Universidad Politécnica
de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Grado en Ingeniería Informática

Trabajo Fin de Grado

Lectura Fácil: Inflexión de Sustantivos y Adjetivos

Autor: Daniel Terrón Martín

Tutora: María del Carmen Suárez de Figueroa Baonza

Cotutor: Juan Acher Blanco Martín

Madrid, Mayo 2024

Trabajo Fin de Grado
Grado en Ingeniería Informática
Título: Lectura Fácil: Inflexión de Sustantivos y Adjetivos
Mayo, 2024

Autor: Daniel Terrón Martín

Tutora:

María del Carmen Suárez de Figueroa Baonza
Departamento de Inteligencia Artificial
ETSI Informáticos
Universidad Politécnica de Madrid

Cotutor:

Juan Acher Blanco Martín
Departamento de Inteligencia Artificial
ETSI Informáticos
Universidad Politécnica de Madrid

Resumen

La lectura fácil es una herramienta esencial para promover la accesibilidad y la inclusión en la comunicación escrita, especialmente para personas con dificultades de comprensión lectora. En un mundo donde la información escrita es fundamental para la participación plena en la sociedad, la lectura fácil facilita el acceso a textos a personas con discapacidad cognitiva, personas que están aprendiendo el idioma, personas mayores y cualquier otra persona que enfrente barreras para la comprensión lectora.

La importancia de la lectura fácil radica en su capacidad para hacer accesible la información a todos, permitiendo que más personas puedan comprender y utilizar textos en su vida diaria. Sin embargo, la adaptación de textos a lectura fácil es un proceso laborioso y costoso. Requiere un profundo conocimiento de las técnicas de simplificación lingüística y una cuidadosa consideración de la coherencia y claridad del texto adaptado. Por eso, con este trabajo estamos buscando una forma de flexionar automática y un proceso de adaptación de palabras a frases semi-automático.

En este contexto, la correcta inflexión de sustantivos y adjetivos juega un papel crucial para mantener la claridad y la coherencia del texto adaptado. La inflexión adecuada asegura que los textos sean fáciles de entender y mantengan su significado original, sin introducir ambigüedades que puedan confundir a los lectores.

Este Trabajo de Fin de Grado de Daniel Terrón (TFG) se centra en analizar y proponer estrategias para la correcta inflexión de sustantivos y adjetivos en materiales de lectura fácil, con el objetivo de mejorar su eficacia y accesibilidad.

A lo largo de la memoria se detallarán los métodos, procesos, técnicas y funciones empleadas para llevar a cabo este estudio.

Abstract

Easy-to-read is an essential tool for promoting accessibility and inclusion in written communication, especially for people with reading comprehension difficulties. In a world where written information is essential for full participation in society, easy reading facilitates access to texts for people with cognitive disabilities, language learners, older people and anyone else who faces barriers to reading comprehension.

The importance of easy reading lies in its ability to make information accessible to all, enabling more people to understand and use texts in their daily lives. However, adapting texts for easy reading is a laborious and costly process. It requires a thorough knowledge of linguistic simplification techniques and careful consideration of the coherence and clarity of the adapted text. Therefore, with this work we are looking for a form of automatic inflectionalisation and a semi-automatic word-to-sentence adaptation process.

In this context, the correct inflection of nouns and adjectives plays a crucial role in maintaining the clarity and coherence of the adapted text. Proper inflection ensures that texts are easy to understand and maintain their original meaning, without introducing ambiguities that may confuse readers.

This Final Degree Project by Daniel Terrón (TFG) focuses on analysing and proposing strategies for the correct inflection of nouns and adjectives in easy-to-read materials, with the aim of improving their effectiveness and accessibility.

Throughout the report, the methods, processes, techniques and functions used to carry out this study will be detailed.

Tabla de contenidos

Índice de Ilustraciones

Índice de Tablas

1. Introducción

La comunicación escrita es y ha sido un pilar fundamental en la sociedad, ya que nos permite compartir información, expresar ideas y participar plenamente en la vida cotidiana. Sin embargo, para muchas personas, especialmente aquellas con dificultades de comprensión lectora, el acceso a la información escrita puede representar un desafío significativo. Esto se debe a la complejidad del lenguaje, que puede incluir vocabulario especializado, estructuras gramaticales complejas y un estilo de escritura que no siempre es accesible para todos los lectores.

La lectura fácil es una estrategia fundamental para garantizar que la información sea accesible y comprensible para una amplia variedad de lectores. En particular, su desarrollo se dirige principalmente a personas con discapacidades cognitivas, niños en proceso de aprendizaje y aquellos que están adquiriendo un nuevo idioma. La lectura fácil implica el uso de un lenguaje claro y directo, la simplificación de las estructuras gramaticales y la inclusión de apoyos visuales cuando sea necesario. Esta estrategia no solo facilita la comprensión, sino que también promueve la inclusión y la participación en la sociedad.

En este trabajo, nos centraremos en un aspecto específico de la lectura fácil: la inflexión de sustantivos y adjetivos, y en menor medida, la de verbos y determinantes. La inflexión se refiere a la modificación de las palabras para expresar diferentes categorías gramaticales como el género, el número, el caso y el grado. En el caso de los sustantivos y adjetivos, las inflexiones más comunes son las de género (masculino y femenino) y número (singular y plural). Comprender cómo funcionan estas inflexiones es crucial para mejorar la capacidad de lectura y escritura, ya que ayuda a los lectores a identificar relaciones gramaticales y a construir oraciones de manera correcta y coherente.

Para una mejor comprensión, es útil conocer las definiciones básicas de algunos términos gramaticales fundamentales sacados del Diccionario de la lengua española de WordReference [1]:

Sustantivo: Parte de la oración con morfemas de género y número y cuya función es ser núcleo del sintagma nominal. Los sustantivos designan semánticamente seres y objetos sobre los que se predican cualidades. Por ejemplo: "libro", "niña", "ciudad".

Adjetivo: Palabra que acompaña al sustantivo, concordando con él en género y número, para limitar o completar su significado. Por ejemplo: "rojo", "grande", "rápido".

Determinante: Parte del sintagma nominal que actualiza al nombre/sustantivo. Ejemplos de determinantes son: "el", "la", "un", "este", "mi", "algunos".

Verbo: Parte conjugable de la oración que expresa la acción y el estado del sujeto y ejerce la función sintáctica de núcleo del predicado. Por ejemplo: "correr", "ser", "hablar".

En resumen, la lectura fácil y el conocimiento de las inflexiones de sustantivos y adjetivos son componentes clave para mejorar la accesibilidad y la comprensión de la información escrita, especialmente para aquellos que enfrentan mayores desafíos en el momento de la lectura.

1.1 Objetivos del Proyecto

El propósito primordial de este Trabajo de Fin de Grado consiste en elaborar un programa que permita la correcta inflexión de sustantivos, adjetivos y determinantes incorporando dos funcionalidades fundamentales: una que produce todas las inflexiones de género y número para un sustantivo, adjetivo o determinante dado, y otra que adapta una palabra a una frase determinada. Con el fin de alcanzar una implementación eficaz de las funcionalidades, se establecen los siguientes objetivos específicos:

- Definición y análisis de la inflexión de sustantivos y adjetivos. Realizar un análisis detallado de la inflexión de los sustantivos y adjetivos en español, incluyendo las normas gramaticales y patrones comunes y no comunes de cambio de género y número.
- Comprensión y adaptación al lenguaje de programación Python. Desarrollar una comprensión profunda del lenguaje de programación Python, así como sus estructuras de datos, funciones, y bibliotecas estándar, para asegurar una implementación eficiente y eficaz del programa.
- Estudio de la librería denominada "spanish_inflections" [2] en Python. Investigar y comprender el uso de la librería 'spanish_inflections', que proporciona herramientas para gestionar las inflexiones en español, evaluando su funcionalidad y aplicabilidad al proyecto.
- Estudio de la librería denominada 'spaCy' [3] para Python: Se requiere explorar la librería denominada "spaCy", un amplio conjunto de herramientas para el procesamiento del lenguaje natural en Python, con el propósito de emplear sus habilidades para el análisis y manipulación de texto.
- Diseño del generador de inflexiones: Planificar y diseñar la arquitectura del generador de inflexiones, estableciendo la integración de las librerías examinadas y la estructuración de las funcionalidades para asegurar una ejecución eficiente y precisa.
- Implementación del generador de inflexiones: Desarrollar e implementar el generador de inflexiones de acuerdo con el diseño propuesto, asegurando que pueda generar correctamente las inflexiones de género y número de sustantivos, adjetivos y determinantes, y que sea capaz de adaptar palabras a frases específicas.

Cada uno de estos objetivos contribuirá al desarrollo de un programa sólido y eficaz, lo que facilitará el acceso a la información escrita a través de la lectura fácil y mejorará la comprensión lectora para diversos usuarios.

1.2 Estructura de la memoria

Este documento seguirá una estructura dividida en capítulos:

- Capítulo 1: Introducción: Breve toma de contacto con el tema del que tratará el trabajo y los objetivos y estructura de el mismo.
- Capítulo 2: Estado del arte: Análisis de trabajos previos, referencias de artículos consultados para el estudio del tema y herramientas utilizadas
- Capítulo 3: Planteamiento de la cuestión: Explicación de funcionalidades
- Capítulo 4: Diseño y desarrollo: Diagramas de flujo de las funcionalidades e implementación de estas.
- Capítulo 5: Pruebas: Comprobación de la ejecución de las funcionalidades.
- Capítulo 6: Conclusiones y Análisis de Impacto: Conclusiones a nivel académico y personal, y posible relevancia del TFG en objetivos de desarrollo sostenible
- Capítulo 7: Líneas de Investigación Futura: Posibles proyectos que mejoren el actual.
- Capítulo 8: Bibliografía: Fuentes, artículos, páginas web utilizadas para la realización del trabajo.
- Capítulo 9: Anexo: Enlace al código del proyecto y capturas de pantalla de los resultados de las pruebas del capítulo 5

2. Estado del arte

Este capítulo trata de contextualizar explicando el proceso de preparación para el posterior desarrollo del trabajo. Además, se explicarán ciertos conceptos teóricos básicos para la comprensión del proyecto, finalmente se detallará las herramientas utilizadas en el mismo, desde las que tuvieron un mayor peso hasta las que inicialmente se usaron, pero no formaron parte de la versión final del trabajo.

2.1 Trabajos previos

Este apartado narra el proceso de preparación seguido para abordar el trabajo, desde la investigación del concepto de Lectura Fácil, hasta el estudio de formación de inflexiones de género y número y de ciertas bibliotecas relacionadas con esto para llevar a cabo las dos funcionalidades principales.

Cabe mencionar implementaciones previas similares que fueron revisadas con el fin de obtener cierta información. IAFlex es un flexionador nominal inteligente elaborado por Daniel Josué Pérez Melián en su TFG [4], consiste en una aplicación web capaz de obtener todas las flexiones de una palabra dada. Para la elaboración de ese proyecto se usó un servicio WCF[1] que permite obtener información morfológica, el cuál es accesible sólo para las personas de la Universidad de Las Palmas de Gran Canaria, por lo que se tuvo que buscar otra alternativa, la cual fue spaCy.

Otra implementación a destacar es el Flexionador Tip del Instituto Universitario de Análisis y Aplicaciones Textuales [5], es un proyecto en desarrollo, su finalización se estima a finales de 2024, el cual genera distintas flexiones dependiendo de la categoría gramatical de la palabra, en sustantivos llega a generar formas despectivas, diminutivos o aumentativos si es que la palabra los tiene, en adjetivos genera las mismas formas que en sustantivos agregando el grado superlativo, y en determinantes flexiona género y número.

Para comenzar el proyecto, se buscó información sobre la Lectura Fácil, para así poder tener un contexto sobre el que trabajar, y entender más allá el objetivo del trabajo. Adicionalmente, se llevó a cabo un estudio de varios artículos recomendados por mi cotutor que explicaban ciertas reglas de formación del femenino y del plural de sustantivos y adjetivos.

Relacionado con la inflexión de número, nos basamos en una librería de GitHub [6] que utilizaba reglas bastante generales para la formación de los plurales en sustantivos y adjetivos para el español, la cual fue ligeramente modificada y adaptada a mi necesidad. A su vez también se analizó un artículo de la RAE que recopilaba reglas de formación del plural [7], para conseguir abarcar todos los casos posibles.

Para el estudio de las inflexiones de género en sustantivos y adjetivos, encontramos un artículo de la RAE con información sobre la formación del femenino en sustantivos relacionados con profesiones [8], pero las reglas que usaba se podían extrapolar a casos más generalizados, por lo que fue de gran ayuda.

El siguiente paso consistió en familiarizarse con el lenguaje de programación Python, esencial para el desarrollo del proyecto. Además, se investigaron diversas bibliotecas y herramientas disponibles que podrían facilitar el procesamiento del lenguaje natural en español, como spaCy y spanish_inflections.

2.2 Sustantivos, adjetivos, determinantes y flexiones

Este apartado da información teórica sobre el marco en el que se ha trabajado, concretamente sobre sustantivos, adjetivos y determinantes y sus respectivas flexiones, detallando casos extraños o distintos de la flexión de los mismos.

2.2.1 Sustantivos

Este apartado da información teórica sobre los sustantivos o nombres.

Un sustantivo es una categoría gramatical o clase de palabra que se utiliza para nombrar un objeto, sujeto, lugar, concepto.

Los sustantivos siempre poseen género gramatical, pudiendo ser masculino o femenino, no existen sustantivos de género neutro, aunque existen sustantivos con forma de género común, que dependiendo del contexto se considera si es masculino o femenino. Por ejemplo, en la frase: "*el artista terminó su gira"* el sustantivo artista se considera masculino puesto que el determinante que le acompaña es masculino, con la frase: *"la artista terminó su gira"*, la palabra artista se considera de género femenino puesto que el determinante que le acompaña es de género femenino.

Cabe destacar que en los casos que los sustantivos son variables en género, es decir que poseen forma femenina y forma masculina, siempre son variables en número, poseen forma singular y forma plural.

Más adelante se explicarán las reglas generales de flexión de sustantivos y ciertos casos extraños

2.2.2 Adjetivos

Este apartado da información teórica sobre los adjetivos.

Los adjetivos son un tipo de palabra que complementa al sustantivo y suministra más información acerca de él, puntualizando cualidades generales, detallando características particulares que por su naturaleza están inseparablemente unidas, o bien delimitando su alcance.

Es importante hacer hincapié en que los adjetivos al modificar al sustantivo deben coincidir en género y número con este, ya que esto tiene un gran peso en la implementación de la funcionalidad de adaptar una palabra en una frase.

A veces el adjetivo no se coloca inmediatamente delante o detrás del sustantivo. En estos casos es muy fácil confundirlo con un adverbio.

2.2.3 Determinantes

Este apartado da información teórica sobre los determinantes.

Son palabras que acompañan a un sustantivo y concretan su significado. Son palabras variables, concuerdan en género y número con el sustantivo que acompañan, y si tiene un adjetivo, también con él. Siempre van delante de un sustantivo, aunque hay algunos determinantes que se usan detrás, en ese caso se les llama 'adjetivos determinativos', que se considera un término medio entre la función de adjetivo y determinante, aunque en el contexto del TFG, spaCy los detecta como determinantes, nunca como adjetivos.

2.2.4 Flexiones

En este apartado se introducirá el término de flexión y se explicaran las reglas comunes de formación de plural y de formación de femenino en sustantivos, adjetivos y determinantes en español.

Una flexión es un proceso mediante el cual se generan distintas formas de una misma palabra en función de diversas informaciones gramaticales (género, número, persona, etc.). Las flexiones nunca alteran la clase de palabra (nombre, adjetivo, verbo, etc.), solo cambia su forma a otro género/numero.

Estas son las reglas generales de la formación del plural en sustantivos y adjetivos que se han seguido durante el desarrollo del proyecto [6]:

- Sustantivos y adjetivos terminados en -ch forman el plural añadiendo -es. Ejemplos: "sándwich→sándwiches".
- Sustantivos y adjetivos terminados en consonantes distintas de -l, -r, -n, -d, -z, -j, -s, -x, -ch forman el plural añadiendo -s. Ejemplos "mamut→mamuts", "comic→comics".
- Sustantivos y adjetivos terminados en -l, -r, -n, -d, -z, -j. Si no van precedidas de otra consonante forman el plural añadiendo -es. Ejemplos: "color→colores", "pan→panes".
- Sustantivos y adjetivos terminados en -i o en -u tónicas forman el plural añadiendo -es. Ejemplos: "bisturí→bisturíes", "tabú→tabúes".
- Sustantivos y adjetivos terminados en vocal átona o en -e tónica forman el plural añadiendo -s. Ejemplos: "papá→papás", "bebé→bebés".
- Sustantivos y adjetivos terminados en -y precedida de vocal forman el plural añadiendo -es. Ejemplos: "ley→leyes", "rey→reyes".
- Sustantivos y adjetivos terminados en -s o en -x forman el plural añadiendo -es. Ejemplos: "fax→faxes", "tos→toses".

En cuanto a la formación del femenino en sustantivos y adjetivos, se ha tenido en cuenta un artículo de la rae cuyas normas eran aplicadas a sustantivos relacionados con profesiones y actividades humanas, pero tras un largo estudio de la formación del femenino se percibió que existían demasiadas excepciones que las reglas de este articulo no contemplaban. Por lo que se implementó una funcionalidad que buscaba el femenino de un sustantivo o adjetivo dado en la RAE, la cual en primera instancia funcionaba a la perfección, pero más tarde se descubrió que no se podía estar continuamente realizando peticiones ya que las bloqueaban.

Tras mucho tiempo analizando la situación se descubrió que en la librería spanish_inflections (librería que funciona mediante amplias listas de palabras en castellano) existía un método llamado fix_word el cual podía suplir esta carencia de formación del femenino, por lo que se decidió ampliar las listas con ciertas excepciones para su que su funcionamiento fuera lo más completo posible.

Para las flexiones de género y de número de determinantes no se encontró ningún artículo que destacará reglas generales como las de sustantivos y adjetivos vistas anteriormente, por lo que se aprovechó la funcionalidad fix_word para tratar de conseguir generar las correctas flexiones de este tipo de palabras.

2.2.5 Casos especiales de género y número

En este apartado se describen casos extraños de flexión de género de flexión de número, los cuales tienen mucha relevancia en el proyecto.

Casos especiales de flexión del género:

- **Heterónimos**: Sustantivo que se opone en género a otra de distinta raíz. Así, se dice que *caballo* y *yegua* son heterónimos, o que entre estas palabras se da una relación de heteronimia. La heteronimia no es frecuente en español, ya que las flexiones de género suelen establecerse con recursos morfológicos, en lugar de léxicos. Así, en la mayor parte de los casos la alternancia masculino-femenino se establece mediante una o más letras añadidas a una misma raíz, como ocurre con niño/niña. Ejemplos: "*toro, vaca*" "*hombre, mujer*"
- **Mismo lexema para ambos géneros**: Palabras que mantienen la misma raíz o base léxica independientemente de su género. En el caso de los sustantivos se les denomina comunes en cuanto al género. Ejemplos: "el/la artista", "el/la periodista", "el pelo azul, la melena azul", "el perro grande, la jirafa grande".

Casos especiales de flexión de número extraídos de un artículo de la UCM [9]:

- Sustantivos que sólo tienen forma singular (**singularia tantum**): la sed, el zodíaco, el cenit, la salud, el caos.

- Sustantivos que sólo tienen forma plural (**pluralia tantum**): bártulos, víveres, enseres, nupcias, entendederas, honorarios, etc.

- **Sustantivos invariables** (llanas o esdrújulas acabadas en –s, o en –x: la/las tesis, el/los tórax, la/las crisis, la/las caries, etc.

En concreto en este proyecto cobran mucha importancia los casos de singularia y pluralia tantum, ya que a la hora de añadir un sustantivo de este tipo a una oración es posible que necesites adaptar también el número del verbo ya que estos sustantivos no poseen más que una forma de número (singular o plural).

2.3 Etiquetas EAGLES

El analizador morfológico para el castellano utiliza una serie de etiquetas con el propósito de representar la información morfológica de las palabras de todas las lenguas europeas, denominadas como etiquetas EAGLES [10].

Estas etiquetas varían su contenido dependiendo de la categoría gramatical de la palabra a etiquetar.

La forma general de las etiquetas es esta:

ETIQUETAS			
Posición	**Atributo**	**Valor**	**Código**
Columna 1	Columna 2	Columna 3	Columna 4

Tabla 1. Estructura etiquetas EAGLES

En la primera columna se encuentra un número que indica la secuencia y posición de los atributos. La segunda columna corresponde a los nombres de los atributos. La tercera columna muestra los posibles valores que cada atributo puede tomar. Finalmente, la cuarta columna contiene los códigos asignados para representar dichos valores. Las etiquetas en cuestión son solo los códigos (columna 4), y se puede identificar a qué atributo pertenecen mediante su posición en la primera columna.

Es importante conocer el comportamiento de estas etiquetas ya que la librería spanish_inflections las utiliza modificando un poco algunas tablas por lo que puede variar ligeramente las formas de las etiquetas con las convencionales, a continuación, se detallaran los cambios en las tablas de sustantivos, adjetivos, determinantes y verbos, ya que son las categorías gramaticales importantes en este proyecto:

- Sustantivos: Elimina el atributo de caso y de género semántico y añade el atributo neclass (clase de sustantivo) con los posibles valores S: persona, G: localización, O: organización y V: otro., aunque nunca suele recibir valor, por lo que es prácticamente igual a la original.
- Adjetivos: Añade al atributo tipo los valores O: ordinal y P: posesivo, al atributo grado le añade los valores S: superlativo y V: evaluativo y elimina el valor A: apreciativo que está en la tabla de etiquetación EAGLE.
- Determinantes: En el caso de los determinantes, el etiquetado difiere al EAGLE convencional, puesto que en este los artículos son tratados como determinantes, mientras que en el EAGLE existen dos tablas de etiquetado distintas, una para artículos, y otra para el resto de los determinantes.
- Verbos: Por último, tendríamos la forma de etiquetar a los verbos, que es prácticamente igual a la EAGLE pero añade el tiempo C: condicional eliminándolo del modo, agrega el género C: común y el tipo S: semiauxiliar.

Cabe destacar que el lema (forma base) de las palabras es idéntico al de las etiquetas EAGLES mencionadas anteriormente

Aquí se muestran distintos ejemplos de la forma de etiquetar palabras de la librería spanish_inflections usando su propio etiquetado que funcionalmente es prácticamente igual al EAGLE:

Palabra	Lema	Etiqueta
hermosa	hermoso	AQ0FS00
los	el	DA0MP0
viviendas	vivienda	NCFP000
perro	perro	NCMS000
comerás	comer	VMIF2S0
peleó	pelear	VMIS3S0

Tabla 2. Ejemplos etiquetación spanish_inflections.

2.4 Herramientas utilizadas

En esta sección se hablará desde el conjunto de herramientas que desde un principio se valoraron como posibles candidatas e incluso se implementaron para la ejecución del trabajo, pero que finalmente no fueron usadas por ciertos problemas, hasta las herramientas que fueron imprescindibles para el correcto funcionamiento de las funcionalidades y del proyecto en sí.

2.4.1 Python

Python es un lenguaje de programación de alto nivel, interpretado y orientado a objetos. Es uno de los lenguajes más populares en la actualidad, particularmente en campos como la inteligencia artificial y el desarrollo web. Esto se debe a su capacidad para manejar estructuras de datos avanzadas, junto con su sintaxis clara y flexible, lo que lo hace ideal para desarrollos rápidos y eficientes.

La importancia de Python en este proyecto es alta ya que es un lenguaje de programación que ofrece bibliotecas especializadas en el procesamiento del lenguaje natural como pueden ser NLTK y spaCy, que facilitan el procesamiento y el análisis del texto.

2.4.2 SpaCy

Para el procesamiento de textos utilizaremos la biblioteca spaCy[3]. SpaCy es una biblioteca gratuita de código abierto para el procesamiento avanzado del lenguaje natural en python la cual tiene pipelines entrenadas para numerosos idiomas, entre ellos el español.

Es muy útil en el contexto de este proyecto, al disponer de funcionalidades como el etiquetado gramatical, la tokenización, la lematización, etc…

Resumen de funcionalidades principales usadas en el proyecto:

Nombre	Descripción
Tokenización	Segmenta el texto en palabras, signos de puntuación, etc. Los convierte en tokens, los cuales poseen distintos atributos que contienen información sobre ellos.

Tabla 3. Función tokenización spaCy.

Una vez aplicada la tokenización, los tokens tienen distintos atributos:

Atributo	Descripción
i	Índice del token en la frase.
tag_	Categoría gramatical del token (sustantivo, adjetivo, verbo…).
morph	Morfología del token, como el género y número.
morph.get(“”)	Devuelve la información morfológica que introduzcas entre “”, como género, número o tiempo, persona, modo dependiendo de la categoría gramatical.
children	Hijos del token, es decir las palabras que dependen gramaticalmente de la palabra que contiene el token.
ancestors	Padres del token: las palabras de las que depende gramaticalmente la palabra que contiene el token.
text	Devuelve el contenido del token
lemma_	Devuelve la forma base de la palabra que contiene el token.

Tabla 4. Atributos token spaCy.

SpaCy posee muchas más funcionalidades como clasificación de textos, capacitación, semejanza, y los tokens tienen más atributos, como dep_, head, etc... Se ha limitado a introducir las funcionalidades y los atributos de los tokens que más uso se ha dado durante el desarrollo del proyecto.

2.4.3 Spanish_inflections

Spanish_inflections [2] es una biblioteca basada en grandes listas de sustantivos, adjetivos, determinante y verbos en español, clasificados con una pequeña variante de etiquetación EAGLE explicada anteriormente en este capítulo. Cada entrada de las listas sigue el siguiente formato:

Palabra	Lema	Código
Palabra	Forma base de la palabra	Etiqueta

Tabla 5. Forma general de las entradas de las listas de spanish_inflections.

Por ejemplo:

Palabra	Lema	Código
Guapa	Guapo	AQ0FS00

Tabla 6. Ejemplo entrada de lista spanish_inflections.

La función principal de esta librería es otorgar información sobre adjetivos y sustantivos y sobre sus flexiones de género y número.

Estas listas eran bastante completas, pero en el caso de la de los sustantivos, faltaban ciertos casos de heteronimia y de pluralia y singularia tantum, los cuales se añadieron durante el transcurso del proyecto.

Las funcionalidades de esta biblioteca son amplias, aun así, se modificaron ciertos comportamientos de las funciones y se agregaron otras:

Nombre	Descripción
rules_adjetive	Reglas que obtiene del archivo de texto que posee la lista de adjetivos en el formato explicado
rules_noun	Reglas que obtiene del archivo de texto que posee la lista de sustantivos en el formato explicado anteriormente
rules_tanc	Reglas que obtiene del archivo de texto que posee la lista de determinantes en el formato explicado anteriormente

rules_verb	Reglas que obtiene del archivo de texto que posee la lista de verbos en el formato explicado anteriormente
Search_rule(rules,Word)	Función que devuelve la palabra, el lemma y el código de "Word" introduciendo las reglas a las que pertenece.
Search_word(rules,lemma,code)	Función que retorna la palabra relacionada con las reglas, forma base y código introducidos.
Search_noun(noun)	Función que retorna la palabra introducida, y sus flexiones de número en caso de que sea sustantivo, si no lo es retorna None.
Search_adjetive(adj)	Función que retorna la palabra introducida, y sus flexiones de número en caso de que sea adjetivo, si no lo es retorna None. Esta funcionalidad no existe en la biblioteca, fue implementada dada su necesidad.
Basic_noun_data(noun)	Función que devuelve el género y el número de la palabra en el caso de que sea un sustantivo, si no lo es retorna None.
Basic_adj_data(adj)	Función que devuelve el género y el número de la palabra en el caso de que sea un adjetivo, si no lo es retorna None. Esta función no se encontraba en la biblioteca inicialmente, se implementó posteriormente tras necesitarla.
fix_verb(rules,word,gender,number)	Función que retorna una flexión de un verbo en relación con el contenido de los parámetros "gender" y "number".
fix_word(rules,word,gender,number)	Función que retorna una flexión de un sustantivo, adjetivo o determinante en relación con el contenido de los parámetros.

Tabla 7. Funcionalidades spanish_inflections.

2.4.4 LanguageTool API

En el español, las flexiones de número suelen seguir ciertas reglas comunes a todos o casi todos los sustantivos y adjetivos. Sin embargo, hay ocasiones en los que la formación del plural de una palabra se sale de estas normas generales, como puede ser el caso de las palabras que terminan en sílaba tónica, que en unos casos la tilde desaparece en la formación del plural y en otros no.

Al pluralizar las palabras con estas reglas generales, este caso que acabo de mencionar era problemático puesto que la tilde cuando debía desaparecer seguía ahí, por eso se utilizó la API de LanguageTool[11].

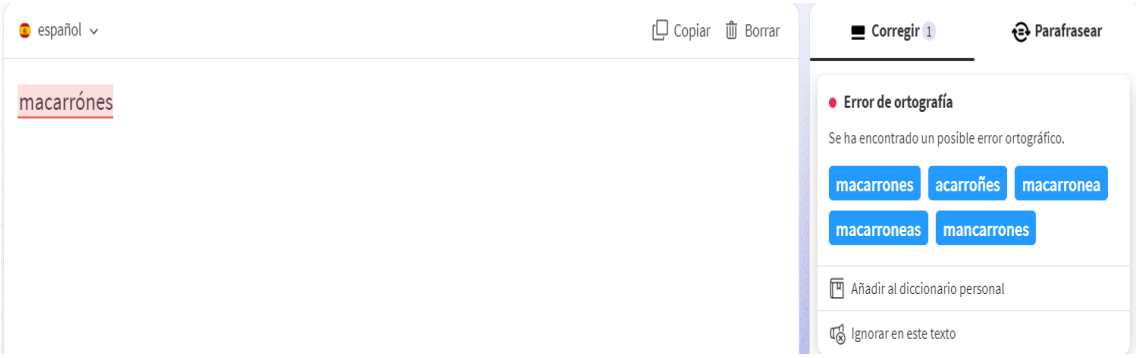

Ilustración 1. Ejemplo de corrección LanguageTool

LanguageTool es un asistente de escritura inteligente, que detecta y corrige errores gramaticales y de ortografía.

La API se puede usar con cualquier lenguaje de programación, ya que se basa en HTTP y devuelve datos JSON de estructura simple, con la limitación de que solamente soporta 20 peticiones de corrección por minuto.

La conexión con la API se ha realizado mediante una función en Python a la que se le pasa por parámetros la palabra a corregir y el idioma, se realiza una solicitud post a la API mediante su url y los parámetros establecidos anteriormente.

Si esta solicitud ha sido exitosa se convierte la respuesta en formato JSON, si no hay coincidencias significa que la palabra pasada por parámetro está bien escrita y se retorna esa misma palabra, si hay coincidencias se extraen las sugerencias de corrección y se retorna la primera sugerencia de la lista.

2.4.5 Herramientas Contempladas

Este apartado resume brevemente las bibliotecas o herramientas que se llegaron a implementar en un principio pero que finalmente fueron sacadas del proyecto:

Scrapping Diccionario de la Lengua Española de la RAE: El Diccionario de la lengua española (DLE) cuenta con la colaboración de las 23 academias de la lengua española existentes en todo el mundo, lo que hace de esta obra lexicográfica una referencia panhispánica. El propósito es recoger el léxico general utilizado en España y en los demás países hispanohablantes y se dirige, principalmente, a aquellas personas cuya lengua materna es el español, quienes encontrarán en él recursos suficientes para descifrar textos escritos y orales.

La Real Academia Española ofrece en internet la vigesimotercera edición del DLE [12]. En un principio se contempló como una buena opción para formar las flexiones de género realizar consultas a la página web del Diccionario de la Lengua Española RAE y extraer la información. Tras implementarlo y comprobar su funcionamiento, al hacer tantas peticiones terminó por dejar de funcionar devolviendo un error en cada consulta, por lo que se tuvo que modificar la forma en la que se obtenían las flexiones de género de las palabras.

```
Error al realizar la solicitud HTTP: 403 Client Error: Forbidden for url: https://dle.rae.es/perro
Error al realizar la solicitud HTTP: 403 Client Error: Forbidden for url: https://dle.rae.es/gato
Error al realizar la solicitud HTTP: 403 Client Error: Forbidden for url: https://dle.rae.es/mesa
Error al realizar la solicitud HTTP: 403 Client Error: Forbidden for url: https://dle.rae.es/puerta
```

Ilustración 2. Error en petición al diccionario de la RAE

Pattern: La biblioteca pattern de Python [13] es una herramienta versátil para trabajar con tareas de procesamiento del lenguaje natural (NLP). Proporciona funcionalidades para tareas como minería de textos, aprendizaje automático y análisis de redes. Pattern tiene un módulo pattern.text.es el cual contiene un etiquetador gramatical de texto en español, funcionalidad que ya estaba cubierta por spaCy, y herramientas para la conjugación de verbos lo cual era la razón por la que instalé esta biblioteca.

El léxico en el que se basan estas herramientas de conjugación contiene unos 600 verbos comunes, y para los verbos no incluidos en este léxico se recurre a un enfoque basado en reglas con una precisión del alrededor del 80%.

Sin embargo, pattern es una biblioteca principalmente enfocada al inglés y los verbos en español tienen distintas formas plurales para cada persona y poseen formas adicionales para el tiempo y para el modo que los verbos en inglés, por lo que la precisión no fue suficiente y se recurrió a otro enfoque para la correcta conjugación de verbos, la función fix_verb de la biblioteca spanish_inflections, explicada anteriormente en este mismo capítulo.

3. Planteamiento del problema

En este apartado se explicará cuáles son las funcionalidades a implementar para añadir contexto previo a la explicación de su diseño e implementación.

3.1 Primera Funcionalidad

En primera instancia, esta funcionalidad se planteó de tal forma que recibiera como parámetro un sustantivo, un adjetivo o un determinante, un género (Masculino/Femenino) y un número (Singular/Plural) y retornara la flexión de la palabra recibida acorde al género y número recibidos. Ejemplos:

Parámetros (Separados por +)	Resultado
"Rojo" + Femenino + Plural	“Rojas”
"Actrices" + Masculino + Singular	“Actor”
“El” + Masculino + Plural	“Los”

Tabla 8. Primera versión de la primera funcionalidad.

Finalmente, tras consultar con los tutores decidimos que la funcionalidad final recibiría como parámetro una palabra, ya sea sustantivo, adjetivo o determinante y retornaría todas las variaciones existentes de esa palabra en el orden [“Masculino Singular”, “Femenino Singular”, “Masculino Plural”, “Femenino Plural”]. Ejemplos:

Parámetros	Resultado
“Rojo”	[“Rojo”, “Roja”, “Rojos”, “Rojas”]
"Actrices"	[“Actor”, “Actriz”, “Actores”, “Actrices”]
“El”	[“El”, “La”, “Los”, “Las”]

Tabla 9. Versión final de la primera funcionalidad.

Tras este retoque a la funcionalidad surgían distintos casos a contemplar en relación con la flexión de palabras:

- Palabras variables en género y número con distintas formas de género: [“Rojo”, “Roja”, “Rojos”, “Rojas”]
- Palabras variables en género y número con mismas formas de género: [“Artista”, “Artista”, “Artistas”, “Artistas”]
- Palabras invariables en género y variables en número: [“Zapato”, “”, “Zapatos”, “”]
- Palabras invariables en género y número: [“”, “”, “Víveres”, “”]

3.2 Segunda Funcionalidad

Esta funcionalidad trata de adaptar una palabra (sustantivo) a una frase de forma que la frase adopte el género de la palabra a introducir, y la palabra a introducir se adapte al número de la frase que dicta el verbo.

Existen casos extraños como los singularia tantum y pluralia tantum cuyo número no puede variar, solo existen en singular o en plural respectivamente. En estos casos sería necesario modificar el verbo, ya que no habría otra forma de adaptar la palabra a la frase, puesto que si difiere en número al ser pluralia o singularia tantum no se puede flexionar.

Tras tiempo de estudio sobre esta funcionalidad, se llegó a la conclusión de la posibilidad de tres casos diferenciados:

- Palabra sustituta concuerda en género y número con la palabra a sustituir.
- Palabra sustituta no concuerda en género con la palabra a sustituir.
- Palabra sustituta modifica el número de la frase.

Caso de palabra sustituta concuerda en género y número:

La funcionalidad recibe una frase, la palabra a sustituir y la palabra sustituta y retorna la frase con la palabra adaptada. Ejemplos:

Parámetros	Resultado
"La mesa es roja" + "mesa" + "silla"	"La silla es roja"
"Las celebridades fueron a la gala" + "celebridades" + "mujeres"	"Las mujeres fueron a la gala"

Tabla 10. Segunda funcionalidad primer caso.

En este caso, al tener concordancia en género y número entre la palabra a sustituir y la sustituta, el único paso a realizar es la sustitución de una por otra, ya que entra directamente en concordancia con la frase.

Caso de palabra sustituta no concuerda en género y número:

La funcionalidad recibe una frase, la palabra a sustituir y la palabra sustituta y retorna la frase con la palabra adaptada. Ejemplos:

Parámetros	Resultado
"La mesa es roja" + "mesa" + "escritorios"	"El escritorio es rojo"
"Las celebridades fueron a la gala" + "celebridades" + "famoso"	"Los famosos fueron a la gala"

Tabla 11. Segunda funcionalidad segundo caso.

En este caso, la concordancia en número es indiferente puesto que las palabras se van a adaptar al número de la frase, pero al no concordar en género se deben modificar todas las palabras que estén relacionadas gramaticalmente con la palabra a sustituir. En el primer ejemplo se puede observar que se modifican tanto el determinante "la" a "el", como el adjetivo "roja" a "rojo", para conseguir la concordancia con la palabra que entra a la frase. En el segundo ejemplo se modifica simplemente el determinante que acompaña al sustantivo.

Caso de palabra sustituta modifica el número de la frase:

La funcionalidad recibe una frase, la palabra a sustituir y la palabra sustituta y retorna la frase con la palabra adaptada. Ejemplos:

Parámetros	Resultado
"El avituallamiento del barco fue saboteado" + "avituallamiento" + "víveres"	"Los víveres del barco fueron saboteados"
"Sus ansias de poder terminaron llevándole a su propia perdición" + "ansias" + "sed"	"Su sed de poder terminó llevándole a su propia perdición"

Tabla 12. Segunda funcionalidad tercer caso.

Cómo se puede observar, en el primer caso el número de la frase es singular, pero la palabra a introducir es un caso de pluralia tantum (víveres), por lo que se adapta el número de la frase al del sustantivo, cambiando el verbo también.

En el segundo caso ocurre lo contrario, el número de la frase es plural, pero se quiere introducir un singularia tantum (sed), por lo que el número de la frase pasa de plural a singular para concordar con la palabra a introducir.

4. Desarrollo

En este capítulo se detallan los pasos seguidos para lograr el correcto funcionamiento de las funcionalidades explicadas en el capítulo anterior, desde su diseño hasta su implementación.

4.1 Diseño

El proyecto consta de dos funcionalidades, en este apartado se mostrará el diseño seguido en ambas.

4.1.1 Flexionador de palabras

El siguiente diagrama de flujo muestra el diseño de la primera funcionalidad:

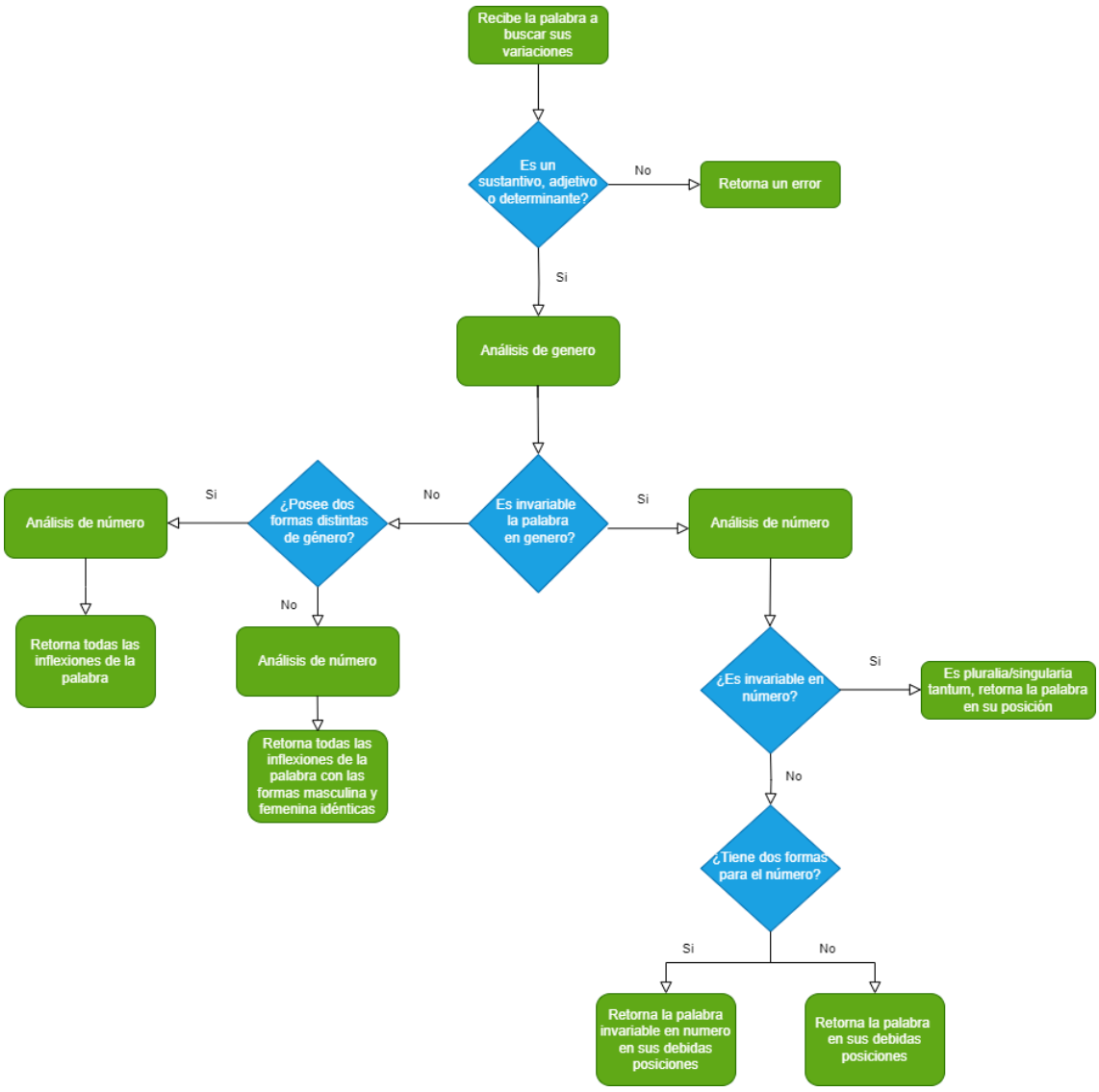

Ilustración 3. Diagrama de flujo primera funcionalidad

Esta funcionalidad solo puede recibir sustantivos adjetivos o determinantes, por lo que el primer paso del algoritmo es filtrar por el tipo de palabra que le llega, si no es ninguno de los mencionados anteriormente, retorna un error.

Una vez pasado el filtro de la categoría gramatical, se procede a realizar el análisis de género, obteniendo las flexiones de género que tiene una palabra. Si la palabra tiene una sola flexión quiere decir que la palabra es invariable en género, en caso de que tenga dos la palabra sería variable en género.

El segundo filtro analiza si la palabra es variable o invariable en género, pudiendo seguir dos caminos:

- La palabra es variable en género: Pasa por el siguiente filtro que analiza si el masculino y el femenino de la palabra son iguales (Masculino: periodista, Femenino: periodista) o distintos (Masculino: niño, Femenino: niña), dando lugar a dos casuísticas:
    - Forma masculina = Forma femenina: Se obtienen las inflexiones de número mediante el análisis de número y se retornan todas las inflexiones de género y numero de la palabra cuyas formas de género son iguales.
    - Forma masculina != Forma femenina: Se obtienen las inflexiones de número mediante el análisis de número y se retornan todas las inflexiones de género y numero de la palabra.

- La palabra es invariable en género: Se realiza el análisis de número, obteniendo las flexiones de número de la palabra, llegando al filtro que divide las palabras invariables en número con las variables:
    - Es invariable en número: Al ser invariable en género y en número la palabra es un singularia/pluralia tantum, se retorna la palabra sin variaciones puesto que no tiene, en su lugar correspondiente.
    - Es variable en número: Se observa si la palabra posee dos formas distintas para el singular y para el plural (Singular: perro, Plural: perros) o si la palabra es la misma en ambas formas de número (Singular: crisis, Plural: crisis):
        - Forma Singular != Forma Plural: Retorna la palabra y su flexión de número en sus debidas posiciones.
        - Forma Singular = Forma Plural: Retorna la palabra y su flexión de número (idéntica a ella) en sus debidas posiciones.

A continuación, se muestra los caminos que seguirían las palabras “periodista” y “sed” en este diagrama:

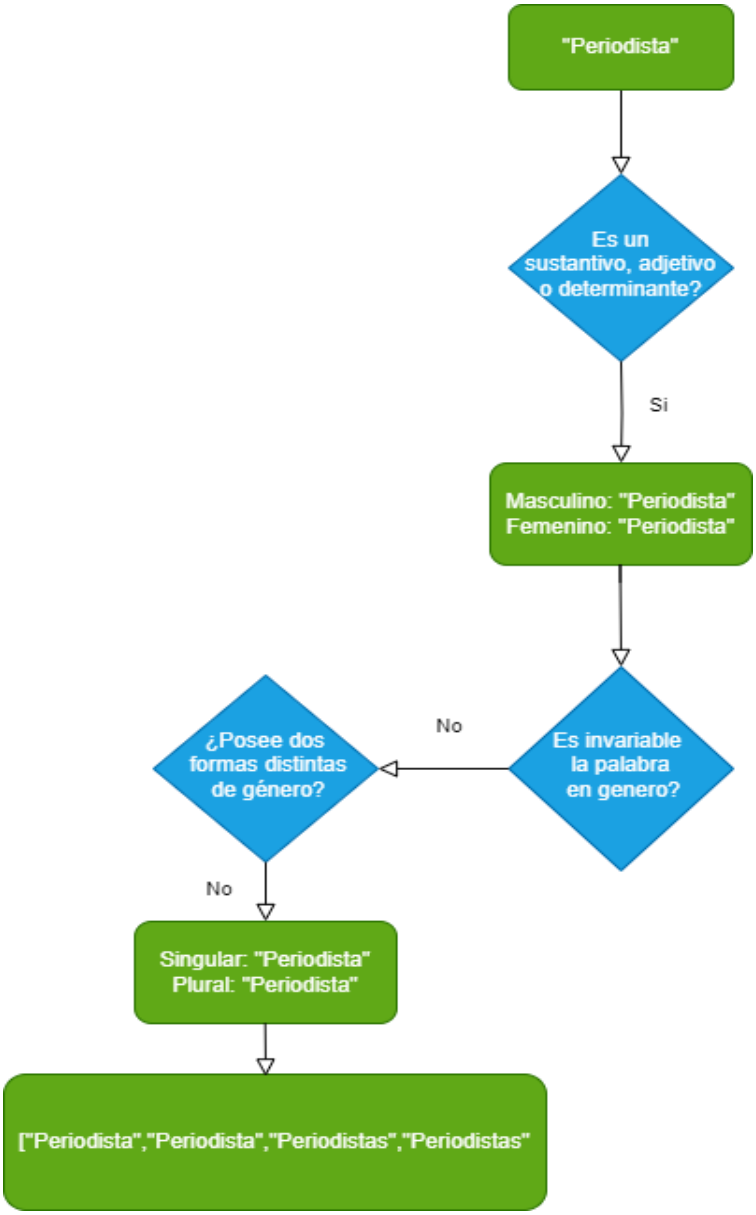

Ilustración 4. Ejemplo 1 primera funcionalidad.

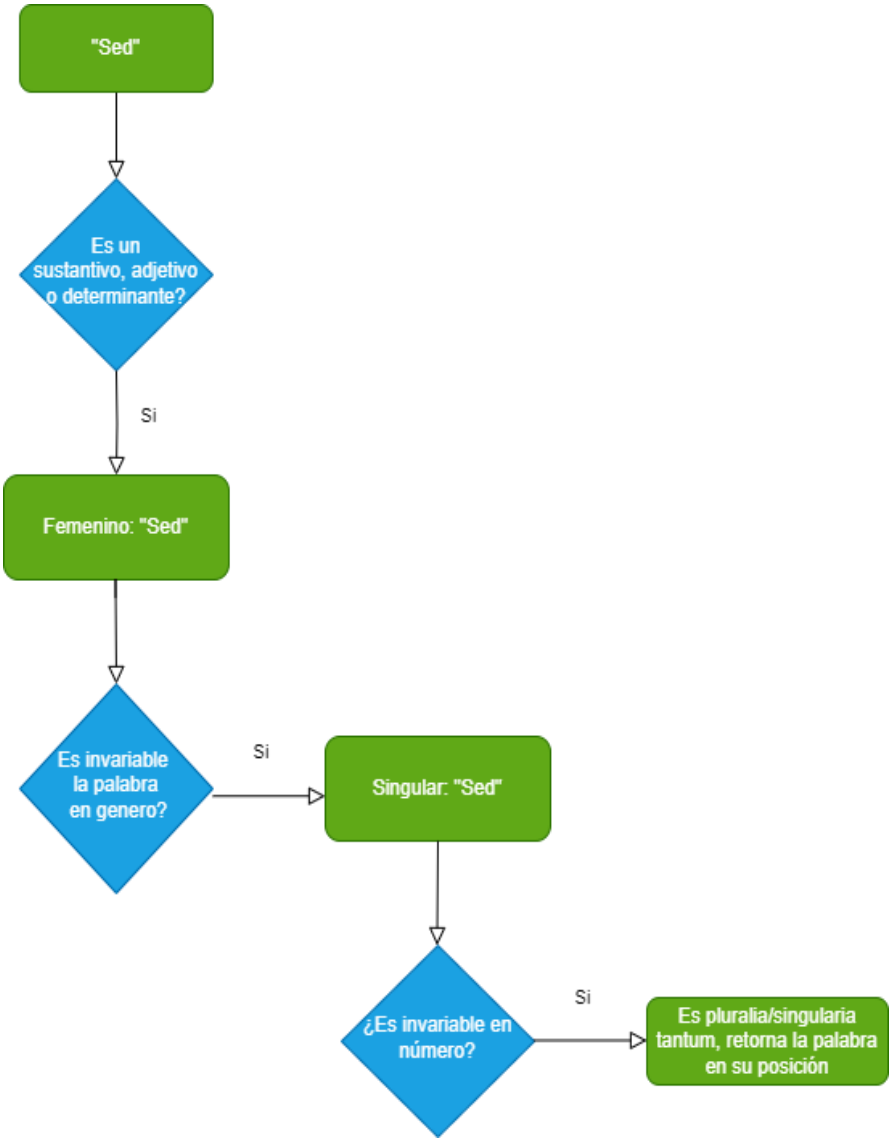

Ilustración 5. Ejemplo 2 primera funcionalidad.

4.1.2 Adaptador de palabra a frase

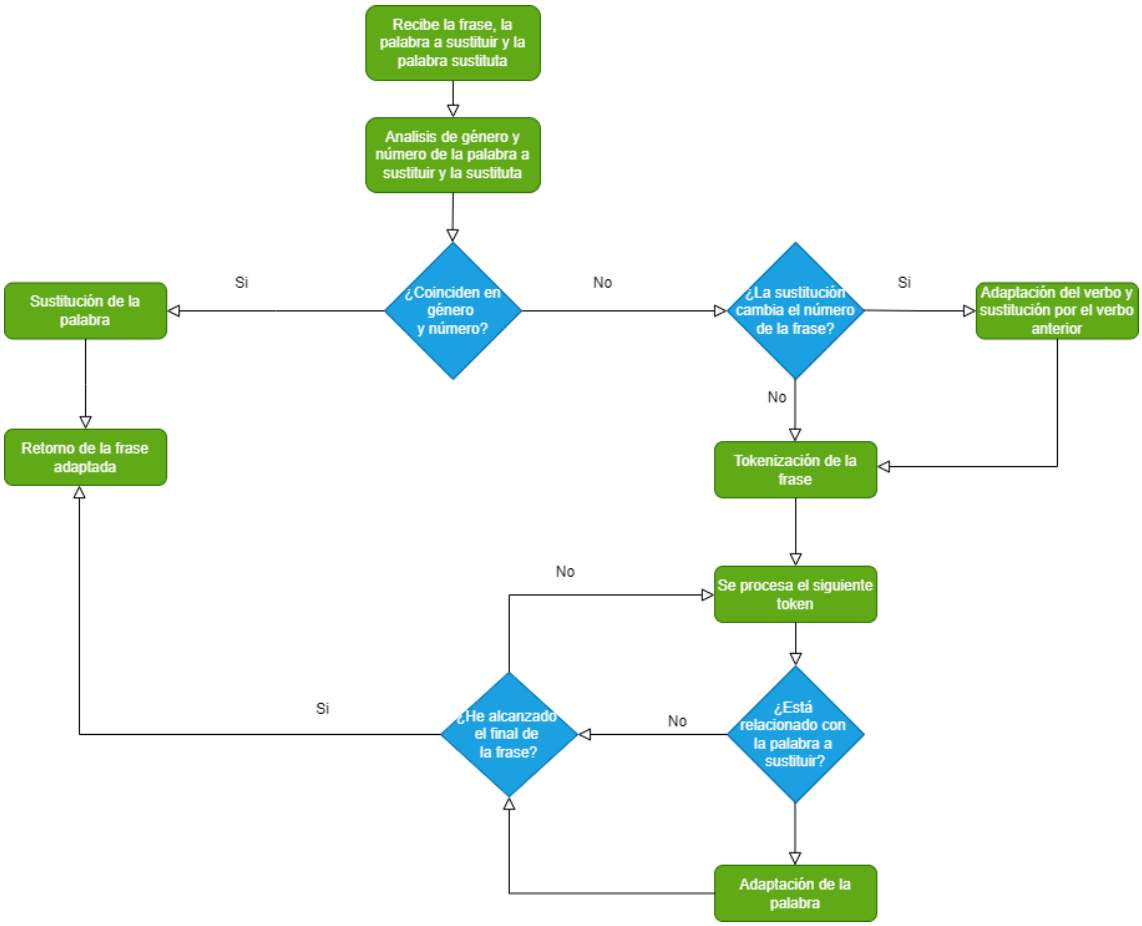

Ilustración 6. Diagrama de flujo segunda funcionalidad.

Paso 1: Esta funcionalidad recibe la frase, la palabra a sustituir y la palabra sustituta, y realiza el análisis de género y de número de ambas, planteando la primera pregunta, ¿Coinciden en género y número?:

Paso 1.1(Final): Coinciden en género y número: Se sustituye la palabra a sustituir por la sustituta y posteriormente se retorna la frase adaptada.

Paso 1.2: No coinciden en género y número: Se observa si el número de la palabra sustituta es variable o invariable, porque en el caso de ser invariable, la sustitución cambiaría el número de la frase por ser singularia/pluralia tantum:

Paso 1.2.1: Sí la sustitución modifica el número de la frase, se adapta el verbo al número de la palabra sustituta y se procede al paso 2.

Paso 1.2.2: Sí la sustitución no modifica el número de la frase, se procede directamente al paso 2.

Paso 2: Se procede a la tokenización de la frase, y se va procesando token a token llegando al siguiente filtro: ¿Está el token relacionado con la palabra a sustituir?:

Paso 2.1: El token está relacionado con la palabra a sustituir: Se adapta la palabra que contiene el token al género de la palabra sustituta y al número de la frase, y pasa al paso 3.

Paso 2.2: El token no está relacionado con la palabra a sustituir: Se pasa directamente al paso 3.

Paso 3: Después de adaptar la palabra contenida en el token, o de comprobar que no estaba relacionado el token con la palabra a sustituir, llegamos al último filtro: ¿He alcanzado el fin de la frase?:

Paso 3.1: No he alcanzado el fin de la frase: Se vuelve al paso 2 después de haber tokenizado la frase, es decir se procesa el siguiente token.

Paso 3.2: Si he alcanzado el fin de la frase: Se retorna la frase adaptada.

A continuación, dos ejemplos que toman caminos distintos en el diagrama:

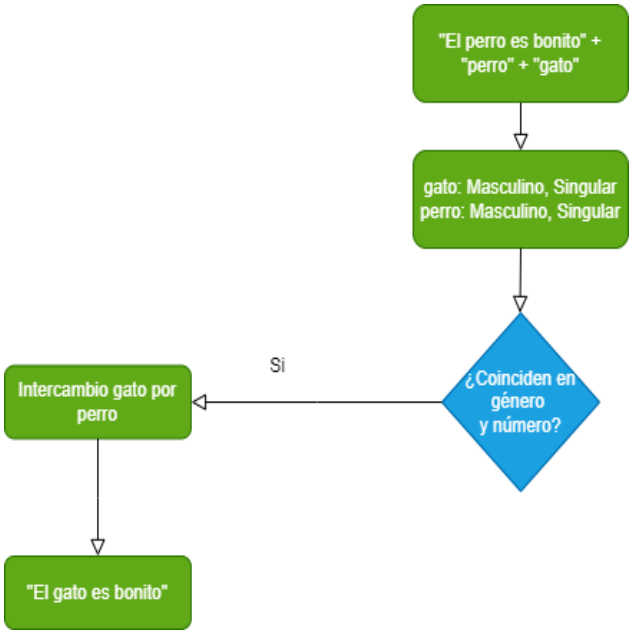

Ilustración 7. Ejemplo 1 segunda funcionalidad.

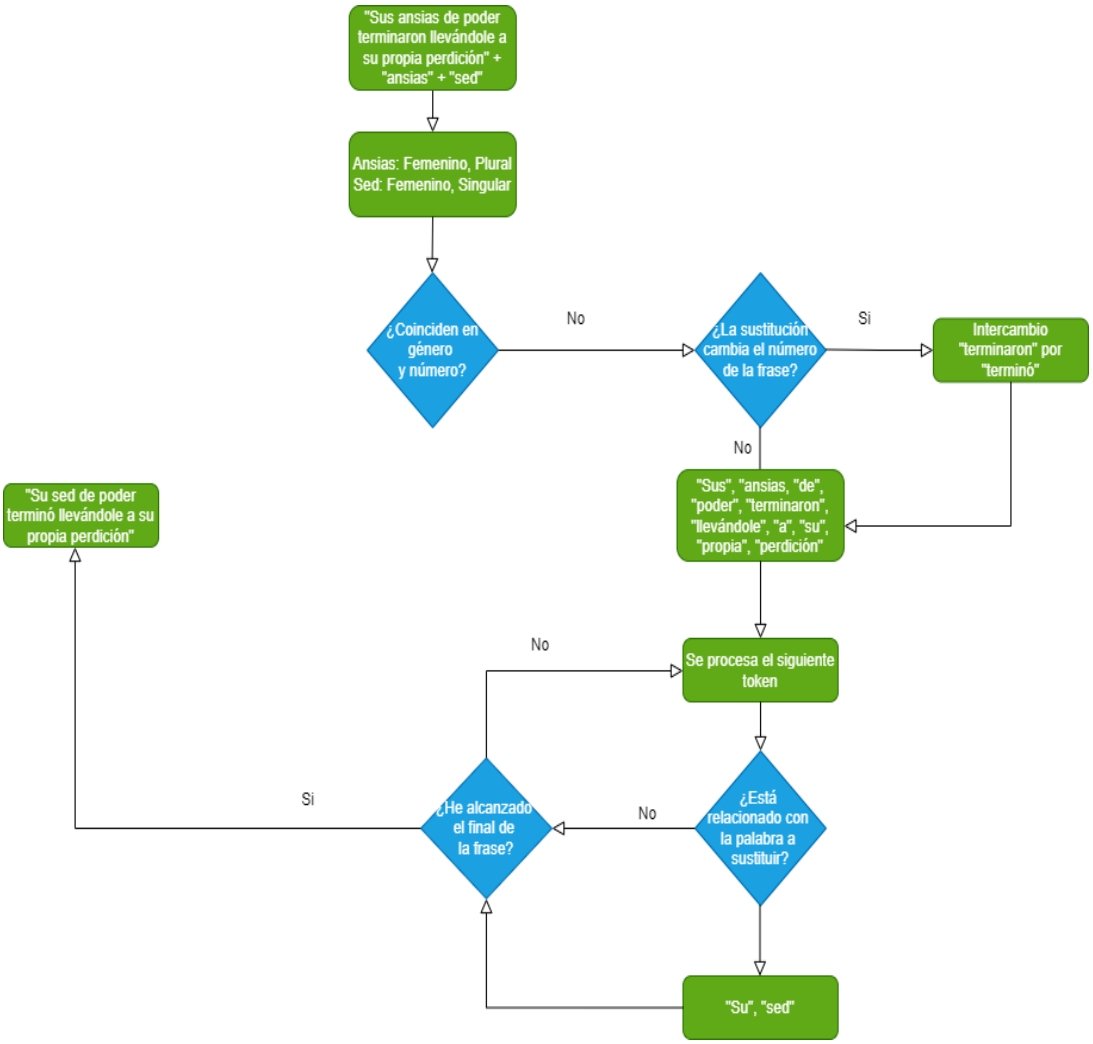

Ilustración 8. Ejemplo 2 segunda funcionalidad.

4.2 Implementación

En este apartado se tratarán las clases, funciones principales y funciones auxiliares utilizadas para la consecución del proyecto.

4.2.1 Clase apiLanguageTool

Esta es la clase en la que se encuentra el método que hace uso de la API de la herramienta LanguageTool explicada en el capítulo 2.4.4

El método principal es check_spelling:

```python
def check_spelling(word, language='es'):
    url = 'https://api.languagetool.org/v2/check'
    params = {
        'text': word,
        'language': language
    }

    response = requests.post(url, data=params)
    if response.status_code == 200:
        result = response.json()
        matches = result.get('matches', [])
        if not matches:
            return word  # La palabra está bien escrita
        else:
            # Devolver la primera sugerencia de corrección encontrada
            corrections = [match['replacements'][0]['value'] for match in matches if 'replacements' in match and match['replacements']]
            return corrections[0] if corrections else word
    else:
        print(f"Error en la solicitud a LanguageTool: {response.status_code}")
        return word
```

Ilustración 9. Método check_spelling.

Esta función recibe como parámetro la palabra a revisar, y un parámetro con valor predeterminado 'es' que es el idioma español.

Se establece la url del servicio de la API de LanguageTool [11], se definen los parámetros para enviar la solicitud POST a la API, estos son:

- 'text': la palabra a revisar
- 'language': idioma en el cual se revisará

Se envía la solicitud POST y se comprueba si la respuesta ha sido exitosa (response.status_code == 200) o si no lo ha sido (else).

Si no ha sido exitosa se imprime un mensaje de error con el código de estado de la respuesta y se devuelve la palabra sin modificar.

Si ha sido exitosa, se procesa la respuesta en formato JSON y se extrae el valor en la variable matches, que devuelve las posibles correcciones si la palabra estaba mal escrita, o no devuelve nada en caso de que estuviera bien escrita.

En caso de que estuviera bien escrita se retorna la palabra sin modificar, en caso contrario se obtiene la primera sugerencia de corrección encontrada y se retorna.

4.2.2 Clase inflexionNumero

Esta clase se centra en la correcta formación de plurales y singulares de palabras dadas.

En primer lugar, tenemos el método is_singular:

```
def is_singular(word):
    if(search_noun(word)!=None):
        if basic_noun_data(word).get("number")  == ['Plur']:
            return False
        else:
            return True
    else:
        if basic_adj_data(word).get("number")  == ['Plur']:
            return False
        else:
            return True
```

Ilustración 10. Método is_singular.

Usando los métodos basic_noun_data y basic_adj_data de la librería spanish_inflections, comprueba si el adjetivo o sustantivo recibido es singular.

También existe el método is_plural:

```
def is_plural(word):
    if(search_noun(word)!=None):
        if basic_noun_data(word).get("number")  == ['Plur']:
            return True
        else:
            return False
    else:
        if basic_adj_data(word).get("number")  == ['Plur']:
            return True
        else:
            return False
```

Ilustración 11. Método is_plural.

La estructura es igual a la de is_singular pero cambiando el retorno de los True y False, esta función no es necesaria puesto que se sabría que una palabra es plural si retorna False en is_singular, pero por cuestiones de claridad de código se implementó.

Método pluralize:

```
def pluralize(word):
    if(is_plural(word)):
        return word
    if word.endswith("ch"):
        return word + "es"
    elif not word.endswith(("l", "r", "n", "d", "z", "j", "y")) and not word.endswith in vocal:
        return word + "s"
    elif word[-1] == "z":
        return word[:-1] + "ces"
    elif  word.endswith(("l", "r", "n", "d", "j")):
        if len(word) > 1 and word[-2] not in consonante:
            return word + "es"
        else:
            return word + "s"
    elif word.endswith(("í", "ú")):
        return word + "es"
    elif word[-1] in "aeiouáéó":
        return word + "s"
    elif word[-1] in consonante and word[-2] in consonante:
        return word + "es"
    elif word.endswith("y") and word[-2] in "aeiouáéíóú":
        return word + "es"
    elif word.endswith(("s", "x")):
        return word + "es"
    else:
        return word + "s"
```

Ilustración 12. Método pluralize..

Este método transforma la palabra que recibe en plural, comprobando previamente que no sea plural ya, lo hace mediante las reglas mencionadas en el capítulo 2.2.3.

Método singularize:

```
def singularize(word):
    if is_singular(word):
        return word
    if word.endswith("ches"):
        return word[:-2]
    elif word.endswith("es"):
            if word[-3] == "c":
                return word[:-3] + "z"
            else:
                return word[:-2]
    elif word.endswith("s"):
            return word[:-1]
    else:
        return word
```

Ilustración 11. Método singularize.

Transforma la palabra a singular comprobando que no sea singular previamente, sigue las mismas reglas de creación del plural, pero a la inversa.

Por último, estos dos métodos que complementan la funcionalidad de singularizar y pluralizar:

```
def pluralizeChecked(word):
    wordPluralized = pluralize(word)
    wordPluralized = check_spelling(wordPluralized)
    return wordPluralized

def singularizeChecked(word):
    wordSingularized = singularize(word)
    wordSingularized = check_spelling(wordSingularized)
    return wordSingularized
```

Ilustración 12. Métodos pluralizeChecked y singularizeChecked.

Su funcionalidad es básica, llaman a singularize y a pluralize y posteriormente pasan el resultado al método check_spelling explicado anteriormente, que comprueba si una palabra está correctamente escrita y si no lo está la corrige, ya que, al formar las flexiones de número mediante reglas generales, hay casos en los que alguna tilde se coloca en una sílaba que no le corresponde.

4.2.3 Clase flexionador

Esta clase contiene el flexionador que implementa las dos funcionalidades principales del proyecto, generar todas las flexiones de palabras dadas y adaptar una palabra a una frase.

En primer lugar, cabe destacar la creación de dos listas que contienen casos de singularia y pluraria tantum que se han ido recogiendo:

```
singularia_tantum = [
    "sed", "salud", "gente", "dinero", "pánico", "agua",
    "aire", "caos", "fama", "amor", "paz", "felicidad", "sufrimiento",
    "odio", "esperanza", "libertad", "justicia", "sabiduría", "fe",
    "miedo", "electricidad", "hambre", "dolor"
]
pluralia_tantum = [
    "tijeras", "gafas", "pantalones", "víveres", "albricias", "enseres",
    "vacaciones", "cenizas", "esponsales", "aduanas", "anales", "exequias",
    "nupcias", "bragas", "catacumbas", "hortalizas", "cosquillas",
    "fauces", "misericordias", "trizas", "vítores", "andas", "modales"
]
```

Ilustración 13. Listas singularia tantum y pluralia tantum.

Estas listas fueron creadas debido a que estos casos cambian la forma en la que actúan las funcionalidades, por lo que era necesario recogerlos.

Las siguientes funciones a analizar son métodos auxiliares cuyo uso facilita la implementación de las funcionalidades principales.

```
def adaptarDeterminante(determinante, numero, genero): #Se le pasa un determinante y el genero y numero al que adaptar ese determinante
    determinante = determinante.lower()
    lemma = search_rule(spanish_inflections.rules_tanc,determinante).get("lemma")
    if (genero == ["Masc"] and numero == ['Sing']) :
        respuesta = spanish_inflections.fix_word(spanish_inflections.rules_tanc,lemma,"M","S")
    elif(genero == ["Masc"] and numero == ['Plur']) :
        respuesta = spanish_inflections.fix_word(spanish_inflections.rules_tanc,lemma,"M","P")
    elif(genero == ["Fem"] and numero == ['Sing']) :
        respuesta = spanish_inflections.fix_word(spanish_inflections.rules_tanc,lemma,"F","S")
    else:
        respuesta = spanish_inflections.fix_word(spanish_inflections.rules_tanc,lemma,"F","P")
    return respuesta
```

Ilustración 14. Método adaptarDeterminante.

adaptarDeterminante: este método auxiliar recibe como parámetros un determinante, un número y un género y trata de devolver la flexión de ese determinante relacionado con el género y el número recibidos. Su funcionamiento es simple, convierte el determinante a minúsculas para evitar errores, posteriormente busca el lemma (forma base) del mismo puesto que es más seguro flexionar desde la forma base, y una vez sacado el lemma comprueba el género y el número al que flexionar, y lo flexiona mediante el método fix_word de spanish_inflections.

```
def buscarDeterminantesRelacionados(frase, Adjetivo):#devuelve el indice del determinante relacionado a un adjetivo
    mi_array = array('i')
    for token in frase:
        if token.tag_=="DET" and token in Adjetivo.children:#si es un determinante y esta relacionado con el adjetivo
            if (token.i not in mi_array):
                mi_array.append(token.i)
    for token in frase:
        if token.tag_=="DET" and Adjetivo in token.ancestors:#si es un determinante y esta relacionado con el adjetivo
            if (token.i not in mi_array):
                mi_array.append(token.i)
    if(len(mi_array)!=0):
        mi_array_sorted = array('i',sorted(mi_array))
        return mi_array_sorted
    return mi_array
```

Ilustración 15. Método buscarDeterminantesRelacionados.

buscarDeterminantesRelacionados: este método recibe una frase y un token de esa frase (normalmente adjetivo) al que se le quiere buscar determinantes que estén en relación con él, la implementación de este método surge para cerciorar que, en la funcionalidad de adaptar palabras a frases, se flexionen todas las palabras necesarias. La funcionalidad se basa en buscar mediante métodos de la librería spaCy, si existe algún determinante con relación de hijo o de padre con el adjetivo, si existen, se añaden al array los índices en los que aparecen en la frase los determinantes relacionados y se retorna, si no se retorna el array vacío.

```
def cambiarPalabraEnFrase(frase,numeroPalabra,palabraSustituta):
    palabras = frase.split()  # Divide la frase en una lista de palabras
    if numeroPalabra <= len(palabras):
        palabras[numeroPalabra] = palabraSustituta  # Reemplaza la palabra en la posición indicada
        return ' '.join(palabras)  # Une las palabras de nuevo en una frase
```

Ilustración 16. Método cambiarPalabraEnFrase.

cambiarPalabraEnFrase: recibe una frase, el índice de la palabra que se desea sustituir, y la palabra sustituta, divide la frase en palabras, y va buscando cuál es la palabra que se corresponde al índice, una vez encontrada se reemplaza y une de nuevo la frase. Este método es consecuencia del anterior puesto que buscarDeterminantesRelacionados devuelve un array de índices, y este método trata de sustituir las palabras correspondientes a los índices con palabras modificas en género y número coincidentes con el del adjetivo de la función anterior.

masc_y_fem: esta función es de suma importancia puesto que es la que flexiona en género una palabra. Recibe como parámetro una palabra (sustantivo/adjetivo), la intenta singularizar porque es más sencillo trabajar con singulares, comprueba su género y mediante fix_word trata de cambiarla de género. La función devuelve una lista con las flexiones de género existentes de la palabra priorizando que la flexión masculina se encuentre la primera y la femenina la segunda, en caso de ser invariable en género se devuelve la lista con una sola entrada.

flexionesGeneroyNumero: Esta es la primera funcionalidad del proyecto. Recibe una palabra y devuelve todas sus flexiones de género y número existentes en este formato: ['MS: ', 'FS: ', 'MP: ', 'FP: ']. Contempla distintos casos:

- Singularia/pluralia tantum: si la palabra se encuentra en una de estas dos listas se retorna directamente en su posición.
- Determinante: si la palabra es un determinante se haya su género y su número y se sacan las flexiones restantes.
- Sustantivo/adjetivo: si la palabra es un sustantivo o un adjetivo se llama al método masc_y_fem para recibir la lista con sus flexiones de género. Se hacen comprobaciones con el tamaño de la lista puesto que si es 1 la palabra es invariable en género y si es 2 es variable, en el caso de ser 1 el tamaño, se comprueba si la palabra es masculina o femenina, se llama al método pluralize y se retornan la flexión singular y plural en sus posiciones, en el caso de ser 2 se pluralizan con el método y se retornan todas las flexiones en sus posiciones.

adaptarPalabraenFrase: Esta es la segunda funcionalidad del proyecto. Recibe una frase, la palabra a sustituir de esa frase, y la palabra sustituta y trata de adaptar la frase al género de la palabra sustituta y la palabra sustituta al número que dicta la frase, retornando una nueva frase en concordancia con la palabra introducida. Contempla distintos casos:

- Palabra sustituta es singularia/pluralia tantum: en este caso, si el número de la palabra difiere del de la frase, la frase se adapta al número de la palabra llegando a modificar el verbo puesto que estos sustantivos solo poseen una flexión de número.
- Género y número palabra sustituta = género y número palabra a sustituir: en este caso simplemente habría que sustituir la palabra puesto que la frase ya se encuentra en concordancia.

- Género y número palabra sustituta != género y número palabra a sustituir: se buscan las palabras relacionadas con la palabra a sustituir y se modifican para concordar con la palabra a introducir.

Estas funcionalidades son muy extensas por lo que se incluirán en otro archivo.

5. Pruebas

En este capítulo se analizan las pruebas realizadas para ambas funcionalidades del proyecto.

5.1 Pruebas unitarias

Las pruebas unitarias son un proceso en el que se verifica la precisión de bloques aislados de código, en este caso de las dos funcionalidades por separado, en primer lugar, el generador de flexiones de palabras y por último el adaptador de palabras en frases.

Estas pruebas son realizadas con el fin de verificar que las funcionalidades se ejecutan según lo esperado.

5.1.1 Pruebas unitarias primera funcionalidad

Pruebas pertenecientes a la funcionalidad de generar todas las flexiones existentes de una palabra.

En este caso las pruebas se dividirán en distintos bloques para asegurar que cubre todos los casos de flexiones de palabras.

Los resultados serán reflejados en tablas porque son más visuales y fáciles de entender, pero en los anexos de pruebas se encontrará la salida del programa para verificar que los datos de la tabla no son inventados.

A su vez, el programa genera los resultados de esta forma: [‘MS: niño’, ‘FS: niña’, ‘MP: niños’, ‘FP: niñas’], siendo MS masculino singular, FS femenino singular, MP masculino plural y FP femenino plural.

5.1.1.1 Pruebas palabras variables en género y número con formas distintas de género.

En este apartado de pruebas pasaremos a la funcionalidad palabras que disponen de todas las flexiones de género y número.

Utilizaremos esta lista de palabras para las pruebas: "niño perro gata hombre maestras rojo doctores bueno mala pequeñas amarillo el lentas un altos bajas blancos estos" y se le irá pasando palabra a palabra a la funcionalidad.

N.º Prueba	Palabra	Respuesta programa	¿Resultado deseado?
1	niño	['MS: niño', 'FS: niña', 'MP: niños', 'FP: niñas']	Correcto
2	perro	['MS: perro', 'FS: perra', 'MP: perros', 'FP: perras']	Correcto
3	gata	['MS: gato', 'FS: gata', 'MP: gatos', 'FP: gatas']	Correcto
4	hombre	['MS: hombre', 'FS: mujer', 'MP: hombres', 'FP: mujeres']	Correcto
5	maestras	['MS: maestro', 'FS: maestra', 'MP: maestros', 'FP: maestras']	Correcto
6	rojo	['MS: rojo', 'FS: roja', 'MP: rojos', 'FP: rojas']	Correcto
7	doctores	['MS: doctor', 'FS: doctora', 'MP: doctores', 'FP: doctoras']	Correcto
8	bueno	['MS: bueno', 'FS: buena', 'MP: buenos', 'FP: buenas']	Correcto
9	mala	['MS: malo', 'FS: mala', 'MP: malos', 'FP: malas']	Correcto
10	pequeñas	['MS: pequeño', 'FS: pequeña', 'MP: pequeños', 'FP: pequeñas']	Correcto
11	amarillo	['MS: amarillo', 'FS: amarilla', 'MP: amarillos', 'FP: amarillas']	Correcto
12	el	['MS: el', 'FS: la', 'MP: los', 'FP: las']	Correcto
13	lentas	['MS: lento', 'FS: lenta', 'MP: lentos', 'FP: lentas']	Correcto
14	un	['MS: un', 'FS: una', 'MP: unos', 'FP: unas']	Correcto
15	altos	['MS: alto', 'FS: alta', 'MP: altos', 'FP: altas']	Correcto
16	bajas	['MS: bajo', 'FS: baja', 'MP: bajos', 'FP: bajas']	Correcto
17	blancos	['MS: blanco', 'FS: blanca', 'MP: blancos', 'FP: blancas']	Correcto
18	estos	['MS: este', 'FS: esta', 'MP: estos', 'FP: estas']	Correcto

Tabla 13. Pruebas unitarias primera funcionalidad primer caso

5.1.1.2 Pruebas palabras variables en género y número con formas iguales de género.

En este apartado de pruebas pasaremos a la funcionalidad palabras que disponen de todas las flexiones de género y número, pero a diferencia de las pruebas anteriores, tienen la misma forma masculina y femenina.

Las palabras usadas para el conjunto de pruebas son: "su cantante tu artista atleta joven triste policía turista dentista".

N.º Prueba	Palabra	Respuesta programa	¿Resultado deseado?
1	su	['MS: su', 'FS: su', 'MP: sus', 'FP: sus']	Correcto
2	cantante	['MS: cantante', 'FS: cantante', 'MP: cantantes', 'FP: cantantes']	Correcto
3	tu	['MS: tu', 'FS: tu', 'MP: tus', 'FP: tus']	Correcto
4	artista	['MS: artista', 'FS: artista', 'MP: artistas', 'FP: artistas']	Correcto
5	atleta	['MS: atleta', 'FS: atleta', 'MP: atletas', 'FP: atletas']	Correcto
6	joven	['MS: joven', 'FS: joven', 'MP: jóvenes', 'FP: jóvenes']	Correcto
7	triste	['MS: triste', 'FS: triste', 'MP: tristes', 'FP: tristes']	Correcto
8	policía	['MS: policía', 'FS: policía', 'MP: policías', 'FP: policías']	Correcto
9	turista	['MS: turista', 'FS: turista', 'MP: turistas', 'FP: turistas']	Correcto
10	dentista	['MS: dentista', 'FS: dentista', 'MP: dentistas', 'FP: dentistas']	Correcto

Tabla 14. Pruebas unitarias primera funcionalidad segundo caso.

5.1.1.3 Pruebas palabras invariables en género y variables en número.

En este apartado de pruebas pasaremos a la funcionalidad palabras que, a diferencia de las pruebas anteriores, disponen de una sola flexión de género y ambas flexiones de número.

Las palabras usadas para el conjunto de pruebas son: "zapato libro silla mesa ventana carro bicicleta casa sofá coche".

N.º Prueba	Palabra	Respuesta programa	¿Resultado deseado?
1	zapato	['MS: ', 'FS: silla', 'MP: ', 'FP: sillas']	Correcto
2	libro	['MS: libro', 'FS: ', 'MP: libros', 'FP:']	Correcto
3	silla	['MS: ', 'FS: silla', 'MP: ', 'FP: sillas']	Correcto
4	mesa	['MS: ', 'FS: mesa', 'MP: ', 'FP: mesas']	Correcto
5	ventana	['MS: ', 'FS: ventana', 'MP: ', 'FP: ventanas']	Correcto
6	carro	['MS: carro', 'FS: ', 'MP: carros', 'FP: ']	Correcto
7	bicicleta	['MS: ', 'FS: bicicleta', 'MP: ', 'FP: bicicletas']	Correcto
8	casa	['MS: ', 'FS: casa', 'MP: ', 'FP: casas']	Correcto
9	sofá	['MS: sofá', 'FS: ', 'MP: sofás', 'FP: ']	Correcto
10	coche	['MS: coche', 'FS: ', 'MP: coches', 'FP: ']	Correcto

Tabla 15. Pruebas unitarias primera funcionalidad tercer caso.

5.1.1.4 Pruebas palabras invariables en género y en número.

En este apartado de pruebas pasaremos a la funcionalidad palabras que, a diferencia de las pruebas anteriores, son invariables en género y en número, es decir son casos de pluralia tantum y singularia tantum.

Las palabras usadas para el conjunto de pruebas son: "sed tijeras gafas electricidad agua dolor hambre bragas vacaciones amor".

N.º Prueba	Palabra	Respuesta programa	¿Resultado deseado?
1	sed	['MS: ', 'FS: sed', 'MP: ', 'FP: ']	Correcto
2	tijeras	['MS: ', 'FS: ', 'MP: ', 'FP: tijeras']	Correcto
3	gafas	['MS: ', 'FS: ', 'MP: ', 'FP: gafas']	Correcto
4	electricidad	['MS: ', 'FS: electricidad', 'MP: ', 'FP: ']	Correcto
5	agua	['MS: ', 'FS: agua', 'MP: ', 'FP: ']	Correcto
6	dolor	['MS: ', 'FS: dolor', 'MP: ', 'FP: ']	Correcto
7	hambre	['MS: ', 'FS: hambre', 'MP: ', 'FP: ']	Correcto
8	bragas	['MS: ', 'FS: ', 'MP: ', 'FP: bragas']	Correcto
9	vacaciones	['MS: ', 'FS: ', 'MP: ', 'FP: vacaciones']	Correcto
10	amor	['MS: ', 'FS: amor', 'MP: ', 'FP: ']	Correcto

Tabla 16. Pruebas unitarias primera funcionalidad cuarto caso.

5.1.2 Pruebas unitarias segunda funcionalidad

Pruebas pertenecientes a la funcionalidad de adaptar una palabra a una frase.

En este caso las pruebas se dividirán en distintos bloques para asegurar que cubre todos los casos de adaptación de palabras.

Los resultados serán reflejados en tablas porque son más visuales y fáciles de entender, pero en los anexos de pruebas se encontrará la salida del programa para verificar que los datos de la tabla no son inventados.

El programa retorna directamente la frase adaptada.

5.1.2.1 Pruebas palabra sustituta concuerda en género y número con palabra a sustituir

En este apartado se tratarán los casos en los que la palabra a introducir coincide en género y número con la palabra a sustituir, por lo que simplemente hay que intercambiar las palabras.

Se van a crear dos tablas, una relacionada con los parámetros de cada prueba y otra con los resultados, de forma que se vea más visual puesto que en una tabla sola no se entiende bien.

Tabla de parámetros:

N.º Prueba	*Parámetros pasados a la funcionalidad*
1	Frase: 'El perro de mi padre adoptivo es bonito' Palabra a sustituir: 'perro' Palabra sustituta: 'gato'
2	Frase: 'Mi primo pequeño tiene 9 años' Palabra a sustituir: 'primo' Palabra sustituta: 'hermano'
3	Frase: 'A mi amigo le gusta el futbol' Palabra a sustituir: 'amigo' Palabra sustituta: 'padre'
4	Frase: 'Las luces se apagaron' Palabra a sustituir: 'luces' Palabra sustituta: 'bombillas'
5	Frase: 'La fiesta se ha terminado' Palabra a sustituir: 'fiesta' Palabra sustituta: 'reunión'
6	Frase: 'Los leopardos son preciosos' Palabra a sustituir: 'leopardos' Palabra sustituta: 'leones'
7	Frase: 'La mesa es roja' Palabra a sustituir: 'mesa' Palabra sustituta: 'silla'
8	Frase: 'He merendado un bocadillo' Palabra a sustituir: 'bocadillo' Palabra sustituta: 'plátano'
9	Frase: 'Los caballos son gigantes' Palabra a sustituir: 'caballos' Palabra sustituta: 'elefantes'
10	Frase: 'Las casas están vacías' Palabra a sustituir: 'casas' Palabra sustituta: 'viviendas'

Tabla 17. Parámetros pruebas unitarias segunda funcionalidad primer caso.

Tabla de resultados:

N.º Prueba	*Respuesta Programa*	*Resultado Obtenido*
1	El gato de mi padre adoptivo es bonito	Correcto
2	Mi hermano pequeño tiene 9 años	Correcto
3	A mi padre le gusta el futbol	Correcto
4	Las bombillas se apagaron	Correcto
5	La reunión se ha terminado	Correcto
6	Los leones son preciosos	Correcto
7	La silla es roja	Correcto
8	He merendado un plátano	Correcto
9	Los elefantes son gigantes	Correcto
10	Las viviendas están vacías	Correcto

Tabla 18. Resultados pruebas unitarias segunda funcionalidad primer caso.

5.1.2.2 Pruebas palabra sustituta no concuerda en género y número con palabra a sustituir

En este apartado nos adentraremos en los casos en los que la palabra a sustituir y la sustituta no concuerden en género, la concordancia en número es secundaria puesto que se adapta al número de la frase.

Se van a crear dos tablas, una relacionada con los parámetros de cada prueba y otra con los resultados, de forma que se vea más visual puesto que en una tabla sola no se entiende bien.

Tabla de parámetros:

N.º Prueba	Parámetros pasados a la funcionalidad
1	Frase: 'El perro de mi padre adoptivo es bonito' Palabra a sustituir: 'perro' Palabra sustituta: 'tortugas'
2	Frase: 'Mi primo pequeño tiene 9 años' Palabra a sustituir: 'primo' Palabra sustituta: 'hermana'
3	Frase: 'A mi hermano pequeño le gusta el futbol' Palabra a sustituir: 'hermano' Palabra sustituta: 'hermana'
4	Frase: 'Las luces se apagaron' Palabra a sustituir: 'luces' Palabra sustituta: 'focos'
5	Frase: 'La fiesta se ha terminado' Palabra a sustituir: 'fiesta' Palabra sustituta: 'evento'
6	Frase: 'Los leopardos son preciosos' Palabra a sustituir: 'leopardos' Palabra sustituta: 'gacelas'
7	Frase: 'La mesa es roja' Palabra a sustituir: 'mesa' Palabra sustituta: 'escritorios'
8	Frase: 'He merendado la tarta de la nevera' Palabra a sustituir: 'tarta' Palabra sustituta: 'plátanos'
9	Frase: 'Los caballos son gigantes' Palabra a sustituir: 'caballos' Palabra sustituta: 'jirafas'
10	Frase: 'Las casas están vacías' Palabra a sustituir: 'casas' Palabra sustituta: 'apartamento'

Tabla 19. Parámetros pruebas unitarias segunda funcionalidad segundo caso.

Tabla de resultados:

N.º Prueba	*Respuesta Programa*	*Resultado Obtenido*
1	La tortuga de mi padre adoptivo es bonita	Correcto
2	Mi hermana pequeña tiene 9 años	Correcto
3	A mi hermana pequeña le gusta el futbol	Correcto
4	Los focos se apagaron	Correcto
5	El evento se ha terminado	Correcto
6	Las gacelas son preciosas	Correcto
7	El escritorio es rojo	Correcto
8	He merendado el plátano de la nevera	Correcto
9	Las jirafas son gigantes	Correcto
10	Los apartamentos están vacías	Incorrecto

Tabla 20. Resultados pruebas unitarias segunda funcionalidad segundo caso.

5.1.2.3 Pruebas palabra sustituta modifica el número de la frase

En este caso nos encontramos con que, la palabra sustituta es un singularia tantum, o un pluralia tantum, que en el caso de no concordar en número con la frase, hay que modificar el número de toda la frase puesto que el de estos no puede modificarse.

Se van a crear dos tablas, una relacionada con los parámetros de cada prueba y otra con los resultados, de forma que se vea más visual puesto que en una tabla sola no se entiende bien.

Tabla de parámetros:

N.º Prueba	Parámetros pasados a la funcionalidad
1	Frase: 'El avituallamiento del barco fue saboteado' Palabra a sustituir: 'avituallamiento' Palabra sustituta: 'víveres'
2	Frase: 'Sus ansias de poder terminaron llevándole a su propia perdición' Palabra a sustituir: 'ansias' Palabra sustituta: 'sed'
3	Frase: 'Los quevedos son algo que se puso de moda a finales del siglo XIX' Palabra a sustituir: 'algo' Palabra sustituta: 'gafas'
4	Frase: 'Las personas se fueron' Palabra a sustituir: 'personas' Palabra sustituta: 'gente'
5	Frase: 'Los miedos se apoderaron de él' Palabra a sustituir: 'miedos' Palabra sustituta: 'pánico'
6	Frase: 'La dentadura del tigre da miedo' Palabra a sustituir: 'dentadura' Palabra sustituta: 'fauces'
7	Frase: 'Las precipitaciones mueven el barco' Palabra a sustituir: 'precipitaciones' Palabra sustituta: 'aire'
8	Frase: 'Las ganas mueven montañas' Palabra a sustituir: 'ganas' Palabra sustituta: 'fe'
9	Frase: 'Los cables dan calambre' Palabra a sustituir: 'cables' Palabra sustituta: 'electricidad'
10	Frase: 'La fiesta fue muy larga' Palabra a sustituir: 'fiesta' Palabra sustituta: 'vacaciones'

Tabla 21. Parámetros pruebas unitarias segunda funcionalidad tercer caso.

Tabla de resultados:

N.º Prueba	*Respuesta Programa*	*Resultado Obtenido*
1	Los víveres del barco fueron saboteados	Correcto
2	Su sed de poder terminó llevándole a su propia perdición	Correcto
3	Los quevedos son gafas que se pusieron de moda a finales del siglo XIX	Correcto
4	La gente se fue	Correcto
5	El pánico se apoderó de él	Correcto
6	Las fauces del tigre dan miedo	Correcto
7	El aire mueve el barco	Correcto
8	La fe mueve montañas	Correcto
9	La electricidad da calambre	Correcto
10	Las vacaciones fueron muy larga	Incorrecto

Tabla 22. Resultados pruebas unitarias segunda funcionalidad tercer caso.

5.2 Análisis de resultados

En este apartado se detallará el análisis de los resultados de las pruebas realizadas anteriormente en este mismo capítulo. Se han realizado un total de 78 pruebas, 48 referentes a la primera funcionalidad y 38 a la segunda.

Resultados de las pruebas de la primera funcionalidad:

N.º Pruebas	Aciertos	Fallos	% de acierto
48	48	0	100%

Estos porcentajes son fiables, pero no exactos, es decir, el léxico español es muy amplio, por completar correctamente 48 pruebas no se puede asegurar que se flexionen bien el 100% de los determinantes, adjetivos y sustantivos del castellano, pero sí que se puede afirmar que la funcionalidad está correctamente implementada.

En las pruebas se cubren casos tales como palabras variables e invariables en género y en número, con la misma forma para el masculino y para el femenino, pluralia y singularia tantum por lo que el rango que cubren las pruebas es amplio y la funcionalidad no falla en ninguno de los casos en concreto, asegurando así que su funcionamiento es óptimo.

Resultados de las pruebas de la primera funcionalidad:

N.º Pruebas	Aciertos	Fallos	% de acierto
30	28	2	93.33%

Como se ha mencionado anteriormente, estos porcentajes pueden ser ligeramente engañosos, y más con esta funcionalidad existe un mayor número de escenarios posibles. Esta funcionalidad presenta un menor porcentaje de acierto que la anterior, aun así, es suficientemente alto como para afirmar que su implementación es buena.

Se puede observar que de 30 pruebas ha fallado en 2, concretamente en:

- Frase: 'Las casas están vacías' Palabra a sustituir: 'casas' Palabra sustituta: 'apartamento' Resultado: 'Los apartamentos están vacías': El resultado falla debido a que la palabra vacías no se modifica en género, este error puede ser causado porque spaCy no considere que la palabra 'vacías' tenga relación con 'casas', cuando realmente sí que la tiene, ya que es un adjetivo que se refiere al sustantivo.
- Frase: 'La fiesta fue muy larga' Palabra a sustituir: 'fiesta' Palabra sustituta: 'vacaciones' Resultado: 'Las vacaciones fueron muy larga': El error se encuentra en que se introduce una palabra invariable en número, y debería modificar el número de toda la frase, sin embargo, modifica al verbo, que pasa de 'fue' a 'fueron' correctamente, pero el adjetivo 'larga' no pasa a 'largas'. Puede ser que el error sea el mismo que el explicado en el caso anterior, que spaCy no considera que la palabra 'larga' tenga relación con 'fiesta' cuando realmente sí que la tiene.

Los posibles errores son estimaciones, la raíz del error no tiene por qué ser esa, puesto que si se supiera con certeza se habría solucionado, es una forma de orientar para posibles líneas de investigación futuras.

6. Conclusiones y Análisis de impacto

En este capítulo se detallarán las conclusiones del proyecto y se relacionará su utilidad con los Objetivos de Desarrollo Sostenible.

6.1 Conclusiones

Los objetivos iniciales del trabajo se centraron simplemente en la correcta flexión de sustantivos y adjetivos, pero una vez comenzado el proyecto y tras las primeras reuniones con mis tutores, se decidió que también sería de gran utilidad flexionar determinantes y verbos, puesto que había ciertos casos que lo requerían. Finalmente, se me asignaron dos funcionalidades principales, una que devolviera todas las flexiones de una palabra, y otra que adaptara una palabra a una frase.

Para la realización de la primera se plantearon dos enfoques. El primero se basaba en coger del diccionario de la RAE las flexiones de género de una palabra y posteriormente pluralizarlas para conseguir así todas sus variaciones. Este enfoque terminó por ser descartado puesto que el diccionario de la RAE de repente dejó de aceptar mis peticiones, por lo que se pasó al segundo enfoque que funcionaba sacando las flexiones de género directamente de una biblioteca con listas de sustantivos, adjetivos, determinantes y verbos en español llamada spanish_inflections, y posteriormente pluralizándolas.

La segunda funcionalidad también adaptó este segundo enfoque ya que usa la misma función auxiliar que la primera para obtener las flexiones de género de una palabra, por lo que el enfoque final fue el de la biblioteca basada en listas (spanish_inflections).

Cabe recalcar que este proyecto depende completamente de la librería mencionada anteriormente y de la librería spaCy de procesado de lenguaje natural, por lo que se ha de tener en cuenta que, si estos factores externos se dejan de actualizar, se modifican, o se eliminan habría que hacer múltiples cambios en la implementación.

A nivel personal, este trabajo ha sido muy enriquecedor, he aprendido a programar en Python, ya que siempre había programado en Java o en C++, he podido entender la metodología de la Lectura Fácil, que también desconocía antes de comenzar con el proyecto y sobre todo el procesamiento de lenguaje natural, que nunca había investigado ni había hecho ningún trabajo en relación con este tema y me ha gustado mucho y ha despertado bastante curiosidad en mí.

En cuanto al tema del proyecto (flexión de sustantivos y adjetivos) inicialmente se puede llegar a pensar que va a ser aburrido, pero al ser un tema en el que los resultados son tan visuales, te termina por enganchar y por hacer que dediques mucho tiempo para conseguir que las funcionalidades se ejecuten correctamente.

Me ha parecido una gran culminación del grado de Ingeniería Informática puesto que han sido necesarios ciertos conocimientos y aptitudes que te proporciona la carrera en su totalidad.

El balance del TFG ha sido realmente positivo, y espero que estas funcionalidades puedan servir en algún otro proyecto de otra persona.

6.2 Análisis de impacto

El estudio de la flexión de sustantivos, adjetivos y determinantes, así como la adaptación de palabras en frases, es fundamental para facilitar la comprensión del español. Este trabajo de fin de grado (TFG) no solo contribuye al campo de la lingüística, sino que también puede tener un cierto impacto en diversos aspectos relacionados con los Objetivos de Desarrollo Sostenible (ODS) establecidos por la ONU [15].

Impacto Lingüístico, Educativo y Social

Educación de Calidad:

- Este TFG puede ser usado como material didáctico en cuestiones de gramática, ayudando especialmente a personas con discapacidades cognitivas, a niños pequeños que se encuentren estudiando este tipo de temas, o para personas extranjeras que pretenden estudiar el español.

Reducción de las Desigualdades:

- Puesto que este proyecto puede ser utilizado para la comprensión del español a niveles muy básicos, puede hacer que personas que estén comenzando a aprender el idioma, o personas con dificultades de compresión lectora rompan ciertas barreras lingüísticas promoviendo así la igualdad.

Impacto Tecnológico y Social

Industria Innovación e Infraestructura:

- La investigación realizada en este TFG puede contribuir al desarrollo de tecnologías lingüísticas avanzadas, como sistemas de procesamiento de lenguaje natural (PLN), que son esenciales para aplicaciones en inteligencia artificial y traducción automática. Esto no solo mejora la eficiencia y precisión de estas tecnologías, sino que también fomenta la innovación en el sector tecnológico.

7. Líneas de investigación futura

Este capítulo pretende abordar las posibles líneas de investigación futura, mejorando la precisión de la actual.

En primer lugar, el proyecto se basa en dos librerías, spaCy y spanish_inflections, las cuales tienen sus limitaciones:

- spaCy: en ciertas ocasiones el etiquetador gramatical da errores, etiquetando adjetivos como sustantivos, sustantivos como verbos etc... Además, una vez hecha la tokenización, hay numerosos adjetivos y sustantivos a los que no clasifica correctamente por género y número induciendo al error, o directamente no proporciona un valor para su género o para su número.
- Spanish_inflections: su funcionamiento depende de sus propias listas, por lo que, si en alguna funcionalidad se llama a una palabra que no existe en ellas, no funcionaría.

El proyecto al estar ligado a dos recursos depende de ellos y de su frecuencia de actualización, puesto que si se dejaran de actualizar quedaría estancado.

Si spaCy usara otro diccionario más completo para el español, y se añadiesen todas las palabras existentes a las listas de spanish_inflections, la funcionalidad del proyecto sería mucho más flexible y se podría utilizar en más ámbitos.

Otro aspecto a mejorar es el número de llamadas a la API de LanguageTool, que ha sido usada en numerosos métodos ya que es un corrector de errores gramaticales y ortográficos. La versión gratuita solo permite 20 por minuto y si el proyecto quiere ser usado en una mayor escala sería limitante. La solución sería adquirir el plan premium que mejora a 80 llamadas por minuto.

La formación de flexiones masculinas y femeninas del proyecto tiene un cierto margen de mejora, porque esta depende de que ambas flexiones aparezcan en las listas de spanish_inflections, y en ocasiones no era así y he tenido que añadir ciertos femeninos que estaban ausentes. La forma de tener una mejora en este aspecto es traerte desde el diccionario de la RAE ambas flexiones puesto que aparecen juntas como se puede observar en esta figura:

Ilustración 17. Consulta al diccionario de la RAE.

Haciendo scrapping se pueden obtener ambas flexiones, el problema que surgió implementando esto, fue que llegado un punto dejaron de funcionar las peticiones, por lo que para mejorar se debería hacer un estudio de porque empiezan a rechazar peticiones, o de qué forma evitar que las rechacen, para poder aprovechar al máximo este recurso.

8. Bibliografía

[1] WordReference, "Diccionario de la Lengua Española," WordReference.com. [Online]. Available: https://www.wordreference.com/es/. [Accessed: 2024].

[2] M. Gatti, “spanish_inflections,” GitHub repository, 2021. [Online]. Available: https://github.com/mathigatti/spanish_inflections. [Accessed: 2024].

[3] Explosion AI, "spaCy: Industrial-strength Natural Language Processing in Python," spaCy.io. [Online]. Available: https://spacy.io/. [Accessed: 2024].

[4] D. J. Pérez Melián, "IAFlex: Flexionador nominal inteligente”, Universidad de Las Palmas de Gran Canaria, 2021.

[5] IATEXT, "Flexionador Tip," tulengua.iatext.ulpgc.es. [Online]. Available: https://tulengua.iatext.ulpgc.es/flexionador/. [Accessed: 2024].

[6] Guapolo, “flexiones_es_mx_spec.rb,” GitHub gist, 2019. [Online]. Available: https://gist.github.com/guapolo/15b05d08cc8ff4dc89bfba1961380a16. [Accessed: 2024].

[7] RAE, "Plural," Diccionario panhispánico de dudas. [Online]. Available: https://www.rae.es/dpd/plural. [Accessed: 2024].

[8] RAE, "Formación del femenino en profesiones, cargos, títulos o actividades humanas”, Diccionario panhispánico de dudas, capítulo 3. [Online]. Available: https://www.rae.es/dpd/género. [Accessed: 2024].

[9] Universidad Complutense de Madrid, "Casos Especiales de Número," PlataformaELE, [Online]. Available: https://www.ucm.es/plataformaele/casos-especiales-de-numero. [Accessed: 2024].

[10] Universitat Politècnica de Catalunya, "Etiquetas morfosintácticas PAROLE," nlp.tools. [Online]. Available: https://www.cs.upc.edu/~nlp/tools/parole-sp.html. [Accessed: 2024].

[11] LanguageTool, "LanguageTool API", LanguageTool.org. [Online]. Available: https://languagetool.org/http-api/. [Accessed: 2024].

[12] Real Academia Española y Asociación de Academias de la Lengua Española, *Diccionario de la lengua española*, 23rd ed.

[13] J. Penia, "Pattern y Python 3.7+", GitHub gist, 2021. [Online]. Available: https://gist.github.com/juanpenia/4ea3f58aedbfaefd44f97ad905992ccf. [Accessed: 2024].

[14] DigiAsset, "pattern.es," DigiAsset.org. [Online]. Available: https://digiasset.org/html/pattern-es.html. [Accessed: 2024].

[15]. Organización de Naciones Unidas. “Objetivos de Desarrollo Sostenible”. Naciones Unidas. [Online]. Available:

https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/ [Accessed: 2024].

9. Anexo

Link al documento de originalidad de la herramienta Turnitin: https://drive.google.com/file/d/1VmMFnOnlTohKVQYr2JsCUhe7V2eIgiaE/view?usp=drive_link

Captura de pantalla asociada a las pruebas realizadas en el capítulo 5.1.1.1

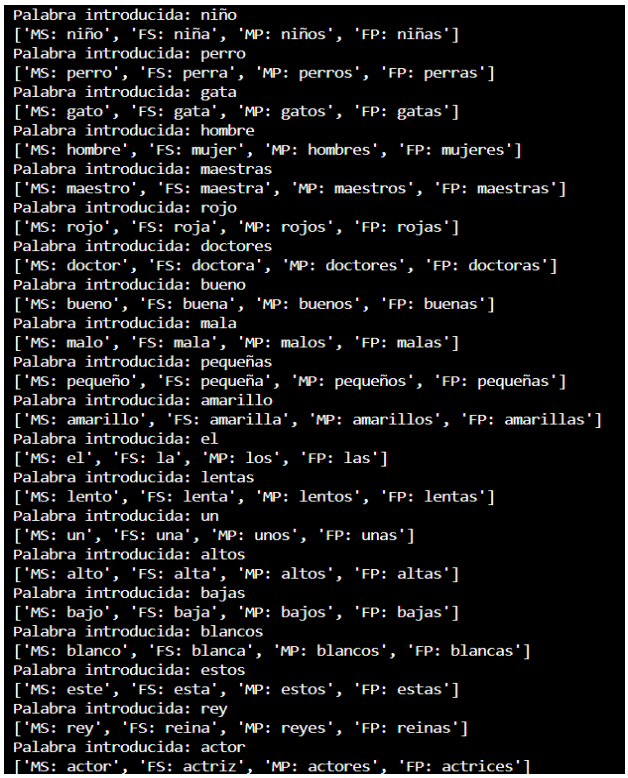

Ilustración 18. Resultado del programa a las pruebas del capítulo 5.1.1.1.

Captura de pantalla asociada a las pruebas realizadas en el capítulo 5.1.1.2

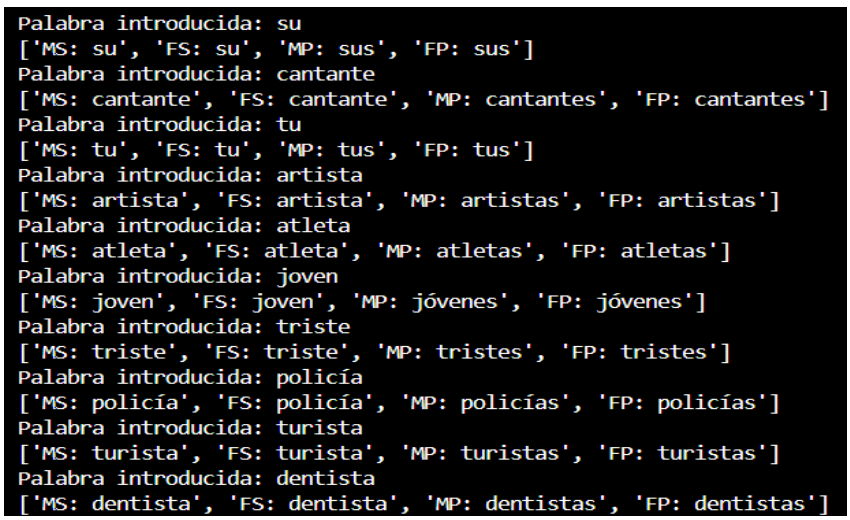

Ilustración 19. Resultado del programa a las pruebas del capítulo 5.1.1.2.

Captura de pantalla asociada a las pruebas realizadas en el capítulo 5.1.1.3

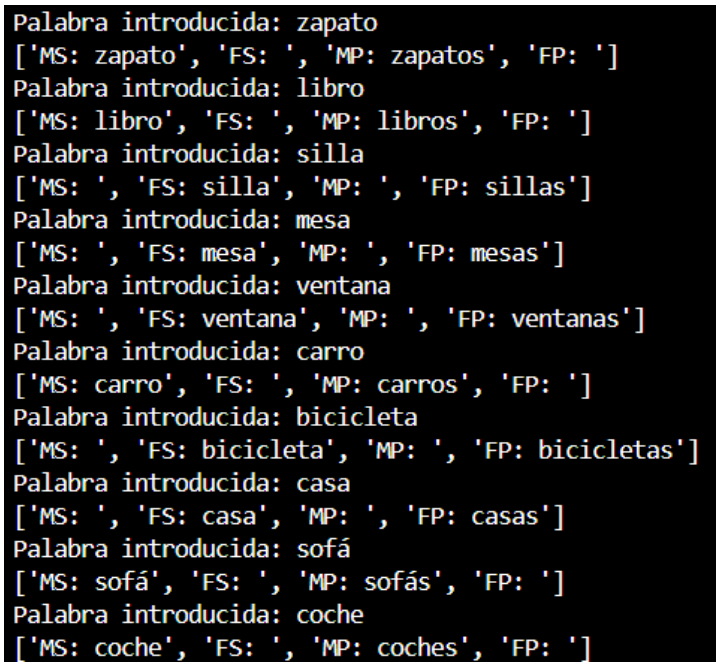

Ilustración 20. Resultado del programa a las pruebas del capítulo 5.1.1.3.

Captura de pantalla asociada a las pruebas realizadas en el capítulo 5.1.1.4

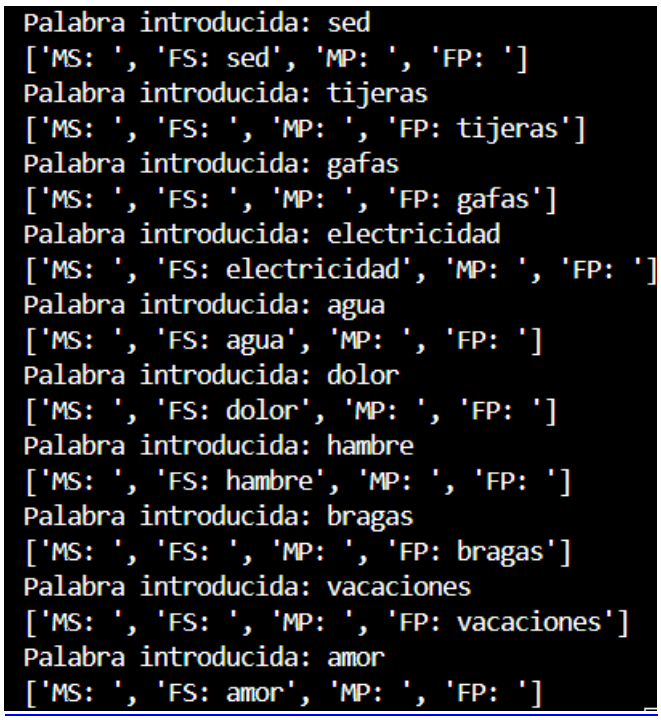

Ilustración 21. Resultado del programa a las pruebas del capítulo 5.1.1.4.

Captura de pantalla asociada a las pruebas realizadas en el capítulo 5.1.2.1

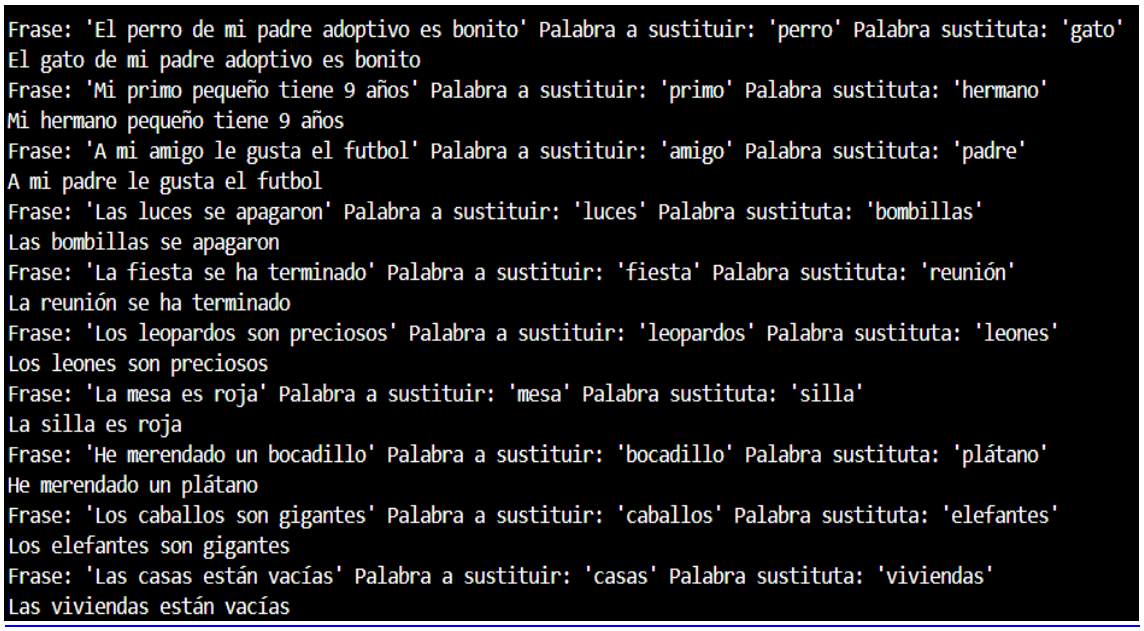

Ilustración 22. Resultado del programa a las pruebas del capítulo 5.1.2.1.

Captura de pantalla asociada a las pruebas realizadas en el capítulo 5.1.2.2

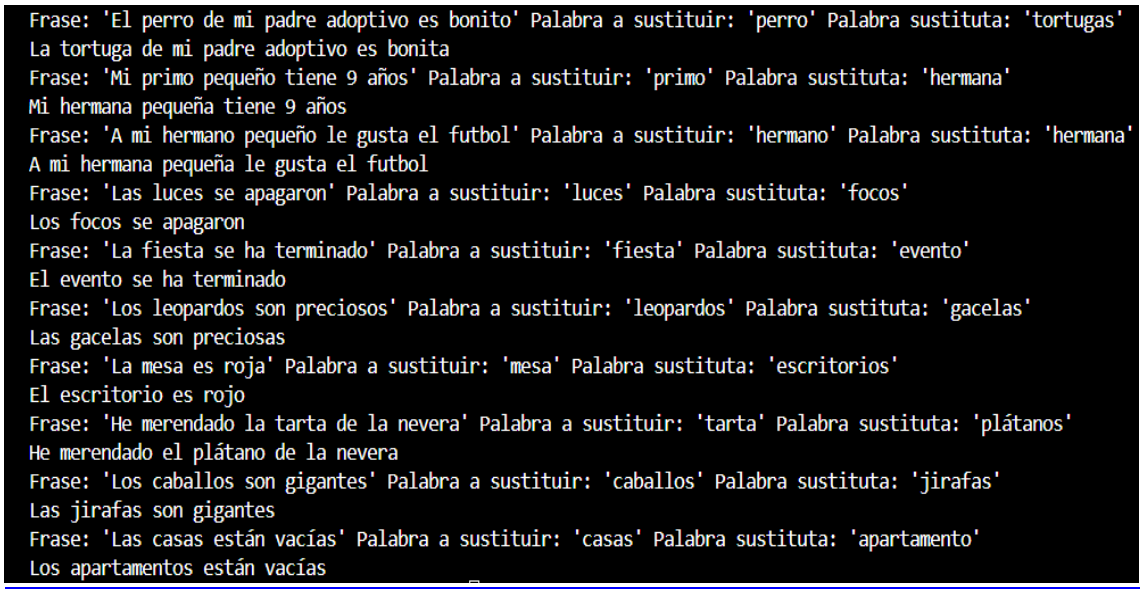

Ilustración 23. Resultado del programa a las pruebas del capítulo 5.1.2.2.

Captura de pantalla asociada a las pruebas realizadas en el capítulo 5.1.2.3

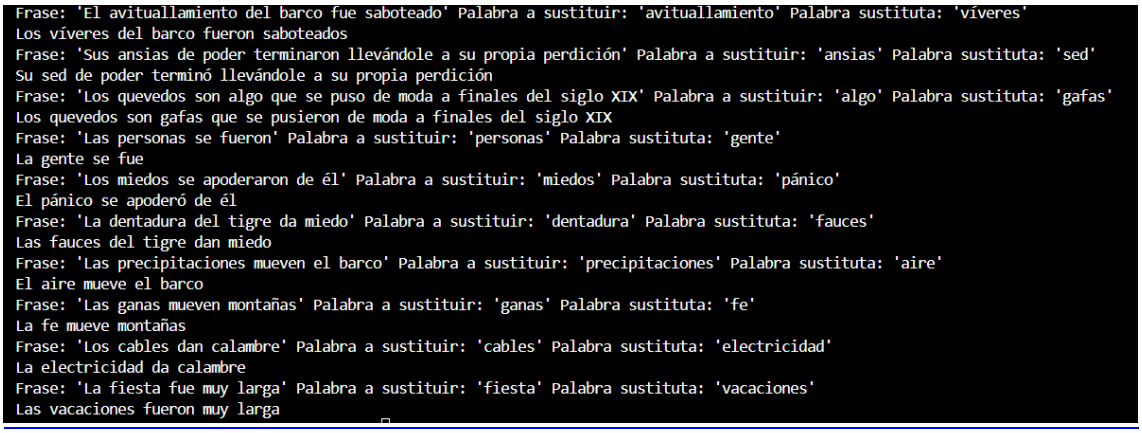

Ilustración 24. Resultado del programa a las pruebas del capítulo 5.1.2.3.

Este documento esta firmado por

	**Firmante**	CN=tfgm.fi.upm.es, OU=CCFI, O=ETS Ingenieros Informaticos - UPM, C=ES
	Fecha/Hora	Mon Jun 03 21:04:23 CEST 2024
	Emisor del Certificado	EMAILADDRESS=camanager@etsiinf.upm.es, CN=CA ETS Ingenieros Informaticos, O=ETS Ingenieros Informaticos - UPM, C=ES
	Numero de Serie	561
	Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)