2018

TRABAJO DE FIN DE GRADO

UTILIZACIÓN DE MEDIAWIKI/WIKIPEDIA Y MODELOS PROBABILÍSTICOS PARA MEDIR LA SEMEJANZA SEMÁNTICA DE DOS TEXTOS ESCRITOS EN DISTINTO IDIOMA

DANIEL KAIWEI CHEE CHAN

Resumen:

Este proyecto tiene como objetivo el desarrollo del estudio de los modelos probabilísticos de tópicos entrenados a partir de artículos de la Wikipedia para ser capaces de medir la semejanza semántica de dos texto en distinto idioma. Para este proyecto los dos idiomas son el inglés y el español.

Las primera fase del proyecto ha sido la recopilación de los datos a partir de la API de MediaWiki y generar dos corpus uno en inglés y otro en español. Los requisitos de esta tarea consistía en recoger en ambos idiomas artículos que pertenezcan a las mismas categorías en ambos idiomas y cada categoría debía de contener un mínimo de artículos para que los modelos tópicos tuvieran conocimiento de forma equitativa sobre cada categoría.

La segunda fase del proyecto ha sido el entrenamiento de los modelos probabilísticos de tópicos. Una vez generados los corpus en ambos idiomas se procedía a mandar cada uno de los textos a una API REST que permite construir el modelo para cada idioma.

La tercera y última fase, consiste en el estudio y análisis estadístico de los resultados obtenidos en los dos idiomas.

Abstract:

This project aims at the development of the study of probabilistic models of topics trained to be able to measure the semantic similarity of two different language text from Wikipedia articles. For this project, the two languages are English and the Spanish.

The first phase of the project has been the compilation of the data from the MediaWiki API and generate two corpus one in English and one in Spanish. The requirements of this assignment was to pick items that belong to the same categories in both languages in both languages and each category should contain a minimum of articles so topical models would have knowledge in an equitable manner on each category.

The second phase of the project has been training topics probabilistic models. Once generated the corpus in both languages was to send each text to a REST API that allows you to build a model for each language.

The third and last phase, consists of the study and statistical analysis of the results obtained in the two languages.

Índice

1. Introducción

Este proyecto tiene como objetivo medir la semejanza semántica de dos texto a partir de sus distribuciones de palabras. Para ello se utilizarán modelos probabilísticos de tópicos entrenados para poder clasificar textos según su temática. Uno de los casos de uso, es proyectar un texto en un espacio vectorial independiente del idioma que permita relacionarlo con otros textos sin necesidad de recurrir a ningún tipo de traducción. Para un futuro, ésta forma de relacionar información puede ser un avance en las tecnologías de la información.

La motivación de este proyecto, es ser capaz de relacionar cualquier tipo de información sin las barreras de idiomas. Otro punto que me parece muy interesante es visualizar cómo ha evolucionado dos idiomas que provienen de un idioma antiguo, “latín”.

En este documente se va a explicar, paso a paso, cómo se ha diseñado la estructura de procesos y propiedades que se han tenido en cuenta. Se probarán diversos corpus para entrenar a los modelos y se compararán los resultados. Para saber de qué forma puede influir el corpus con los modelos. También se va exponer los problemas encontrados a lo largo del proyecto.

2. Estado del Arte

Actualmente en Internet, la cantidad de información que hay es enorme. Uno de los portales más grandes de información es Wikipedia. Esta plataforma ofrece información que ha sido aportada por la comunidad. Cualquiera puede editar, añadir o revisar la información. Al ser un proyecto contribuido por la comunidad, la Wikipedia inglesa ya contiene más de 5 millones de artículos y en la Wikipedia española 1,4 millones de artículos.

Entre los dos portales en diferentes idiomas se puede observar que la cantidad de información que existe en un idioma frente al otro es mucho mayor. Por lo tanto en la mayoría de casos no existe un artículo equivalente a otro en otro idioma, es decir, una traducción directa.

También puede ser útil para relacionar de forma automática los artículos en distinto idioma, en vez, de depender de la comunidad.

Aunque no sólo se aplica a Wikipedia, si se puede aplicar a motores de búsqueda como Google, Bing, Yahoo!, …. Esto ayudaría mucho a encontrar información relacionada con las palabras claves de la búsqueda cuando se refieran a temas, porque la base del funcionamiento que tiene los modelos tópicos. Los modelos tópicos busca si un texto contiene ciertas palabras claves para clasificar el texto. A partir de las palabras claves de la búsqueda, se puede buscar a qué tópico pertenece y buscar textos que pertenezcan al tópico.

3. Objetivo del proyecto

El objetivo del proyecto es ser capaz de a partir de un texto en español (o inglés) encontrar el texto más parecido correspondiente en inglés (o español). Sin haber recurrido a ningún tipo de traducción, solamente mediante los modelos de tópicos y la semejanza semántica. Pero sobretodo es ver qué resultados se han obtenido y sacar conclusiones sobre ellos.

4. Herramientas utilizadas

Se ha elegido como lenguaje de programación Java por recomendación del tutor académico. Además por tener más conocimiento con el lenguaje frente a otros.

Como IDE se ha utilizado Eclipse, por el lenguaje de programación Java.

Se ha intentado utilizar librerías básicas que tiene Java por defecto, para evitar tener dependencias con librerías externas. Se ha utilizado la librería externa GSON que tiene como base la librería Jackson, para procesar los paquetes JSON que se envían o se reciben. También se ha utilizado la librería JSONObject para crear los cuerpos de las llamadas a los servicios y para procesar las respuestas.

Como programa de soporte se ha utilizado el programa “Postman” para probar las llamadas a los servicios API.

Como programa para analizar los datos, crear gráficos y estadísticas se ha utilizado Microsoft Excel, ya que ofrece funciones y fórmulas que pueden ayudar.

5. Desarrollo del proyecto

A continuación se ha dividido el proyecto en diferentes módulos que se explicarán una a una en detalle.

En principio, el proyecto está constituido por tres módulos:

- Creación del corpus, dónde se recogen información de Wikipedia para entrenar a los modelos tópicos.
- Entrenamiento de modelos tópicos, a partir de los corpus, se encargará de pasarlos a los servicios de los modelos tópicos encargados de entrenarlos.
- Resultados y pruebas, que será donde realizaremos pruebas y obtendremos los resultados.

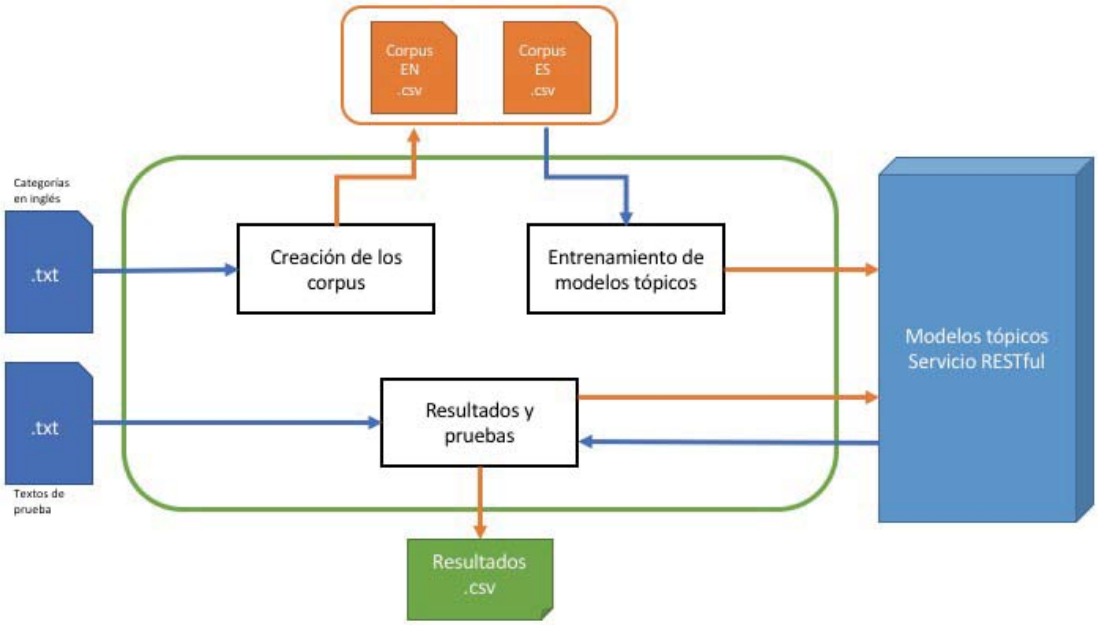

Ilustración 1: Diagrama del proyecto

5.1 Creación del corpus

Como base partiremos de 50 categorías, cada categoría tendrá que tener por lo menos 50 artículos en ambos idiomas (inglés y español). Por lo que en total serán 2500 artículos. El objetivo de esta fase que crear dos fichero .csv para cada idioma donde en cada línea tenga el siguiente formato:

$$identificadorArticulo;;nombreArtículo;;categoria;;textoArtículo$$

El criterio que se ha utilizado para escoger las categorías es elegir categorías más específicas sobre un campo, es decir, que no sea un categoría muy genérica. Por ejemplo, se ha preferido la categoría “Física atómica” ante la categoría “Física”. La razón de este criterio es obtener información mucho más concreto y específico, de esta forma, se diferencia de forma más significativa ante otras categorías. Otro criterio a la hora de escoger categorías es elegir categorías que en todos los temas posibles para poder abarcar el mayor conocimiento posible. Por ejemplo se ha escogido categorías como: “Comida callejera”, “Comercio electrónico”, “Videojuegos Post-apocalíptico”, etc.

Problemas encontrados a la hora de seleccionar las categorías: El número de páginas en la Wikipedia inglesa es mucho más mayor que la española. Esto ha supuesto un problema a la hora de escoger las categorías ya que pocas categorías no cumplen esa condición. Además esas mismas categorías son categorías muy genéricas, criterio que queríamos evitar si fuera posible.

Se ha utilizado el servicio API de MediaWiki, la cual tiene un amplio abanico de servicios para la obtener información de la Wikipedia. Para obtener información de la Wikipedia según el idioma, solamente hay que cambiar la URL base de MediaWiki:

- https://es.wikipedia.org/w/api.php
- https://en.wikipedia.org/w/api.php

Las llamadas al servicio de MediaWiki se realizaban mediante parámetros pasado como “query string”.

En todas las llamadas que se han realizado, se ha utilizado *action=query* al ser todas de consulta y *format=json* para que nos devuelva la información en formato JSON.

Tipos de consulta:

- Obtener información de una categoría o artículo.

CAMPO	PARÁMETRO	EXPLICACIÓN
PROP	info	Para especificar que solicitamos información (info) de la página
TITLES	nombre	Se inserta aquí el nombre de artículo o categoría. Hay que tener en cuenta de que en caso de que sea una categoría se debe informar de la siguiente manera: “Category:nombre” El formato de la cadena de caracteres debe estar en Percent-encoding.

Un ejemplo de llamada sería la siguiente:

"$https://es.wikipedia.org/w/api.php?action=query\&format=json\&prop=info\&titles=Category:Comercio\%20electr\%C3\%B3nico$"

Respuesta en json:


```
{
    "batchcomplete": "",
    "query": {
        "normalized": [
            {
                "from": "Category:Comercio electrónico",
                "to": "Categoría:Comercio electrónico"
            }
        ],
        "pages": {
            "86197": {
                "pageid": 86197,
                "ns": 14,
                "title": "Categoría:Comercio electrónico",
                "contentmodel": "wikitext",
                "pagelanguage": "es",
                "pagelanguagehtmlcode": "es",
                "pagelanguagedir": "ltr",
                "touched": "2018-04-22T04:19:53Z",
                "lastrevid": 105144327,
                "length": 198
            }
        }
    }
}
```


De la respuesta podemos obtener el identificador y el nombre de la categoría.

- Obtener la lista de artículos de una categoría.

CAMPO	PARÁMETRO	EXPLICACIÓN
LIST	categorymembers	Para especificar que solicitamos una lista de los artículo de la categoría
CMLIMIT	500	Límite de resultados.
CMSORT	timestamp	Existen dos opciones de ordenar los resultados: Sortkey: ordenación por clave Timestamp: por fecha de modificación desde la más reciente hasta el más antiguo. Se ha elegido la opción "timestamp" para conseguir los temas más actuales.
CMTITLE	nombre	Se inserta aquí el nombre de la categoría. Hay que tener en cuenta que al ser una categoría se debe informar de la siguiente manera: "Category:nombre" El formato de la cadena de caracteres debe estar en Percent-encoding.

Con la siguiente llamada:

$$"https://es.wikipedia.org/w/api.php?action=query\&format=json\&list=categorymembers\&cmlimit=500\&cmsort=timestamp\&cmtitle=Category:Comercio\%20electr\%C3\%B3nico"$$

Se obtiene la siguiente respuesta en json:


```
{
   "batchcomplete": "",
   "query": {
       "categorymembers": [
           {
               "pageid": 77360,
               "ns": 0,
               "title": "Subasta electrónica"
           },
           {
               "pageid": 86364,
               "ns": 0,
               "title": "B2B2C"
           },
           {
               "pageid": 86367,
               "ns": 0,
               "title": "E-Fullfilment"
            …
            …
```


De la respuesta json, se recogen los nombres de los artículos

- Obtener la categoría correspondiente en español

CAMPO	PARÁMETRO	EXPLICACIÓN
PROP	langlinks	Se especifica que solicitamos todos los enlaces interlenguaje de las página dada.
LLANG	es	Se especifica que código de lenguaje queremos.
PAGEIDS	Id. De la página	Se inserta el identificador de la página.

Con la siguiente llamada:

$$"https://en.wikipedia.org/w/api.php?action=query\&format=json\&prop=langlinks\&lllang=es\&pageids=7570884"$$

Se obtiene la siguiente respuesta json:

```
{
    "batchcomplete": "",
    "query": {
        "pages": {
            "7570884": {
                "pageid": 7570884,
                "ns": 0,
                "title": "Asian barbet",
                "langlinks": [
                    {
                        "lang": "es",
                        "*": "Megalaimidae"
                    }
                ]
            }
        }
    }
}
```


Esta llamada nos resulta útil, para encontrar la categoría en español a partir del identificador de la categoría o artículo en inglés.

- Obtener el texto de un artículo.

CAMPO	PARÁMETRO	EXPLICACIÓN
PROP	extracts	Para especificar que solicitamos la extracción de un texto.
EXPLAINTEXT	true	Por defecto, devuelve el texto en HTML pero poniendo este parámetro en “true” devuelve el texto plano.
EXLIMIT	1	En el parámetro “titles” se puede especificar varios títulos pero cómo solo queremos un artículo por llamada, se define como 1.
TITLES	nombre	Se inserta aquí el nombre del artículo.

Con la siguiente llamada:

$$"https://es.wikipedia.org/w/api.php?action=query\&format=json\&prop=extracts\&explaintext=true\&exlimit=1\&titles=Gato\%20siam\%C3\%A9s"$$

Se obtiene la siguiente respuesta json

```
{
    "batchcomplete": "",
    "query": {
        "pages": {
            "216803": {
                "pageid": 216803,
                "ns": 0,
                "title": "Gato siamés",
                "extract": "La siamesa es una raza de gato. Dentro de
dicha raza se distinguen dos variedades: por un lado el siamés moderno o
siamés propiamente dicho, y por otro el siamés tradicional o Thai.\n\n\n==
Siamés moderno ==\nEl siamés moderno es una raza de gato proveniente del
antiguo reino de Siam, actualmente Tailandia. En 1880 fueron llevados a
Inglaterra y en 1890 a Estados Unidos.\nEste tipo de siamés, desde 1950,
fue ganando protagonismo y resultó ser el elegido por los criadores y
jueces de exposiciones felinas. Tal vez sea por esto que se acuñó el
nombre \"siamés\" para el siamés moderno, ya que es la variedad que
durante todas estas décadas ha participado a nivel de competición.\nEl
estándar del siamés moderno o siamés estilizado indica un cuerpo elegante,
esbelto, estilizado, flexible y bien musculoso, con un esquema de color
denominado pointed y en otros casos, colourpoint. Su cabeza es de forma
triangular, el hocico fino, los ojos son almendrados y oblicuos, las
orejas son grandes, el cuello delgado y largo, del mismo modo que su
cuerpo y su cola. Su pelo es corto, brillante, fino, suave, apretado y
adherido al cuerpo. El siamés se caracteriza por su esquema de color
pointed típico, es decir, por una coloración más oscura en los puntos
donde la temperatura corporal es menor (extremidades, cola, cara y
orejas), que constrasta con el resto del cuerpo.\n\n\n== Siamés
tradicional o Thai ==\n\nLos orígenes del gato siamés son imprecisos,
aunque hay …
```


A partir de la respuesta json, podemos obtener el texto del artículo pero como se puede ver el texto contiene caracteres especiales que será necesario “limpiar” el texto.

En el texto se puede apreciar caracteres especiales, como por ejemplo los saltos de línea “\n”. Todos estos caracteres de espacios es necesario quitarlos ya que no nos aporta nada de información. También aparecen símbolos igual “=”, los cuales sirven para identificar las secciones del texto que tienen el siguiente formato:

$$==< nombre\ de\ la\ seccion >==$$

Hemos dejado el nombre de las secciones en el corpus por lo que solamente hemos quitado los símbolos igual.

En algunos artículos aparecen objetos como: ecuaciones, tablas, imágenes, … Al obtener el texto plano estos objetos aparecen como *“{\\diplaystyle …”*

A continuación se muestra una parte del texto que contiene dos ecuaciones.

```
Output carry and sum typically represented by the signals Cout and S,
where \n  \n    \n      \n        \n          sum\n        \n        =\n
2\n        ×\n        \n          C\n          \n            out\n
\n        \n        +\n        S\n      \n    \n    {\\displaystyle
{\\text{sum}}=2\\times C_{\\text{out}}+S}\n   in decimal system.\nA full
adder can be implemented in many different ways such as with a custom
transistor-level circuit or composed of other gates. One example
implementation is with \n  \n    \n      \n        S\n        =\n
A\n        ⊕\n        B\n        ⊕\n        \n          C\n          \n
in\n          \n        \n      \n    \n    {\\displaystyle S=A\\oplus
B\\oplus C_{\\text{in}}}\n
```

Así es como se vería el texto en la página web de Wikipedia

component in a cascade of adders, which add 8, 16, 32, etc. bit binary numbers. The circuit produces a two-bit output. Output carry and sum typically represented by the signals C_{out} and S, where $\text{sum} = 2 \times C_{\text{out}} + S$ in decimal system.

A full adder can be implemented in many different ways such as with a custom transistor-level circuit or composed of other gates. One example implementation is with $S = A \oplus B \oplus C_{\text{in}}$ and $C_{\text{out}} = (A \cdot B) + (C_{\text{in}} \cdot (A \oplus B))$.

In this implementation, the final OR gate before the carry-out output may be replaced by an XOR gate without altering the resulting logic. Using only two types of gates is convenient if the circuit is being implemented using simple IC chips which contain only one gate type per chip.

Ilustración 2: Captura de un artículo de Wikipedia

Para quitar todos los caracteres o símbolos que no son necesarios ni aportan información, se ha utilizado expresiones regulares (regex).

A continuación se muestra el diseño de alto nivel de la creación del corpus, como se relacionan todas las llamadas que hemos mencionado antes.

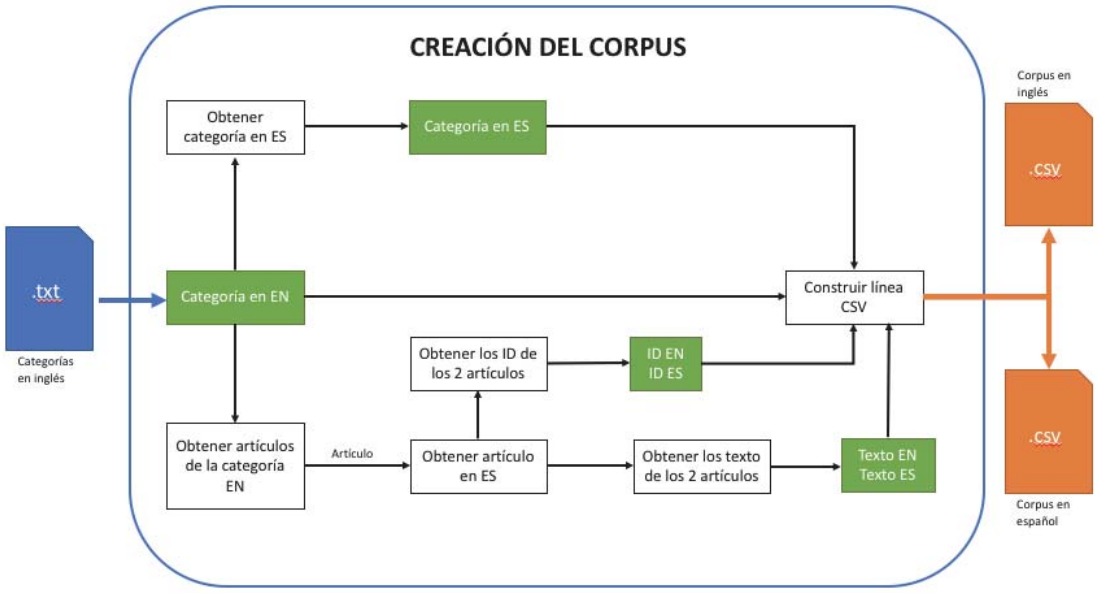

Ilustración 3: Diagrama de la creación del corpus

5.2 Entrenamiento de los modelos tópicos

En este módulo, se preparan los textos de los corpus para enviarlos al servicio RESTful de entrenamiento de los modelos probabilísticos de tópicos. En el servicio, se guardan los textos (o documentos) que se van enviando. Existen peticiones de borrado (Delete), enviar (Post) y obtener información (Get).

Existen dos endpoints para cada idioma de modelos tópicos:

- Modelo Tópicos en Español: http://librairy.linkeddata.es/dkaiwei-es-topics
- Modelo Tópicos en Inglés: http://librairy.linkeddata.es/dkaiwei-en-topics
- Espacio Vectorial de tópicos: http://librairy.linkeddata.es/dkaiwei-space

Para acceder a los servicios se requiere unas credenciales.

Cada vez que se inicia el proceso de entrenamiento, se eliminan los textos guardados en el sistema mediante la llamada de un servicio DELETE /documents. Posteriormente se va leyendo línea a línea el corpus para ir enviando mediante el servicio POST /documents. Finalmente se llama al servicio GET /documents para verificar que se han enviado todos los textos correctamente y llamar al servicio POST /dimensions para que empiece el entrenamiento.

A continuación se muestra el diseño de alto nivel del módulo.

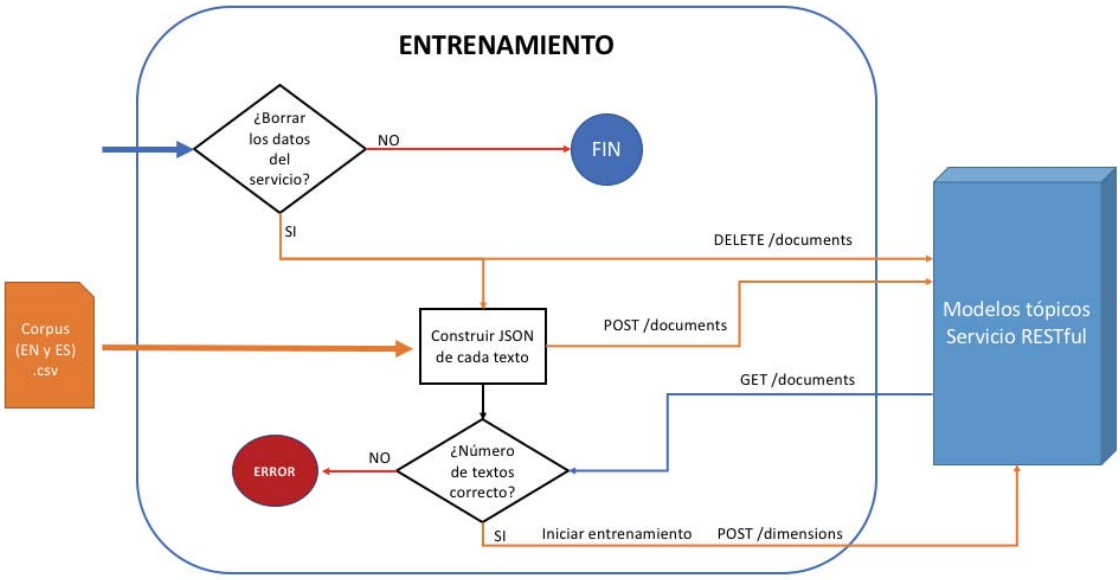

Ilustración 4: Diagrama del módulo de entrenamiento

Para enviar los documentos o textos, se envían el cuerpo, en formato JSON de la llamada POST /documents con la siguiente estructura:

```
{
  "id": "string",
  "labels": [
    "string"
  ],
  "name": "string",
  "text": "string"
}
```

En el campo “id” se introduce el identificador del artículo, en el campo “labels” se ponen las categorías a las que pertenece el artículo, para este proyecto solo se ha puesto una categoría. En el campo “name”, se pone el título del artículo y finalmente en el campo “text”, todo el texto del artículo.

Una vez, enviado todos los artículos y verificado que el número de artículos corresponde, se procede a llamar al servicio POST /dimensions que se encarga de entrenar a los modelos tópicos. Para ello se le pasan los siguiente argumentos en el cuerpo de la llamada:

```
{
   "parameters":{
      "algorithm":"llda",
      "language":"en",
      "email":"your@email.com"
   }
}
```

El parámetro “language” va en función del lenguaje, así que en caso de inglés es “en” y para el español es “en”. Más adelante se verán más parámetros y opciones a la hora de entrenar el modelo. En el parámetro “algorithm” se especifica el algoritmo que se va a usar, en nuestro caso va a ser el algoritmo LLDA, que corresponde con la implementación del algoritmo LabeledLDA orientado a manejar textos etiquetados.

Después de hacer la llamada el sistema empezará a crear los modelos tópicos, esto suele tardar según el tamaño y la cantidad de textos pasados anteriormente. Con 2500 artículos en cada idioma ha tardado aproximadamente 5 minutos. Rellenando el campo ”email” con un correo electrónico llegará un aviso a este cuando el modelo de tópicos esté listo.

Cuando estén listo los modelos tópicos, con la llamada GET /dimensions, obtendremos la siguiente respuesta con todos los tópicos:

```
{
  "dimensions": [
    {
      "id": 0,
      "name": "Plasma_physics",
      "description":
"plasma,field,energy,electron,lamp,sun,ray,temperature,x,ion"
    },
    {
      "id": 1,
      "name": "Semantics",
      "description":
"word,example,language,meaning,sentence,term,logic,value,knowledge,theo
ry"
    },
    {
      "id": 2,
      "name": "Cryptography",
      …
```


A partir de este momento los modelos están listos para las pruebas.

5.3 Pruebas y resultados

En este módulo, se preparan los textos para enviarlos al servicio de espacio vectorial para los tópicos. Con la llamada POST /shape pasándole un texto, este nos devuelve un vector de tamaño 50 que corresponde con el número de tópicos que habíamos establecido. Este vector nos muestra valores de 0 a 1. Este valor nos dice el grado que el texto pasado pertenece a un tópico. Por ejemplo: si le pasamos el texto1 y este nos devuelve el siguiente vector:

$$vector: [0.0078, 0.003, 0.0437, 0.0934, \dots \dots]$$

El texto1 tiene mayor probabilidad de que pertenezca al tópico 4 ya que es la posición del vector con mayor valor 0.0934 y tiene poca probabilidad de que pertenezca al tópico 2. La operación se realizará para el mismo artículo tanto en su versión española como inglesa. Por lo tanto los vectores de ambos artículos, en teoría se deben de parecer.

Luego se envían todos los vectores obtenidos de los artículos de prueba al servicio POST /points al cual le pasaremos un identificador y nombre del vector, el vector y el tipo. En el tipo especificamos si el vector es de un artículo español o inglés, para ello utilizamos estas dos nomenclaturas: “wiki-EN” y “wiki-ES”.

Teniendo los pares de vectores se llamará al servicio POST /comparisons que compara los dos vectores, y este nos devolverá su valor de semejanza que tiene valor de 0 a 1 también.

Una vez estén todos los vectores en el sistema de SPACE, se podrá llamar al servicio POST /points/{id}/neighbours pasándole el identificador del vector, el número de vecinos y el tipo de vecinos. En el caso de buscar los vecinos de un vector de un artículo español, nos interesa buscar un vecino de tipo inglés para saber si corresponde con el vector correspondiente pero en inglés. Hay que tener en cuenta que el identificador de un artículo en ingles y español son distintos por lo que cada vector tendrá su propio identificador. La llamada nos devolverá los vecinos más próximos, es decir, el más parecido. Cabe destacar que la llamada nos devolverá una lista ordenada de mayor a menos puntuación, por lo que podemos buscar la posición del vector correspondiente al otro idioma. En un mundo ideal, nos debería salir como vector más próximo el vector del artículo correspondiente al idioma opuesto.

Finalmente, con los resultados obtenidos del servicio, se guardarán en un fichero con formato “csv” para poder abrirlo en Microsoft Excel que nos facilita el análisis estadístico.

A continuación se muestra el diseño de alto nivel:

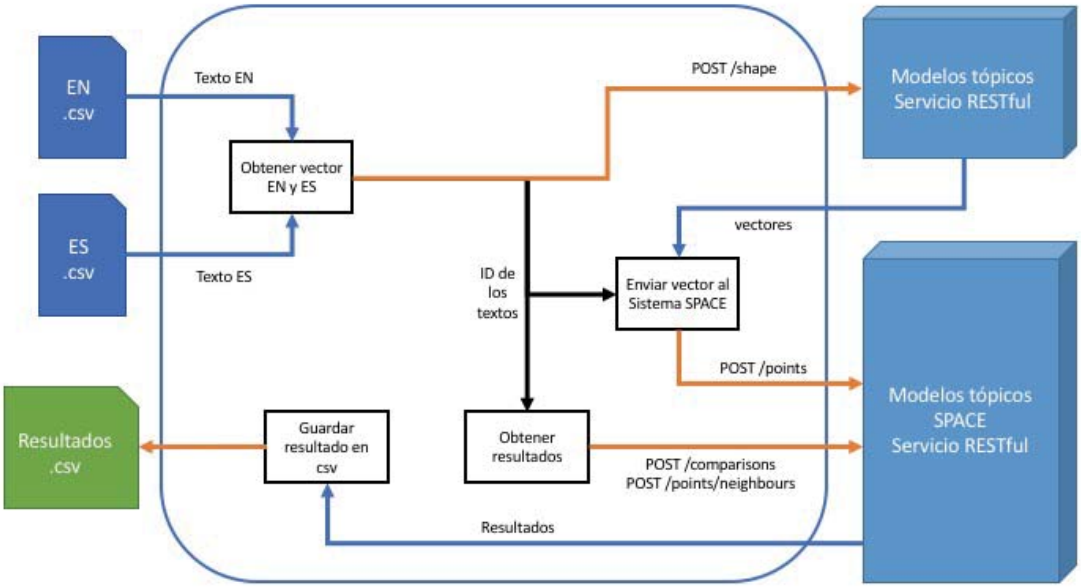

Ilustración 5: Diagrma del módulo de pruebas y resultados

En la llamada POST /shape, le pasamos el siguiente cuerpo:

```
{
  "text": "string"
}
```

Este nos devuelve un vector de tamaño 50 que corresponde con el número de tópicos:

```
{
  "vector": [
    0.007285714285707967,
    0.007142857142850956,
    0.0675714285714219955,
    0.007214285714279467,
    0.027499999999993496,
    0.037285714285707978,
      …
```

Con el vector, ya podemos enviar este vector al sistema de SPACE. Es un sistema donde se encarga de comparar vectores. Para ello llamamos al servicio POST /points con el siguiente cuerpo:

```
{
  "id": "string",
  "name": "string",
  "shape": [
    0
  ],
  "type": "string"
}
```

En los campos: “id”, “name”, y “type”, les pasamos el identificador, el nombre y el idioma del artículo respectivamente. En el campo “shape” el vector que hemos obtenido anteriormente.

Con la llamada POST /comparisons pasándole el siguiente cuerpo:

```
{
  "shape1": [
    0.0008076923076915482,
    0.0005549450549445356,
    0.23339010988989053,
    …
  ],
  "shape2": [
    0.028150668286735302,
    0.0002448359659779598,
    0.043451397326821906,
    …
  ]
}
```

Nos devuelve el resultado numérico con valor entre 0 y 1. Por ejemplo: 0.71502806610

Con la llamada POST /points/{id}/neighbours, pasamos el identificador en la URL y el cuerpo de la llamada contiene:

```
{
  "number": 10,
  "force": true,
  "type": ["wiki-EN"]
}
```

En el campo “number” se especifica el número de vecinos más próximos que se desea obtener y en el campo “type” se especifica que tipo de vecino se desea obtener. En nuestro caso es obtener el vecino del idioma opuesto.

Nos devuelve la siguiente lista ordenada:

```
{
   "neighbours":[
      {
         "id":"454916",
         "name":"Propafenone",
         "type":"wiki-EN",
         "score":0.8569560000742156
      },
      {
         "id":"2892141",
         "name":"Horseball",
         "type":"wiki-EN",
         "score":0.8483232909159333
      },
     …
```

6. Resultados y conclusiones

En este apartado se expondrán los resultados obtenidos, analizar estadísticamente y sacar conclusiones sobre ello. En las pruebas se utilizarán los 2500 artículos utilizados en el entrenamiento y otros 500 artículos seleccionados aleatoriamente. Los resultados que se van a tener en cuenta son: el valor de comparación (del servicio POST /comparisons) entre el mismo artículo con ambos idiomas, la posición en la lista de vecinos al buscar al artículo en español y viceversa, y el número de palabras de los artículos.

Como primera versión v0.1 se han entrenado los modelos tópicos con captando todos tipos de palabras: sustantivos, verbos, adjetivos y adverbios. Estos son los 5 resultados que han obtenido mayor puntuación de semejanza con los textos entrenados.

ID EN	ID ES	Score	Position EN -> ES	Position ES -> EN	EN Words	ES Words
31929	6431	0,947749689011944	411	0	2160	1298
11034	1969279	0,926661662252167	1052	0	3377	472
46112	25280	0,913558737318356	15	8	9274	2961
24931	72369	0,903198598305524	54	0	6603	2472
338960	2635156	0,903115956418651	37	5	9240	7228

Estos son los 5 resultados que han obtenido la peor puntuación de semejanza

21060202	7785709	0,355893585012216	2339	2461	27	157
7218004	7781104	0,340753986952934	2191	2469	39	126
8257916	2476451	0,336130250717175	2175	2472	49	161
21060162	7802904	0,317317573096179	2159	2474	69	171
21060139	7784494	0,304945055652058	2427	2473	30	133

A primera vista, la puntuación de semejanza parece estar relacionado con la cantidad de palabras de los artículos. Si se observan los 5 mejores puntuaciones, la cantidad de palabras en los artículos en inglés es siempre mayor a los artículos en español incluso en algunos caso de duplica o triplica. La cantidad de palabras en un artículo parece que también se relaciona con la posición en la lista de vecinos. Cuando el artículo español tiene menos palabras, y desde el artículo inglés se busca el artículo español. Este aparece mucho más lejos que buscando de español a ingles.

A continuación se muestra la información agrupado por categorías

					Neighbours	
ID	Categories	Average Score	Score Median	Score Variance	Position EN -> ES	Position ES -> EN
12	Post-apocalyptic video games	0,793815132020811	0,79620095372872	0,00355191476922	926	546
33	Human behavior	0,788945694611004	0,78866919904465	0,00437004580223	1055	635
43	World Health Organization essential medicines	0,781186141416454	0,78909714410959	0,00194556510628	1014	658
39	Ballet composers	0,779694081535370	0,79613314864921	0,00696426667170	970	629
32	Stock characters	0,778092074867402	0,78269758407810	0,00453569142795	1017	770
46	Spanish-language newspapers	0,660495443708327	0,66292402250637	0,01330863254540	941	1758
7	Plant morphology	0,658313983815436	0,67949295764583	0,01286227127706	1264	1627
49	Triplophysa	0,623111780447478	0,62371332893599	0,00265878601267	157	2291
48	Impact craters on the Moon	0,611246636177984	0,60921025571443	0,00259714643989	909	2148
18	National handball teams	0,538936885359646	0,56057631815072	0,01508058672772	1704	1998

Aquí se muestran los resultados obtenidos utilizando los textos aleatorios (no han sido utilizado como entrenamiento)

Los 5 mejores resultados:

ID EN	ID ES	Score	Position EN -> ES	Position ES -> EN	EN Words	ES Words
5843419	1173	0,925536531083220	4	0	19929	10640
42542	245957	0,913061409590863	2	3	8329	4000
766678	719655	0,905972456492161	1	2	4719	2407
602667	750369	0,897506786257194	1	22	3670	4651
23435	35488	0,895860651897382	19	2	21000	4251

Los 5 peores resultados:

39991679	7228410	0,440494816896943	397	491	41	170
13397061	2596014	0,438591826040079	427	493	22	217
37317755	5482393	0,433230997686620	455	489	108	189
5190792	1062916	0,396603333888816	482	495	131	190
41489362	6453327	0,394219966215303	492	487	133	339

Como se puede observar en estos resultados también se ve afectado por la cantidad de palabras en los artículos.

En los modelos tópicos se aparecen algunas de las siguiente palabras como las más frecuentes en el tópico:

Cryptography	Bird_families	Alcohols	Rare_dog_breeds
• Key	• Species	• Effect	• Dog
• Not	• Genus	• Also	• Breed
• Security	• Bird	• Other	• Not
• System	• Family	• Drug	• Club
• Message	• Most	• Naloxone	• Coat
• Attack	• Other	• Not	• Other
• Function	• Not	• Cause	• Also
• Hash	• Include	• Alcohol	• Kennel
• Such	• Also	• Such	• Well
• Other	• Warbler	• Treatment	• White

Éstas palabras que sean resaltado en amarillo, como se puede apreciar son muy genéricas cualquier texto puede contener. Lo mismo pasa con los modelos tópicos en español.

Aerodinámica	Embalaje	Personajes_tipo	Terminología_musical
• V	• Producto	• Personaje	• Música
• Velocidad	• Plástico	• También	• Nota
• T	• Material	• Hombre	• Musical
• Flujo	• Envase	• Mujer	• Ritmo
• R	• Utilizar	• Estereotipo	• Voz
• Aire	• Proceso	• Ejemplo	• También
• D	• También	• Historia	• Parte
• M	• Botella	• Hacer	• Ejemplo
• Presión	• Forma	• Término	• Obra
• Fluido	• Tipo	• Tipo	• Decir

En el caso de los modelos tópicos españoles, se añade a que aparecen palabras de una sola letra.

Para la versión 0.2 solo se van a tener en cuenta los sustantivos durante el entrenamiento. Para ello se llamará al servicio /POST dimensions con el siguiente cuerpo:

```
{
  "parameters": {
    "language": "es",
    "algorithm": "llda",
    "pos": "NOUN",
    "email":"dk.chee@alumnos.upm.es"
  }
}
```

A continuación se muestra un gráfica comparativa de los resultados de los textos que se han utilizado para entrenar a los modelos tópicos agrupados en categorías

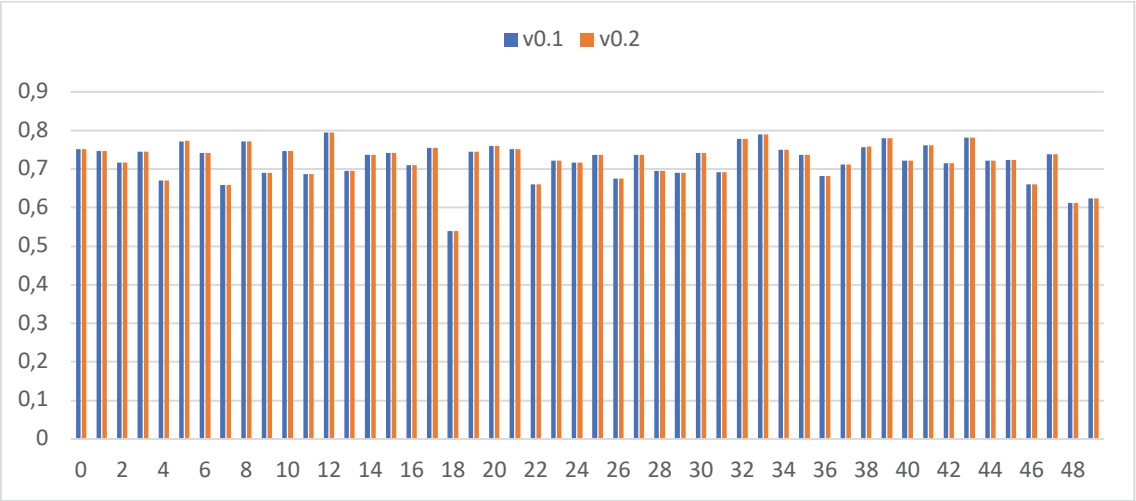

Ilustración 6: Gráfico de comparación de los resultados de las categorías entre versiones 0.1 y 0.2

En el eje X se muestra las 50 categorías y en el eje Y el valor medio de las semejanzas. Como se puede observar la versión 0.2 no mejora mucho los resultados.

A continuación se muestra otra gráfica comparativa de los resultados de textos aleatorios. En la gráfica solo se han puesto los resultados de 16 textos. Tampoco ha habido una diferencia notable.

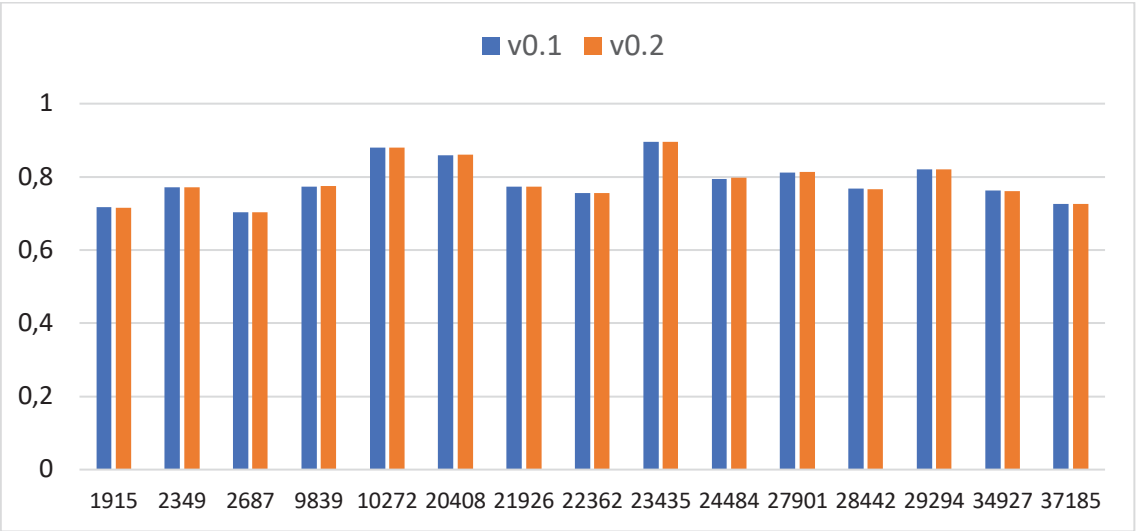

Ilustración 7: Gráfico de comparación de los resultados de textos aleatorios entre versiones 0.1 y 0.2

Comparando el modelo tópico inglés que habíamos visto antes

Cryptography v0.1		Cryptography v0.2	
• Key • Not • Security • System • Message	• Attack • Function • Hash • Such • Other	• Key • Security • System • Message • Attack	• Function • Encryption • Algorithm • Number • Signature

Se han sustituido algunas palabras genéricas que pueden generar ruido en los resultados pero los resultados no han sufrido ningún cambio significativo.

Para la versión v0.3 se han definido unas palabras a los modelos tópicos para que no los tengan en cuenta que son los “stopwords”.

En ambos idiomas, se han restringido las palabras de 1 sola letra pero en los dos idiomas tienen diferentes “stopwords” ya que en cada idioma las palabras se utilizan de forma diferentes.

Para el español, se han restringido las palabras: tipo, forma, vez, año, ejemplo y parte.
Para el inglés, se han restringido las palabras: “type”, “year” y “example”.

Para ellos se ha llamado al servicio POST /dimensions con los siguientes cuerpos:

```
{
   "parameters":{
      "algorithm":"llda",
      "language":"es",
      "pos":"NOUN",
      "email":"dk.chee@alumnos.upm.es",
      "stopwords":"a b c d e f g h i j k l m n ñ o p q r s t u v w
x y z tipo forma vez año ejemplo parte"
   }
}
```

```
{
   "parameters":{
      "algorithm":"llda",
      "language":"en",
      "pos":"NOUN",
      "email":"dk.chee@alumnos.upm.es",
      "stopwords":"a b c d e f g h i j k l m n o p q r s t u v w x
y z example year type"
   }
}
```

Aquí se muestran la gráfica comparativa de los mismos textos aleatorios:

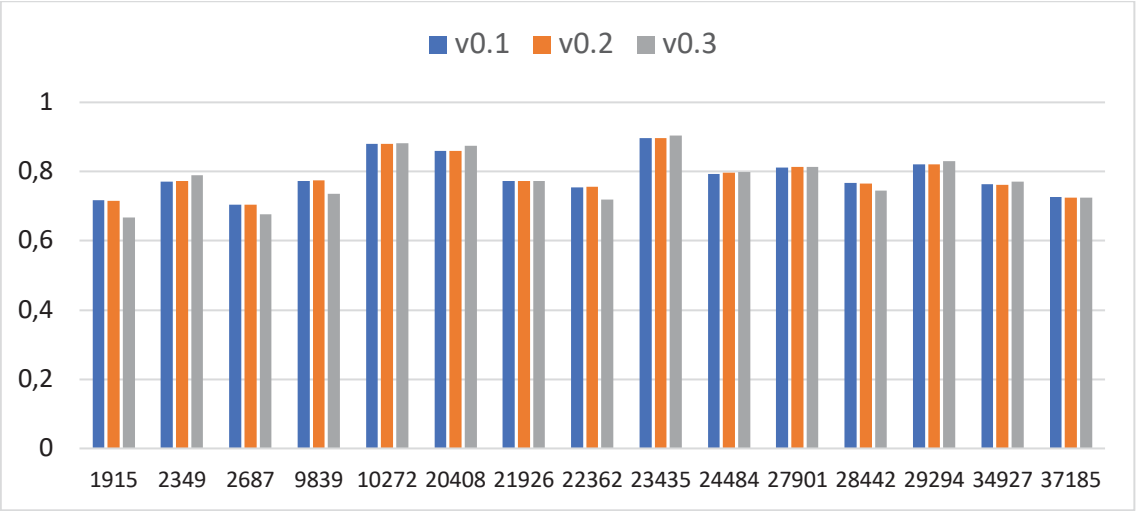

Ilustración 8: Gráfico de comparación de los resultados de los textos aleatorios entre versiones 0.1, 0.2 y 0.3

En algunos textos mejoran, otros empeoran y algunos se quedan igual, es decir, no hay ningún cambio significativo. Como consecuente también, los textos entrenados ocurre exactamente lo mismo (se muestra los resultados agrupados por categorías)

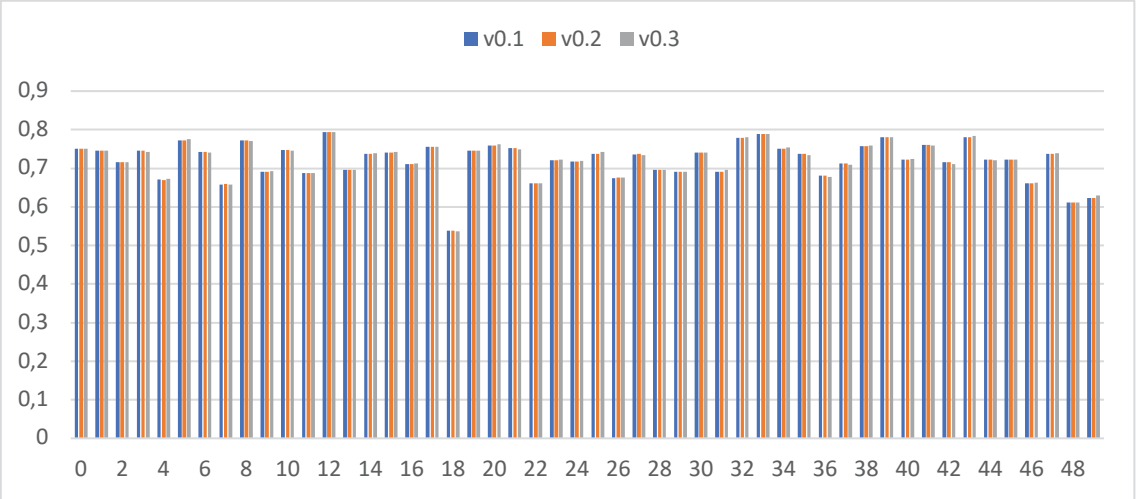

Ilustración 9: Gráfico de comparación de los resultados de las categorías entre versiones 0.1, 0.2 y 0.3

A continuación vamos a ver si las palabras de los tópicos han cambiado mucho respecto a la versión 0.2:

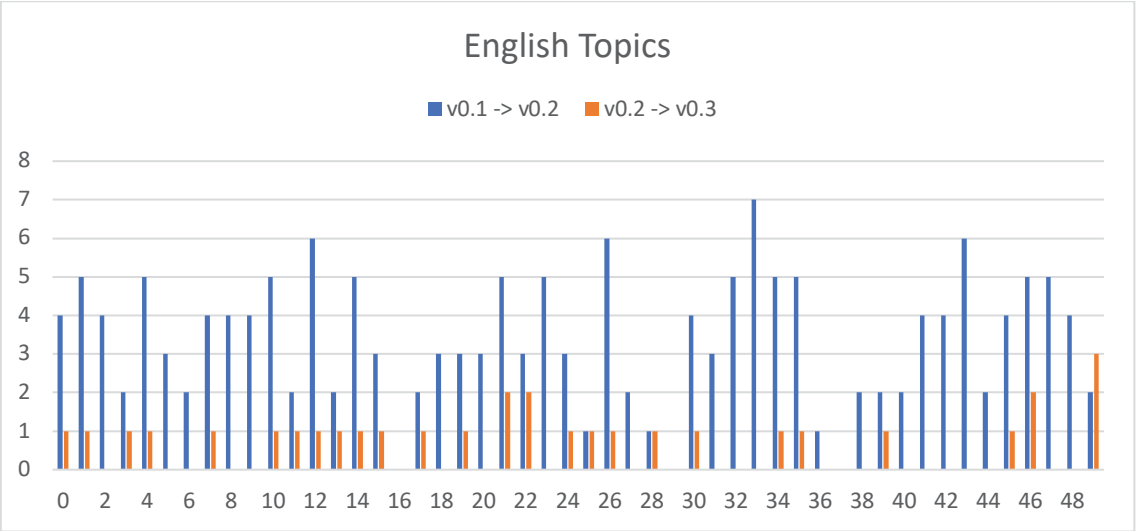

Ilustración 10: Gráfico comparativos de cambios en los tópicos en inglés

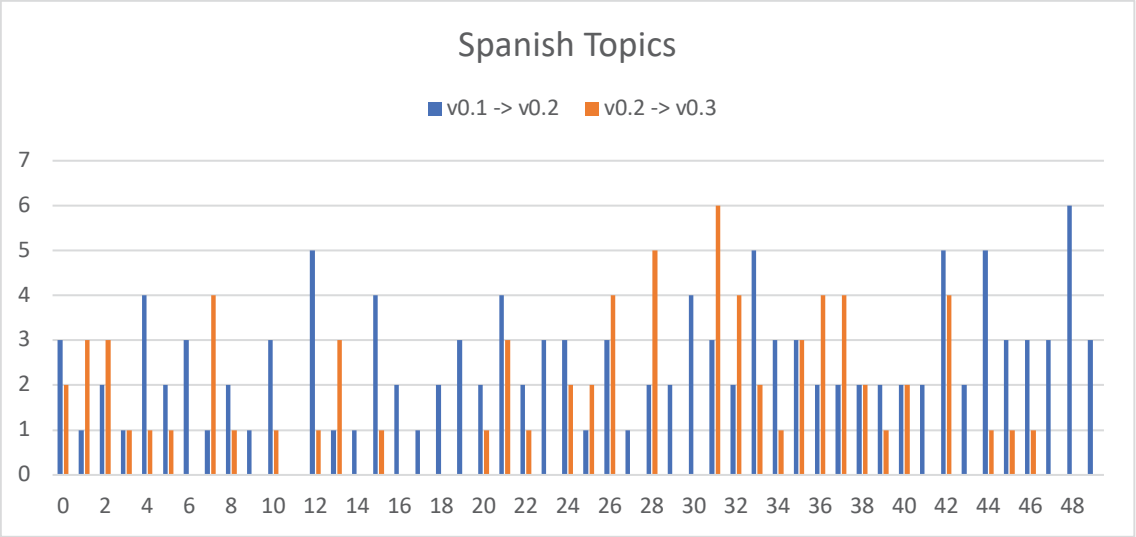

Ilustración 11: Gráfico comparativos de cambios en los tópicos en español

Las palabras en los tópicos han cambiado más de la versión 0.1 a 0.2 que en la versión 0.2 a 0.3. En los tópicos españoles ha habido mucho más cambios que en los tópicos ingleses. También cabe destacar que en los tópicos en español se ha restringido más palabras. Aún así las tres versiones muestran resultados muy similares. Con esto podemos concluir que los cambios en los tópicos no están teniendo un impacto notable.

Esto nos conduce de nuevo a la longitud de los textos. A continuación se va a mostrar una gráfica de dispersión, con el eje Y el grado de semejanza entre los dos idiomas del artículo y en el eje X, la cantidad de palabras entre idiomas del artículo.

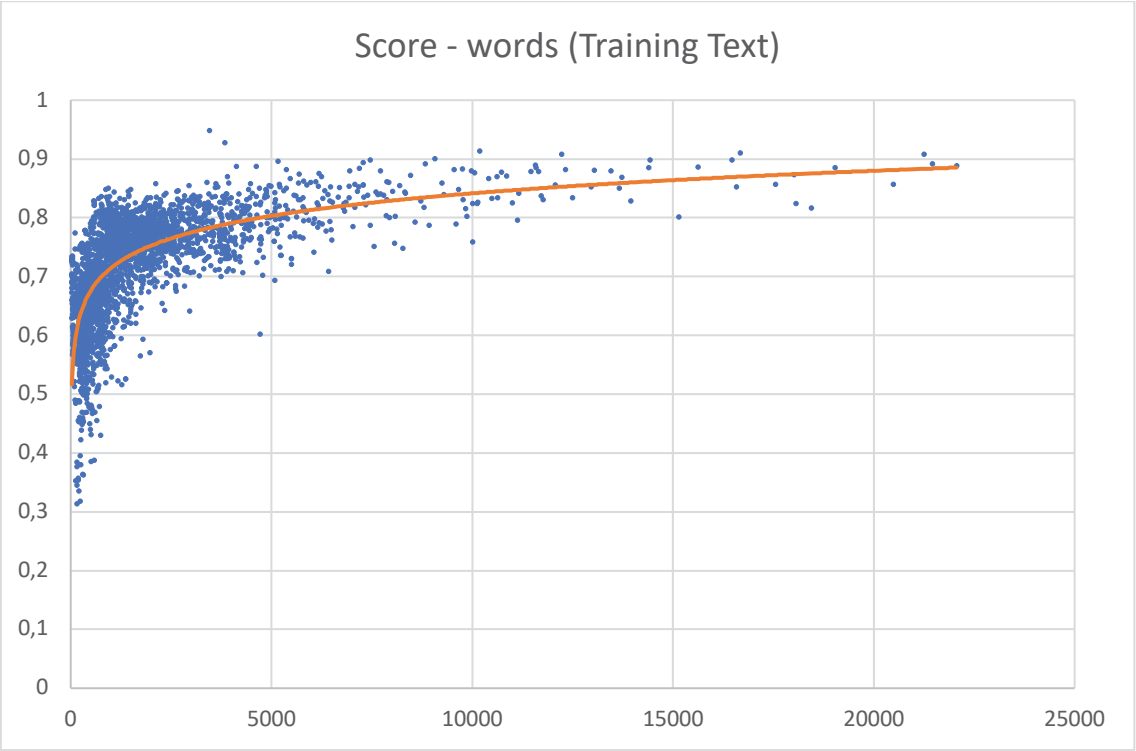

Ilustración 12: Gráfico de dispersión con textos entrenados

Como se puede ver en la gráfica, los puntos de dispersión siguen una tendencia logarítmica cuánto mayor sea la cantidad de palabras en los textos se consigue una mayor semejanza.

Aquí se muestran la misma gráfica pero con los artículos aleatorios, en el que pasa exactamente lo mismo.

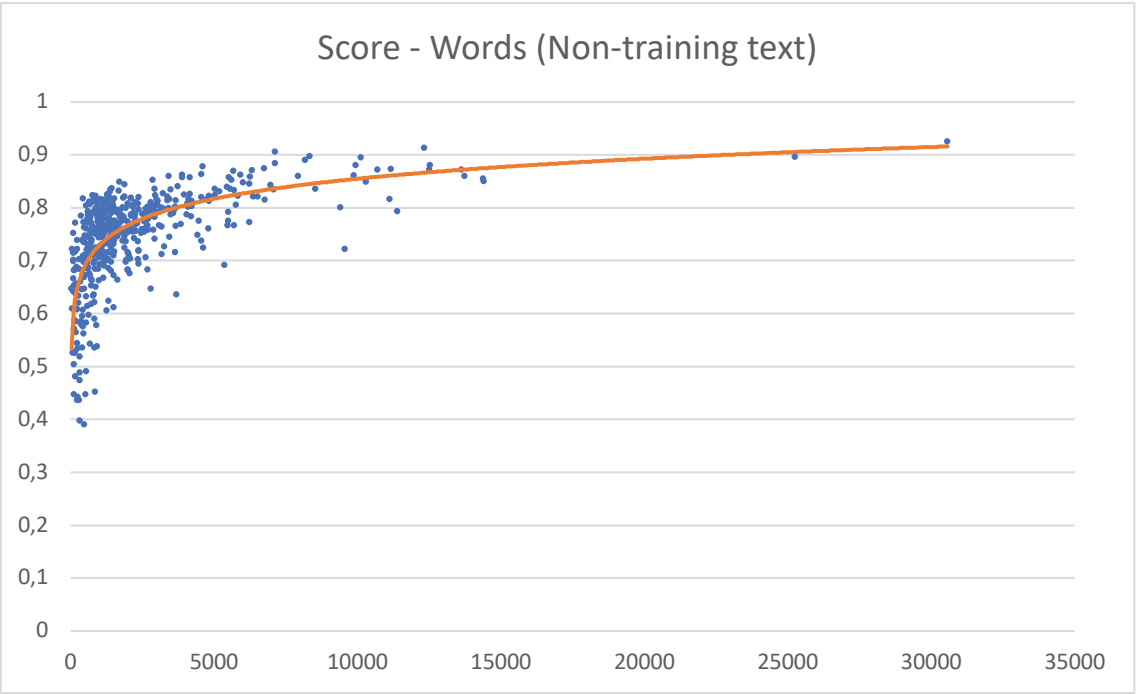

Ilustración 13: Gráfico de dispersión de textos aleatorios

A partir de 5000 palabras es cuándo los valores mejoran sustancialmente y a partir de las 10000 palabras ya no influye tanto. Con una cantidad de 30.000 palabras entre los dos artículos (inglés y español) se ha alcanzado el mejor valor de semejanza de 0,925.

En conclusión, con estos datos, podemos decir que los modelos tópicos son más efectivos si la cantidad de texto de mayor a 30.000 si queremos que al menos los valor de semejanza estén por encima del 0,9.

Los datos concuerdan con la base del algoritmo utilizado, LLDA, el cual se basa en la repetición de las palabras para conocer las palabras importantes. Como consecuente, en los textos cortos no se llega a identificar las palabras claves del texto por no haber suficientes repeticiones de las palabras clave y por eso no se llega a un valor de semejanza fiable.

Tal vez para un futuro, sea mejor utilizar otro algoritmo que sea capaz de identificar estas palabras claves en textos cortos.

7. Futuras líneas de trabajo

Existen muchas formas de ampliar este proyecto en un futuro para seguir avanzando y mejorar los modelos tópicos:

- Entrenar los modelos tópicos con diferentes algoritmos, según la longitud de los textos. También se podría hacer lo mismo para el calcular la semejanza.

- Automatizar el entrenamiento de los modelos tópicos basados, según los artículos se vayan actualizando o aparezcan nuevos artículos, que el sistema sea capaz de detectar esos cambios y volver a entrenar los modelos tópicos.

- Crear una herramienta con interfaz de usuario para realizar pruebas y para el uso real.

- Estudio de otros algoritmos para analizar textos cortos.

- Recopilación de datos desde otras fuentes para el entrenamiento de modelos tópicos.

8. Bibliografía y referencias

[1]. Documentación de JSONObject:
https://docs.oracle.com/javaee/7/api/javax/json/JsonObject.html

[2]. Documentación de JSONArray:
https://docs.oracle.com/javaee/7/api/javax/json/JsonArray.html

[3]. Documentación para las conexiones HTTP:
https://docs.oracle.com/javase/7/docs/api/java/net/HttpURLConnection.html

[4]. Consulta de servicio de API de MediaWiki:
https://www.wikidata.org/w/api.php

[5]. Funciones y fórmulas de Microsoft Excel: https://support.office.com/es-es/article/funciones-de-excel-por-categor%C3%ADa-5f91f4e9-7b42-46d2-9bd1-63f26a86c0eb

[6]. Tester & Debugger de Expresiones Regulares Online (Regex):
https://regex101.com/

[7]. Autentificación http: https://stackoverflow.com/questions/4883100/how-to-handle-http-authentication-using-httpurlconnection

GitHub del proyecto:
https://github.com/danick6/Topic_Models

Modelos de tópicos exportados a Docker (es necesario tener Docker instalado.)

- v0.1:
  - español: docker pull Librairy/librairy/wiki-es:0.1
  - ingles: docker pull librairy/wiki-en:0.1
- v0.2:
  - español: docker pull librairy/wiki-es:0.2
  - ingles: docker pull librairy/wiki-en:0.2
- v0.3:
  - español: docker pull librairy/wiki-es:0.3
  - ingles: docker pull librairy/wiki-en:0.3

9. Referencias de figuras

Este documento esta firmado por

Firmante	CN=tfgm.fi.upm.es, OU=CCFI, O=Facultad de Informatica - UPM, C=ES
Fecha/Hora	Mon Jun 04 16:09:20 CEST 2018
Emisor del Certificado	EMAILADDRESS=camanager@fi.upm.es, CN=CA Facultad de Informatica, O=Facultad de Informatica - UPM, C=ES
Numero de Serie	630
Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)