systems will be reached thanks to those extra analyses of section 5.2.5.3. In any case, we refer the reader to section 2.2 to find works dealing with LM adaptation in ASR systems, where we also mentioned works that take into account, to a limited extent, OOV terms ((Naptali et al., 2012) and (Currey et al., 2016)).

5.2.5.1 Evaluation corpora

As we advanced, we employ two test corpora to be used as input speech to the dynamic TD strategy, both in Spanish language and with spontaneous speech. The interest of testing over two corpora resides on their different features, especially in LM terms, as we will mention.

The first one is the **test partition of TCSTAR**, which was already used in the previous chapter for the experimental framework of the static TD strategy (section 4.5), and also for the evaluation of the WVPS (section 4.7), so a detailed description can be found in section 4.5.1. The duration of this corpus is about 27h, and the domain is clearly political. The vocabulary used in the speech adds up to 15.2K.

It is important to emphasize that the kind of language employed in this corpus shows a high level of formality, where the utterances occur in an ordered fashion, and where no more than one speaker is talking at the same time. Finally, although the topics vary multiple times, they share the political domain. However, since this test corpus is composed of utterances from the European Parliament and the Spanish *Congreso de los Diputados*, there are notable differences among both divisions. First, the topics in the first division are focused on the European sphere, while in the second one, on the Spanish realm. And second, the quality of the audios is much better for the European Parliament, which it is likely to affect the recognition performance at the front-end (as experienced in the previous chapter, commented in section 4.5.5).

The second corpus is the **test partition of a compilation of talk shows from the Spanish TV**. The same test partition was used for the evaluation of the WVPS (section 4.7.3.3), and the train partition of this corpus was used for training additionally the AM of the LVCSR systems of this chapter (section 5.2.1.1). The programs included in the test partition sum 27 hours, as for the previous test corpus. And the vocabulary used now adds up to 17.1K.

The speakers gathered in these programs debate about current issues, predominating the topics related to the Spanish political scene. However, very different topics can appear, which comprehend from politics or economics to culture or arts. Also, the kind of language present here can get considerably emotional, so it is not unusual that two or more speakers are talking at the same time, which will have a very noticeable effect on the recognition performance (which is also affected by the different acoustic conditions in which the AM of the ASR systems was trained, the train partition of TCSTAR).

The different features among the two test corpora can be also noticed by the distinct vocabulary sizes employed in them, 15.2K versus 17.1K, respectively, which reflects the variety of topics, greater for the TV test corpus. Also, the talking speed is greater for the TV test corpus, as the number of total words in here is 285K, while for the TCSTAR test corpus is 243K words.

The two corpora were divided in speech portions so as to serve as iterative inputs to the ASR systems inside the dynamic TD strategy. This division has to attend to the succession of utterances so as not to cut inadequately the speech, thanks either to the timestamps of the ground-truth transcription (only for experimentation scenarios where this transcription is available, like in here) or, for the general use case, the non-speech segments marked by a VAD system (Voice Activity Detector). Also, the average length of each portion should be in accordance with the rate of vocabulary and LM updates we find reasonable to experience, since the dynamic arrangements are to be made after each portion (although not every portion has to trigger such updates, especially because of efficiency reasons). Therefore, the average length of each portion should be in the order of few minutes, so the automatic adaptations can occur in time.

Thus, the first corpus was divided in 4,431 portions, although we will pay special attention to leave enough margin between successive updates. Such small portions were intended for studying the possibility of more frequent updates, but in practice are used in groups of ten. As for the second corpus, we directly divided it in 392 portions, which is already in the preferred order of magnitude.

We would like to remark that the dynamic vocabulary strategy is evaluated separately for each of the corpora, so we can obtain conclusions per type of corpus, as their differences are so notable. For convenience, we indicate in table 5.7 the most important features of both corpora.

Table 5.7: Characteristics of the two evaluation corpora for the dynamic vocabulary strategy

Corpus	Duration	Vocabulary	No. words	Style	Topics
TCSTAR (test partition)	27h	15.2K	243K	Formal	Homogeneous
Spanish TV programs	27h	17.1K	285K	Informal	Heterogeneous

5.2.5.2 Evaluation metric and comparative systems

As for the evaluation of the static TD strategy of the previous chapter, the preferred metric to employ is the WER, computed separately for each of the test corpora, and taking into account all their corresponding utterances, so we can study an **overall WER per corpus**.

Since this overall WER will be computed over a large amount of speech (27 hours per test corpus), we find this metric adequate for assessing the performance of our dynamic TD strategies, which should have had enough time for propitiating an adaptation to the input speech so that is reflected in WER performance, at least for the systems with moderated vocabulary sizes.

As we commented before (section 5.2.1.1), we implemented ASR systems with the dynamic TD strategy for three different sizes of the vocabulary: 12.5K, 20K and 50K, so we will show results for all of them. The dynamic TD strategies that they will be equipped with will count with all the proposed modules (see figure 5.1), including the reconsideration mechanism (with the limitations mentioned throughout this section 5.2).

In order to be able to observe improvements, we implemented other ASR systems to act as **comparative systems**, in a similar manner we did in the previous chapter. In here, we designed four different systems for each of the three sizes of the vocabulary.

The **first comparative system** would be a system identical to the systems with the dynamic TD strategy, emphasizing that they would have the same vocabulary and LM (either for 12.5K, 20K or 50K), but for which the dynamic TD strategy is deactivated, not even a static TD strategy. This baseline system would serve to mark the upper bound in WER terms, which the systems with a dynamic TD strategy should surpass, provided that such strategy is able to offer improvements in performance, as we want to demonstrate. We call this systems as *raw baseline*.

The **second comparative system** would be the same system as before but equipped with a static TD strategy, so as to observe the benefits contributed by a static TD strategy. Thus, the performance of this system should be at least similar to the previous baseline and, hopefully, better. And compared to the systems with a dynamic TD strategy, the performance should be inferior, showing that the dynamic systems offer additional advantages than just employing a static TD strategy. We call this system as *TD system*.

The third comparative system, together with the fourth one, are intended to show performances of systems in conditions very favorable for a specific test corpus, getting to know how far from these performances our dynamic systems are. This means that the third and fourth comparative systems, apart from being diversified by vocabulary size, will be also diversified by test corpus.

For the **third comparative system**, we designed a system which would be identical to the *raw baseline* (so not even a static TD strategy is employed), except for the words that are included in the vocabulary. In the two first comparative systems, and also the dynamic systems, the vocabulary was selected to be the most frequent words from a huge Text Corpora, so as to achieve the desired universality. For this third comparative system, the words will be taken from the vocabulary that actually occurred in each of the test corpora instead (we detail this process below), while the LM is computed from the same universal texts as the other systems, but constraining the word counts to this new vocabulary. This way, we can explore how well we can perform if we had chosen a perfectly tuned vocabulary (although not a perfectly tuned LM). We call this system as ***tuned vocabulary oracle***.

The construction of the vocabularies for this third system was made as follows. We wanted to preserve the same size of 12.5K, 20K and 50K. Therefore, for the second and third cases, we could include all the vocabulary present in a test corpus (either 15.2K or 17.1K, as seen in table 5.7). In these cases, the remaining words for reaching the 20K or 50K were taken from the universal vocabulary from which we originally built the 12.5K, 20K and 50K vocabularies, choosing the most frequent words that were not already included in the vocabulary of a test corpus. And for the case of 12.5K, we took the most frequent words in the vocabulary of a test corpus (observing the ground-truth transcriptions), so in this case the vocabulary is not complete, although this should not have considerable consequences, as we will see in the results section.

And as regards the **fourth comparative system**, we intended to construct the **fair best-performer** system (since the previous system is *unfairly* watching the test vocabularies) which would serve to mark the lower bound in WER terms (although the previous system is already likely to be very hard to surpass by a dynamic system, given the relatively short length of the test corpora). As for the third comparative system, the construction of this one was distinct per test corpus.

For the TCSTAR test corpus, we already had a great system whose LM was trained in the political domain (train partition of TCSTAR plus EUROPARL), and which was used in the experiments of the previous chapter (named *full-vocabulary system*). This system had a vocabulary of 135K, and a sufficiently large and tuned LM to achieve a great WER performance, as we saw then.

And for the TV test corpus, since we could not design a system with a LM so well tuned as before (due to the lack of adequate additional training sources, as the transcriptions from the TV train partition refers to just 49 hours of speech), we decided to divide this test corpus in three parts, so we could train a LM with the other two test parts (and the mentioned train partition) for decoding the remaining test part, and so on. For further enriching these three LMs, we interpolated them with the LM obtained from the EUROPARL corpus (whose domain, political, is a bit more related than the universal texts we used for the other systems). The vocabulary for each of these LMs is composed of the words contained in the other two thirds of the TV test corpus (less than 10K), plus the most frequent words from the EUROPARL corpus so as to reach the 40K words.

We call this fourth comparative system as ***best oracle***.

5.2.5.3 Dynamic behavior studies

To end this section, we describe some interesting studies that we will carry out for analyzing the behavior of the dynamic vocabulary strategy along the decoding time.

First, we will study the **evolution of the progressive WER** as the speech portions are decoded, separately per test corpus. This progressive WER will compute the WER only for the input speech that was already processed (so it will be more trustful as more input speech is processed). Thus, it will become the overall WER mentioned before when all the speech portions of a corpus are decoded. The interest of studying this evolution can be noticed as follows.

If we compare the evolution of the progressive WER among the different systems (including the comparative systems), we will be able to observe how fast or slow the performances get separated, and at which moments of the speech the separation accelerates or decelerates. For example, comparing a dynamic system with its corresponding *raw baseline*, we can observe whether the automatic adaptation occurs at the beginning of a test corpus or whether it occurs more steadily along the whole corpus, since it results evident that an improvement in WER would be due to a successful adaptation to the input speech (at least until some extent).

Second, we will study the **evolution of the word addition**, in order to observe which of the added or removed words did really appear in a test corpus. Thus, we can have a notion about the proportion of useful terms that we were able to introduce into the vocabulary, and at what rate through the decoding of a test corpus, which would be influenced by the dynamic models we are handling.

And third, we will study the **evolution of the LM**, so as to get information about how the LM is being modified over time. We are interested in modifications that affect the log-likelihood of the 1-grams present in the LM, since they can acknowledge about an adaptation to the input speech. We will also pay attention to the growth in size of the LM, since it might affect the efficiency of the strategy, although for our experimental scenario, and with the proper configuration, it should not represent a danger, given the medium-sized duration of our test corpora, 27 hours each (in longer corpora, the successive adaptations could eventually cause a more dangerous growth of the LM).

5.3 Evaluation results

We present the evaluation results of the implemented ASR systems equipped with the proposed dynamic TD strategy.

In section 5.3.1, we present and discuss the overall WER results per test corpus obtained with the dynamic systems, together with the results obtained by the comparative systems indicated in section 5.2.5.2. This evaluation constitutes the principal assessment of the improvements that can be offered by the dynamic TD strategy.

Then, from sections 5.3.2 to 5.3.4, we carry out three studies related with the behavior over time of the dynamic TD strategy, already presented in section 5.2.5.3, and which are focused on the evolutions of the progressive WER, the word addition and the LM of the systems, respectively.

5.3.1 Overall WER results

The overall WER results are indicated in tables 5.8, 5.9 and 5.10 for the systems with a vocabulary of 20K, 50K and 12.5K, respectively.

As a whole observation, we can confirm that the proposed dynamic TD strategy is capable of offering, at least, a positive tendency for improving the performance of the ASR systems designed in this experimental scenario and, in most of the cases, a significant improvement is achieved, as we will indicate in detail below.

We can also observe that the *Best* and *Tuned* oracles, in both test corpora, are not beaten by any other system whose LM and vocabulary are started on a more universal domain of the language. In the case of TCSTAR, we can reach very low WERs with these two best oracles (specially for the *Best* oracle, indicating that the language of TCSTAR is pretty predictable provided the right LM), unlike what happens for TV (although we had less quantity of proper training material to build its best LM). These benchmarks already reflect the distinct levels of difficulty among both corpora, as we advanced before (see table 5.7). Nonetheless, the *Tuned* oracle is usually very close to the *Best* oracle (specially for the TV test corpus), which encourages the idea of finding a small but tuned vocabulary for achieving a proper performance with limited resources, as the dynamic TD systems attempts to do (in this case, also tuning at the same time the LM).

Regarding the results obtained by the *TD* systems, it is interesting to point out that they retrieved significant improvements in some cases of this experimental configuration, different from the previous chapter (remember that in the previous chapter, the LM and the vocabulary were tuned to the domain of the test corpus). Thus, although for the case of 12.5K the benefits are significant, they get attenuated for the case of 20K (where only for TCSTAR the improvement remains significant) and finally neutralized for the case of 50K. This can be explained by the different rates of OOV appearances for the three cases, so the effects from the OOV resolution of the TD strategy may become more or less noticeable accordingly. Also, for the case of the TV test corpus, the sole TD strategy has more difficulties for dealing with OOVs in the more hostile language conditions of this corpus. In any case, when we allow the systems to evolve through time, the benefits offered by the static TD strategy get emphasized by the propitiated adaptation, reaching more notable improvements (in most of the cases, with a significant separation from these *TD* systems). This a strong point for the dynamic TD strategy.

In section 5.3.2 and successive, we will gain more insight about the adapting capacities of the dynamic TD systems over time, which will help us to understand better these results. Now, we discuss in detail the results obtained by the dynamic systems separately for each of the vocabulary sizes, focusing on their improvements over the *Raw* baseline and the *TD* system.

Systems with 20K vocabulary, table 5.8. This size of vocabulary is reasonable for the intended purposes of our dynamic TD strategy in this experimental framework (provided the smaller though similar vocabulary sizes of the test corpora), and so the strategy should offer significant improvements over the *Raw* baseline and *TD* system. Indeed, the ASR systems equipped with this strategy experimented notable improvements over both systems. In the case of the TCSTAR corpus, the improvements reached a relative 5.9% and 4.1% significant WER improvements over *Raw* and *TD*, respectively, after the 27 hours of the corpus. And in the case of the TV corpus, we obtained a significant difference up to a relative 1.9% and 1.3% WER improvements over *Raw* and *TD*, respectively, after the 27 hours of the corpus.

Systems with 50K vocabulary, table 5.9. This size of vocabulary was chosen as an upper bound, so it is likely too large for the dynamic TD strategy to offer improvements in a short amount of time (as most of the frequent vocabulary present in the speech is likely to be already handled by the systems), so the 27 hours of each test corpus might fall short. For TCSTAR, we did achieve a significant difference, a relative 2.2% and 2.0% WER improvements over *Raw* and *TD*, respectively, after the 27 hours of the corpus. In contrast, for TV, the improvements did not

get significant. Nonetheless, a positive tendency is observed here, as the dynamic systems keep lowering through time the WER obtained by the *Raw* baseline (this will be explicitly shown in section 5.3.2). Thus, we might expect significant improvements after longer amounts of time.

Table 5.8: Overall WER results for the systems with a vocabulary of 20K

System	WER test TCSTAR	WER test TV
<i>Raw baseline</i> 20K	$19.51 \pm 0.16\%$	$34.43 \pm 0.17\%$
<i>TD system</i> 20K	$19.14 \pm 0.16\%$	$34.23 \pm 0.17\%$
DYN 20K	$18.35 \pm 0.15\%$	$33.79 \pm 0.17\%$
<i>Tuned vocabulary oracle</i> 20K	$14.34 \pm 0.14\%$	$29.94 \pm 0.17\%$
<i>Best oracle</i> {135K, 40K}	$11.48 \pm 0.13\%$	$29.08 \pm 0.17\%$

Table 5.9: Overall WER results for the systems with a vocabulary of 50K

System	WER test TCSTAR	WER test TV
<i>Raw baseline</i> 50K	$16.19 \pm 0.15\%$	$31.49 \pm 0.17\%$
<i>TD system</i> 50K	$16.16 \pm 0.15\%$	$31.51 \pm 0.17\%$
DYN 50K	$15.84 \pm 0.15\%$	$31.27 \pm 0.17\%$
<i>Tuned vocabulary oracle</i> 50K	$14.45 \pm 0.14\%$	$30.03 \pm 0.17\%$
<i>Best oracle</i> {135K, 40K}	$11.48 \pm 0.13\%$	$29.08 \pm 0.17\%$

Table 5.10: Overall WER results for the systems with a vocabulary of 12.5K

System	WER test TCSTAR	WER test TV
<i>Raw baseline</i> 12.5K	$22.44 \pm 0.17\%$	$36.70 \pm 0.18\%$
<i>TD system</i> 12.5K	$21.86 \pm 0.16\%$	$36.30 \pm 0.18\%$
DYN 12.5K	$20.59 \pm 0.16\%$	$36.14 \pm 0.18\%$
<i>Tuned vocabulary oracle</i> 12.5K	$15.41 \pm 0.14\%$	$31.37 \pm 0.17\%$
<i>Best oracle</i> {135K, 40K}	$11.48 \pm 0.13\%$	$29.08 \pm 0.17\%$

Systems with 12.5K vocabulary, table 5.10. This size of vocabulary was chosen as a lower bound, so there is a risk that such a small vocabulary leads to a level of confusion in the ASR systems so that not even the static TD strategy works (e.g. the TD strategy might not be able to detect properly the start and end times of the OOV occurrences, as the surrounding words might be detected as part of a POOV SS), which may prevent the addition of useful terms into the vocabulary and consequently no adequate adaptation to the input speech could be experienced. As a matter of fact, the effects that the dynamic TD strategy propitiated empirically were shown to be quite different depending on the test corpus, as we indicate as follows.

For the TCSTAR corpus, the dynamic system was able to reach significant improvements over both *Raw* and *TD*, a relative 8.2% and 5.8% WER improvements, respectively, after the 27 hours of the corpus. The benefits were observed more notably for the EPPS division of TCSTAR (the first 16.2

hours), as we will see more clearly in section 5.3.2. Thus, a relative 10.8% WER improvement over the *Raw* baseline is obtained after those 16.2 hours (the *Raw* baseline gets a WER of $20.79 \pm 0.21\%$ while the dynamic system $18.55 \pm 0.21\%$). This can be understood due to the cleaner acoustic conditions and the kind of language of this part of the TCSTAR test corpus, which facilitates the OOV detection, apart from the not so big difference among the vocabulary sizes involved (15.2K for the TCSTAR corpus and 12.5K for the dynamic systems).

And for the TV test corpus, the dynamic system is still able to reach significant improvements over the *Raw* baseline, although less notable, a relative 1.5% WER improvement after the 27 hours of the corpus. Nonetheless, no significant improvement over the *TD* system is achieved now. We associate these results to the characteristics of this second corpus and to the smallness of the system’s vocabulary, especially now that the difference among vocabulary sizes (17.1K for this TV corpus and 12.5K for the dynamic systems) is quite considerable.

5.3.1.1 Speech redecoding with the adapted systems

In the next sections, we carry out different studies of the temporal behavior and evolution of the dynamic systems, so we can understand in more detail all the previously discussed results. Nonetheless, we would like to show now some additional overall WER results that can easily give insight about the adequateness of the adaptation that occurred inside the dynamic systems.

These results were obtained with ASR systems designed in the same manner as before, but for which we chose the vocabulary and LM of the final state of a dynamic system (after the 27 hours of a test corpus), so both vocabulary and LM are supposed to be already adapted to one of the test corpora, at least to some extent. Then, we performed a redecoding of the speech of both corpora, in a raw manner (i.e. without making use of the static TD strategy nor the dynamic TD strategy).

If the adaptation occurred in a dynamic system was successful, a raw redecoding with its final state should show, on the one hand, similar performance in overall WER when the redecoded corpus is the same as for which the adaptation was produced (assuming that in the 27 hours of a test corpus the different topics that were discussed share a considerable amount of their associated vocabulary). And on the other hand, the performance in overall WER should be harmed when the redecoded corpus is not the same as for which the adaptation was produced, since now the vocabularies among the two corpora are not shared notably. Also, take into account that the TV test corpus has a bigger vocabulary than the TCSTAR test corpus, which indicates more topic-variety and difficulty for TV than for TCSTAR, as we will notice below. Thus, both vocabularies share 7.9K words, which for TCSTAR represents the 52% of all its vocabulary, while for TV just 46%.

We performed the mentioned raw redecodings for the best dynamic systems with 20K, as it appeared to be the vocabulary size that allowed a more noticeable adaptation. The results of the redecodings are shown in table 5.11.

As can be seen, the expected behavior of a successful adaptation is held. Thus, we can observe how the raw redecoding with a system adapted to TCSTAR performed almost equally for TCSTAR as the corresponding dynamic system did ($18.36 \pm 0.15\%$ vs $18.35 \pm 0.15\%$, respectively), while for TV the performance is far away from the dynamic system adapted to TV ($35.08 \pm 0.18\%$ vs $33.79 \pm 0.15\%$, respectively). Indeed, this last redecoding performance is significantly worse than the *Raw* baseline for TV ($34.43 \pm 0.17\%$), since the vocabulary of TV is larger and more topic-heterogeneous, as we noted before.

Table 5.11: Raw redecoding WER results comparison for the systems with a vocabulary of 20K

System	WER test TCSTAR	WER test TV
<i>Raw baseline</i> 20K	$19.51 \pm 0.16\%$	$34.43 \pm 0.17\%$
DYN 20K	$18.35 \pm 0.15\%$	$33.79 \pm 0.17\%$
Raw redecoding w/ DYN 20K after test TCSTAR	$18.36 \pm 0.15\%$	$35.08 \pm 0.18\%$
Raw redecoding w/ DYN 20K after test TV	$19.76 \pm 0.16\%$	$34.26 \pm 0.17\%$

Regarding the raw redecoding with a system adapted to TV, the performance for TV is also similar to the performance offered by the corresponding dynamic system ($34.26 \pm 0.17\%$ vs $33.79 \pm 0.17\%$, respectively), although slightly worse. Indeed, a significant deterioration was measured, which may account for the diversity of topics present in this corpus, where the final state of the dynamic system is likely to be adapted to the last topics present in the speech, which may not result as useful for decoding the rest of the topics that appeared before. Regarding the performance for TCSTAR, it is, as well as before, far away from the dynamic system adapted to TCSTAR ($19.76 \pm 0.16\%$ vs $18.35 \pm 0.15\%$, respectively). However, the differences now with the *Raw* baseline for TCSTAR ($19.51 \pm 0.16\%$) are not significant. This can be due to the smaller size and more topic-homogeneous features of the vocabulary of TCSTAR, which may have been learned partially when learning the different topics of TV.

5.3.2 Progressive WER evolution study

In this first study, we analyze the evolution of the progressive WER as the ASR systems decode the input speech, separately for each test corpus. We will start with the comparative systems to get a first overview. Then, we will focus on the dynamic systems with vocabularies of 20K, 50K and 12.5K.

Comparative systems. We plotted the progressive WER evolution for the comparative systems (except for the *TD* system, for visibility reasons) in figure 5.6, for the TCSTAR (up) and TV (down) test corpora. We omitted the first few hours of the evolution, since the transient nature of that period is not very informative for a progressive WER (this will be done too for the evolution of the other systems).

In the case of the TCSTAR test corpus, upper part of figure 5.6, we drew a vertical line at the 16.4 hour indicating the change from the European Parliament part to the Spanish *Congreso de los Diputados* part, of greater difficulty, as the growth in progressive WER shows for all the systems. For this figure, we would like to make the following remarks.

First, we can clearly observe the separation between three groups of comparative systems: the *Best* oracle, the *Tuned* oracles (specially for 20K and 50K) and the *Raw* baselines (especially for 20K and 12.5K). It is normal that the *Tuned* oracles for 20K and 50K behave equally, as the vocabulary of this corpus, 15.2K, is inferior to those values.

Second, the *Tuned* oracle for 12.5K is still performing better than the *Raw* baseline for 50K, which, again, reinforces the importance of a fine-tuned vocabulary, even if the LM is built with texts of more universal domains.

And third, the smooth evolution of the progressive WER is another evidence of the homogeneity of this corpus, which affects both to acoustic and language conditions.

As regards the TV test corpus, lower part of figure 5.6, we can see important differences with the previous test corpus. First, the *Best* oracle is not as separated from the best *Tuned* oracles (20K and 50K) as before, reflecting the difficulty of building an optimal ASR system for this corpus (partially due to the lack of proper training material for building the LM). Second, the *Tuned* oracle for 12.5K is not performing significantly better than the *Raw* baseline for 50K, as they are found very close one of each other. And third, the evolution of the progressive WER is not smooth but it presents notable oscillations which count for the heterogeneity of this corpus in both acoustic and language terms.

Figure 5.6: Progressive WER for the comparative systems without TD strategy. Up: TCSTAR test corpus. Down: TV test corpus

Systems with 20K vocabulary. Their progressive WER evolutions are plotted in figure 5.7, for the TCSTAR (up) and TV (down) test corpora.

Figure 5.7: Progressive WER for the systems with $V=20K$. Up: TCSTAR test corpus. Down: TV test corpus

In the TCSTAR test corpus, we can see how the dynamic system gets away from the *Raw* baseline, which allowed it to achieve significant improvements inside the 27 hours of duration of this corpus. In fact, this dynamic system achieves significant improvements over the *Raw* baseline just after 3.2 hours (recomputing the confidence intervals counting only the words already uttered; this means that the less hours processed, the more difficult to be significant).

Also, we can see that the WER separation at the end of the European Parliament part (i.e. at 16.4 hours) is maximum, due to the more favorable conditions of this part, so it is normal that the separation gets smaller, though significant, for the Spanish *Congreso de los Diputados* part, as there is more noise in the ASR systems.

For the TV test corpus, the separation between the dynamic system and the *Raw* baseline occurs similarly as for the other corpus, although now is slightly slower. This dynamic system is able to achieve significant improvements after 12 hours of speech, accounting for the more challenging conditions of this corpus.

For illustrative purposes, we show in figure 5.8 a variant of this WER study for the 20K systems, computing now the WER with a sliding window, instead of the progressive WER of before. We chose a sliding window that averages the WER obtained for the previous 4 hours. This means that for the case of TCSTAR, from 16.4 to 20.4 hours (bounds not included), the window would consider both European Parliament and *Congreso de los Diputados* parts (since the first part ends precisely at 16.4 hours).

As can be seen, the WER curves experiment now more notable changes over time, accounting for the different levels of difficulty that each segment of a test corpus offers (due to changes in the acoustic conditions or the topics present in the input speech, for example). As expected, we find more drastic WER changes for the TV test corpus. As a matter of fact, consider the different ranges of the vertical axis between the plots in the previous figure 5.7 and figure 5.8.

In any case, the benefits offered by the dynamic TD strategy can still be observed in this study variant, specially for the case of TCSTAR, where is clearer and more consistent the positive tendency of enlarging the separation between the dynamic system and the *Raw* baseline, occurring for the European Parliament part and also for the *Congreso de los Diputados* part. For this second part, the great separation between the dynamic system and the *Raw* baseline achieved during the first part is partially lost, mainly due to the considerable change of linguistic conditions, but nevertheless the separation is gradually enlarged again, as we could expect.

Because of the mentioned noisy condition of the WER curves, and the difficulty of choosing the most illustrative window width, we preferred to study the progressive WER over time, and this is the reason why we are showing the progressive WER instead of the windowed WER for the remaining systems.

Figure 5.8: Windowed WER (last 4 hours) for the systems with $V=20K$. Up: TCSTAR test corpus. Down: TV test corpus

Systems with 50K vocabulary. Figure 5.9 shows their progressive WER evolutions for the TCSTAR (up) and TV (down) test corpora.

Figure 5.9: Progressive WER for the systems with $V=50K$. Up: TCSTAR test corpus. Down: TV test corpus

In the case of the TCSTAR test corpus, we can observe in the dynamic system a tendency of improvement over the *Raw* baseline. Indeed, the differences start to get significant after 22.1 hours, which is much more time than the 3.2 hours that needed the system with 20K vocabulary when processing the same corpus. This seems reasonable as now the rate of OOV appearances

For the TV test corpus, we can also observe the same positive tendency as for the other corpus, although slower (as occurred for the case of 20K vocabulary). Thus, more hours of input speech would be needed for the dynamic system to achieve significant improvements over

Systems with 12.5K vocabulary. We plotted in figure 5.10 their progressive WER evolutions, for the TCSTAR (up) and TV (down) test corpora.

In the TCSTAR test corpus, the dynamic system performed notably better than the *Raw* baseline, specially for the European Parliament part, as we commented before. The benefits offered by the dynamic TD strategy can be noticed even in the first hour of speech of this corpus, reaching significant improvements over the *Raw* baseline only after 0.7 hours, which is a very small amount of time. This last result evidences the ability of the proposed dynamic TD strategy of performing very fast adaptations to the input speech when the conditions are favorable. Nevertheless, it has to be taken into account that a vocabulary of just 12.5K is likely to be little functional in most realistic ASR scenarios.

And for the TV test corpus, we can observe benefits despite the fact that we are employing a small vocabulary in comparison with the vocabulary of this corpus (12.5K vs 17.1K). This difficulty makes the dynamic system to experiment a less clear positive tendency of improvement. Thus, the differences start to get significant after 6.3 hours, but they loose and win this condition intermittently until 16.2 hours, a moment after which the significant condition is not lost anymore. This may indicate that the propitiated adaptation with such a small vocabulary is prevented from being complete, so the benefits perceived are far from the potential benefits that we could experiment with greater vocabularies, as it occurred for the case of 20K.

After these separated reviews, we finish this first study by collecting in table 5.12 the time to significant improvements (TSI) that the different dynamic systems needed to outperform their corresponding *Raw* baseline. The TSI indicated in the table for the case of 12.5K and the TV corpus refers to the moment in which the improvements start to get significant, 6.3 hours, despite their intermittent condition until 16.2 hours, as commented before.

Table 5.12: Time to significant improvements (TSI) for the dynamic systems, per test corpus

System	TSI test TCSTAR	TSI test TV
DYN 20K	3.2 hours	12.0 hours
DYN 50K	22.1 hours	>27 hours
DYN 12.5K	0.7 hours	6.3 hours

Figure 5.10: Progressive WER for the systems with $V=12.5K$. Up: TCSTAR test corpus. Down: TV test corpus

5.3.3 Word addition evolution study

In this second study, we analyze how many words we add to the vocabulary over the time for each test corpus. We distinguish three types of added words (all of them OOV terms for the vocabulary of the ASR system), considered separately for each of the test corpora. As we will see below, the two first types are terms *ready for update*, which are originally proposed by the TD strategy. And the third type are extensions to the words of the previous types, built from an *ad hoc* mechanism, unrelated with the TD strategy.

More specifically, the first type are words *ready for update* that were actually uttered in the speech of a test corpus, anywhere in the 27 hours of speech of each corpus (information obtained for this study by looking at the ground-truth transcription). Therefore, they are words we are interested in adding to the vocabulary, specially if they are used frequently, although we did not make such frequency distinctions. We call this type **natural OOVs**. Keep in mind that such type of word can result from correct completions of some POOV SSs (for which the natural OOV that was uttered there was actually retrieved), or from mistaken completions of some POOV SSs (for which it was retrieved a natural OOV not present there but which is actually employed in some other part of the input speech), although this last case rarely happens.

The second type are the rest of the words *ready for update*, which means they are words that were never uttered during the speech of a test corpus (again, information obtained by looking at the ground-truth transcription). In principle, we are interested in reducing this type of added words to the minimum, since they are words that were born from mistaken and unfavorable completions of POOV SSs. However, there is a small chance that an OOV of this type results useful for decoding ulterior speech, as when the term is related to the same topic of the speech, so the corresponding word sequences (containing such term) that will be modeled in the LM might help to enrich the LM towards the proper direction. We call this type **unexpected-OOVs**.

And the third type are morphological variants of the natural/unexpected-OOVs that we are going to add at some time t_i , so we also introduce into the vocabulary these variants at t_i . The interest of adding these variants was already discussed at the end of section 5.1.2.2, where different kinds of variants were suggested. For this experimentation, we limited the variants to the morphological level, in terms of gender and number, thanks to the use of simple morphological rules. We call this type **morphological variants**. It is possible that a variant of a natural/unexpected-OOV is indeed a natural OOV. In this lucky case, such added word is considered for this analysis to be a natural OOV.

Now, we show the evolution of the addition of words of the first type, natural OOV, since these are the words that mostly reflect the success of the dynamic TD strategy. Later, we will also pay attention to the total additions of the words of the others types, so as to verify that no dangerous behavior is occurring (for example, if the number of unexpected-OOVs added is much greater than the number of natural OOVs added).

The mentioned evolution related to natural OOVs is plotted in figure 5.11, for the TCSTAR (up) and TV (down) test corpora. In both cases, we show the data of all the three dynamic systems (with vocabulary sizes of 20K, 50K and 12.5K), so we can easily compare them. There are two remarks that should be considered in order to understand the figure.

Figure 5.11: Presence of natural OOVs in vocabulary (without starting offsets) for all dynamic systems. Up: TCSTAR test corpus. Down: TV test corpus

First, we are showing the **presence of natural OOVs** in the systems as a function of time, which means that the number of natural OOV (vertical axis) will **grow** as we add natural OOVs to the vocabulary. Likewise, whenever we remove natural OOVs from the vocabulary (in principle, this should not happen often), the number of natural OOVs will **decrease**. Of course, both effects are considered together, so if at some time the same number of natural OOVs are added and removed, the function will remain the same value (keep in mind that this behavior can also be produced, and more likely, because no natural OOV is added nor removed).

And second, for the sake of easier comparison, all functions were started at zero, which means that we have subtracted the initial number of natural OOVs that were already included in the original vocabulary of 20K, 50K and 12.5K (i.e. the number of shared words between a original vocabulary of some size and the ground-truth vocabulary of a test corpus). These starting **offsets** are indicated in table 5.13, together with the **representativeness** they imply. This representativeness is computed as the proportion of the occurrences of those shared words (i.e. those that count for the previous offset), out of the total occurrences of all words uttered in a test corpus, given its ground-truth transcription.

We can see in table 5.13 that the different representativeness are usually very high, indicating that the natural OOVs of each test corpora constitute a difficult and realistic set of OOVs, since in real scenarios a natural OOV is not a frequent term. Thus, it is probable that the terms in the test corpora that are not shared with the starting vocabularies belong to the specific topics discussed in the testing speech, so the term discoveries to make have to be very accurate.

Table 5.13: Starting offsets and representativeness of natural OOVs present in the original vocabulary of 20K, 50K and 12.5K, per test corpus

System	test TCSTAR		test TV	
	Offset	Representativeness	Offset	Representativeness
20K	8.9K	95.5%	9.8K	94.7%
50K	12.1K	98.0%	13.3K	97.7%
12.5K	7K	93.4%	7.7K	92.7%
Ground Truth	15.2K	100% (243K)	17.1K	100% (285K)

In the figure, we can observe that the number of natural OOVs added is always larger in TCSTAR than in TV, reflecting once again the harder predictability and unfavorable conditions of the TV test corpus. We will give more details later about the specific number of overall word additions/deletions per system.

This difference of difficulty among test corpora is also reflected by analyzing the **slopes** of this evolution for the case of 20K vocabulary. A constant slope may indicate that the ASR systems are continuously surprised with new terms, which indicates a heterogeneous stream of topics in the input speech, as occurs in the TV corpus, where the slopes of the pertinent evolution, despite some small ripple, remain more constant. However, in the TCSTAR corpus, more homogeneous, the 20K dynamic system experiments a more varying behavior of the slopes, being greater at the beginning (first five hours) than in the rest of the corpus. This indicates that the system was able to acquire a great quantity of natural OOVs in the first five hours, which were already useful for decoding the remaining speech, as the slopes get smoother since then.

Now, we comment some details of this evolution shown in the previous figure 5.11, separately by vocabulary size.

Systems with 20K vocabulary. The evolution of the presence of natural OOVs in both corpora happens in the expected increasing fashion. As we have just commented, this increasing fashion is quite constant for both corpora, although in the case of TCSTAR the dynamic system was able to introduce a large amount of natural OOVs in the first five hours, decreasing afterwards the slopes of the evolution. This occurs except for very specific moments when the slope is increased for a very short time (also for the TV corpus), which may indicate changes of topic inside the speech.

Systems with 50K vocabulary. The evolution of the presence of natural OOVs experimented pretty constant rates of increasing, although very small. These low slopes for this vocabulary size can be understood due to the fact that this vocabulary contains 79.6% and 77.8% of the vocabulary in the TCSTAR and TV test corpora, respectively, representing the 98.0% and 97.7% of total word occurrences, respectively (and as shown in table 5.13), so the remaining natural OOVs are words of low frequency and therefore more difficult to retrieve. In any case, the dynamic TD strategy is still able of taking advantage of the retrieved terms to carry out a beneficial adaptation of the system, where a positive tendency was observed (and with already significant improvements for the case of TCSTAR, needing more hours of input speech for the case of TV so as the improvements get significant).

Systems with 12.5K vocabulary. The evolution occurs also in an increasing fashion, with the highest slopes when compared to the other vocabulary sizes, as we could expect since more natural OOVs were present in the input speech. This behavior indicates that the dynamic TD strategy is still able to introduce appropriate words into the vocabulary of the systems despite the unfavorable conditions of such a small vocabulary size (and the hostile acoustic and language conditions of the TV corpus).

Now, to finalize this second study, we collected in table 5.14 the number of words introduced into the vocabulary by the different dynamic systems, distinguishing per type of added OOV and test corpus (*Nats* refer to natural OOVs of a test corpus, *Vars* refer to morphological variants and *Unexps* refers to unexpected-OOVs). In contrast with what we did for the two previous figures, we are counting all the natural OOVs that a system was able to learn throughout the decoding of a whole test corpus, so not subtracting the natural OOVs that the SWD process might have deleted. The numbers of overall deletions made by this last process are indicated in table 5.15, where we make only two distinctions between the types of removed words: either natural OOVs of a test corpus (again denoted as *Nats*), or any other word chosen from the current vocabulary that is not a natural OOV (denoted as *Others*), which might include any morphological variant and unexpected-OOVs that we have introduced before.

As a general observation considering both tables 5.14 and 5.15, we can verify that the number of added natural OOVs constitutes a large proportion with respect to the deleted natural OOVs for the best-performing systems (20K vocabulary), where this proportion is 3.1 in TCSTAR and 2.0 in TV. In the case of the 12.5K systems, this proportion remains as favorable. And in the case of the 50K systems, which hardly remove natural OOVs from the vocabulary, this proportion appears to be excellent, although it has to be taken into account that, for such a large vocabulary, the word removal process is easier as we can find many words which are very unrelated to the topics present in the speech.

Lastly, we would like to make a couple of remarks about the numbers related to the added words (table 5.14). First, the number of unexpected-OOVs added is much lower than the number of natural OOVs added, especially for the best systems. Thus, the proportion of natural OOVs over unexpected-OOVs takes the value of 3.7 in TCSTAR and 2.8 in TV, for the 20K systems. For the other systems, a proper proportion is found too for the case of 12.5K (much better, indeed), while

for the case of 50K the proportion is not as favorable. This last result can be explained due to the small number of natural OOVs that are not included in such a large vocabulary, which most probably indicates that the missing words are very specific terms from different topics which are hard in the first place to be discovered by the TD strategy.

And regarding the morphological variants, they are similar in number to the natural OOVs. This should not be alarming since it just means that, on average, we are adding to the vocabulary less than one variant per word *ready for update*, and indeed we believe they can be both useful for an adequate enrichment of the LM (when they are variants of natural OOVs, and these are majority) and for handling properly most of the phonetic competitors that otherwise might get masked.

Table 5.14: Number of added words to the vocabulary by the dynamic systems, per type of OOV and test corpus

System	test TCSTAR				test TV			
	Nats	Vars	Unexps	TOTAL	Nats	Vars	Unexps	TOTAL
DYN 20K	1080	892	289	2261	967	942	349	2258
DYN 50K	291	183	190	664	249	146	140	535
DYN 12.5K	1617	1323	101	3041	1617	1761	22	3400

Table 5.15: Number of deleted words from the vocabulary by the dynamic systems, per type of word and test corpus

System	test TCSTAR			test TV		
	Nats	Others	TOTAL	Nats	Others	TOTAL
DYN 20K	347	2238	2585	492	2274	2766
DYN 50K	8	668	676	1	287	288
DYN 12.5K	436	1568	2004	649	1780	2429

5.3.4 LM evolution study

In this third and last study, we analyze the internal changes that are produced in the LM of the dynamic systems, which are caused by the language adaptation that is produced dynamically.

As the LMs employed are based in n-grams (of order 3), we can explicitly measure the log-likelihoods of each n-gram in the model. Specifically, we are more interested in the log-likelihoods of the unigrams, since this would reflect the importance of a term in the vocabulary at some time.

Observing which **specific** terms increase/decrease their log-likelihood over time, we can know which terms are believed by the system to gain/lose usefulness for the input speech at each decoding moment (this is information was already used as one of the criteria for selecting words to remove from the vocabulary). Furthermore, this information could be employed to automatically predict which topics are more likely to arise in ulterior speech, so we could incorporate in advance the most representative words of that semantic direction that were not already present in the system's vocabulary (e.g. just by computing the KNNs of the words that experimented most recently the larger increase in their log-likelihoods).

In any case, we have focused this study on the evolution experienced by the distribution of the unigrams' log-likelihood as a whole, **disregarding the specificity** of the terms, since such distribution offer a more holistic view of the importance of all unigrams at the same time. Thus, if we observed that the distribution evolves towards greater values of log-likelihoods, it would mean that the terms are acquiring more importance inside the LM, which in the case of our dynamic TD strategy, may be caused by a tuning of the modeled language where useless terms are removed and terms that fits in that language are added. Therefore, this would be the desired evolution we would like to experience.

We plotted this evolution in figures 5.12, 5.13 and 5.14, for the dynamic systems with a vocabulary of 20K, 50K, 12.5K, respectively. The selected LMs comprise the one of the corresponding *Raw* baseline system (which does not evolve over time) and the LM of the corresponding dynamic system, sampled at different moments of the decoding process (about ten samples equally spaced along the 27 hours of each test corpus), whose distributions were drawn darker the later in time they were sampled.

Systems with 20K vocabulary. The evolution of the distribution, as seen in figure 5.12, experiments very similar changes for both test corpora. These changes are in consonance with the ones that the dynamic TD strategy should propitiate, as commented before.

Thus, we can see how the low log-likelihoods (infrequent words) keep disappearing from the distribution, so the LM only keeps those terms that are meaningful for the inherent language that is modeled at each moment. Also, higher log-likelihoods gain more relevance, indicating that, first, we were able to emphasize the most useful terms for the evolving language, and second, the new words that we kept adding fitted adequately in the LM.

These observations suggest that the LM is adapting automatically to more specific domains and topics. If we consider this together with the overall WER results, we can conclude that the LM, and also the vocabulary, experiment adaptations that are actually in accordance with the topics of the input speech, as it was intended.

Figure 5.12: Distribution evolution of the unigram log-likelihood of the systems' LM, $V=20K$. The darker the later in time the distribution was sampled. Up: TCSTAR test corpus. Down: TV test corpus

Systems with 50K vocabulary. The observed tendency in figure 5.13 is similar as for the previous system. However, as the incorporation of new words (and then the LM adaptation) is so slow in comparison (as it involves fewer terms), the effects are not as visible as before (this was also translated in smaller WER improvements). Nonetheless, when employing the dynamic system for the TCSTAR corpus, a more promising WER tendency was observed than when employing it for the TV corpus, as seen in the previous studies. Now, we can notice that this system was capable

of flattening most of the lower log-likelihoods, and thus orienting better the semantics of its LM towards the input speech.

Figure 5.13: Distribution evolution of the unigram log-likelihood of the systems' LM, $V = 50K$. The darker the later in time the distribution was sampled. Up: TCSTAR test corpus. Down: TV test corpus

Systems with 12.5K vocabulary. In figure 5.14 we can acknowledge a very similar behavior as for the systems with 20K vocabulary. We can see that for TCSTAR, the adaptation occurs

very fast, due to the successful adaptation to the European Parliament part, as we discussed before, maintaining the same shape for the rest of the decoding, although with big fluctuations on the higher log-likelihoods, which in turn is associated to the adaptation to the *Congreso de los Diputados* part. As for TV, we can see how a similar shape is reached but more smoothly and with even bigger fluctuations on the higher log-likelihoods, as the adaptation that takes place now is more difficult.

Figure 5.14: Distribution evolution of the unigram log-likelihood of the systems' LM, $V = 12.5K$. The darker the later in time the distribution was sampled. Up: TCSTAR test corpus. Down: TV test corpus

For illustrative purposes, we will show the same unigram distribution evolution for a LM that is artificially evolved via a random selection of words to add to its vocabulary. This sole LM would constitute a non-ASR system which we can denote as **Random system**. This LM would be initialized with the already employed baseline LM handling an universal vocabulary of 20K, the same as for the dynamic systems of 20K. Then, we carry out iterative updates where we add and remove the same number of words as it occurred in the updates produced in the dynamic systems of 20K. These updates are continued until we added/removed a similar total number as for these dynamic systems (about 2.3K added words, as indicated in table 5.14). Nevertheless, the word addition and deletion processes differ notably from the processes inside the dynamic systems.

Regarding the word addition process, we would select now words from an universal vocabulary of 100K in Spanish (the 100K most frequent words from the Text Corpora employed in the previous experimentation, obtained in a very similar way as we did for obtaining the vocabularies of 12.5K, 20K and 50K). The selection of words to add from this 100K vocabulary is occurring in a total **random** way, so no guiding from input speech or other source is produced. This means that the selected words may belong to very diverse topics. In order to allow the introduction of a word, we require that the term is an OOV for the current vocabulary handled by the LM, and also that enough text material is found in the Text Corpora for retraining the LM, in the same manner as the LM retraining was produced for the dynamic systems. And as regards the word deletion process, we would only employ the criterion based on low log-likelihoods, since we do not have word statistics of usage as this *Random* system is not an ASR system.

If we measure the perplexity of the final LM of the *Random* system for the ground-truth transcription of the TCSTAR test corpus, for example, we achieve a tiny improvement of a relative 1.6% from the *Raw* baseline, mainly due to the additional training materials with which we retrained the LM, and which propitiated an increase of the size of the LM. When we measure the perplexity of the final LM of the 20K dynamic system after processing the TCSTAR test corpus, this relative improvement is instead 4.4%, considerably better, as it could be expected.

We plotted the unigram distribution evolution of this *Random* system (with a constant vocabulary of 20K) in figure 5.15 (upper part), to be compared with the evolution of the 20K dynamic systems presented in figure 5.12. We can observe a very similar behavior of the evolution for the lowest log-likelihoods, as the removal criterion remained almost the same as for the dynamic systems. However, we can notice some differences in the evolution affecting **higher log-likelihoods**. In order to observe properly these discrepancies, we plotted in the lower part of figure 5.15 the differences of the final unigram distribution between this *Random* system and the *Raw* baseline, and also between the TCSTAR dynamic system and the *Raw* baseline, all with a V of 20K. This way, we can measure which changes did the two systems make to the *Raw* baseline distribution. We would like to comment two aspects here, in an attempt to offer a plausible interpretation.

First, we can notice that the higher log-likelihoods (from -4.5 to -3.5) experienced a greater increase for the dynamic system than for the *Random* system. This may be associated to the better ability of the dynamic system to incorporate words that are present in the speech of a test corpus, which can be circumscribed in a certain bunch of topics with some relation among them. Thus, most of the added words by this dynamic system are expected to belong to similar semantic areas. Furthermore, the training materials for retraining the LM with these words are likely producing an adaptation towards those semantic areas that affects other IV terms. This would finally propitiate that the LM is tuned to certain topics, permitting that the important words behind experiment an increase of their likelihood, as they should be properly trained. Nonetheless, this is not expected to occur for the *Random* system, as it is incorporating words that most probably belong to very diverse semantic areas, so the resulting LM may not reinforce as much a specific set of words that

are important to all those semantic areas at the same time, since this set of words is expected to be much smaller (e.g. we can think of a great number of words important for the political domain, but not for words that have to be important for the political, biological and artistic domains together).

Figure 5.15: Up: distribution evolution of the unigram log-likelihood of the *Random* system's LM. Down: differences of the final unigram distribution between the TCSTAR dynamic system and the *Raw* baseline, and also between the *Random* system and the *Raw* baseline. $V = 20K$

The second remark refers to log-likelihoods comprised between -5.15 and -4.6 (approximately). It

can be clearly seen that the *Random* system propitiated a very notable increase of the likelihoods in this interval, in contrast to the dynamic system. This may be associated to the kind of words that the *Random* system is adding to its vocabulary, words that belong to very diverse semantic areas, as we said. Thus, the LM of this system is prevented from modeling a properly delimited semantic area, but impelled to model multiple and not clearly delimited semantic areas. Therefore, there would probably be a lot of words that do not fit in such LM as nicely as the correctly-tuned words of the LM of the dynamic system. In turn, the LM of the dynamic system, as it is more confident about its handled words, could increase instead the higher log-likelihoods, as we commented in the previous remark. Also, it seemed to be more confident to move words not fitting properly the LM to a log-likelihood of around -5.5, not allowing them to introduce noise in higher log-likelihoods, which should be reserved for the important words of the modeled semantic area (this is reflected in the peak around -5.5, which is higher for the dynamic system than for the *Random* system).

5.3.4.1 LM size growth analysis

We end this third study by showing the **size growth** that is experimented by the different LMs, an effect that we already advanced in section 5.1.3.3. The sizes are indicated in table 5.16.

This growth occurred in a very linear fashion, which makes it easier to control the size from becoming too big so as to affect efficiency. Firstly, we have to pay attention to which maximum growth can be expected for a certain configuration and a certain amount of hours, so we can design the ASR systems more properly. This slope highly depends on the size of the vocabulary, as we already observed that the bigger is the vocabulary, the lower is the rate of added words. Thus, for the case of 20K, we experimented a **maximum growth** of 82.8% after 27 hours of speech, while for 50K, this was just 10.7%, and for 12.5K, we reached a notable 175.1% growth.

Then, depending on the computational effort we wish to dedicate, we may set differently the tolerable percentage of growth, executing the size-control operations accordingly and in the most efficient cadence (as these operations may require considerable extra computational effort). For example, and as a line of future work, we could get rid of the lower log-likelihood n-grams of higher order (e.g. bigrams and trigrams), until a more preferable size is reached.

Such operations would make more compact the LM, so it could be possible that the WER performance get affected to some extent. For this reason, it would be desirable to make use of another mechanism that processes the available metadata of the input speech, whenever it exists, or to employ additional classifiers for this purpose otherwise. Thus, as soon as a notable variation on the speech's characteristics is indicated (e.g. long separation in time or spacial location with respect to the previous speech), we could proceed with those size-control operations, getting rid of the LM specificity that now it might not be as useful as for the past speech.

Table 5.16: LM size growth for the different dynamic systems after decoding entirely a test corpus

System	test TCSTAR		test TV	
	Size at Start	Size at End	Size at Start	Size at End
DYN 20K	151.5MB	276.2MB	151.5MB	277.0MB
DYN 50K	173.7MB	192.2MB	173.7MB	189.9MB
DYN 12.5K	134.5MB	357.7MB	134.5MB	370.0MB

Conclusions

In this thesis work we have presented strategies which permit the ASR systems to experience an automatic unsupervised adaptation to the input speech, in terms of vocabulary and LM. This adaptation is based on the occurrences of OOV words in the speech being processed, since they can indicate change of topic or domain and therefore their resolution allows us to anticipate the kind of language that is going to be used in subsequent speech.

Being capable of reacting to OOV appearances permits the systems to have a medium-sized vocabulary that is fine-tuned to the most recent speech. And with such moderated vocabulary, the involved LM should acquire a sufficiently precise prediction ability for the possible sequences of the handled words, without harming efficiency. Thus, when such LM is tuned to the input speech, it can offer an optimal performance despite its limited scope. In order to prevent damages to the performance when the input speech changes of domain or topic, we should make use of strategies like the ones proposed in this thesis work so the tuning to the speech is continuously happening, and thus exploiting the accuracy of a fine-tuned LM.

The strategies that can achieve such adaptation can be understood at two levels of operation: the local, static level and the dynamic, evolving level. The operations involved at the local level tempts to firstly detect new, OOV terms directly from the input speech, as they are uttered; and secondly, to retrieve the most adequate candidate word that may suit those detected regions of speech, improving thus the output transcription. These operations conform by themselves a sub-strategy we called **static Term Discovery** (TD).

And the operations involved at the dynamic level are intended to learn the most useful terms for the input speech that were found by the previous TD strategy. This learning implies also to adapt the LM of the ASR systems towards the recently-processed speech, in an attempt of preparing the systems for a better performance in decoding ulterior speech. Both levels of operation conform what we called **dynamic vocabulary** strategies.

We have contributed at both levels of operation, proposing a new static TD strategy and also a dynamic vocabulary strategy, where for this last one we also proposed an exhaustive mathematical model. Apart from this work at the **theoretical** plane, we have developed separate **experimental frameworks** for demonstrating some of the capabilities of our proposal for such strategies. Therefore, we present next the **features** of the proposed strategies that comply with the distinct

objectives pursued in our Thesis, as well as the conclusions obtained after the experimentation with each of the strategies.

6.1 On Static Term Discovery Strategies

The proposed static TD strategy gathers a series of **features** that makes our strategy a novel contribution to the field. These characteristics were conceived to comply with the **objectives** pursued in the first part of our Thesis, as stated in chapter 3. Then, we associate below each of the corresponding objectives with the strategy's features:

- **Objective:** *to detect OOV occurrences in speech, with a sufficiently precise temporal delimitation of them.*

Feature: The TD strategy can perform the detection of OOV words by two means working concurrently, each of them indicating the precise start and end timestamps of an OOV region detected.

The first way, by making use of a single OOV word model that introduces a special term $\langle UNK \rangle$ in the vocabulary of the system, so it is reflected in both language and acoustic models (for this last one, we constructed a mumble phone model with a special topology).

The second way, by performing special confidence and output graph analysis proposed in this thesis work. In such analysis, we compare pairs of best-paths from the output graph, which are obtained for different weights of the LM, so one best-path would be linguistically emphasized whereas the other would be acoustically emphasized.

- **Objective:** *to find the most appropriate candidates for those speech segments believed to contain some OOV term, putting special effort towards the correctness of their orthographic representation by making use of external knowledge sources.*

Feature: the two following features are associated with this objective.

First, the orthographic representation of the candidates is almost always correct as they are looked for in external knowledge sources (where however some typos might exist), instead of constructing them by just some correspondence between subword units and graphemes.

And second, the strategy makes use of external knowledge sources for where to find those candidates. Moreover, it can work with open corpora such as Linked Open Data (LOD). This allows a wider search in multiple domains, instead of depending on more limited sources like for example news broadcasts. Also, LOD allows a completely free access, so we do not have to be limited, if so, by the crawling restrictions of search engines (like Google), although the TD strategy could also make use of the knowledge behind such search engines if the access is not restricted.

- **Objective:** *to explore the use of distributed representations (i.e. embeddings) of different kinds of entities (like words or knowledge base resources) that may enrich the search for the previous candidates, given the semantic power that embeddings are able to hold.*

Feature: the search for candidate words is done in two manners that complement each other. The first one is based on the phones found in a OOV region, so we can build words from them and validate their existence in external knowledge sources like Wiktionary.

The second one is based on LOD resources, selected accurately after a syntactic-semantic analysis that takes into account the context words of each OOV region detected. For this analysis, we can make use of two types of embeddings: words and resources. The first ones can be obtained with Skip-gram Word2Vec and the second ones with a scheme that we also proposed in this thesis work, Resource2Vec. Then, inside the content of the selected LOD resources, we look for words that might fit in the pertinent OOV region.

Alongside with the Resource2Vec scheme, we also proposed several quantitative analysis for this kind of embeddings, so their quality can be measured in objective ways.

- **Objective:** *to perform corrections in the output transcription of an ASR system, for those segments where a found candidate is disposed to substitute the original content.*

Feature: the strategy employs a scoring mechanism for assessing the convenience of a candidate for substituting the content of an OOV region, with multiple scores that focus on the acoustics, syntax and semantics. A final score for each candidate is computed so we can decide more adequately whether to make a correction to the output transcription.

Furthermore, the strategy does not require a second-pass audio decoding for obtaining the OOV words and correcting the transcription.

Additionally, the TD strategy holds an extra feature that is worth to mention as it comes from another contribution we made in this thesis work:

- The strategy can make use of a Word Vector Predictor System (WVPS) to improve the search for candidates based on LOD resources. We proposed in this work three different architectures of WVPS based on recurrent neural networks. One of the architectures is also equipped with attention mechanisms, which appeared to outperform the others in a experimental framework designed specifically for evaluating the implemented WVPSs.

In order to fulfill the objectives related with the implementation and evaluation of this part of the Thesis, we designed and implemented a **experimental framework** where to evaluate the proposed TD strategy.

For the **implementation**, we used ASR systems, speech corpora and knowledge sources whose characteristics are in consonance with realistic scenarios, and which are listed below. The language employed in all the systems and external sources of this experimental framework is Spanish.

- The ASR systems were implemented with a DNN-HMM architecture, and they were trained with speech corpora belonging to the political domain: a large training partition (85 hours) of the TCSTAR corpus (for the AM and LM) and the whole EUROPARL corpus (only for the LM). The test partition of TCSTAR was used for the evaluation of the TD strategy. The associated speech of these corpora presents a natural and spontaneous style as in the modern applications of ASR.

In order to force the appearance of more OOVs in the test speech, enlarging thus the room for improvement, we deleted from the vocabulary of the ASR systems a number of important words for the different topics.

- The knowledge sources employed comprise a collection of text corpora and a LOD corpus. The text corpora is composed of texts from a recent version of the Spanish Wikipedia, literature texts and political records, summing more than 2.2G words. It was used for the training of word embeddings, so acquiring a universal flavor of the language. As for the LOD corpus, we used the LOD version of Wikipedia, called dbpedia, and which contains more than 1.5M resources. This source was employed for both the training of resource embeddings and the search for candidates that occurs in the TD strategy.
- Apart from a baseline system (a plain LVCSR system without the TD strategy), we designed three TD systems to be evaluated: one system that only searches for candidates by building words from phones, and two systems that complement the previous type of search with another based on LOD resources, guiding this last type of search with either word embeddings or resource embeddings.

Finally, the **evaluation** was done by studying the results obtained from the experimentation in the previous framework, which offered interesting conclusions about the suitability of our TD strategy, as they are indicated next.

- The ASR systems equipped with our TD strategy obtained a significantly better WER than the baseline system without such strategy, up to a 7.3% of relative WER improvement.
- Both TD variants involving some type of embeddings (Word2Vec or Resource2Vec) behaved better than the phone-based TD system, with at least a 5.5% of relative WER significant improvement.
- Although both TD systems with word and resource embeddings behaved similarly, it seems that Resource2Vec may outperform Word2Vec in general TD scenarios, in which even more challenging OOVs than those appearing in the test corpus are expected to arise in spontaneous speech (where very different domains and topics are likely).

We support this idea due to the ability of Resource2Vec to carry out a potentially more exhaustive and precise search of OOVs, because it is able to learn the semantics of the resources involved properly.

Numerically, the Resource2Vec TD system was able to reach a relative 6.9% WER significant improvement from the baseline system without the TD strategy. If we consider only the utterances in which some OOVs appeared, this relative improvement rises to 8.3%, without significantly damaging the performance on other utterances.

- Although the OOV detection rises a considerable number of false alarms (due to the difficulty of the OOV detection task), the scoring mechanism of the candidates allows the strategy to avoid most of the erroneous substitutions in the transcription.

Thus, we can affirm that the proposed TD strategy is suitable to be equipped in ASR systems as it offers a way of dealing explicitly with the OOV words that might arise in the input speech. If the presence of OOVs is considerable (as it might occur with systems tuned into some specific domain or topic decoding arbitrary speech), the strategy is able to offer significant improvements that can be reflected in the output transcription. And in the contrary case, the strategy would not harm significantly the performance of the ASR system, which expands the use cases of the strategy.

6.2 On Dynamic Vocabulary Strategies

We proposed a dynamic vocabulary strategy (or, interchangeably, dynamic TD strategy) that is able to adapt automatically the vocabulary and LM of the ASR systems to the speech being decoded at each time. For the detection and resolution of OOVs, it employs the previous static TD strategy which has already shown to offer such capabilities. And for the dynamic arrangements, we proposed a set of modules that can perform operations in compliance with the objectives we established in chapter 3 for this part of the Thesis.

Thus, we can associate each of the corresponding **objectives** with the **features** offered by the dynamic TD strategy, as we indicate below:

- **Objective:** *to iteratively collect the terms discovered in the previous part, accompanied by a scoring scheme that give signs of their usefulness for the language used in the current speech.*

Feature: the dynamic TD strategy maintains a term catalog of the best candidates retrieved by the static TD strategy, whenever their associated score surpass a threshold. The proposed term catalog is subject to the following arrangements.

Each entry of the term catalog contains a word, an accumulative score and a counter of times the word was collected into the catalog. When a word enters the catalog the first time, we take the same score assigned by the TD strategy. This score is then accumulated as the word is retrieved more times by the TD strategy, considering also the same scores that this strategy assigned to it at every time the word was retrieved.

Also, all the scores in the term catalog experiments a decay over time, so the score can reflect the usefulness of a word for the most recent input speech.

- **Objective:** *to dynamically update the dictionaries of the ASR systems, making use of a selection scheme over the previously collected terms through time. Also, any update might involve the deletion of a set of terms believed not to contribute anymore to the performance of the systems.*

Feature: the dynamic TD strategy performs a selection of words to introduce into the dictionaries of the ASR systems following a scheme we proposed in this thesis work, which functions in three stages. This scheme will be subject to different models that we can configure and which will be mentioned later.

In the first stage, an initial selection of words from the term catalog is carried out, choosing all the words that surpass some new threshold, or with a more sophisticated scheme that also considers their counter of times they were collected into the catalog.

In the second stage, the previous selection of words are subject to a search for texts containing them, out from a large text corpora that is updated periodically, so the language style and words employed is up to date. The strategy only keeps those words for which enough text is available so as to learn their syntactic and semantic use in the language. The rest are included into a quarantine of terms so we prevent the static TD strategy from proposing them as candidates at least in the next future. The terms can leave the quarantine after some amount of time passed, or whenever there is new text for them available after updating the text sources.

And in the third stage, we could opt for performing an additional evaluation of the surviving words, by reconsidering the past modifications to the transcription made by them.

For this reconsideration, the strategy builds and *ad hoc* LM with the text retrieved in the second stage, compacted thanks to the use of word classes. If the LM finds inadequate most of the corrections made by a term, we can discard it and introduce it into the quarantine temporarily.

When the selection of words to add is finished, we could opt for extending them with additional OOV terms, coming from just morphological variations or from more sophisticated analysis that try to predict which semantically related words might be used in ulterior speech.

As regards the removal of terms from the vocabulary, the strategy is capable of selecting the proper number depending on the models that we will mention below. This selection of words to remove is based on two complementary means. First, by selecting the words that were not uttered recently, observing the output transcription. And second, by choosing the words with the lowest log-likelihood according to the current LM employed by the ASR system. Both criteria work in a combined fashion and allow the incorporation of new ones.

The thresholds involved here and in the collection for the term catalog, as well as the number of words to remove, can dynamically react to the models that are configured in the dynamic TD strategy (e.g. the thresholds can turn more or less strict). These models would be part of a proposed controller whose ultimate goal is to maintain the real vocabulary size as close as possible to a **vocabulary size model**. Thus, the involved actuators that manipulates the word addition and removal processes will be modulated by this controller. Apart from a vocabulary size model, the controller is handling as well two distinct **models of word addition/deletion rates**. All the models are designed to take into account different circumstances related with the current input speech, which may indicate that a vocabulary size increase/decrease is advisable, or that the two word rates should be modified in some manner. For these word rate models, we will also consider how far is the real vocabulary size from the associated model, so as to not increase such difference.

We proposed a couple of circumstances to be taken into account by the models. First, the rate of OOV regions detected, which can indicate change of topic in the input speech when high (we will propitiate then an increase in the vocabulary size and the word rates). And second, which proportion of the OOV regions detected are not retrieving a proper word candidate, which will be used for stimulating an extra increase to the models when high, as it might indicate the appearance of new and difficult topics. Finally, the word rate models will also employ a naive model of word add rate that would inform us which addition rate (and, accordingly, which deletion rate) is generally advisable for a certain vocabulary size.

- **Objective:** *to dynamically adapt the language models of the ASR systems, driven by the updated dictionaries, and accessing to external knowledge sources where to learn the proper use, in actual language, of the newly incorporated words.*

Feature: the dynamic TD strategy updates the LM of the ASR systems by performing an interpolation between two models. The first one is the most recent LM used by the ASR system, which we want to update. The second one is a LM constructed with four different types of texts, where the most important type refers to the texts containing the new words that we want to add to the vocabulary at some moment.

Then, the interpolation weight is chosen with a proposed scheme that takes into account the approximate length of the texts from which the two LMs were built.

The update of the LM requires a compilation of a new decoding graph for the ASR system, which can harm efficiency if performed too often. Thus, the strategy can decide whether to proceed with the update by taking into account, first, when was the last update carried out;

second, how many words ready for introduction into the vocabulary are available; and third, how big is the difference between the real vocabulary size of the system and the vocabulary size model that we mentioned before. This way, we try to keep a temporal margin between updates, for efficiency, without disregarding the different factors that can vary over time.

We designed and implemented a new **experimental framework** where to evaluate the proposed dynamic vocabulary strategy, fulfilling thus the objectives related with the implementation and evaluation of this second part of the Thesis.

For the **implementation**, we used slightly different ASR systems and different speech/text corpora (for both training and testing) than before, so as to adapt the framework to the new characteristics found in realistic scenarios where this strategy is to be employed. These new systems and corpora are indicated below. We maintained, however, the same external knowledge sources as before, since they already complied with the requirements of diversity, size and present time necessary for dealing with the conditions of such hostile scenarios. Again, the language in this experimental framework is Spanish.

- The ASR systems were implemented with the same DNN-HMM architecture as before.

Regarding the AM, we extended its speech training material in order to make this system more robust. Thus, apart from the previous train partition of TCSTAR (85 hours), we employed 49 hours of speech from the train partition of a second corpus, a compilation of Spanish TV programs where talk shows are being held.

Regarding the LM, it was trained with other sources where the use of the language is not constrained to the political domain nor to any other, so some universality is reached. Thus, the material employed was taken from random fragments of literature texts, texts which were used also as one of the knowledge sources, as we did before.

Coming from an universal language modeling, we limited the vocabulary (and therefore the ASR systems) to the 20K, 50K or the 12.5K most frequent words in the overall text corpora mentioned before. The last two vocabulary sizes were chosen as a higher and lower bound of functional sizes, respectively, for the current experimental framework.

Regarding the speech test corpora, we employed two different ones with distinct characteristics so as to evaluate the behavior of the strategy in two dissimilar scenarios. They were divided in small portions so as to deliver them in an iterative fashion to the ASR systems, so the strategy can decide whether to perform an update after a portion is processed.

The first test corpus is the test partition of TCSTAR (27 hours), which presents a formal speech style with homogeneous domain, although the European Parliament part is much cleaner in audio than the *Congreso de los Diputados* part. The second test corpus is the mentioned set of Spanish TV programs where talk shows are being held (also 27 hours), which presents informal speech (sometimes two or more speakers are talking at the same time, which complicates the evaluation) and multiple and diverse topics, complicating considerably the task.

- We designed three dynamic TD systems with the previous 20K, 50K and 12.5K vocabularies, respectively. For each one, we implemented a corresponding baseline system without any TD strategy and another comparative system with only the static TD strategy, to be beaten by the dynamic systems. For illustrative purposes, two oracle systems were constructed for each of the vocabularies and test corpora: one with the proper vocabulary and LM tuned to a test corpus, and one where only the vocabulary was tuned like this.

Finally, the **evaluation** of the dynamic systems involved the analysis of the results obtained by these and the comparative systems in the described experimental framework. Additional studies about the temporal behavior of the dynamic TD strategy were carried out in order to obtain more insight. The conclusions resulted from these studies are indicated below.

- In general, the dynamic TD strategy showed a positive tendency for improving the performance of the ASR systems, reaching significant improvements in most of the cases.

In the case of 20K vocabulary, the improvements are significant, with a relative 5.9% and 1.9% WER improvements over the baseline without our proposed strategies (denoted as *Raw* baseline), after the decoding of the whole TCSTAR and TV test corpus (27 hours each one), respectively. Indeed, significant improvements were already achieved after 3.2 hours and 12 hours for TCSTAR and TV, respectively. Apart from this, we also find significant improvements over the comparative system that only makes use of a static TD strategy (denoted as *TD* system), a relative 4.1% and 1.3% WER improvements for TCSTAR and TV, respectively.

As regards the 50K vocabulary, a positive tendency can be observed by studying the progressive WER over time. Since this vocabulary already contains a notable number of the words in the test corpora, the potential benefits of the dynamic strategy are thus attenuated. Nonetheless, significant improvements are found for TCSTAR, a relative 2.2% and 2.0% WER improvements over *Raw* and *TD*, respectively, after the 27 hours of the corpus. The differences with the *Raw* baseline start to get significant after 22.1 hours of speech processing. In contrast, for TV, the improvements did not get significant over those comparative systems. Nonetheless, a positive tendency is observed here, so, with more hours of input speech, significant improvements are to be expected.

And regarding the 12.5K vocabulary, we are still able to reach significant improvements, despite the challenging conditions of such a small vocabulary. Thus, for TCSTAR, we find a relative 8.2% and 5.8% WER improvements over both *Raw* and *TD*, respectively, after the 27 hours of the corpus. The benefits were observed more notably for the EPPS division of TCSTAR (the first 16.2 hours), where a relative 10.8% WER improvement over the *Raw* baseline is obtained. Furthermore, the differences with the *Raw* baseline for this corpus start to get significant just after 0.7 hours of speech processing. And regarding the TV corpus, the dynamic system is still able to reach significant improvements over the *Raw* baseline, although less notable, a relative 1.5% WER improvement after the 27 hours of the corpus. The differences with this *Raw* baseline start to get significant after 6.3 hours, winning and loosing this condition intermittently until 16.2 hours, after which the differences remain significant. Nonetheless, no significant improvement over the *TD* system is achieved now. We associate these results to the characteristics of the TV corpus and to the smallness of the system's vocabulary, especially now that the difference among vocabulary sizes (17.1K for this TV test corpus and 12.5K for the dynamic systems) is quite considerable (taking into account that the TCSTAR test corpus has a vocabulary of 15.2K).

- The words that the strategy introduces into the vocabulary are mostly the natural OOVs that are present in the test corpora, together with their morphological variants. The number of natural OOVs added is very similar to the number of the variants added, which means that when we introduce a natural OOV, we introduce one variant of it, on average, strengthening the vocabulary and LM (as the variants are very likely to be related to the topics found in the input speech).

The rest of words that we added to the vocabulary (unexpected-OOVs), which in principle are undesirable words as they do not appear in a test corpus, are introduced much less often

in comparison with natural OOVs. For example, for the case of 20K, the proportion of natural OOVs added over unexpected-OOVs added takes the value of 3.7 in TCSTAR and 2.8 in TV.

- The words selected by the strategy to be removed from the vocabulary at each update are words that do not appeared in the overall speech of the test corpus being decoded. For example, for the case of 20K, the proportion of natural OOVs added over natural OOVs deleted is 3.1 in TCSTAR and 2.0 in TV. This was propitiated by the proposed removal criteria which prevented any word that was recently transcribed or relevant for the evolving LM from being deleted, which can be better understood by considering the item below.
- The LMs of the dynamic systems experimented an adequate direction of adaptation to the input speech, in consonance with the commented tendencies in WER performance. This was observed by studying the distribution of the unigram log-likelihoods established in the LMs. Thus, the low log-likelihoods kept disappearing from the distribution as more speech was decoded, gaining more relevance the higher log-likelihoods at the same time.

This behavior indicates that the strategy was able to emphasize the most useful terms for the evolving language, rejecting the least interesting ones. Also, it indicates that the new terms that were introduced into the vocabulary seemed to fit adequately in the LM, since no abrupt changes to the previous distribution were experimented.

The resulting shape of the unigram distribution is then a narrower Gaussian with only one tail on the right (towards more-frequent words), instead of a wide Gaussian with long tails at both sides. Such shape allows the ASR systems to focus the decoding on the most useful words for the current speech. Therefore, we are avoiding a more exhausting search over numerous (and less interesting) possibilities as when having a Gaussian distribution with a long tail on the left, where the less-frequent words are found.

All in all, we can affirm that the proposed dynamic TD strategy is capable of evolving the dictionaries of words managed by ASR systems, triggering successive updates of their vocabularies and LMs towards an automatic adaptation to the current speech being decoded.

With a very moderated size of the vocabulary and a LM that models a universal use of the language, the dynamic TD strategy is able to retrieve relevant OOV words present in the speech and learn about their syntax and semantics, offering to the ASR systems an updated vocabulary and LM more in consonance with the current speech, which results in positive tendencies of WER improvement.

The longer the strategy is active, the greater benefits it can offer, being able to report significant improvements in few hours for vocabularies rounding the 20K, as more OOVs are expected. Smaller vocabularies (e.g. 12.5K), although in our experimentation they got notably benefited by the strategy (reporting indeed significant improvements in fewer hours), there is a risk that, under unfavorable conditions, they harm the system's performance if the difficulties in recognition become notable (so not even the static TD strategy could work properly). And for bigger vocabularies (e.g. 50K or more), which may be preferred as they usually offer a better starting performance, the strategy is still capable of propitiating a beneficial though slower adaptation, whose gains may become notable in the long run, due to the lower rate of OOV appearances that is to be expected.

Future Work

In this chapter we present and discuss some lines of future work that are born from the research and experimentation carried out in this Thesis. The different lines of future work are shown separately per each of the two strategies we have proposed: the static TD strategy and the dynamic vocabulary strategy (also denoted as dynamic TD strategy).

Each of these lines can be grouped in one out of **three different categories**. In the first one, we will suggest variations to the approaches presented in each of our TD strategies. In the second one, we will indicate which additional experimental scenarios would be interesting to consider so as to extend our insight about the capabilities of the proposed strategies. And in the third one, we will suggest other application fields where it might be interesting to incorporate these strategies.

7.1 On Static Term Discovery Strategies

First, we start with the lines of future work that belong to the **first category**, involving variations to the scheme proposed for the static TD strategy:

- The OOV detection is the first process of the TD strategy (see section 4.3.1), and it is crucial for not missing the OOVs uttered in the speech, since not detecting an OOV implies we will not start a search for candidates, so it will never be solved (we would have to wait for the next occurrence and try to detect it again). For this reason, the OOV detection modules are configured to obtain a high recall even in spite of a large number of false alarms. Then, further improvements in this OOV detection could increase the recall while lowering the false alarms.

We suggest a third OOV detection mechanism that could work in conjunction with the two mechanisms proposed in this thesis work. This new mechanism would imply the use of a LM of mixed words and classes, very similar to the compact LMs employed in the reconsideration mechanism of the dynamic TD strategy (see section 5.1.4.1).

For every word class i in this LM, we would have the pertinent special $\langle UNK-C_i \rangle$ word, so the ASR would acquire a more precise idea about the OOV words it may encounter (apart from the more general $\langle UNK \rangle$ word). In the same way, the TD strategy would be able to

propose specific candidates that belong to such word class, candidates that could even be listed in advance, so the gain in efficiency would be also notable. Such LM could be built in the following manner:

- Firstly, we want to group together all the words that could be replaced in a sentence without losing much syntactic nor semantic meaning (e.g. English male names, Portuguese family names, country names, adjectives or verbs with similar meanings, etc.). We believe that most of the OOVs we may find in the speech belong to this kind of word classes, as they might be arbitrarily large. These classes can be obtained, for example, by clustering word embeddings, as we commented in section 5.1.4.1.
- Second, we would train the LM over training texts, where any appearance of a member of the i th class can be counted as an appearance of a corresponding $\langle \text{Class-}C_i \rangle$ word, so all the syntax and semantics related to a class can be learned in the pertinent $\langle \text{Class-}C_i \rangle$ term. Thus, the LM would be composed of two kinds of tokens: words that do not belong to any of the defined classes, and the $\langle \text{Class-}C_i \rangle$ terms. In turn, we have to define sub-LMs that would be used to expand each of the $\langle \text{Class-}C_i \rangle$ terms, explained in the next step.
- Third, the sub-LMs associated to each $\langle \text{Class-}C_i \rangle$ term can be as simple as a 1-gram LM containing a reduced set of members (from the i th class) that we want to handle in the ASR system and a special unknown word $\langle \text{UNK-}C_i \rangle$. If the set of members for each class is not reduced, the resulting vocabulary of the ASR system is likely to turn inoperably large, as some of the classes would be able to admit an arbitrarily large number of members, as we said. The special unknown word particular of each of the classes, $\langle \text{UNK-}C_i \rangle$, would refer then to the rest of the members of that class that we dismissed as well as other members that we were not aware of initially, but which we might learn afterwards in some manner (e.g. by updating the word classes with new word embeddings that we will cluster again). Lastly, in the sub-LM of each class, we can set probabilities to the surviving members of every class e.g. just by considering the frequency of these members in the training texts, reserving some probability to the special unknown word of that class.

Thus, an ASR system equipped with such LM would be able to become aware, for some OOV occurrences, that a word of a certain class had to be uttered at some moment because it really fits the sequence of words learned for that class, but for which the ASR does not know the proper member of the class fitting also acoustically. Then, it can indicate an $\langle \text{UNK-}C_i \rangle$ in the output transcription, so the TD strategy could easily deal with such OOV occurrence, as it is already determined in time and whose semantics is also already circumscribed.

For illustrative purposes, we depicted in figure 7.1 a toy LM of this kind, in graph form, together with the sub-LMs associated to each of the defined classes, also in graph form.

- We suggest an improved acoustic score for the candidates that the TD strategy finds for each POOV SS detected (see section 4.3.2.3, where we denoted this score as Levenshtein score). As we commented then, the acoustic score was one of the most important as it takes into account the acoustic evidence of the POOV SS, which can restrict crucially the candidates that are adequate. This score was a Levenshtein distance between the sequence of phones in the POOV SS (obtained by a Phone ASR system) and the sequence of phones of the candidate (obtained by a grapheme-to-phone algorithm).

Now, we would take into account a more exhaustive phonetic transcription of the POOV SS. For doing so, we would keep all the output graph (for some reasonable beam size) that is provided by the Phone ASR system as a transcription solution for the POOV SS, instead of just computing the best-path from this graph as we did before. This way, multiple phonetic

Figure 7.1: Example of a LM of mixed words and classes, in graph form. Sub-graphs are presented for each of the class tokens

alternatives can be considered for the transcription solution. Since this output graph is a WFST, we could process the sequence of phones of the candidate with it.

Therefore, we will be able to score more precisely the different candidates that we find for the POOV SS. If the sequence of phones of a candidate coincides with the best path of this output graph, we can associate a maximum acoustic score. Then, if the sequence of phones of a candidate corresponds with a detour from that best-path, we will be still able to score accordingly the candidate, without too much penalization as it may have occurred with the Levenshtein-based original scoring scheme.

Furthermore, we could consider the employment of a confusion matrix of phones so as to take into account which score penalties should be applied for different substitution errors between pairs of phones, specially useful when the sequence of phones of a candidate does not correspond with an existing path in the mentioned output graph (in such cases, we should also devise a manner of treating insertion and deletion errors so a proper alignment between the output graph and the sequence of phones of the candidate is produced). In such confusion matrix, for example, we could account for the fact that to substitute a vowel phone by a consonant phone is more dangerous than to substitute two relatively similar consonant phones, like /d/ and /t/. In any case, this confusion matrix may be trained directly by using the ASR system for decoding speech belonging to some development set, so we can observe which phone recognition errors are more tolerable than others, and thus the same criteria would be taken into account for the computation of this improved acoustic score.

- We suggest an improvement for the Resource2Vec scheme (see section 4.4.1). In the original scheme, all the 1-hop resources of a resource R_a are considered equally as neighboring resources, which implied to ignore the specificity of the properties of every connection (e.g. *capitalOf*, *bornIn* or *releasedIn*).

Now, we would like to not miss the information provided by these properties, since they can determine more precisely the semantics of every resource (e.g. it is not the same to be born in a certain country, which can be applied to people, than to be released in that country, which can be applied to films). For doing so, we would have to list all the different kinds of properties and create extra neurons in the output layer of the network to code them, training

again the network properly. This way, the network is trained not to only predict a 1-hop resource given a resource R_a on its input, but also which kind of property is behind such connection, which would allow us to achieve embedded representations of resources that are richer semantically, so their usefulness should be increased.

- In order to facilitate a conjunction between word and resource embeddings, it would be interesting to represent both types of entities in a same vectorial space. This way, the LOD search of the TD strategy could be benefited as it would allow us to employ both kinds of representation to compute a weighted centroid that abstracts semantically the context of a OOV (see section 4.3.2.2), instead of having to choose between one of the two types.

For achieving such joined representation, we could opt to either train both embeddings together from scratch or to employ some algorithm (or even a neural network) that tries to project every entity from each of the spaces into a third, joined space.

In both cases, we would need to define enough word-resource correspondences that associate a word with a resource so they serve as anchors that indicate *inter-type* closeness, i.e. which entities of different types should appear close in the final space, apart from the relations between entities of the same type that would indicate a *intra-type* closeness (with the original criteria of Word2Vec and Resource2Vec).

While to set these correspondences can be a relatively easy task (e.g. by lemmatisation and upper casing, like in *politician* $\longleftrightarrow$ *Politics*), to join multiple spaces of distributed representations remains an open area of research.

- The OOV regions that the TD strategy detects can be produced by an acoustic and/or language decoupling with respect to the training material of the ASR system. This means that only those regions triggered by some language decoupling can hide an actual OOV, in the best case. Thus, if we skip the processing of the regions triggered only by an acoustic decoupling we can make more efficient the TD strategy. With this purpose, we suggest the introduction of an additional mechanism to classify the origin of the OOV regions detected, so we process only those regions with some language decoupling origin. This system could be based, for example, on a phone recognizer that computes the likelihood of each of the decoded phones in an OOV region with respect to the pertinent phone models, so as to measure how strange is the decoded sequence of phones. In case this sequence is considerably reliable, we can assume that the OOV region was triggered by a language decoupling, so we can process it. Nonetheless, the phonotactic rules of such recognizer should be properly trained (apart from its AM) so as to achieve an adequate phone recognition.

We proceed now with the lines of future work that correspond to the **second category**, involving the design of new experimental scenarios for the static TD strategy:

- It would be interesting to evaluate the TD strategy under a richer variety of test corpora, apart from the TCSTAR test corpus. The ASR systems would face then OOVs from very distinct topics and domains, not only the political domain of TCSTAR (which however was enough for demonstrating the capabilities of such strategy). In any case, this kind of experimentation was already performed for evaluating the dynamic TD strategy, as two very different test corpora were employed.

Also, we could assess which topics or domains are easier or more difficult to retrieve candidates from, so, for future applications of the ASR systems, we could prepare the starting vocabulary of them so as to balance these difficulties. This way, with a sufficient

general vocabulary, the systems would be more able to detect the OOVs that may appear in the different domains (due to the less confusion when decoding), and to better retrieve candidates for them (due to the semantic contextualization provided by the surrounding words belonging to the current domain).

- In order to provide better semantic representation of resources and also extend the number of represented entities, we could train the Resource2Vec scheme over multiple LOD corpora (not only DBpedia). Thus, we would take advantage of the interconnected nature of the resources from corpora that belong to the LOD project. Having entities from very diverse corpora (and likely also domain), the search for candidates of the TD strategy would be able to consider a great variety of fields, incrementing the chances of finding the precise OOV that was uttered.

And finally, we end with a line of future work that belongs to the **third category**, involving other application fields for the static TD strategy:

- Since the proposed Resource2Vec representations appeared to help in the TD strategy, they could be used to improve as well systems that face other related semantic tasks, as occurs with NLP and NLU.

For example, when it is necessary to abstract semantically a piece of text (like in Query & Answer systems), a centroid operation is usually computed with the embeddings of the words that appeared in the text. Then, this centroid could be semantically enriched if we also take into account the resource representations by Resource2Vec of those words that are associated to some resource. Also, when studying the neighboring entities of the centroid, it may result advantageous to find both word and resource entities, since the knowledge obtained would be richer. These operations could work optimally if both types of entities are located in the same vectorial space, as we suggested before.

7.2 On Dynamic Vocabulary Strategies

We indicate firstly the lines of future work that can be grouped in the **first category**, involving variations to the scheme proposed for the dynamic vocabulary strategy:

- The strategy makes use of a large text corpora where to find sequences of words that contains the new terms we are interested in adding to the vocabulary. Therefore, the corpora should be kept up to date so as to not miss terms of recent creation or new usages of the words.

We suggest to take new text corpora that is even more related to the present time, like collections of the most recent news or speech transcriptions from radio and TV shows, which could be chosen even in a day-by-day basis. This way, we ensure that we will have enough training material to learn the syntax and semantics of the words to add, and otherwise we can put them in the quarantine with more confidence (but still paying attention to whether there is trainable material for them in the future).

Furthermore, we could give preference factors to each of the text corpora, for example depending on the creation date, so the search for texts occurs in a more ordered fashion. So, a text corpus with a very low preference factor could even be removed from the collection,

specially if a considerable amount of time has already passed without the strategy making use of its content.

- For each term that we are interested in adding to the vocabulary, we take a number of lines of texts so as to learn its syntax and semantics. The lines are chosen as they are found in the text corpora. These texts are shuffled, so we can find sequences from very different sources which may enrich the knowledge about the term. However, it could be possible that part of these sequences belong to so different domains that they deteriorate the proper notion about the term. Therefore, we suggest to filter these sequences so as to keep only those that are in the semantic direction to which we are evolving the language.

For example, we could select only those sequences that are composed of words already present in our vocabulary, or at least of words whose embeddings (in case we have one) are sufficiently close to other words in our vocabulary, so we restrict the semantic area and force the language adaptation to be smooth.

- Every time the strategy adds a new term to the vocabulary, we can also add morphological variants of the term, so as to get prepared in advance for new words that are likely to happen. Then, we suggest that we also introduce other words into the vocabulary that are semantically related not just to the word we are about to add, but taking into account the set of words that were recently incorporated. Such a set can inform us more explicitly about which topic is being talked about in the current speech, so we can anticipate other words that may occur in the recent future. This set of words might be extended with the words most frequently used recently (ignoring function words) and also the words whose LM log-likelihoods are experimenting an increase, indicating that they are becoming more relevant for the current speech.

For example, we could compute a centroid with the embeddings of the words of such set. Then, the neighbors of the centroid would be words belonging to the current topic that might be interesting to add to the vocabulary or at least to consider them in a cache of words so as to propose them as candidates for the upcoming POOV SSs.

- In order to control the size growth of the LM, we can remove the n-grams containing any of the words that we decided to remove from the vocabulary at some moment. Apart from this, we could also remove, when needed, other n-grams of order >1 containing IV words. Thus, we suggest the incorporation of a mechanism that selects a number of this kind of n-grams, as it can reduce arbitrarily the size of the LM (depending on our needs). For example, we can choose n-grams whose log-likelihoods are the lowest ones, or those that contains IV words that were not used at all in the recent past.
- In this thesis work, the proposed strategies were applied to DNN-HMM ASR systems with statistical LMs. Therefore, it remains to explore their application to the recent End2End architectures based entirely on NNs, specially regarding the dynamic TD strategy, for which the benefits are larger. Since with this strategy we aim to manipulate the LM, we could think of an End2End architecture where the overall NN has a separated LM (which can be, in turn, a NN of its own, based for example on recurrent NNs). Thus, apart from the usual encoder-attention-decoder scheme, the decoder could communicate with this separated LM to refine its output. Thus, at each time, the current output of the decoder would be used as both the current transcription solution and as the next input to the NN LM (in turn, the current output of the NN LM would serve as the next input to the decoder, apart from the current output of the attention mechanism). In any case, the whole architecture could still be trained together through backpropagation. Under this configuration, the actions of the dynamic TD strategy would focus on this NN LM.

Firstly, the vocabulary of the End2End system, supposing it is based on words and not characters or subword units, would be in the form of an inventory of word embeddings, and so the outputs of both decoder and LM should be in this format too (like our proposed WVPS, section 4.7). In this format, we can just train new embeddings for the new words that we want to handle, in such a way that the new vectors fit properly the existing word embeddings space. Also, to remove a term would be as simple as to remove it from the inventory of word embeddings.

Then, the NN LM should be retrained with text materials containing the new terms to add, retrieved in some external knowledge sources. This way, we can expect that the NN LM is able to evolve through time learning the new words employed in the input speech and about their associated topics. Nonetheless, it would be necessary to study which retraining scheme would be more successful for this expected behavior to happen. Besides, the decoder should also be included in this retraining scheme so it becomes able to handle the new inventory of word embeddings, for which some kind of acoustic information about the new words should be included in some manner.

We continue with the lines of future work that correspond to the **second category**, involving the design of new experimental scenarios for the dynamic vocabulary strategy:

- The AM of the ASR systems could be trained with many more hours and distinct training corpora than we did because of availability. We believe that a more robust AM would impact positively the overall performance of both TD strategies, as it would benefit very important processes: better output transcription of the LVCSR, better OOV detection, better phonetic transcription of POOV SSs, better scoring of the candidates for the POOV SSs (which would ultimately have effects on the substitute words chosen to correct the transcription and also on the words we decide to introduce into the vocabulary) and better determination of the circumstances Θ that can influence our models of vocabulary size and word addition/deletion rates (e.g. the rate of POOV SSs detected (see section 5.1.1.2), which now would be more reliable).

Despite we are limiting the sources to train the LMs of the ASR systems (towards some extent of universality), there are no limitations regarding the sources employed to train the AMs, so we can take advantage of multiple speech corpora, whenever they are **available**. Then, the acoustic conditions of the test speech would harm the performance of our strategies as least as possible, so we could get rid of the effects of such a troublesome variable.

- It would be very interesting to evaluate the dynamic TD strategy over test corpora with more hours of speech (e.g. >50 hours), and where very different topics are treated. Thus, we would be able to ascertain two new features of the strategy.

First, we could study the evolution of the dictionaries of the ASR systems to verify whether they correspond to the topics discussed in the input speech, observing which topics are more difficult to adapt to than others. Studying the reasons behind such differences might offer more insight about the internals of the strategy, so we could find additional improvements that can address the detected dangers.

And second, we could correspondingly enlarge the starting dictionaries of the ASR systems, since although the bigger is the vocabulary, the more likely we encounter less OOVs in the speech, the offered benefits of an automatic adaptation would still be perceived in the long run, which in this scenario would not be a problem. To have a medium-large size vocabulary

(e.g. 50K or 100K) would allow the ASR systems to show performances that compare to full-vocabulary and adapted LM benchmarks, in comparison with smaller vocabularies which would require a great adaptation to the input speech so as to compete with such benchmarks.

Therefore, under such experimental scenario, we could build ASR systems that already offer a good performance (so the distance with the oracle systems might be inexistent), and which may show additional improvements after our dynamic TD strategy is working for longer periods of time, as this strategy has shown the capability of producing automatic adaptations to the current speech.

And lastly, we finalize with a line of future work that belongs to the **third category**, involving other application fields for the dynamic vocabulary strategy:

- This strategy was conceived for allowing an ASR system to evolve towards the current speech it is decoding. However, we could think of an ASR scenario where it is more interesting to have multiple ASR systems, so each one is dedicated to a specific domain, geolocation or a certain speaker, to name some of the possibilities. Then, each ASR system would be able to experience a more precise adaptation which would be useful for the speech it was designed to listen to (e.g. in the case of geolocation diversification, the vocabulary would adapt to the local people names and businesses or the words that characterize the most frequent activities in there). By avoiding the need of adapting to very diverse domains or multiple local vocabularies, the lapses of time needed for the adaptations would be minor, so we keep exploiting a LM that is carefully tuned to the input speech.

For such scenario, the design of various ASR systems would be required, for example one per city of operation, and also a module in charge of distributing the speech to decode. This module would need to use some kind of metadata or classifier so as to make a decision about to which ASR system some speech is to be delivered. The richer this metadata is (e.g. location, age of speaker, day of the week, etc.) or the better the classifier is, the more accurate decision the module can make, so as to achieve the optimal recognition offered by the most adequate ASR system.

In the same manner, the available metadata can also be used to update the text corpora where the candidates and LM training material are to be searched for. For example, if we receive a notification of an incident in real time which is close to the speech's location, we could update the text corpora and give more relevance to the sources that inform about it (e.g. news), so the candidates would be searched there with preference, and also their LM training material in case we decide to add it to the vocabulary.

Publications

We list here the publications that are a direct result of the work developed in this Thesis.

Journal article

A. Coucheiro-Limeres, J. Ferreiros-López, R. San-Segundo and R. Córdoba. "Resource2Vec: Linked Data distributed representations for term discovery in automatic speech recognition". *Expert Systems with Applications*, Vol. 112, pp. 301-320, December 2018.
DOI: 10.1016/j.eswa.2018.06.039.

International conferences

A. Coucheiro-Limeres, F. Fernández-Martínez, R. San-Segundo and J. Ferreiros-López, "Attention-based word vector prediction with LSTMs and its application to the OOV problem in ASR". *Interspeech 2019*, September 2019, Graz, Austria.

A. Coucheiro-Limeres and J. Ferreiros-López, "GTH-UPM System for Albayzin 2016 Search on Speech Evaluation". *ALBAYZIN 2016 Search on speech evaluation*, *Iberspeech 2016*, November 2016, Lisbon, Portugal.

Other publications

Next, there is a list of other publications which also involve, to some extent, the work we have developed in this Thesis.

J. D. Echeverry-Correa, J. Ferreiros-López, A. Coucheiro-Limeres, R. Córdoba, J. M. Montero, "Topic identification techniques applied to dynamic language model adaptation for automatic speech recognition". *Expert Systems with Applications*, Vol. 42, pp. 101-112, January 2015.

J. Tejedor, D.T. Toledano, P. López-Otero, L. Docio-Fernández, L. Serrano, I. Hernáez, A. Coucheiro-Limeres, J. Ferreiros-López, J. Olcoz and J. Llombart, "ALBAYZIN 2016 spoken term detection evaluation: an international open competitive evaluation in Spanish". EURASIP Journal on Audio Speech And Music Processing, September 2017.

J.D. Echeverry-Correa, A. Coucheiro-Limeres and J. Ferreiros, "GTH-UPM System for Search on Speech Evaluation" . Albayzin Evaluation Special Session, Proceedings of the Iberspeech 2014, pp. 299-305, November 2014, Las Palmas de Gran Canaria, Spain.

J. Tejedor, D. Torre, P. López-Otero, L. Docio-Fernández, C. García-Mateo, A. Cardenal, J.D. Echeverry-Correa, A. Coucheiro-Limeres, J. Olcoz and A. Miguel, "Spoken term detection ALBAYZIN 2014 evaluation: overview, systems, results, and discussion". EURASIP Journal on Audio, Speech, and Music Processing, August 2015.

M. Gil-Martín, R. San-Segundo, S. Lutfi, A. Coucheiro-Limeres, "Estimating gravity component from accelerometers". IEEE Instrumentation and Measurement Magazine, Vol. 22, pp. 48-53, February 2019.

References

- Asadi, A., Schwartz, R., and Makhoul, J. (1990). Automatic detection of new words in a large vocabulary continuous speech recognition system. In Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on, pages 125–128. IEEE.
- Audhkhasi, K., Sethy, A., and Ramabhadran, B. (2016). Semantic word embedding neural network language models for automatic speech recognition. In Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on, pages 5995–5999. IEEE.
- Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.
- Bansal, M., Gimpel, K., and Livescu, K. (2014). Tailoring Continuous Word Representations for Dependency Parsing. In ACL (2), pages 809–815.
- Bazzi, I. (2002). Modelling out-of-vocabulary words for robust speech recognition. PhD thesis, Massachusetts Institute of Technology.
- Bazzi, I. and Glass, J. R. (2002). A multi-class approach for modelling out-of-vocabulary words. In INTERSPEECH.
- Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. Journal of machine learning research, 3(Feb):1137–1155.
- Bian, J., Gao, B., and Liu, T.-Y. (2014). Knowledge-powered deep learning for word embedding. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pages 132–148. Springer.
- Bisani, M. and Ney, H. (2005). Open vocabulary speech recognition with flat hybrid models. In Interspeech, pages 725–728.
- Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. Journal of machine Learning research, 3(Jan):993–1022.
- Burget, L., Schwarz, P., Matejka, P., Hannemann, M., Rastrow, A., White, C., Khudanpur, S., Hermansky, H., and Cernocky, J. (2008). Combination of strongly and weakly constrained recognizers for reliable detection of OOVs. In 2008 IEEE International Conference on Acoustics, Speech and Signal Processing, pages 4081–4084. IEEE.
- Celikyilmaz, A., Bosselut, A., He, X., and Choi, Y. (2018). Deep communicating agents for abstractive summarization. arXiv preprint arXiv:1803.10357.

- Chen, P., Sun, Z., Bing, L., and Yang, W. (2017a). Recurrent attention network on memory for aspect sentiment analysis. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 452–461.
- Chen, S., Soni, A., Pappu, A., and Mehdad, Y. (2017b). DocTag2Vec: An Embedding Based Multi-label Learning Approach for Document Tagging. arXiv preprint arXiv:1707.04596.
- Chen, S. F. and Goodman, J. (1996). An empirical study of smoothing techniques for language modeling. In Proceedings of the 34th annual meeting on Association for Computational Linguistics, pages 310–318. Association for Computational Linguistics.
- Chen, S. F. and Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. Computer Speech & Language, 13(4):359–394.
- Chen, X., Tan, T., Liu, X., Lanchantin, P., Wan, M., Gales, M. J., and Woodland, P. C. (2015). Recurrent neural network language model adaptation for multi-genre broadcast speech recognition. In Sixteenth Annual Conference of the International Speech Communication Association.
- Clarkson, P. R. (1999). Adaptation of statistical language models for automatic speech recognition. PhD thesis, Citeseer.
- Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. Journal of Machine Learning Research, 12(Aug):2493–2537.
- Currey, A., Illina, I., and Fohr, D. (2016). Dynamic adjustment of language models for automatic speech recognition using word similarity. In 2016 IEEE Spoken Language Technology Workshop (SLT), pages 426–432. IEEE.
- DBpedia (n.d.). <https://www.dbpedia.org>. Accesed November 25, 2019.
- de Melo, G. (2017). Inducing conceptual embedding spaces from Wikipedia. In Proceedings of the 26th International Conference on World Wide Web Companion, pages 43–50. International World Wide Web Conferences Steering Committee.
- Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. Journal of the American society for information science, 41(6):391–407.
- Echeverry-Correa, J. D., Ferreiros-López, J., Coucheiro-Limeres, A., Córdoba, R., and Montero, J. M. (2015). Topic identification techniques applied to dynamic language model adaptation for automatic speech recognition. Expert Systems with Applications, 42(1):101–112.
- Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In Kdd, volume 96, pages 226–231.
- Ghannay, S., Estève, Y., Camelin, N., Dutrey, C., Santiago, F., and Adda-Decker, M. (2015). Combining continuous word representation and prosodic features for ASR error prediction. In International Conference on Statistical Language and Speech Processing, pages 84–95. Springer.
- Grave, E., Joulin, A., and Usunier, N. (2016). Improving neural language models with a continuous cache. arXiv preprint arXiv:1612.04426.
- Graves, A. and Jaitly, N. (2014). Towards end-to-end speech recognition with recurrent neural networks. In International Conference on Machine Learning, pages 1764–1772.

- Grover, A. and Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pages 855–864. ACM.
- Harris, Z. S. (1954). Distributional structure. Word, 10(2-3):146–162.
- Hazen, T. J. and Bazzi, I. (2001). A comparison and combination of methods for OOV word detection and word confidence scoring. In Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on, volume 1, pages 397–400. IEEE.
- Hinton, G. E. (1984). Distributed representations.
- Hofmann, T. (1999). Probabilistic Latent Semantic Indexing. In 22nd ACM SIGIR Forum, pages 50–57. ACM.
- Hori, T., Watanabe, S., Zhang, Y., and Chan, W. (2017). Advances in joint CTC-attention based end-to-end speech recognition with a deep CNN encoder and RNN-LM. arXiv preprint arXiv:1706.02737.
- Huang, E. H., Socher, R., Manning, C. D., and Ng, A. Y. (2012). Improving word representations via global context and multiple word prototypes. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1, pages 873–882. Association for Computational Linguistics.
- Inaguma, H., Mimura, M., Sakai, S., and Kawahara, T. (2018). Improving OOV Detection and Resolution with External Language Models in Acoustic-to-Word ASR. In 2018 IEEE Spoken Language Technology Workshop (SLT), pages 212–218. IEEE.
- Iyer, R. M. and Ostendorf, M. (1999). Modeling long distance dependence in language: Topic mixtures versus dynamic cache models. IEEE Transactions on speech and audio processing, 7(1):30–39.
- Kataria, S. and Agarwal, A. (2015). Distributed Representations for Content-Based and Personalized Tag Recommendation. In Data Mining Workshop (ICDMW), 2015 IEEE International Conference on, pages 1388–1395. IEEE.
- Kemp, T. and Jusek, A. (1996). Modelling unknown words in spontaneous speech. In Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on, volume 1, pages 530–533. IEEE.
- Klakow, D., Rose, G., and Aubert, X. (1999). OOV-detection in large vocabulary system using automatically defined word-fragments as fillers. In Sixth European Conference on Speech Communication and Technology.
- Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In MT summit, volume 5, pages 79–86.
- Kuhn, R. and De Mori, R. (1990). A cache-based natural language model for speech recognition. IEEE transactions on pattern analysis and machine intelligence, 12(6):570–583.
- Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In Proceedings of the 31st International Conference on Machine Learning (ICML-14), pages 1188–1196.
- Levy, O. and Goldberg, Y. (2014). Dependency-Based Word Embeddings. In ACL (2), pages 302–308.

- LinkedOpenData (n.d.). <https://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>. Accessed November 25, 2019.
- Lu, S., Wei, W., Fu, X., and Xu, B. (2012). Translation model based cross-lingual language model adaptation: from word models to phrase models. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pages 512–522. Association for Computational Linguistics.
- Lucas Cuesta, J. M. (2013). Contributions to the contextualization of human-machine spoken interaction systems. PhD thesis, Telecomunicacion.
- Merity, S., Keskar, N. S., and Socher, R. (2018). An analysis of neural language modeling at multiple scales. arXiv preprint arXiv:1803.08240.
- Merity, S., Xiong, C., Bradbury, J., and Socher, R. (2016). Pointer sentinel mixture models. arXiv preprint arXiv:1609.07843.
- Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
- Mikolov, T., Karafiát, M., Burget, L., Černocký, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In Eleventh annual conference of the international speech communication association.
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems, pages 3111–3119.
- Mittal, S., Joshi, A., and Finin, T. (2017). Thinking, Fast and Slow: Combining Vector Spaces and Knowledge Graphs. arXiv preprint arXiv:1708.03310.
- Moriokai, T., Tawara, N., Ogawa, T., Ogawa, A., Iwata, T., and Kobayashi, T. (2018). Language model domain adaptation via recurrent neural networks with domain-shared and domain-specific representations. In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6084–6088. IEEE.
- Mostefa, D., Hamon, O., Moreau, N., and Choukri, K. (2007). Evaluation report for the Technology and Corpora for Speech to Speech Translation. TC-STAR Project. Deliverable n. 30.
- Naptali, W., Tsuchiya, M., and Nakagawa, S. (2012). Class-based n-gram language model for new words using out-of-vocabulary to in-vocabulary similarity. IEICE TRANSACTIONS on Information and Systems, 95(9):2308–2317.
- Padmanabhan, M. and Picheny, M. (2000). Towards super-human speech recognition. In ASR2000-Automatic Speech Recognition: Challenges for the new Millenium ISCA Tutorial and Research Workshop (ITRW).
- Palmer, D. D. and Ostendorf, M. (2005). Improving out-of-vocabulary name resolution. Computer Speech & Language, 19(1):107–128.
- Parada, C., Dredze, M., Filimonov, D., and Jelinek, F. (2010a). Contextual information improves OOV detection in speech. In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pages 216–224. Association for Computational Linguistics.

- Parada, C., Sethy, A., Dredze, M., and Jelinek, F. (2010b). A spoken term detection framework for recovering out-of-vocabulary words using the web. In Eleventh Annual Conference of the International Speech Communication Association.
- Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pages 1532–1543.
- Perozzi, B., Al-Rfou, R., and Skiena, S. (2014). Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 701–710. ACM.
- Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., and Vesely, K. (2011). The Kaldi Speech Recognition Toolkit. In IEEE 2011 Workshop on Automatic Speech Recognition and Understanding. IEEE Signal Processing Society. IEEE Catalog No.: CFP11SRW-USB.
- Rastrow, A., Sethy, A., Ramabhadran, B., and Jelinek, F. (2009). Towards using hybrid word and fragment units for vocabulary independent LVCSR systems. In Interspeech, pages 1931–1934.
- Réveil, B., Demuynck, K., and Martens, J.-P. (2014). An improved two-stage mixed language model approach for handling out-of-vocabulary words in large vocabulary continuous speech recognition. Computer Speech & Language, 28(1):141–162.
- Rodriguez-Fuentes, L. J., Varona, A., Penagarikano, M., Bordel, G., and Diez, M. (2014). High-performance query-by-example spoken term detection on the SWS 2013 evaluation. In Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on, pages 7819–7823. IEEE.
- Sahlgren, M. (2006). The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations. PhD thesis.
- Salamea, C., D’Haro, L. F., de Córdoba, R., and San-Segundo, R. (2016). On the use of phone-gram units in recurrent neural networks for language identification. Odyssey 2016, pages 117–123.
- Schaaf, T. (2001). Detection of OOV words using generalized word models and a semantic class language model. In Seventh European Conference on Speech Communication and Technology.
- Sheikh, I., Fohr, D., Illina, I., Linares, G., Sheikh, I., Fohr, D., Illina, I., and Linares, G. (2017). Modelling Semantic Context of OOV Words in Large Vocabulary Continuous Speech Recognition. IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP), 25(3):598–610.
- Singh, A., Saini, S., Shah, R., and Narayanan, P. (2016). Learning to hash-tag videos with Tag2Vec. In Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing, page 94. ACM.
- Smídl, L. and Psutka, J. V. (2006). Comparison of keyword spotting methods for searching in speech. In Ninth International Conference on Spoken Language Processing.
- Stolcke, A. (2002). SRILM-an extensible language modeling toolkit. In Seventh international conference on spoken language processing.
- Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. (2015). Line: Large-scale information network embedding. In Proceedings of the 24th International Conference on World Wide Web, pages 1067–1077. International World Wide Web Conferences Steering Committee.

- Tejedor, J. and Toledano, D. T. (2018). The ALBAYZIN 2018 Search on Speech Evaluation Plan. In IberSPEECH'18.
- Tejedor, J., Toledano, D. T., Lopez-Otero, P., Docio-Fernandez, L., Serrano, L., Hernaez, I., Coucheiro-Limeres, A., Ferreiros, J., Olcoz, J., and Llombart, J. (2017). ALBAYZIN 2016 spoken term detection evaluation: an international open competitive evaluation in Spanish. EURASIP Journal on Audio, Speech, and Music Processing, 2017(1):22.
- Thomas, S., Audhkhasi, K., Tüske, Z., Huang, Y., and Picheny, M. (2019). Detection and Recovery of OOVs for Improved English Broadcast News Captioning. Proc. Interspeech 2019, pages 2973–2977.
- Tiedemann, J. (2012). Parallel Data, Tools and Interfaces in OPUS. In LREC, volume 2012, pages 2214–2218.
- Tillmann, C. and Ney, H. (1996). Selection criteria for word trigger pairs in language modeling. In International Colloquium on Grammatical Inference, pages 95–106. Springer.
- Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In Proceedings of the 48th annual meeting of the association for computational linguistics, pages 384–394. Association for Computational Linguistics.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In Advances in Neural Information Processing Systems, pages 5998–6008.
- Versteegh, M., Anguera, X., Jansen, A., and Dupoux, E. (2016). The zero resource speech challenge 2015: Proposed approaches and results. Procedia Computer Science, 81:67–72.
- Wang, W. and Stolcke, A. (2007). Integrating MAP, marginals, and unsupervised language model adaptation. In Eighth Annual Conference of the International Speech Communication Association.
- Weiss, N. A. (2017). Introductory statistics. Pearson.
- Wiktionary (n.d.). <https://www.wiktionary.org>. Accesed November 25, 2019.