Sistema para la calificación de documentos en base a su completitud

Document qualification system based on its completeness

Trabajo de fin de máster

Autor: Gabriel Sellés Salvà
Director: Alberto Díaz Esteban
Calificación Obtenida: 9
Convocatoria: Junio 2020

Máster en Ingeniería Informática, Facultad de Informática, Universidad Complutense de Madrid

Curso académico 2019/2020

Agradecimientos

Querría agradecer la ayuda y el apoyo recibido por parte de un gran número de personas sin las cuales este proyecto no se habría llevado a cabo.

En primer lugar, agradecer a Alberto Díaz Esteban que me haya ofrecido este proyecto y por la confianza depositada en mí. También querría agradecerle su colaboración y apoyo en todo momento.

En segundo lugar, agradecer el apoyo e interés mostrado por mis familiares y amigos, que han estado presentes cuando más los he necesitado.

Por último, agradecer a los profesores que me han ayudado a alcanzar mis metas, formándome y proporcionándome conocimientos día a día.

Resumen

Debido a la constante creación de documentación en la industria, aparece la necesidad de aplicar las revisiones correspondientes para evaluar el contenido de la documentación generada. Esta labor implica un enorme esfuerzo temporal, económico y humano. Por lo tanto, es de especial interés implementar un sistema que automatice este proceso, liberando así a los profesionales de esta carga de trabajo.

Por esta razón, en este trabajo se propone un sistema que tiene la capacidad de analizar el contenido de distintos documentos y de realizar estas revisiones de forma automática. El sistema propuesto, para que sea adaptable y escalable, se ha implementado de forma que puede adaptase a distintos dominios. Su funcionamiento no se ajusta a un único tipo de documentos.

El sistema propuesto se implementa utilizando distintas técnicas de procesamiento de lenguaje natural, de extracción de información y de aprendizaje automático. En este documento se describe tanto el funcionamiento de estas técnicas como su presencia y relevancia en la industria.

Este trabajo está relacionado con un proyecto de colaboración con la empresa *ECIX Group*, que plantearon esta necesidad y han proporcionado todos los recursos necesarios.

Abstract

Due to the constant creation of documentation in the industry, there is a need to apply the corresponding revisions to evaluate the content of the documentation generated. This work implies an enormous temporary, economic, and human effort. Therefore, it is important to implement a system that automatatizes this process, thus freeing professionals from carrying out this task.

For this reason, in this project we propose a system that has the ability to analyze the content of different documents, and to realize these reviews automatically. The proposed system, to be adaptable and scalable, has been implemented so that it can be adapted to different domains. Its operation does not conform to a single type of documents.

The proposed system is implemented by using different Natural Language Processing, Information Extraction, and Machine Learning techniques. This document describes how these techniques work, its presence in the industry, and its relevance.

This work is related to a collaboration project with the company *ECIX Group*, which raised this need, and has provided all the necessary resources.

Palabras clave

- Procesamiento de Lenguaje Natural
- Calificación de documentos
- Completitud de un documento
- Extracción de Información
- Semejanza Semántica
- Semántica Distribucional
- Word Embeddings
- Word2Vec
- Spacy

Keywords

- Natural Language Processing
- Documents qualification
- Documentation completeness
- Information Extraction
- Semantic Similarity
- Distributional Semantics
- Word Embeddings
- Word2Vec
- Spacy

Índice

Índice de figuras

Capítulo 1

Introducción

Contenido: En este apartado se define la motivación del trabajo realizado junto a los objetivos planteados al inicio del mismo. También incluye una visión general del documento que facilita su comprensión.

1.1. Motivación

De entre todas las tareas que se pueden realizar aplicando técnicas de Procesamiento de Lenguaje Natural (PLN) y de Aprendizaje Automático (AA), el análisis de la calidad de distintos documentos se presenta como uno de los retos más prometedores y de mayor utilidad dentro de la industria en la actualidad.

Esto se debe a múltiples factores. Uno de ellos es la constante creación de documentación presente en la industria. Con la implantación, por ejemplo, de nuevas leyes como el Reglamento General de Protección de Datos (RGPD, abril 2016), aparece la necesidad de producir documentación que se adapte al nuevo contexto impuesto por la ley correspondiente, ya sea actualizando documentos obsoletos o creando nuevos.

La creación de toda esta documentación también implica la necesidad de realizar múltiples análisis y revisiones sobre los documentos producidos. Mediante estas revisiones, se puede determinar tanto su calidad como su validez frente al nuevo contexto establecido, evitando así el uso de dicha documentación en el caso de que este incluya aspectos que no se ajusten adecuadamente.

Actualmente, para determinar si un documento cumple con las restricciones impuestas por un cierto contexto (ya sea una ley, un estándar de calidad, una serie de recomendaciones, etc), es necesario que los responsables del texto lo analicen de forma manual. También es necesario que identifiquen los

distintos aspectos que determinan si un documento es válido o se adecua al contexto concreto.

Este análisis implica un gran esfuerzo tanto económico como humano. Por lo tanto, sería de especial interés experimentar con distintas técnicas que nos permitan implementar aproximaciones que reduzcan el esfuerzo requerido a la hora de realizar esta tarea.

Este contexto ha propiciado que *ECIX Group* determine la necesidad de un sistema de esta índole. Gracias a su colaboración en el proyecto, en este trabajo se propone un sistema que, mediante distintas técnicas de procesamiento de lenguaje natural, tiene la capacidad de realizar estas revisiones de forma automática.

Un sistema como el propuesto automatiza este proceso, liberando considerablemente a los profesionales en la materia correspondiente de su carga de trabajo.

1.2. Objetivos

El objetivo principal de este trabajo es la construcción de un sistema que tenga la capacidad de calificar distintos documentos en base a su completitud mediante el uso de distintas técnicas de procesamiento de lenguaje natural.

Este sistema se construirá aprovechando tanto los recursos sobre la evaluación de documentos presentes en Internet como los proporcionados por la empresa *ECIX Group*. Además, aplicará distintas técnicas de procesamiento de lenguaje natural, extracción de información y aprendizaje automático utilizadas en la literatura actual.

Para que el sistema sea escalable y, además, pueda utilizarse frente a distintos tipos de documentos, el sistema debe ser adaptable a distintos dominios. Esto implica que su uso no debe restringirse a una única temática o tipo de documento.

Para ello, el sistema permitirá el uso de distintos criterios para realizar dicha evaluación. Estos se podrán especificar de forma manual y, además, se podrán extraer de forma automática.

Las tareas que se abarcan en este trabajo podrían clasificarse dentro de los siguientes campos de la informática:

- **Procesamiento de lenguaje natural**: el sistema aplicará distintas técnicas para analizar el contenido textual presente en los documentos evaluados y para realizar las transformaciones correspondientes sobre él.
- **Extracción de información**: el sistema tendrá la capacidad de analizar conocimiento textual en el que se especifican los criterios que determinan la calidad de un cierto tipo de documentos con la finalidad de extraer la información relevante para realizar evaluaciones en un futuro.
- **Comparación de conocimiento textual**: con la finalidad de buscar la información que determina la calidad de un documento (los criterios), el sistema deberá comparar fragmentos de contenido textual, determinando qué conocimiento comparten, en qué difieren, etc.
- **Aprendizaje Automático**: el sistema tendrá la capacidad de ajustar su comportamiento para así calificar distintos tipos de documentos de forma más verosímil. Este ajuste se realizará mediante el análisis de evidencias previamente evaluadas de forma manual.

1.3. Visión general del documento

El documento está dividido en varios apartados que se indicarán a continuación:

1. Análisis de la tarea: se define y se analiza la tarea a resolver, indicando qué factores repercuten en ella, su presencia en la literatura actual y otras tareas relacionadas de interés.
2. Fundamentos: se exponen los fundamentos que dan lugar al sistema propuesto. Incluye explicaciones sobre estos, sobre su presencia en la literatura y sobre las distintas técnicas o recursos que engloban y son de especial utilidad para la resolución del problema propuesto.
3. Funcionamiento del sistema propuesto: se describe, tanto a alto nivel como de forma detallada, las distintas operaciones que dan lugar al sistema propuesto y sus principales usos.
4. Componentes del sistema. Se describen los distintos componentes que interactúan entre sí para dar lugar a las funcionalidades del sistema propuesto, indicado sus principales funciones, cómo se relacionan entre sí y las tecnologías externas utilizadas para ello.

5. Resultados obtenidos: se indican y se analizan los resultados obtenidos al utilizar el sistema propuesto frente a dos colecciones de documentos diferentes. Además, se compara el rendimiento del sistema utilizando tanto *WordNet* como *Word2Vec* (en la comparación de términos) sobre una tercera colección.

6. Conclusiones y trabajo futuro: incluye distintas conclusiones que se pueden extraer sobre el trabajo realizado una vez se ha implementado el sistema propuesto y se han analizado los resultados obtenidos. También se indican distintas modificaciones sobre el sistema propuesto que mejorarían su comportamiento.

7. Introduction: Versión traducida al inglés de este capítulo. Se define la motivación del trabajo realizado junto a los objetivos planteados al inicio del mismo. También incluye una visión general del documento que facilita su comprensión.

8. Conclusions and future work: Versión traducida al inglés del capítulo "Conclusiones y trabajo futuro".

9. **Apéndice**: Acceso e instalación del sistema. Se indica el repositorio en el que se almacena el sistema propuesto. Además, incluye información de utilidad sobre su instalación y uso.

Capítulo 2

Análisis de la tarea

Contenido: En este apartado se define y se analiza la tarea a resolver, indicando qué factores repercuten en ella, su presencia en la literatura actual y otras tareas relacionadas de interés.

2.1. Calificación de un documento

2.1.1. Análisis de la calidad de un documento

La calificación de un documento, independientemente de su tipo, dependerá de los factores que se analicen a la hora de realizar dicha calificación y de la naturaleza del propio documento.

En los enfoques presentes en la literatura actual, la calidad de un documento depende principalmente de dos aspectos: del cumplimiento de los aspectos formales asociados al lenguaje utilizado y del grado de satisfacción de un conjunto de atributos que determinan la calidad de un documento.

Por un lado, los aspectos formales son aquellas normas utilizadas en la lengua escrita que se consideran necesarias para que el mensaje que se transmite sea recibido adecuadamente por el receptor. Este factor engloba reglas, prácticas o recomendaciones de distinta índole. Incluyen desde normas ortográficas que deben cumplirse siempre que se redacte un texto hasta aspectos estilísticos que determinan la distribución de los elementos que componen un documento (márgenes, sangría, orden de escritura, etc).

También incorpora un conjunto de recomendaciones que, en el caso de que se apliquen, a priori, facilitan el entendimiento del mensaje transmitido. Se recomienda, por ejemplo, evitar utilizar términos ambiguos (ya que

facilitan que el receptor entienda un mensaje distinto al intencionado) o sentencias muy largas (ya que puede dificultar la lectura del texto por parte del receptor).

Por otro lado, existen otros aspectos que determinan la calidad de un documento. Estos se recogen en la literatura en forma de propiedades o atributos que cualquier documento, independientemente de su ámbito o funcionalidad, debe cumplir para poder considerarse de calidad. Una de las primeras investigaciones que establecía los atributos a evaluar es la realizada por (Wang y Strong, 1996). En este trabajo, se propuso una jerarquía de propiedades medibles (figura 2.1) que están presentes en cualquier documento considerado de calidad.

En la actualidad, gran parte de las aproximaciones relacionadas con la evaluación de la calidad de un documento utilizan jerarquías similares a esta. De hecho, la mayoría consisten en variantes de la misma.

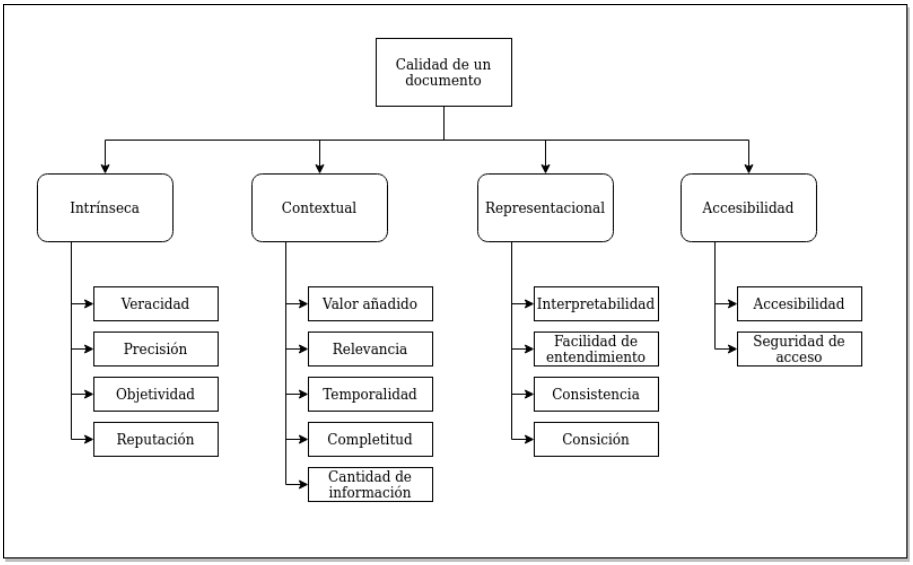

Figura 2.1: Taxonomía de propiedades establecidas por (Wang y Strong, 1996).

Por lo tanto, para determinar la calidad de un documento se evalúa el cumplimiento de múltiples propiedades medibles. Para determinar su cumplimiento, se establecen distintas métricas asociadas a cada una. Por ejemplo, algunas de las métricas utilizadas para analizar la facilidad de entendimiento de un texto son el número de abreviaturas y acrónimos que contiene ya que, el exceso de ambos, dificulta el entendimiento por parte del receptor.

Luego, una vez se han obtenido las distintas métricas de un texto asociadas a cada una de estas propiedades, se extrae un veredicto sobre el cumplimiento de cada propiedad para así poder estimar la calidad del propio documento.

Estas propiedades son las que contemplan, por ejemplo, los sistemas propuestos por (Génova et al., 2011), (Ferschke, 2014) o (Cozza et al., 2016).

2.1.2. Completitud de un documento

De entre todas las propiedades que determinan la calidad de un documento, una de las más importantes es la completitud. Este atributo es el que determina que el contenido de un documento cubre todos los aspectos necesarios para que el propio texto pueda realizar su función correctamente.

A diferencia de otras propiedades, este atributo depende del ámbito o de la funcionalidad del documento. Para considerar que un documento de una cierta temática es de calidad (en base a su completitud) es necesario que incluya una serie de conceptos o aspectos asociados a su temática que permitan que cumpla con su propósito correctamente. Por ejemplo, en el caso de un contrato de alquiler de una vivienda en el que no figuran los nombres del arrendatario o del arrendado, podemos afirmar que no es de calidad ya que falta una información clave para que cumpla su propósito.

Ya que no existe un conjunto de contenidos universales que debe incluir cualquier tipo de documento para ser considerado de calidad, es necesario establecer, para cada tipo de documento, los contenidos o aspectos que determinan su calidad. Estos requisitos o criterios pueden consistir en condiciones de diferente tipo (que aparezcan los siguientes conceptos, que aparezca la definición de una serie de términos, etc) y dependen de la naturaleza y del propósito del tipo de documentos concreto.

Tal y como decíamos anteriormente, en un contrato de arrendamiento debe aparecer el nombre del arrendatario y del arrendado mientras que en una ficha técnica de un ordenador portátil, por ejemplo, debería aparecer su capacidad de memoria, la distribución de su teclado o el tamaño de la pantalla. Estos serían ejemplos de criterios de calidad para dos tipos de documentos distintos.

Además, existen casos en los que los criterios que determinan la calidad de un texto se especifican en otros textos.

Un ejemplo en el mundo de la docencia sería una plantilla para la corrección de un examen en la que, para cada pregunta, se indica qué contenidos deben aparecer en una respuesta para afirmar que se ha respondido correctamente. Si un alumno incluye los contenidos indicados en la plantilla, podemos afirmar que, respecto al contenido esperado, la pregunta se ha respondido correctamente.

Debido a la existencia de este tipo de documentos, es de especial interés analizar si es posible extraer información de estos documentos para evaluar la calidad de otros documentos relacionados. De este modo, no sería necesario especificar de forma manual los distintos criterios que determinan la calidad de un documento. El sistema resultante, si los criterios deben especificarse de forma manual, sería menos escalable y requeriría más esfuerzo humano, el cual queremos reducir lo máximo posible.

La evaluación de un documento en base a su completitud es una tarea que, en comparación con la calificación en base a aspectos formales o a las demás propiedades descritas, no tiene mucha presencia en la literatura actual. Debido a ello, en este trabajo, vamos a experimentar con distintas técnicas para determinar la calidad de un documento en base a su completitud.

2.1.3. Requisitos de calidad

Tal y como se indica en el apartado anterior, la calidad de un documento en función de su completitud depende del tipo del documento evaluado y de los criterios asociados al propio tipo. No se califica del mismo modo una ficha técnica de un dispositivo que un documento respecto a una cierta ley ya que, en cada uno, destacamos aspectos diferentes.

Por lo tanto, para implementar cualquier aproximación que trate de evaluar la calidad de un documento en base a su contenido, es necesario establecer primero qué son los criterios y cómo se representan.

Como evaluamos la calidad de un documento en base a su completitud, un criterio consiste en un aspecto o concepto que debe contemplarse en el contenido del documento para que este sea de calidad y pueda cumplir su propósito.

Una forma visual y entendible de representar los criterios sería a través de sentencias en lenguaje natural. En este caso, las sentencias deberían contener la información de los conceptos que determinan la calidad de un documento.

Entonces, un sistema que resuelva la tarea propuesta debería buscar la presencia, de forma implícita o explícita, del contenido de los criterios dentro de los documentos a evaluar para, en función de los criterios/sentencias encontrados, indicar cuál es la calidad de cada documento.

El principal problema de esta representación es que una única sentencia no tiene porqué incluir toda la información necesaria para describir un concepto que debe aparecer en un documento. Por ejemplo, el criterio "Identificar correctamente a un individuo" no especifica qué implica identificar a un individuo. Para ello, sería necesario, por ejemplo, indicar su nombre completo, su número de identificación (DNI, en el caso de España), su domicilio actual y su número de teléfono.

Una posible solución para resolver este problema sería complementar los criterios con otros conceptos que añadan la información que, de serie, no incluyen los propios criterios. Estos nuevos conceptos serían subcriterios del criterio correspondiente y también se representarían como sentencias en lenguaje natural (tal y como se muestra en la figura 2.3).

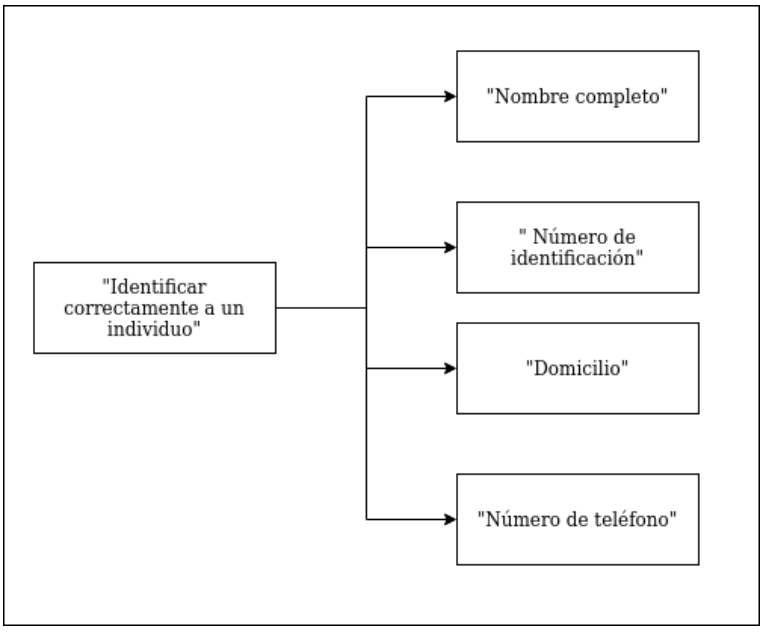

Figura 2.2: Ejemplo de subcriterios asociados a un criterio.

De esta forma, un sistema que resuelva la tarea, buscaría en un documento tanto los criterios que determinan su calidad como los subcriterios asociados a cada criterio. En función de los subcriterios encontrados, podríamos determinar si se cumple un criterio y, en función de si se cumple dicho

criterio, podríamos generar un veredicto respecto a la calidad de un cierto documento en base a su completitud.

2.2. Tareas relacionadas

En la literatura actual, existen muchas tareas resueltas aplicando técnicas de procesamiento de lenguaje natural. Algunos ejemplos son la traducción automática, la extracción de información de un documento, el análisis de sentimientos asociados a un texto, la creación automática de resúmenes, etc. Cada una de estas tareas responde a una necesidad concreta y requiere, en la mayoría de los casos, un enfoque particular para alcanzar los resultados deseados.

Aún así, los planteamientos utilizados para resolver un problema, en algunos casos, pueden reutilizarse para resolver parcial o totalmente otro problema. Incluso, algunos de los problemas tradicionales del PLN pueden plantearse como un conjunto de subproblemas de los cuales algunos ya están resueltos en mayor o menor medida.

Un ejemplo es la tarea de la traducción automática. Cuando un sistema que realiza esta tarea se encuentra con un término ambiguo como "planta", debe resolver el problema de la resolución de ambigüedad antes de realizar la traducción del mismo. En este caso, como se puede ver, se utilizan soluciones a tareas distintas que después nos permitirán resolver otra más grande.

Por lo tanto, es de especial interés identificar los problemas o tareas de PLN que están relacionados con el que se trata en este trabajo. A partir de los enfoques utilizados en estos problemas, es probable que aparezcan técnicas o aspectos que faciliten la resolución del problema propuesto.

Por otro lado, la calificación de documentos en base a su contenido es una tarea que, tal y como indicábamos antes, depende de los criterios asociados a la naturaleza del documento.

Entonces, a grandes rasgos, esta tarea consiste en dos subtareas:

- La extracción de criterios asociados a un cierto tipo de documentos (en el caso de que no se especifiquen de forma manual).
- El análisis tanto del significado de los criterios a buscar como del propio texto a calificar para así poder determinar si el texto cumple con los criterios que determinan su calidad.

Respecto a la extracción de criterios asociados a un cierto tipo de documentos, la tarea de PLN más parecida y a su vez más relevante es la extracción de información (*Information Extraction* en inglés).

Esto se debe a que ambas tareas consisten en la búsqueda y extracción de información relevante dentro contenido textual. En nuestro caso, nuestro objetivo sería extraer información relevante de los documentos que especifican los criterios (y subcriterios) asociados a un cierto tipo de documentos.

Por otro lado, respecto a la búsqueda de criterios dentro de un documento a evaluar, viendo las características del problema propuesto, los problemas que pueden ser de mayor utilidad son la semejanza semántica, la implicación textual y la detección de paráfrasis en textos.

La principal razón de esta semejanza es que, a grandes rasgos, las cuatro tareas consisten en comparar el contenido de dos textos para determinar su semejanza y así extraer una conclusión. Para nuestra tarea, sería relevante evaluar la semejanza entre los criterios (y subcriterios) y los fragmentos de un documento para así determinar su cumplimiento.

2.2.1. Extracción de información

En los último años, el volumen de documentación digitalizada presente en la industria ha incrementado considerablemente. Por esta razón, las tareas de PLN relacionadas con la extracción de información cada vez han adquirido una mayor relevancia.

La extracción de información engloba un numeroso conjunto de técnicas que permiten extraer, de forma automática, información relevante y estructurada a partir del contenido de distintos recursos desestructurados. Los recursos sobre los cuales se aplican estas técnicas tienden a ser documentos descritos en lenguaje natural.

En algunos casos, los recursos desestructurados incluyen aspectos atípicos no presentes en el lenguaje natural. Estos elementos pueden ser desde elementos multimedia (como fotos, vídeos, música, etc) hasta formatos inusuales en un documento pero que, a su vez, facilitan el cumplimiento de su objetivo (recursos semi-estructurados).

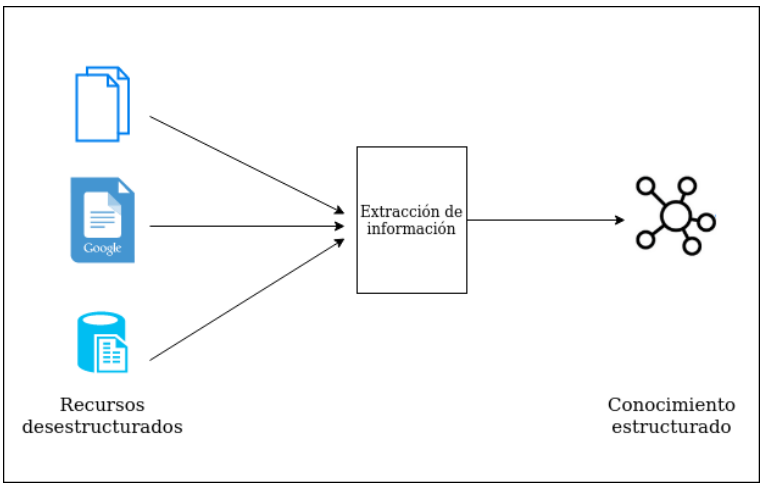

Figura 2.3: Representación de la extracción de información.

La extracción de información agrupa múltiples subtareas y sus técnicas correspondientes. Algunos ejemplos son la extracción de entidades (*Named Entity Recognition*), la extracción de palabras clave (*Keyword Extraction*) o la extracción de temas asociados a un documento (*Topic Extraction*).

A su vez, las técnicas que engloba se aplican para resolver otros problemas como la traducción automática, la asistencia automática digital (presente en sistemas como *Siri* o *Cortana*) o la generación automática de resúmenes.

Como la extracción de información es una tarea amplia que engloba numerosas subtareas con propósitos distintos, no existe una premisa común en todas las aproximaciones que resuelvan dicha tarea.

El planteamiento de cada aproximación depende principalmente del tipo de información que se pretenda extraer. En la literatura, existen aproximaciones que extraen información a partir del análisis sintáctico y semántico del contenido del texto, implementaciones que estudian la distribución de los términos que lo componen, etc.

2.2.2. Semejanza semántica

La semejanza semántica, con el paso del tiempo, se ha convertido en una de las principales tareas relacionadas con PLN debido a sus numerosas aplicaciones en experimentos y sistemas. Algunas de sus principales aplicaciones son la clasificación de textos, la generación automática de resúmenes, la búsqueda de respuestas a cuestiones en lenguaje natural o desambiguación de términos.

Consiste en el análisis de dos textos con la finalidad de obtener el grado de similitud existente entre ambos.

La similitud entre ambos textos se calcula analizando relaciones semánticas existentes entre sus fragmentos, tal y como se muestra en la figura 2.4. Este tipo de relaciones pueden ser de distinto tipo: equivalencia, sinonimia, antonimia, hiperonimia, campo semántico, familia semántica, etc. A cada tipo de relación se le asigna una relevancia dentro del cómputo del resultado, de modo que no todos los tipos de relaciones tienen la misma importancia.

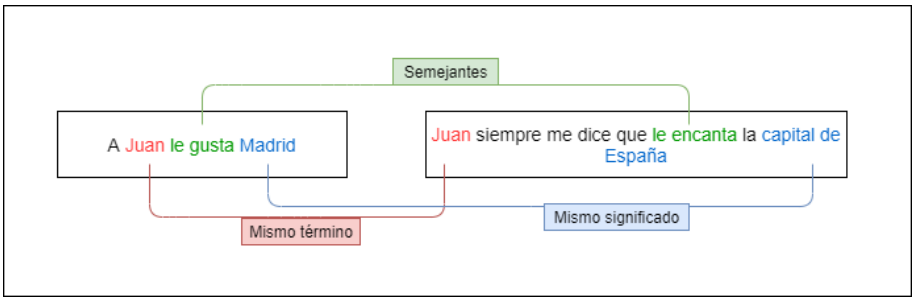

Figura 2.4: Ejemplo de relaciones existentes entre los términos de dos sentencias.

El nivel de semejanza obtenido se calculará en función de un rango predefinido de valores a los cuales se les ha otorgado un cierto significado previamente.

La semejanza semántica consiste en una operación conmutativa en la que la propia semejanza se cumple en ambas direcciones. Por definición, la semejanza entre dos textos A y B es la misma que entre B y A. Esto sucede porque el cálculo se realiza atendiendo al contenido de ambos por igual: buscando las semejanzas del primer texto con el segundo y viceversa.

La mayoría de los sistemas o aproximaciones que resuelven esta tarea calculan la semejanza semántica del mismo modo. La semejanza entre dos textos (sentencias, párrafos, etc) se obtiene a partir de la asociación de términos semánticamente similares (o relacionados semánticamente) de ambos. Luego, en función de la semejanza de los términos asociados y de la relevancia de la asociación, se realiza el cálculo oportuno y se obtiene la semejanza entre los dos textos. En la figura 2.5 se muestra un ejemplo de este proceso.

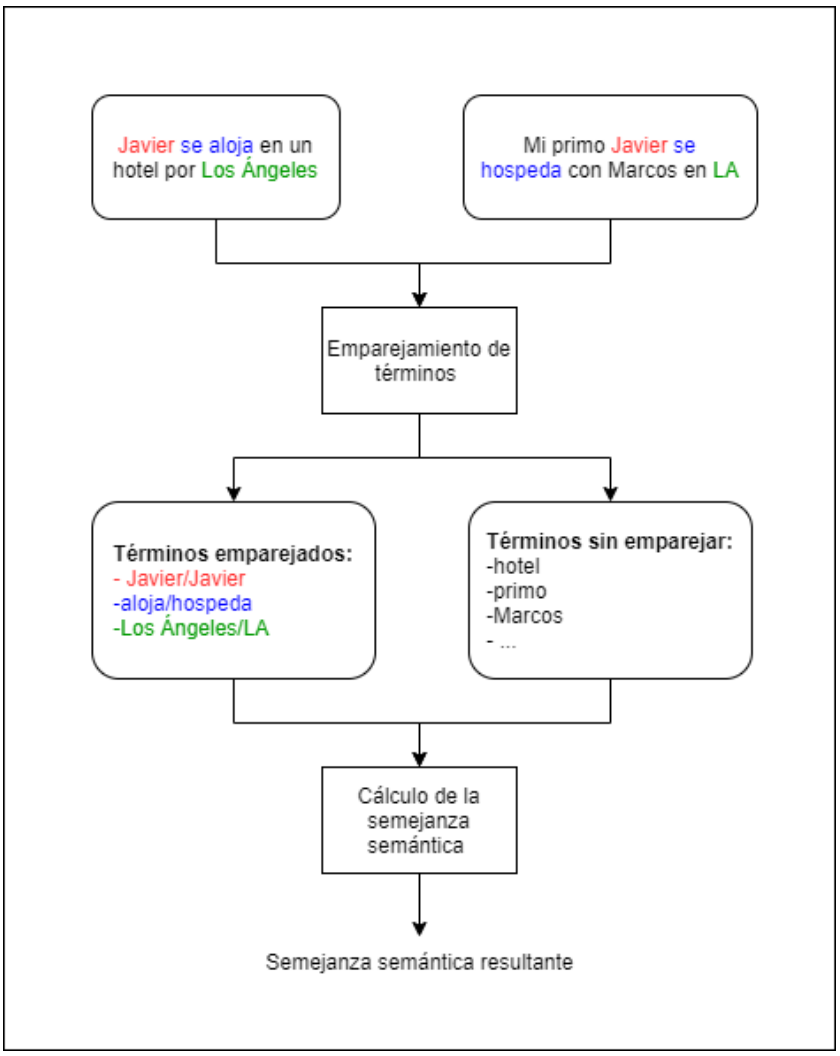

Figura 2.5: Ejemplo de asociación de términos de dos sentencias.

Tal y como se indica arriba, el valor resultante depende del rango de valores establecidos y sus significados. Los valores establecidos dependerán, sobre todo, de la naturaleza del problema a resolver.

Una representación de los resultados recurrente en la industria es la propuesta por SemEval. SemEval (*Semantic Evaluation*) consiste en una competición organizada por ACL-SIGLEX en la que, anualmente, se proponen distintas tareas relacionados con el PLN para que los participantes del evento las resuelvan.

En las convocatorias donde explícitamente se trataba el problema de la semejanza semántica, se establecía que los resultados serían enteros entre 0

y 5. En la figura 2.6 se describe el significado de cada valor junto a ejemplos concretos de cada uno.

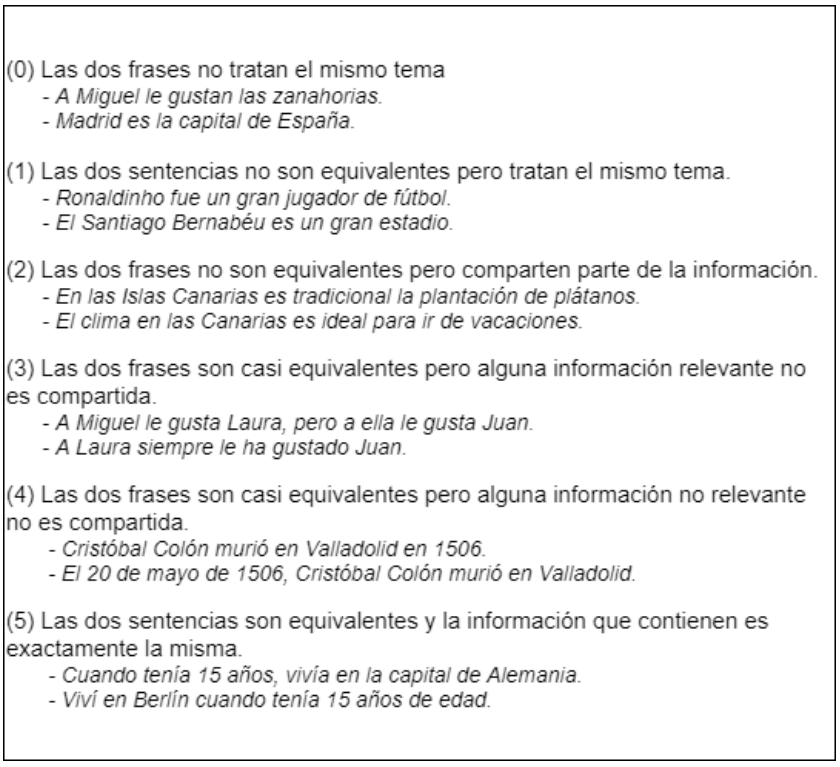

Figura 2.6: Grados de semejanza establecidos por SemEval junto a ejemplos concretos.

Esta tarea tiende a relacionarse con la implicación textual (*Textual Entailment*) y la detección de paráfrasis en textos (*Paraphrase detection*). Aún así, estas dos tareas presentan varios aspectos diferenciales respecto a la semejanza semántica, de modo que se plantean como tareas diferentes e independientes.

2.2.3. Implicación textual

Esta tarea también ha alcanzado una relevancia considerable debido a la multitud de utilidades que comparte con la semejanza semántica (corrección de respuestas automáticas, traducción automática de textos, generación de resúmenes, etc).

La implicación textual analiza si existe una relación entre dos sentencias

H (hipótesis) y T (texto). T consiste en un texto que contiene un cierto conocimiento mientras que H representa una hipótesis que potencialmente se puede deducir a partir del contenido de T. La tarea de un sistema que resuelva la implicación textual es determinar si, a partir del contenido de un cierto T, podemos deducir un cierto H.

La mayoría de las aproximaciones que resuelven esta tarea contemplan dos posibles resultados: hay implicación entre H y T o no la hay. Aún así, en función de la finalidad de cada solución, esto puede variar. En algunos casos, pueden presentarse niveles de implicación intermedios.

Para determinar si existe implicación textual, se utilizan técnicas similares a las utilizadas en la semejanza semántica. Gran parte de las soluciones presentes en la literatura asocian términos de la hipótesis con otros del texto que estén relacionados semánticamente para luego analizar si la información contenida en la hipótesis también se refleja en el texto (figura 2.7).

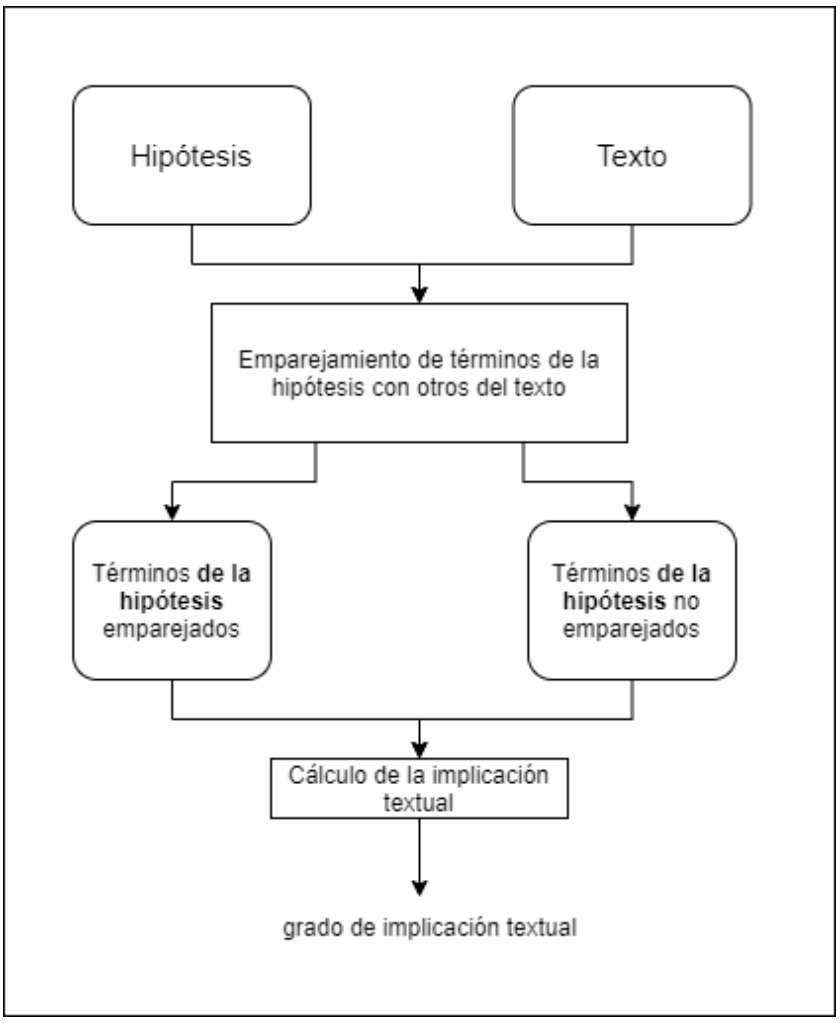

Figura 2.7: Diagrama que representa cómo se calcula la implicación textual.

Aunque, a priori, esta tarea puede parecerse a la semejanza semántica, la implicación textual tiene algunas características que la diferencian.

Mientras que la semejanza semántica es una operación conmutativa, la implicación textual no lo es. En esta última tan solo se analiza si el texto T contiene la hipótesis H, ignorando si T contiene más información que la planteada en H.

Tal y como se puede ver en la figura 2.8, si intercambiásemos los valores de H y T, no necesariamente obtendríamos los mismos resultados. Por lo tanto, no es una operación conmutativa.

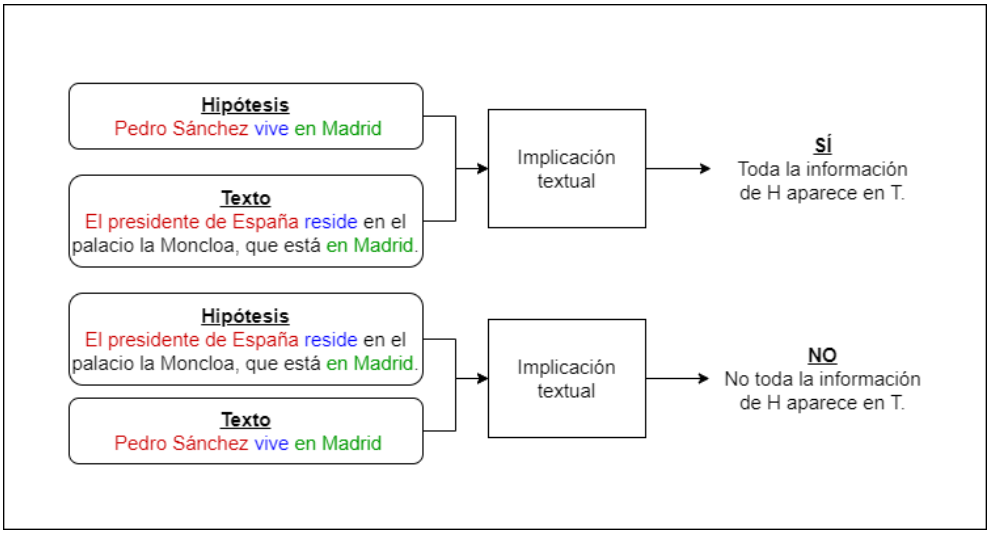

Figura 2.8: Ejemplo concreto de implicación textual.

2.2.4. Detección de paráfrasis en textos

La paráfrasis consiste en la reescritura o el replanteamiento de un documento (total o parcialmente) con la finalidad de crear uno nuevo. Para su detección, se aplican distintas técnicas para analizar el contenido de un texto para así buscar dicho contenido en otros textos e inferir que se ha producido la paráfrasis.

Para realizar esta tarea, también suelen aplicarse algunas de las técnicas descritas en el apartado de la semejanza semántica. En la mayoría de los casos, las aproximaciones que tratan de resolver este problema asocian términos o fragmentos de un texto con otros fragmentos de otro texto (que potencialmente parafrasean al original) y estudian si semánticamente son equivalentes. Si esto sucede, se infiere que existe paráfrasis respecto al texto original.

Tal y como se puede ver, viendo la naturaleza del problema que se trata, tiene muchos aspectos en común con la semejanza semántica. Esta es la razón por la que se utiliza en muchas de las tareas en las que también se utiliza la semejanza semántica: extracción de información, generación automática de resúmenes, etc.

Aún así, el propósito de los sistemas que resuelven esta tarea difiere considerablemente de los que resuelven la semejanza semántica. La principal diferencia radica en que, en la mayoría de los casos, los resultados devueltos pueden suelen tener dos posibles valores: existe paráfrasis o no. Por otro

lado, en el caso de la semejanza semántica, suelen especificarse distintos grados de semejanza, estableciendo más valores intermedios que en la detección de paráfrasis. En la figura 2.9 se ejemplifica tanto la detección de paráfrasis como esta diferencia.

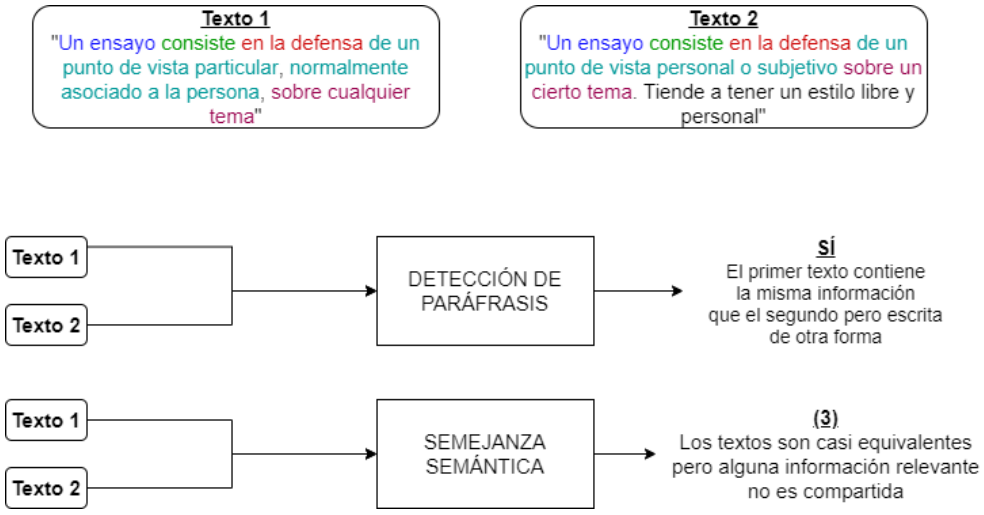

Figura 2.9: Ejemplo de detección de paráfrasis y diferenciación con semejanza semántica.

Capítulo 3

Fundamentos

Contenido: En este se exponen los fundamentos que dan lugar al sistema propuesto. Incluye explicaciones sobre estos, sobre su presencia en la literatura y sobre las distintas técnicas o recursos que engloban y sonde especial utilidad para la resolución del problema propuesto.

3.1. Similitud entre palabras

Las tareas descritas en el apartado anterior (sin incluir la extracción de información) tienen varios aspectos en común. Uno de los más destacables es que, para resolver el problema correspondiente, la mayoría de las aproximaciones propuestas tienden a asociar términos de los dos textos evaluados en función de su semejanza (ya sea semántica o formal). Para cada término de un texto (si es posible) se asocia el término del otro texto con el que tiene un mayor grado de semejanza semántica.

Entonces, cómo se asocian o cómo se calcula la semejanza entre términos de distintas sentencias es un aspecto clave de cada implementación. Qué método se utiliza repercutirá en los resultados obtenidos y los recursos requeridos.

En la literatura, se contemplan principalmente tres métodos para determinar el grado de semejanza existente entre dos palabras.

El método más trivial se basa en los caracteres que contienen los términos a comparar, ignorando su significado, el contexto en el que aparece, etc. En función de los caracteres comunes de ambos términos, del sufijo común más largo o del número de transformaciones necesarias para obtener uno a partir

del otro, estos métodos calculan el nivel de semejanza entre ambas palabras.

Por otro lado, existen los métodos topológicos que, para comparar dos términos, acceden a bases de conocimiento estructurado para obtener información sobre ambos y, a partir de esa información, calculan su semejanza.

Por último, existen los métodos basados en corpus. Estos, en lugar de acceder a bases de conocimiento estructurado, analizan distintas fuentes de conocimiento desestructurado para obtener información a partir de la cual puedan calcular la semejanza entre los términos correspondientes.

En las siguientes secciones (3.2, 3.3 y 3.4) se describen de forma detallada los principales aspectos asociados a cada uno de estos métodos.

3.2. Métodos basados en la semejanza entre cadenas de caracteres

Son aquellos métodos que realizan la comparación de dos términos centrándose exclusivamente en las cadenas de caracteres que los representan. Se basan en la premisa de que si dos términos tienen una representación similar (es decir, las cadenas de caracteres que los componen son similares) los propios significados de los términos también lo serán.

Para realizar la comparación, no evalúan los potenciales significados de los términos. En su lugar, modifican y comparan fragmentos o caracteres de los términos a analizar para así poder calcular el grado de similitud semántica que presentan entre sí.

Esta categoría incluye, en la mayoría de los casos, aproximaciones menos efectivas y complejas que las que veremos en las secciones 3.3 y 3.4. Aún así, algunas de estas son de especial utilidad para suplir las carencias de los demás enfoques.

Si no disponemos de información suficiente o directamente no tenemos ninguna información para comparar dos términos en base a su significado, la comparación no se puede realizar correctamente utilizando aproximaciones como las que se explican abajo. En este caso, se suelen utilizar aproximaciones como las que se muestran en este apartado.

De entre todos estos métodos, podemos destacar los siguientes:

- **Longest Common Substring (LCS)**: algoritmo en el que la semejanza entre dos términos se determina a partir del *substring* más largo común entre ellos. Se obtiene a partir de la fórmula de la figura 3.1, donde *s1* y *s2* son las cadenas de caracteres que se comparan, m y n son sus tamaños (respectivamente) y *LCSuff* es una función que obtiene el sufijo común más largo de ambas cadenas.

$$LCSubstr\,(S1, S2) = \max_{1 \le i \le m, 1 \le j \le n} LCSuff\,(S1_{1\ldots i}, S2_{1\ldots j}),$$

Figura 3.1: Fórmula para calcular el LCS.

- **Damerau-Levenshtein**: calcula la distancia existente entre dos términos en función del número de transformaciones requeridas para transformar uno de los términos en el otro. Las transformaciones contempladas son inserciones y eliminaciones de caracteres o la transposición de dos caracteres adyacentes.
- **Jaro**: define una distancia donde 0 implica la ausencia total de similitud y 1 significa similitud absoluta. Se obtiene a partir de la fórmula de la figura 3.2, donde *s1* y *s2* son las cadenas de caracteres que se comparan, m es el número de caracteres que encajan entre ambos y t es la mitad del número de transposiciones.

$$d_j = \begin{cases} 0 & \text{if} m = 0 \\ \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right) & \text{otherwise} \end{cases},$$

Figura 3.2: Fórmula para calcular la distancia JARO.

- **n-gramas**: se caracteriza por su simpleza y su escalabilidad. Consiste en el uso de un modelo para predecir el siguiente caracter de una secuencia de (n-1) caracteres. Se utilizan los resultados de las predicciones para realizar la comparación de los términos evaluados.

3.3. Métodos topológicos o basados en conocimiento estructurado

También conocidos como aproximaciones topológicas, incluyen un conjunto de técnicas que acceden a conocimiento estructurado para determinar

la relación existente entre distintos términos.

El conocimiento estructurado utilizado por estos métodos se almacena en bases de conocimiento. En su mayoría, tienden a estructurar su contenido en base a las relaciones semánticas (sinonimia, hiperonimia, etc) que existen entre los elementos (términos, palabras, textos, etc) que contemplan, tal y como se muestra en la figura 3.3. En función de las relaciones que se establezcan entre dos conceptos en una base de conocimiento, podemos determinar cuál es su similitud.

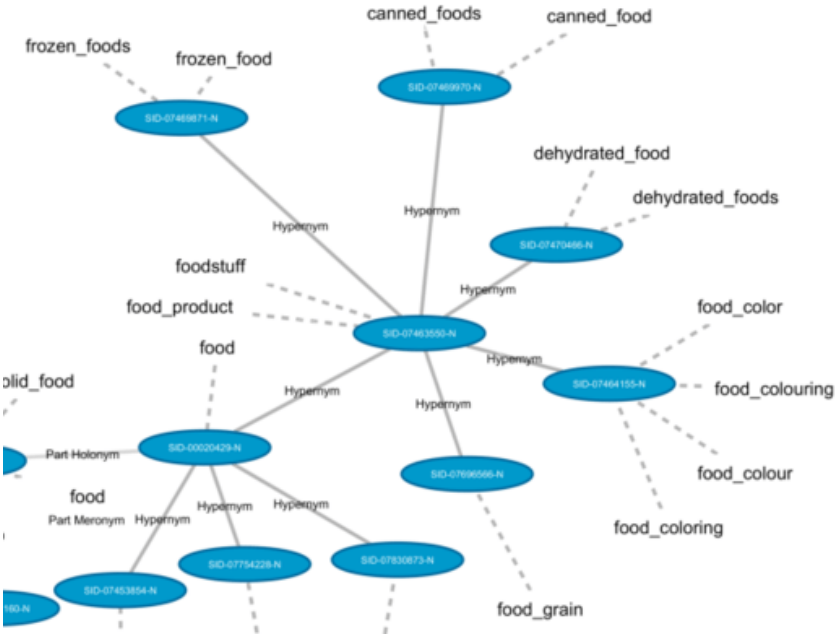

Figura 3.3: Ejemplo de conocimiento (conceptos) estructurado en función de su relación semántica. Imagen encontrada en [7].

Aunque la mayoría de las bases de conocimiento (asociadas a estos métodos) compartan esta característica, la forma de representar los elementos de cada una cambia considerablemente en cada caso. Podemos encontrar elementos representados mediante ontologías, como elementos de un grafo o una red (*WordNet*), como elementos de un diccionario, etc.

La mayoría de estas bases de conocimiento se han creado de forma manual. Gracias al esfuerzo, tanto material como humano, de los responsables de las bases de conocimiento, este tipo de métodos se han convertido en una de las opciones más utilizadas en tareas de extracción de información (*Information Retrieval*), cálculo de semejanza semántica, etc.

Entonces, atendiendo a la información de dichas bases y a cómo se posicionan los conceptos a comparar, podemos calcular la semejanza semántica entre ambos. Podemos diferenciar principalmente entre dos metodologías tradicionales para el cálculo de la semejanza entre dos conceptos a partir de conocimiento estructurado:

3.3.1. Centrada en nodos

Se establecen taxonomías IS-A entre los conceptos que contiene una cierta base de conocimiento. Cada nodo representa un concepto único. Para calcular la semejanza entre dos nodos, se evalúa cuanta información comparten en común. Esto implica buscar (de forma ascendente) su clase en común.

La clase en común de dos conceptos (dos nodos de la taxonomía) consiste en aquel primer nodo común de la misma que ambos van a alcanzar si se desplazan por el grafo de forma ascendente. Es aquel primer concepto (que se encuentra a una mayor profundidad) que engloba ambos conceptos en la taxonomía IS-A.

Tal y como están planteadas este tipo de taxonomías, cuanto más profunda sea la clase en común de dos conceptos, más información en común tendrán entre ellos. En la figura 3.4, ”euro” y ”céntimo” tienen más en común que ”euro” y ”tarjeta de crédito” ya que la profundidad del concepto/clase en común en el primer caso (”Moneda”) es mayor que en el segundo (”Método de pago”).

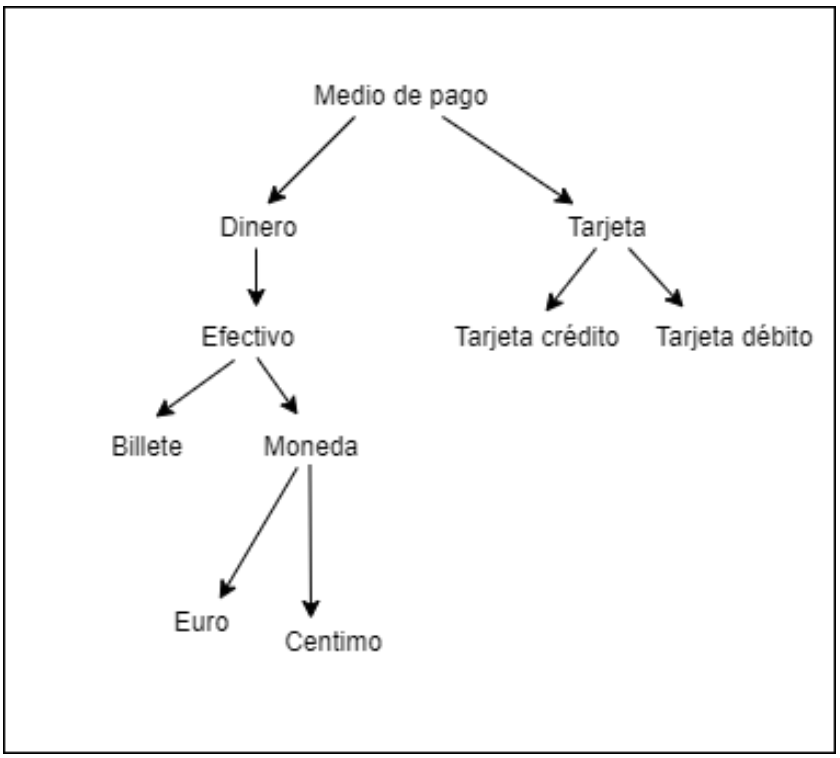

Figura 3.4: Ejemplo de taxonomía IS-A.

3.3.2. Centrada en las aristas

Consiste en la forma más directa de calcular la semejanza semántica entre dos términos ubicados en una cierta taxonomía o estructura (por ejemplo, la que muestra la figura 3.5). La semejanza se calcula a partir del número de aristas que existan entre los conceptos que se comparan.

No todas las aristas tienen el mismo peso en el cálculo. Su valor dependerá de factores como la relación que establecen (sinonimia, instancia de, etc), la profundidad del nodo, de densidad del grafo, etc. Es importante destacar que pueden aparecer varios caminos que relacionen dos conceptos. El seleccionado será aquel cuyo valor calculado sea mejor.

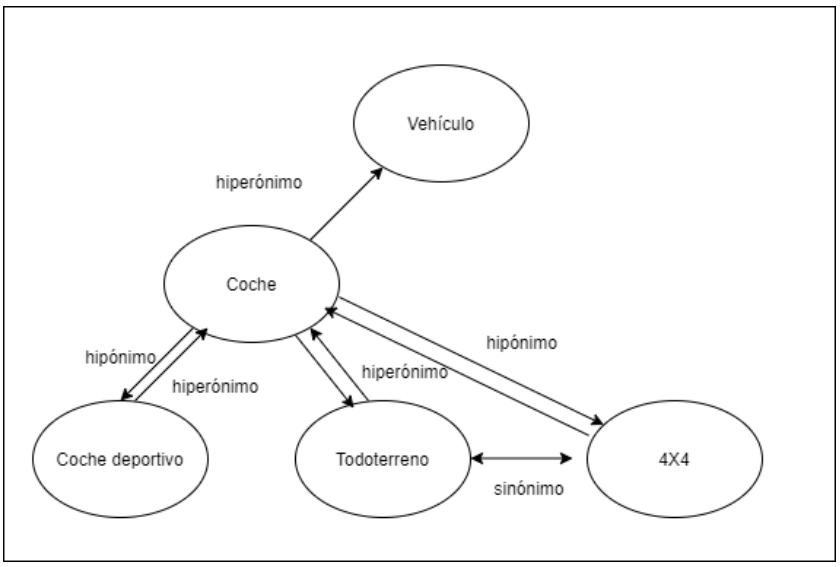

Figura 3.5: Ejemplo de relaciones que se establecen entre conceptos.

Este tipo de aproximaciones, con el paso del tiempo, han alcanzado una relevancia mayor de la que tienen las aproximaciones centradas en nodos, haciendo que estén prácticamente en desuso. Esto se debe, principalmente, a su accesibilidad y manejabilidad, a su estructuración de alto nivel y al encaje que tienen con los recursos estructurados presentes en la literatura actual.

3.3.3. Recursos estructurados

Existen distintos recursos estructurados, tanto de uso general como más especializado, que pueden aplicarse para aplicar los métodos descritos anteriormente. De uso general, de entre todos estos, podemos destacar dos: Wikipedia (Wales y Sanger, 2001) y WordNet (Fellbaum et al., 1990).

3.3.3.1. Wikipedia

Wikipedia (Wales y Sanger, 2001) no consiste en una base de conocimientos que almacene información léxica de forma explícita. En su lugar, consiste en una colección de artículos en varios idiomas y sobre distintas materias que se han escrito desde 2001 de forma colaborativa por voluntarios de la plataforma.

Lo que hace que Wikipedia sea de especial interés es que, entre otros aspectos, desde su creación se han establecido relaciones entre los artículos que la componen. Los artículos de Wikipedia están relacionados entre sí de

distintas formas (temática similar, atributo de, referencia a, etc) de modo que es posible navegar entre los distintos artículos que incluye utilizando estas relaciones.

Gracias a este factor, en la literatura existen múltiples aproximaciones o técnicas (principalmente basadas en aristas) que la utilizan para calcular la semejanza entre dos conceptos:

- **Wikicorpus** (Reese et al, 2010): Corpus trilingüe (catalán, español e inglés) que almacena el significado y la definición (en formato WordNet) de más de 750 millones de términos. Dicha información se ha extraído a partir de múltiples fragmentos de Wikipedia que se han procesado utilizando distinta información lingüística. Es un ejemplo de aproximación basada en aristas ya que las relaciones que se establecen entre los elementos del corpus son de hiperonímia, sinonímia, etc. y a su vez, a la hora de calcular la semejanza entre dos elementos, las distintas relaciones que contempla no tienen el mismo peso en el cálculo.
- **WikiRelate!** (Strube y Ponzetto, 2006): Sistema que busca todos los artículos de Wikipedia en los que aparecen los términos a comparar y aplica una variación del algoritmo de *Lesk* (Lesk, 1896) para determinar su semejanza. En este caso, realiza un cálculo sobre los fragmentos de texto que se solapan en los distintos artículos para calcular la semejanza entre ambos términos.
- **Wikipedia Link Vector Model** (Milne y Witten, 2008): A partir de los enlaces y títulos de los artículos de Wikipedia, genera un modelo vectorial que incluye los distintos términos que aparecen en los artículos. Para posicionar cada término, se utiliza la probabilidad asociada a los enlaces de los artículos.
- **WikiWalk** (Yeh et al., 2009): Utiliza las relaciones entre artículos para generar un grafo. Cada artículo se considera un nodo del grafo y los enlaces son las aristas. A partir de esta información, aplica *Explicit Semantic Analysis* (Gabrilovich y Markovitch, 2007) y una versión modificada del algoritmo de *Pagerank* para generar un modelo vectorial que contemplará todos los términos de los artículos.

3.3.3.2. WordNet

Por otro lado, WordNet (Fellbaum et al., 1990) es una base de datos léxica que, con el paso del tiempo, se ha convertido en el recurso estructurado más utilizado en las tareas de PLN que se describen en este documento.

Inicialmente, tan solo almacenaba conocimiento léxico para términos en inglés. En la actualidad, existen múltiples implementaciones de WordNet que incluyen otros idiomas: *EuroWordNet*, *BalkaNet*, *Asian WordNet*, *Open Multilingual WordNet* (Bond y Foster, 2013), etc. Este último destaca por ser una de las versiones más utilizadas en la literatura actual y por soportar más de 200 idiomas distintos (entre los cuales incluye el español).

La mayoría de estas implementaciones se han creado de forma manual a lo largo del tiempo por múltiples instituciones.

La unidad de conocimiento que plantea WordNet son los *synsets*. Un *synset* consiste en un significado o una realidad concreta. También se define como un conjunto de palabras que tienen la misma POS (*Part of Speech*) y hacen referencia a una realidad (a un significado) concreto.

Como los términos de un *synset* tienen el mismo significado, pueden intercambiarse entre sí en un cierto contexto. Por otro lado, como una palabra puede tener múltiples significados, puede suceder que un término aparezca en múltiples *synsets*. Un ejemplo sería el término "banco". En este caso pertenecería al *synset* que hace referencia a la agrupación de peces, a la institución financiera, al mobiliario urbano para sentarse, etc.

Los *synsets* que componen la base de datos léxica se relacionan entre sí mediante 26 posibles relaciones: hiperonimia/hiponimia, sinonimia/antonimia, meronimia, consecuencia lógica, atributo de, etc. El resultado de los conceptos (*synsets*) y sus relaciones es una red (un grafo) de conocimiento léxico estructurado.

Para calcular la semejanza entre dos conceptos, se analizan los caminos que unen ambos *synsets* y se realiza el cálculo correspondiente. Este cálculo se realiza aplicando distintas técnicas basadas en aristas (apartado 3.3.2). En la literatura, aparecen principalmente 7 aproximaciones:

- **Path-Length Measure** (Rada et al., 1989): funciona bajo la premisa de que cuanto más corto sea el camino entre dos *synsets*, más parecidos son entre ellos. El resultado es un número real entre 0 (no tienen ningún parecido) y 1 (son exactamente iguales).

- **Leacock-Chodorow Measure** (Leacock y Chodorow, 1998): extiende la medida anterior priorizando los caminos más cortos entre la hoja (el *synset*) y la raíz de la jerarquía establecida. La semejanza entre los dos términos consiste en el valor generado por la fórmula *-log(p/2d)*,

siendo p el camino más corto entre los *synsets* y d la profundidad de la taxonomía. Los resultados obtenidos, por lo tanto, no tienen porque ubicarse entre 0 y 1.

- **Wu-Palmer Similarity** (Wu y Palmer, 1994): se basa en la profundidad de los *synsets* en la taxonomía y en su LCS (*Least Common Subsumer* o en el antecesor común más específico). El resultado devuelto es un valor real entre 0 y 1 (no tienen ningún parecido y son exactamente iguales, respectivamente).

- **Resnik Similarity Measure** (Resnik, 1995): Se basa en la información que comparten los conceptos a comparar. Cuanta mayor sea la información que comparten, mayor será su semejanza. Para realizar el cálculo, se basa en el IC (*Information Content*) de su LCS. Los resultados obtenidos no se acotan a los presentes en el rango de reales de 0 a 1.

- **Lin Similarity Measure** (Lin, 1997): Consiste en una extensión de la técnica anterior pero incluyendo los conceptos de *commonality* (aspectos en común) y *difference* (diferenciación). Utiliza la siguiente fórmula para lograr su propósito: 2 * IC(lcs) / (IC(s1) + IC(s2)). Los resultados obtenidos se encuentran en el rango de reales de 0 a 1.

- **Jiang-Conrath Distance** (Jiang y Conrath, 1997): A diferencia de las demás medidas, realiza el cálculo en base a la falta de relación entre los dos términos a comparar. Utiliza la siguiente fórmula para lograr su propósito: 1 / (IC(s1) + IC(s2) - 2 * IC(lcs)). Los resultados obtenidos se encuentran en el rango de reales de 0 a 1.

- **Hirst-St.Onge Measure** (Hist y St. Onge, 1998): Define varios niveles de fuerza en las relaciones entre conceptos dentro de la red de WordNet. También define distintas direcciones en los desplazamientos dentro de la red. Calcula la semejanza entre dos conceptos a partir de la fuerza de las relaciones que se encuentre y del número de cambios de dirección que realice a lo largo del camino.

Utilizando el grafo presentado por WordNet, se pueden realizar distintas aproximaciones para calcular la semejanza semántica entre dos términos. (Hugues y Ramage, 2007) aplican un algoritmo de *random walk* sobre el contenido de WordNet y otros como (Agirre et al., 2009) utilizan la información de WordNet para establecer una potencial distribución de los términos en un espacio vectorial y así poder calcular su similitud a partir de la semejanza de cosenos.

3.4. Métodos estadísticos o basados en corpus.

Esta categoría engloba un conjunto de técnicas o métodos que tienen la capacidad de extraer información de fuentes de conocimiento desestructurado. En PLN, el concepto de conocimiento desestructurado consiste, en su mayoría, en colecciones de textos o documentos en lenguaje natural.

El área de investigación que desarrolla y estudia este tipo de técnicas recibe el nombre de semántica distribucional (*Distributional Semantics*). Esta disciplina (y por lo tanto las técnicas relacionadas) se basa en la hipótesis distribucional.

Dicha hipótesis afirma que los contextos en los que aparecen los términos (y otros aspectos como la frecuencia de aparición, su discriminación entre documentos, etc) determinan el significado de los propios términos. Por lo tanto, los términos que se usan y aparecen en los mismos contextos, tienden a significar lo mismo.

Basándose en esto, estas técnicas aplican técnicas estadísticas y analizan las propiedades distribucionales (contexto en el que aparecen, frecuencia de uso, etc) de una cierta colección para extraer información sobre los términos que la componen.

3.4.1. Latent Dirichlet Allocation (LDA)

Consiste en una técnica de aprendizaje no supervisado (Blei et al., 2003) utilizada principalmente para la extracción de temas asociados a una colección de documentos. También tiene otros usos como el análisis del sentimiento de un texto, localización de objetos en imágenes, etc. Con el paso del tiempo, también se ha utilizado la información extraída mediante esta técnica para determinar la semejanza entre dos términos.

Dada una colección de documentos, LDA tiene la capacidad de identificar los temas aparecen en la misma. Para ello, se basa en las siguientes premisas:

1. Cada documento consiste en una mezcla de temas (figura de la izquierda de 3.6).
2. Cada tema consiste en una mezcla de palabras (figura de la derecha de 3.6).

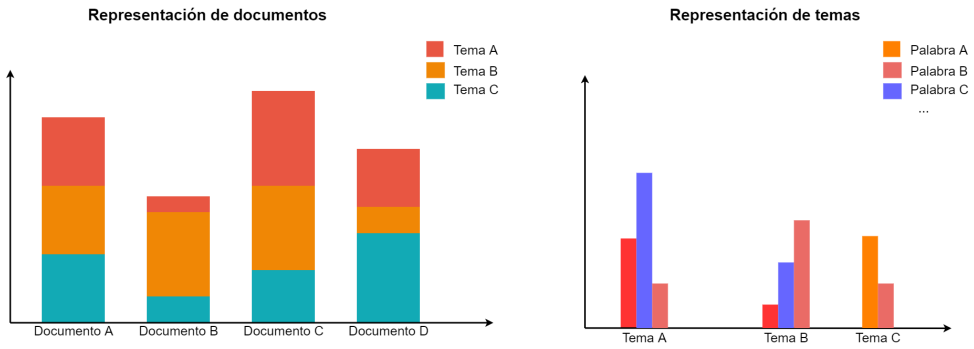

Figura 3.6: Representación de documentos y temas utilizando LDA.

Entonces, basándonos en las distribuciones de las palabras de un fragmento o de la totalidad de un documento, podemos determinar el tema (o los temas) que contiene. Tan solo debemos analizar el tema que tiene la distribución más parecida y así tendremos su tema asociado.

Al utilizar este método, obtenemos tanto los temas asociados a un documento como las palabras asociadas a cada tema. Esta última información es la que nos puede ser de utilidad para determinar la semejanza entre dos términos.

Aunque hay distintos enfoques, la mayoría de las aproximaciones que utilizan la información extraída de LDA concluyen en que dos términos son semejantes si son del mismo tema o en función del número de temas que comparten. Si no tienen asociado ningún tema en común, no son similares.

Debido a su simpleza a la hora de calcular la semejanza entre dos términos, esta técnica está prácticamente en desuso para esta tarea. Por otro lado, con el tiempo, ha tomado una cierta relevancia a la hora de determinar la relevancia de los términos de un cierto vocabulario y a la hora de aplicar una reducción de las dimensiones de un espacio vectorial.

3.4.2. Latent Semantic Analysis (LSA)

También consiste en una técnica de aprendizaje no supervisado. Su principal objetivo es el de obtener representaciones del contenido de una colección de documentos en función de información de las *hidden layers* o *topics*. Para ello, esta técnica aplica dos fases diferenciadas:

La primera consiste en la obtención de la matriz de palabras por documentos. En esta, se indica qué términos del vocabulario de la colección

utilizada aparecen como mínimo una vez en cada documento.

La segunda fase, llamada *Singular Value Descomposition*, es la encargada de reducir la matriz generada en la fase anterior y obtener tantos *topics* como se especifique en la implementación. Estos consisten en las columnas resultantes del truncamiento de la matriz original (la matriz de palabras por documentos).

El resultado de este proceso es una nueva matriz que asocia cada documento con un valor para cada *topic*. A partir de ello, se puede obtener el peso de cada término del documento en cada *topic*, lo que a su vez consiste en una representación de los términos.

Esta representación, en forma de vector cuyos elementos son los términos asociados a cada *topic*, se puede utilizar para determinar la semejanza entre los términos (tal y como veremos en la sección 3.4.5).

El proceso descrito en los últimos párrafos se ejemplifica en la siguiente figura:

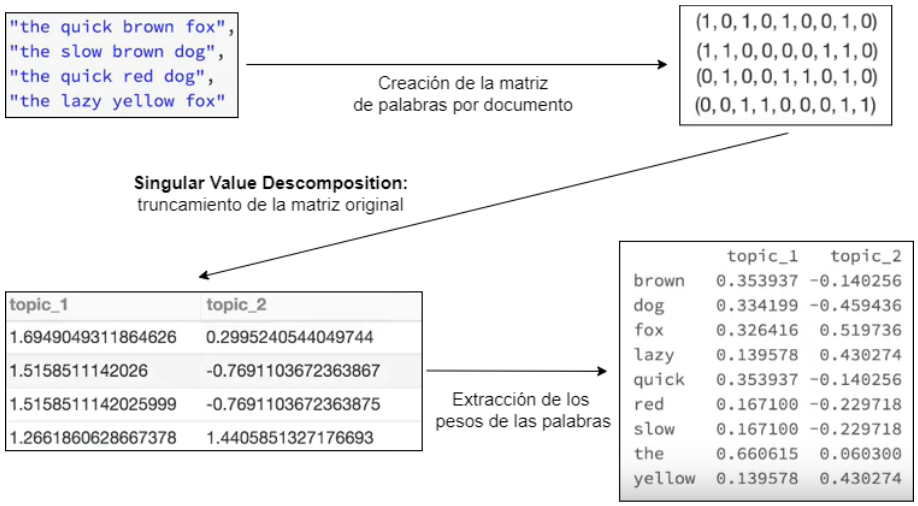

Figura 3.7: Diagrama que ejemplifica las fases de LSA.

3.4.3. Hyperspace Analogue to Language (HAL)

Consiste en una técnica (Lund y Burgress, 1996) de aprendizaje no supervisado que realiza su función analizando agrupaciones de palabras de un corpus a las que llama ventanas (*window*).

Su premisa principal es que las palabras con significados similares tienden a aparecer juntas. Por lo tanto, es de interés determinar cómo co-ocurren los términos para así obtener su significado.

Para ello, crea una matriz de co-ocurrencia (figura 3.8) entre los términos que incluye la *window*. En esta, para un término *a*, se indica cuántas veces aparece junto al término *b*. Estas apariciones, se miden con distintos pesos. Por ejemplo, cuando dos palabras aparecen una al lado de la otra, esa co-ocurrencia suma X unidades, mientras que cuando dos términos aparecen separados por otro término, suma X-1 unidades.

	Studying	animal	vocalizations	is	fun
Studying	0	0	0	0	0
animal	4	0	0	0	0
vocalizations	3	4	0	0	0
is	2	3	4	0	0
fun	1	2	3	4	0

Figura 3.8: Ejemplo de una matriz de co-ocurrencia.

El resultado, tal y como se puede ver en la imagen, es una colección de vectores que representan los términos de la *window*. Estos vectores, pueden ubicarse en un espacio vectorial y, además, es posible calcular su semejanza utilizando las técnicas del apartado 3.4.5.

3.4.4. Word2Vec

Al igual que las demás técnicas descritas en este apartado, Word2Vec (Mikolov et al., 2013) permite generar representaciones en forma de vectores en un espacio vectorial a partir de términos en lenguaje natural utilizando información estadística y contextual de los mismos.

Estas representaciones reciben el nombre de *word embeddings*. Este término no es exclusivo de esta aproximación. Muchas técnicas, como las descritas en los apartados anteriores entre otras, también lo utilizan.

Para realizar esta labor, Word2Vec utiliza redes neuronales. Concretamente, utiliza redes neuronales de tres capas (las capas de entrada, de proyección y de salida) tal y como se muestra en la figura 3.9. Mediante el

aprendizaje, analizan y aprenden distinta información sobre el contexto lingüístico en el que aparecen los términos que componen una colección de datos para generar representaciones en forma de vector.

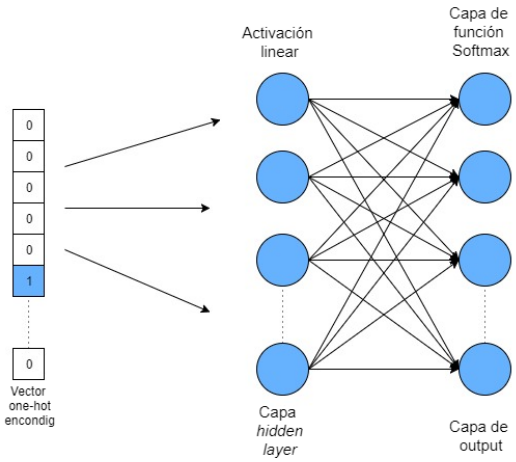

Figura 3.9: Representación de la red neuronal de tres capas utilizada por *Word2Vec*. La primera capa es la capa de entrada. Imagen encontrada en [1].

Esta es la aproximación que tiene más presencia en la literatura actual. Esto se debe, entre otros aspectos, a su eficiencia nivel computacional, a la calidad de resultados que ofrece y a su gran capacidad de personalización y de ajuste.

Al utilizar redes neuronales, por lo tanto, este tipo de modelos requieren entrenamiento previo. Durante este entrenamiento, el modelo obtiene los *word embeddings* asociados a cada término del vocabulario utilizado.

Una vez realizado este proceso, el modelo puede traducir de forma instantánea cualquier término (presente en el vocabulario del aprendizaje) a su representación en forma de vector.

Entonces, la operación más relevante de estos modelos es la de aprendizaje. Una vez se han obtenido los *embeddings* durante el aprendizaje, simplemente se limita a ”traducir” en función de las asociaciones que ha hecho en el proceso de aprendizaje.

Para realizar este proceso, Word2Vec propone dos procesos distintos pero algorítmicamente similares: *Continous Bag-Of-Words* (CBOW) y *Skip-gram*.

3.4.4.1. Continous Bag-Of-Words

Este modelo se caracteriza por generar las representaciones (*word embeddings*) de los términos en base al contexto en el que aparecen. Dado un término, obtiene su representación en forma de vector en función de los términos que lo rodean en las distintas ubicaciones donde aparece.

El número de términos que se utilizan depende de un parámetro configurable: el tamaño de la ventana o *window*. Este determinará cuántos términos se utilizan para realizar esta codificación. Si, por ejemplo, el tamaño es 4, se utilizarán los 4 términos que preceden a la palabra a codificar y los 4 que aparecen a continuación de él.

La red neuronal que realiza este aprendizaje y que genera las representaciones descritas se muestra en la siguiente imagen:

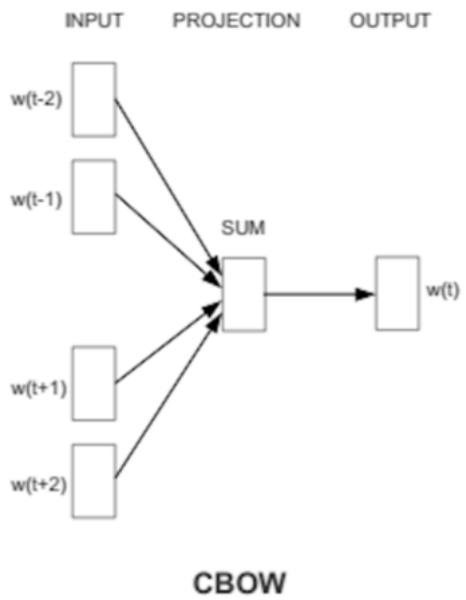

Figura 3.10: Representación de la red neuronal de tres capas utilizada por CBOW. Imagen encontrada en [19].

Consiste en una red neuronal de tres capas: la capa de entrada (*input*), la cada de proyección (*projection*) y la capa de salida (*output*). Aunque estos modelos son altamente personalizables, en la mayoría de las implementaciones:

- La capa de entrada tiene tantas neuronas como términos utilizados

para generar la representación. En la imagen, el tamaño de la ventana es 2.

- En la capa de proyección se combina la información asociada a los términos que rodean a la palabra a evaluar. Esta combinación consiste en la suma de las representaciones.
- La capa de salida es única y común para todos los términos. Esto se debe a que, durante el aprendizaje, este modelo tan solo genera una salida (la correspondiente a la codificación del término a evaluar).
- En la mayoría de las aproximaciones de este tipo, el orden de los términos que rodean a la palabra evaluada no afecta a los resultados obtenidos.

CBOW tiende a ofrecer mejores resultados frente a colecciones de datos pequeñas y su tiempo de ejecución (respecto a *Skip-gram*) es considerablemente menor.

3.4.4.2. Skip-gram

Representa el proceso inverso a CBOW. Durante el aprendizaje, en lugar de predecir la representación de un término en función de su contexto, predice la representación de las palabras del contexto en función de la palabra original.

Dado un término, predice el conjunto de que le rodearían. El número de términos a predecir también dependerá de un parámetro configurable llamado el tamaño de la ventana o *window*. Normalmente, se predicen los cuatro términos que preceden al texto evaluado y los cuatro que le siguen.

La red neuronal que utiliza este modelo es una red neuronal de tres capas:

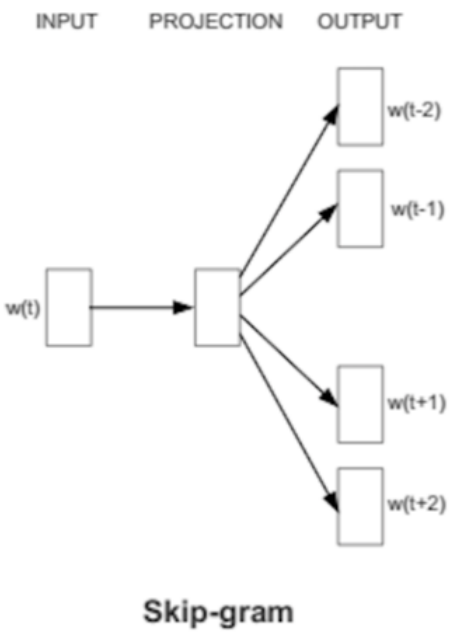

Figura 3.11: Representación de la red neuronal de tres capas utilizada por Skip-gram. Imagen encontrada en [19].

Consiste en una red neuronales de tres capas: *input*, *projection* y *output*. En la mayoría de las implementaciones:

- La capa de entrada (*input*) tan solo tiene un elemento. Este será el término a partir del cual se inferirá el resto (el contexto).
- La capa de proyección (*projection*) aplica la función de *softmax* jerárquica (o su modificación posterior llamada *negative sampling*).
- La capa de salida (*output*) tiene un elemento por término del contexto a inferir. Utiliza el resultado de la capa de proyección (de la función *softmax* o *negative sampling*).

Skip-gram ofrece mejores resultados frente a corpus grandes y representaciones que requieran muchas dimensiones. Por otro lado, su tiempo de ejecución necesario es mayor.

3.4.5. Espacio vectorial y distancia entre dos términos

Como se puede observar, la mayoría de las aproximaciones descritas en el apartado 3.4 analizan el contenido de un corpus de documentos para extraer información sobre los términos que contienen. Con esta información,

que consiste en un conjunto de atributos, generan representaciones de los propios términos.

Dichos métodos se implementan de modo que dos términos que se parezcan tendrán valores similares en los atributos. Esto se debe a que se basan en la hipótesis distribucional y en otros datos estadísticos.

Entonces, si ubicamos los términos en un espacio vectorial en función de los valores de sus atributos, cuanto más cerca se encuentren dos términos en el espacio vectorial más similares serán, mientras que cuanto más distantes se ubiquen, más diferentes serán entre sí.

Por lo tanto, para medir la semejanza entre términos que están ubicados en un espacio vectorial, debemos calcular la distancia existente entre ellos. Existen distintas métricas o formas de medir la distancia:

- **Block distance**: también llamada distancia *Manhattan*. Consiste en la distancia entre dos puntos (p y q), siendo estos puntos los marcados por los vectores asociados a los términos.

$$d_1(p,q) = \|p-q\|_1 = \sum_{i=1}^{n} |p_i - q_i|.$$

Figura 3.12: Fórmula de *Block distance*.

- **Cosine Similarity**: consiste en el coseno del ángulo que se genera entre dos vectores: los vectores asociados a cada término (en la siguiente fórmula, A y B). Dos términos ubicados en la misma posición del espacio vectorial tendrán semejanza de cosenos de 1, mientras que si están en un ángulo de 90 grados, tendrán una semejanza de 0. Es la técnica más utilizada en la literatura actual.

$$cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

Figura 3.13: Fórmula de *cosine similarity*.

- **Soft Cosine Similarity**: consiste en una variante de la medida anterior. En ella, también se tiene en cuenta la similitud de los distintos atributos o dimensiones del espacio vectorial generado.

$$oft\_cos(\theta) = \frac{\sum_{i,j=1}^{n} s_{ij} A_i B_j}{\sqrt{\sum_{i,j=1}^{n} s_{ij} A_i A_j} \sqrt{\sum_{i,j=1}^{n} s_{ij} B_i B_j}},$$

Figura 3.14: Fórmula de *soft cosine similarity*.

- **Euclidean Distance**: consiste en la fórmula clásica para calcular la distancia entre dos puntos (en la imagen 3.15, los puntos s y t) en un espacio euclídeo.

$$d(s,t) = d(t,s) = \sqrt{\sum_{i=1}^{n} (t_i - p_i)^2}.$$

Figura 3.15: Fórmula para calcular la distancia euclídea.

- **Jaccard Index**: es una medida que sirve para calcular la diversidad y la similitud entre dos conjuntos (en la figura 3.16, los conjuntos A y B) finitos de elementos. Tiene su origen en la estadística pero, con el paso del tiempo, se ha acabado utilizando también para determinar la semejanza entre distintos términos. Cada término se representa como un conjunto cuyos componentes son los elementos del vector.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

Figura 3.16: Fórmula para calcular el *Jaccard Index*.

- **Simple Matching Coefficient (SMC)**: coeficiente que sirve para determinar la similitud y la diversidad entre dos objetos. Un objeto, en nuestro caso, puede consistir en una colección de atributos binarios.

$$SMC = \frac{Number\ of\ Matching\ Attributes}{Total\ no.\ of\ Attributes} = \frac{a_{00} + a_{11}}{a_{00} + a_{01} + a_{10} + a_{11}},$$

Figura 3.17: Fórmula para calcular el SMC.

- **Overlap Coefficient**: también sirve para medir la similitud entre dos conjuntos finitos de elementos. Tal y como sucedía con el *Jaccard Index*, se empezó utilizando en estadística para después utilizarse para determinar la semejanza entre dos términos ubicados en un cierto espacio vectorial. Dados dos términos A y B (representados como conjuntos cuyos componentes son los elementos del vector), aplica la siguiente fórmula:

$$overlap\left(A, B\right) = \frac{|A \cap B|}{\min(|A|, |B|)}.$$

Figura 3.18: Fórmula para calcular el *Overlap Coefficient*.

3.5. Similitud entre sentencias

Para resolver la tarea propuesta, es necesario calcular el nivel de semejanza semántica existente entre dos sentencias. Las técnicas descritas en el apartado anterior funcionan a nivel de palabra. Por lo tanto, sirven tanto para determinar el significado de un término como para calcular la semejanza entre dos palabras. Por otro lado, aunque permitan calcular la semejanza existente entre términos, no sirven para calcular la similitud entre dos sentencias completas.

Para ello, se utilizan distintas técnicas que se aplican a nivel de sentencia. La mayoría de estos métodos calculan la semejanza entre dos sentencias a partir de los significados concretos de los elementos que componen los textos. Estos significados se pueden extraer con las técnicas descritas en las secciones 3.2, 3.3 y 3.4.

En la literatura actual, existen principalmente dos enfoques para obtener la semejanza semántica de dos sentencias: la semejanza por pares de palabras (*Pairwise similarity*) y la semejanza por codificación de sentencias (mediante el uso de *Autoencoders*).

3.5.1. Semejanza por pares de palabras

Este tipo de métodos calculan la semejanza semántica entre dos sentencias comparando pares de términos que componen las sentencias a comparar. Los pares que se establecen en función de la semejanza semántica de los propios términos. Cada término de cada sentencia se asociará (formará una

pareja) con el término de la otra sentencia con el que tenga un mayor nivel de semejanza semántica.

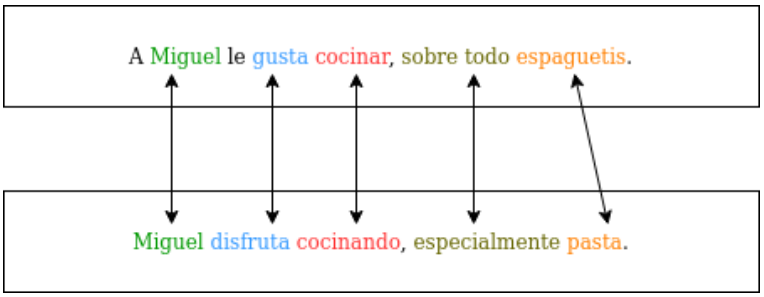

Figura 3.19: Ejemplo de emparejamiento de términos en función de su semejanza semántica.

Para determinar la semejanza semántica entre los términos y así poder establecer las parejas se utilizan los métodos descritos en el apartado 3.1.

Existen múltiples métodos de este tipo dentro de la literatura actual. Algunas de las primeras aproximaciones presentes en esta establecían la semejanza entre dos sentencias como la media aritmética de las semejanzas de los pares que componen ambas frases.

$$sim(S_1, S_2) = \frac{1}{2}\left(\frac{\sum_{w_i \in S_1} \text{máx}_{w_j \in S_2}\, sim(w_i, w_j)}{|S_1|} + \frac{\sum_{w_i \in S_2} \text{máx}_{w_j \in S_1}\, sim(w_i, w_j)}{|S_2|}\right)$$

Figura 3.20: Cálculo de la semejanza semántica utilizando la media aritmética de la semejanza de los términos.

Otros como el método propuesto por (Mihalcea y Strapparava, 2006) (figura 3.21) realizan un cálculo similar pero también atienden a la relevancia y a la especificidad de los distintos términos que componen las sentencias a comparar.

En este método, el peso que cada término tendrá en el cálculo de la semejanza dependerá de su *idf* (*Inverse Document Frecuency* o frecuencia inversa del documento). De este modo, los términos más específicos y que tiendan a aparecer menos tendrán más peso en la comparación mientras que los términos que hagan referencia a conceptos genéricos o que tiendan a aparecer en la mayoría de textos no influirán tanto en el resultado.

$$sim(S_1, S_2) = \frac{1}{2}\left(\frac{\sum_{w_i \in S_1} (idf(w_i) * \max_{w_j \in S_2} Sim(w_i, w_j))}{\sum_{w_i \in S_1} idf(w_i)} + \frac{\sum_{w_j \in S_2} (idf(w_j) * \max_{w_i \in S_1} Sim(w_j, w_i))}{\sum_{w_j \in S_2} idf(w_j)}\right)$$

Figura 3.21: Cálculo de la semejanza semántica propuesta por (Mihalcea y Strapparava, 2006).

En otras aproximaciones presentes en la literatura se utilizan planteamientos parecidos a este pero utilizando otras métricas como la frecuencia de aparición del término (*tf* o *Term Frecuency*), la longitud de los términos a comparar, el tipo de términos comparados, etc.

Incluso, en función del método seleccionado pueden aparecer distintas restricciones a la hora de realizar la comparación de pares. En el propio método propuesto por (Mihalcea y Strapparava, 2006) solo se pueden comparar términos con el mismo PoS (*Part-of-Speach*).

Es importante matizar que este tipo de métodos también sirven para comparar sentencias representadas a través de estructuras complejas. Estos métodos también pueden aplicarse sobre sentencias representadas mediante árboles, grafos (como en la figura 3.22) u otras estructuras, tal y como sucede en la propuesta de (Pakray et al., 2016). La única condición que debe cumplirse para ello es que los términos que componen las sentencias tengan la misma representación y sean comparables.

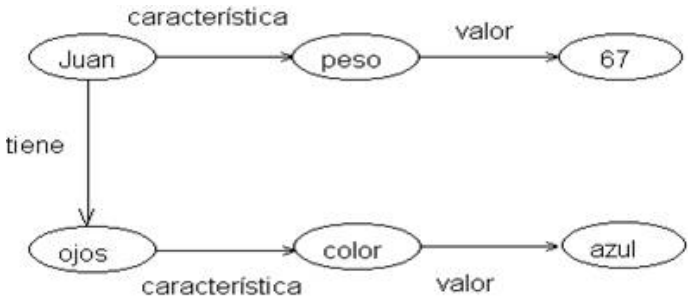

Figura 3.22: Representación mediante un grafo de la sentencia "Juan pesa 67 y tiene los ojos de color azul".

3.5.2. Semejanza por codificación de sentencias

Los métodos basados en comparación de pares utilizan las representaciones y los significados de los términos de dos sentencias para compararlas pero no tienen la capacidad de generar un significado asociado a la totalidad

de la sentencia. El tipo de métodos descritos en este apartado sí.

Los métodos que utilizan la codificación de sentencias para calcular la semejanza semántica tienen la capacidad de representar sentencias completas y de asociarles un significado. Esta representación la obtienen a partir de las representaciones de los términos que componen dichas sentencias.

Para utilizar este tipo de técnicas es necesario que la sentencia que se desea representar cumpla una condición: los términos que contiene deben estar codificados o representados mediante vectores dentro de un espacio vectorial y su ubicación debe identificar unívocamente su significado. No se pueden utilizar estas técnicas sobre sentencias cuyos términos estén representados, por ejemplo, mediante cadenas de caracteres.

Los términos representados mediante vectores son el resultado del uso de métodos estadísticos y basados en corpus (sección 3.4). Estos, para cada término, obtienen tanto una representación del mismo (un vector) como su significado (la posición que ocupan dentro del espacio vectorial). La propia representación del término como vector es el significado de este. Este es el hecho que utilizan este tipo de técnicas para obtener representaciones (y por lo tanto significados) de sentencias completas.

El principal exponente de este tipo de métodos son los *Recursive Autoencoders* o RAE (Pollack, 1990; Socher et al., 2010 y 2011). Estas aproximaciones obtienen la representación de una sentencia uniendo las representaciones de los términos de la misma en función de la jerarquía que establece su árbol de dependencias.

Cuando se unen dos representaciones (por ejemplo, la asociada a ”casa” y a ”verde”) se genera una nueva que incluye la información de ambas (se generaría un vector cuyo significado sería ”casa verde”).

Inicialmente, se unen los términos que tengan dependencia directa (las hojas del árbol). El resultado de esta unión será una representación parcial de la sentencia. A esta se unirá, en función de las dependencias existentes, otros términos hasta obtener la representación total de la sentencia.

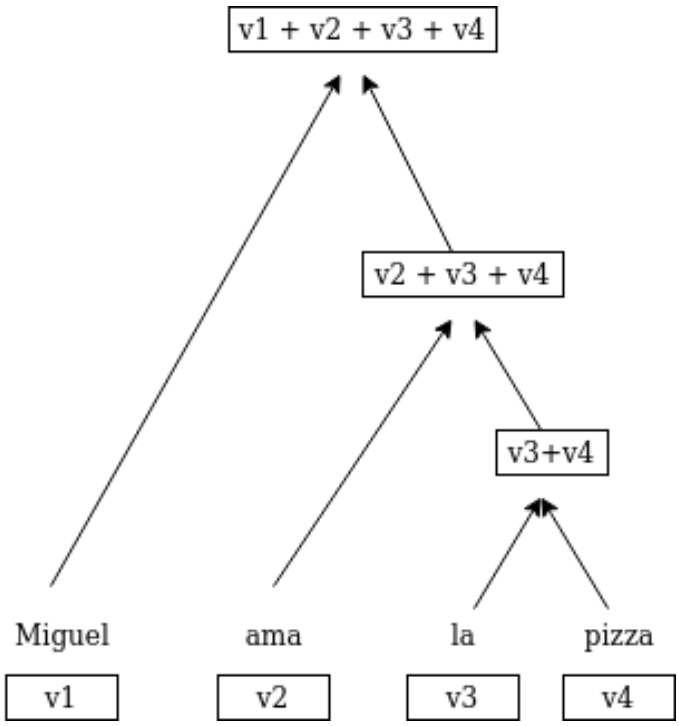

Figura 3.23: Representación de una sentencia a partir de los vectores de los términos que contiene.

Una vez realizadas todas las unificaciones, se obtiene una representación de la totalidad de la sentencia. Esta consiste en un vector cuya posición en su espacio vectorial representa el significado de la sentencia.

Entonces, para obtener la semejanza semántica entre dos representaciones de sentencias generadas a través de un RAE, como ambas sentencias consisten en vectores dentro de un espacio vectorial y su posición identifica su significado, tan solo es necesario calcular la distancia entre los dos vectores utilizando las técnicas descritas en la sección 3.4.5.

3.6. Extracción de información

En este apartado tan solo se analizan las técnicas que nos permiten extraer fragmentos de textos o información relevante del mismo. Por lo que implica la tarea propuesta, no se analizan otras técnicas relacionadas con, por ejemplo, la resolución de correferencia, el análisis exhaustivo de las relaciones existentes entre los términos del texto o la clasificación temática de estos.

3.6.1. Análisis de *collocations*

Las *collocations* consisten en grupos de dos o más palabras que tienden a coocurrir. Además, aunque esto no necesariamente tiene por qué suceder, estos grupos de términos, al coocurrir, pueden tomar un nuevo significado.

Por ejemplo, los términos que componen ”paso de cebra”, por separado, tienen su propio significado individual y, en el caso de aparezcan juntos, toman uno nuevo que no está relacionado con propios significados individuales (como podría suceder, por ejemplo, en ”casa de campo”).

La identificación de las *collocations* que más aparecen en un documento es una de las formas más simples pero efectivas de extraer información que, potencialmente, describe el contenido del mismo.

Para ello, la mayoría de las aproximaciones que realizan esta tarea: especifican el tamaño de las *collocations* a extraer, eliminan la información irrelevante (*stopwords*, determinantes, etc) y calculan el número de coocurrencias existentes entre los distintos términos del texto evaluado. Una vez se ha realizado este proceso, el resultado consistirá en las *collocations* que más coocurren.

El cálculo del número de coocurrencias dependerá del número de términos que contengan las *collocations* que queremos extraer.

Una aproximación común e independiente del tamaño de las *collocations* extraídas será dividir el texto en tuplas de n elementos (siendo n el tamaño deseado) y contar la frecuencia de aparición de cada tupla presente en el mismo.

3.6.2. Rapid Automatic Keyword Extraction

Rapid Automatic Keywords Extraction o RAKE (Rose et al., 2010) es uno de los algoritmos de extracción de información o de extracción de palabras clave más utilizados en la literatura actual.

Esto se debe a que, a pesar de su simpleza, es un algoritmo especialmente eficiente que además ofrece una precisión adecuada. También destaca por su capacidad de operar sobre distintos documentos por separado sin necesidad de utilizar un corpus o una colección de términos que los incluya.

Este algoritmo hace una distinción entre términos:

- **Relevantes**: son aquellos términos que pertenecen al texto y ofrecen información de interés. También reciben el nombre de *content words*.
- **Irrelevantes**: son aquellos que, aunque son necesarios para la construcciones de una sentencia o de un texto, no ofrecen ninguna información sobre el significado del mismo. Estos son, por ejemplo, las palabras vacías o *stopwords* asociadas a un idioma, los determinantes, artículos, etc.

Una vez se ha realizado esta distinción (que normalmente se hace de forma manual), el algoritmo empieza su labor bajo la siguiente premisa: la información relevante o las *keywords* a destacar están formadas por una o varias *content words*. Por lo tanto, para extraer términos que describen la información del texto correspondiente, es necesario analizarlas junto a las relaciones existentes entre ellas.

Dado un texto, este algoritmo:

1. Separa el documento en una lista o *array* de palabras. La separación se hace en base a distintos delimitadores como espacios en blanco, puntos, etc.
2. Agrupa las palabras de la lista en secuencias de términos contiguos, siendo el límite la separación el primer término irrelevante o *stopword* encontrado. Estas secuencias serán potenciales palabras clave o *candidate keywords*.
3. Calcula las puntuaciones asociadas a cada término de las *candidate keywords*.
4. Calcula las puntuaciones asociadas a las *cadidate keywords* utilizando las puntuaciones de los términos que las componen.
5. Extrae los resultados en función de las puntuaciones obtenidos.

La puntuación asociada a cada término se calcula mediante la siguiente fórmula:

$$punt(palabra) = \frac{grado(palabra)}{frecuencia(palabra)}$$

Figura 3.24: Fórmula que determina la puntuación asociada a cada término relevante.

La frecuencia de un término, en este algoritmo, consiste en el número de veces que dicha palabra aparece en la colección de candidatos. Por otro lado, el grado de un término analiza la frecuencia en la que el término coocurre con otros términos de la colección de candidatos.

Como se puede observar en la fórmula que determina la puntuación asociada a cada candidato, la frecuencia de un término es inversamente proporcional a la puntuación obtenida mientras que el grado es justo lo contrario.

Por lo tanto, este algoritmo ofrece mejores puntuaciones a aquellos términos que tienden a coocurrir con varios términos en palabras candidatos y penaliza aquellos que forman parte de candidatos de dos o menos términos.

De este modo, se mide tanto el número de apariciones que tiene dentro del texto como su relevancia dentro de él, analizando con cuánta información relevante se relaciona y de qué modo.

Por otro lado, la puntuación asociada a cada candidato o *candidate keyword* se calcula como la suma de las puntuaciones de los términos que lo componen.

Utilizando estos resultados, ya es posible extraer los términos o fragmentos del texto que mejor describen su contenido. Depende del problema a tratar, el resultado puede consistir en los candidatos que superen una cierta puntuación o los n candidatos con mejor puntuación.

3.6.3. Algoritmo TextRank

Consiste en un algoritmo creado por (Milhacea y Tarau, 2004) que comúnmente se utiliza para la extracción de términos clave y para la generación de resúmenes automáticos. Su funcionamiento se basa en el algoritmo *PageRank* (Lawrence, 1998).

Inicialmente, se separa el texto a evaluar en sentencias y se analizan los términos que componen cada una de estas. Una vez analizados, se eliminan los términos irrelevantes que contienen. Estos son aquellos que no ofrecen información sobre el contenido del propio texto como *stopwords*, determinantes, artículos, etc.

Una vez el algoritmo dispone de las sentencias que forman el texto y se ha eliminado la información irrelevante, se especifica un tamaño de ventana (*window*). Esta determinará las divisiones que se realizarán dentro de estas

sentencias y cómo los distintos términos se relacionan entre sí. Esta ventana se especifica de forma manual (por parte del usuario).

Cada sentencia se fragmenta en función del número de ventanas especificadas tal y como se muestra en la siguiente imagen:

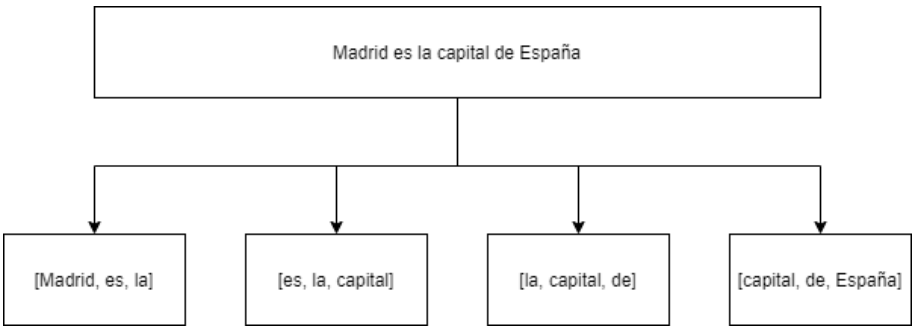

Figura 3.25: Fragmentación de una sentencia en función del tamaño de la ventana. En este ejemplo, la ventana tiene tamaño 3.

Utilizando esta información, el algoritmo tiene la capacidad de generar un grafo a partir del cual realizar un proceso similar al que sucede en *PageRank*.

Cada término relevante consistirá en un nodo del grafo. Cada vez que dos términos coocurren en uno de los fragmentos (establecidos por el tamaño de ventana), se genera una arista que los conecta.

Una vez se construye el grafo, se calcula la puntuación asociada a cada término del grafo. Para ello, se utiliza la siguiente fórmula:

$$punt(palabra_i) = (1-d) + d * \sum_{palabra_j \in Doc} \frac{num(palabra_j)}{C}$$

Figura 3.26: Fórmula básica para el cálculo de puntuaciones asociadas a cada término en el algoritmo *TextRank*.

Donde:

- Doc es el documento evaluado.
- d es el *damping factor*. Consiste en un valor entre 0 y 1. Representa, en *PageRank*, la probabilidad de que una persona que haya pulsado en un

link pulse en otro. En *TextRank* representa lo análogo. Es la probabilidad de que dos términos sean subsecuentes. Es un valor introducido manualmente por el usuario.

- num(palabra) consiste en el número de veces que un término aparece en el documento evaluado (Doc).
- C indica el número de términos que aparecen en el documento evaluado.

En algunas aproximaciones de este algoritmo se asignan distintos pesos a las relaciones existentes entre términos. Estas pueden depender, por ejemplo, de si dos términos comparten *PoS* (o estos suelen estar ligados), de si pertenecen a campos parecidos, etc.

En este caso *num* y C se calcularían en función de la relación existente entre los términos a comparar (los pesos asociados).

Utilizando las puntuaciones asociadas a cada término, podemos extraer los términos más relevantes del documento analizado. Aquellos términos cuya puntuación sea mayor, serán más descriptivos del mismo.

A diferencia de las dos técnicas descritas en los apartados anteriores, esta aproximación tan solo devuelve *keywords* compuestas por un único término. Por lo tanto, es de especial interés combinarla con otras (como las descritas anteriormente) para extraer información de forma más verosímil y completa.

Capítulo 4

Funcionamiento del sistema propuesto

Contenido: En esta sección se describe, tanto a alto nivel como de forma detallada, las distintas operaciones que dan lugar al sistema propuesto y sus principales usos.

4.1. Introducción

Tal y como se indica en los apartados previos, uno de los objetivos del trabajo es la implementación de un sistema que, aplicando distintas técnicas de procesamiento de lenguaje natural, permita analizar la calidad de un documento en base a su completitud.

Un documento completo es aquel que incluye la información o los conceptos necesarios para que pueda realizar correctamente su funcionalidad. Esta información se representa mediante criterios (y los subcriterios asociados).

Entonces, para determinar la calidad de un documento en base a su completitud, el sistema propuesto analizará si el documento evaluado incluye la información correspondiente. Luego, en función de la información presente en el documento, el sistema generará un veredicto sobre su calidad.

Además, el sistema permitirá extraer potenciales criterios (y subcriterios) para evaluar la calidad de un cierto tipo de documentos a partir de textos en los cuales se especifique la información que determina su completitud. Por ejemplo, a partir de un texto en el que se indica la respuesta esperada en un examen, el sistema tendrá la capacidad de extraer la información relevante (los criterios) de dicha respuesta para después poder evaluar las respuestas

proporcionadas por los alumnos.

De este modo el sistema implementado permitirá ahorrar enormes cantidades de recursos, tanto humanos como temporales, y será más escalable a distintos tipos de documentos.

En este apartado se describen las distintas funcionalidades que ofrece el sistema propuesto y cómo todas estas interactúan entre sí para dar lugar a sus principales usos.

4.2. Funcionamiento general del sistema

De entre todos los usos que permite el sistema propuesto, podemos destacar dos: la evaluación de un documento en base a su completitud y la autoconfiguración del sistema.

En este apartado vamos a ver cómo las distintas funcionalidades que ofrece el sistema interactúan entre sí para dar lugar a cada uno de estos usos.

Es importante matizar que el sistema propuesto está implementado de modo que la mayoría de las funcionalidades descritas pueden utilizarse de forma independiente. Por lo tanto, no es necesario utilizarlas en el orden descrito a continuación.

Aún así, este es el orden más lógico y que aprovecha las capacidades del sistema para, en la mayoría de los casos, ofrecer los mejores resultados posibles.

4.2.1. Evaluación de un documento en base a su completitud

Consiste en la búsqueda de la información almacenada en los criterios (y subcriterios) que determinan la completitud de un cierto tipo de documentos en el texto correspondiente con la finalidad de generar un veredicto.

En primer lugar, el texto se procesa con la finalidad de eliminar la información irrelevante que contiene y de obtener una representación interpretable por parte de los demás componentes del sistema. Este procesamiento se describe de forma detallada en la sección 4.3.2. Luego, una vez procesado el documento, el sistema analiza qué criterios se cumplen y cuáles no.

Para cada criterio, el sistema busca la información que almacena dicho

criterio en las distintas partes que componen el texto a evaluar (frases, títulos, etc). El proceso de búsqueda de dicha información se describe en detalle en la sección 4.3.3.

La información que almacena un criterio son los subcriterios asociados. Por lo tanto, el sistema busca si la información almacenada en el texto que representa cada subcriterio aparece en el documento correspondiente.

Una vez el sistema ha analizado qué subcriterios aparecen de cada criterio evaluado, genera un veredicto sobre la completitud del documento. Para cada criterio evaluado, indicará su cumplimiento diferenciando entre dos niveles: se cumple o no.

El sistema determinará el cumplimiento de un criterio en función de si el porcentaje de subcriterios encontrados es mayor que un cierto umbral. Dicho umbral:

- Es único para cada criterio y se obtiene aplicando las distintas técnicas de autoconfiguración del sistema que se describen en los apartados posteriores.
- Tomará un valor predeterminado en el caso de que el sistema no tenga ningún valor asociado al umbral del criterio analizado.

En la figura 4.1 aparece un diagrama que muestra la sucesión de operaciones que dan lugar a la evaluación de un documento en base a su completitud por parte del sistema propuesto.

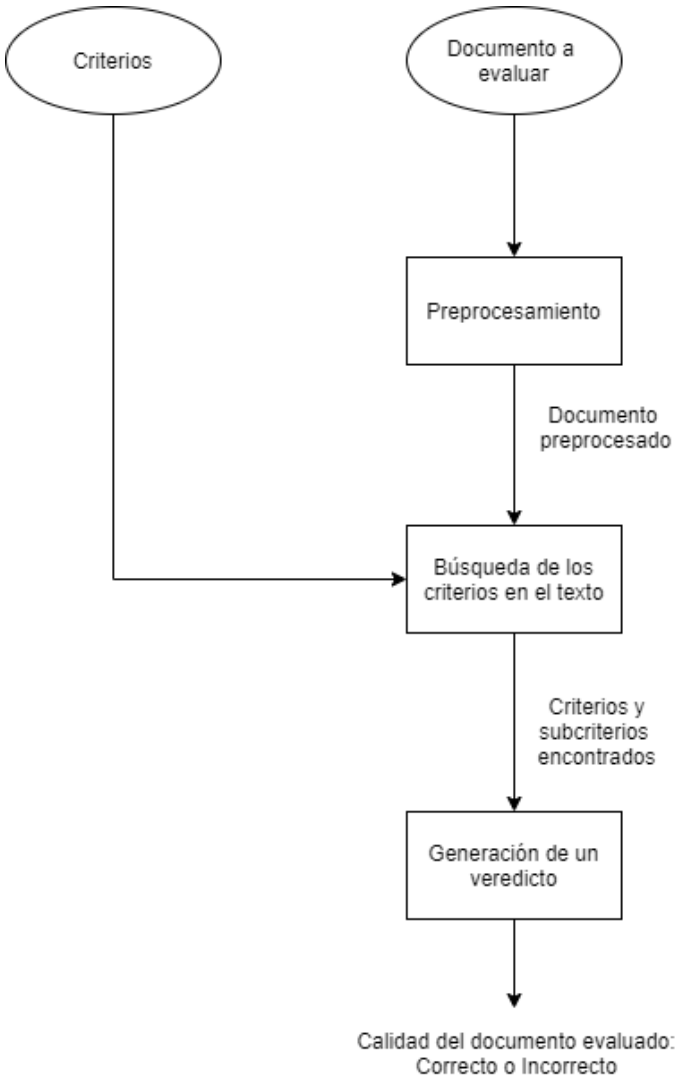

Figura 4.1: Diagrama que representa la evaluación de un documento por parte del sistema propuesto.

4.2.2. Autoconfiguración del sistema

Consiste en la modificación de la configuración del sistema a partir del análisis de los resultados obtenidos frente a una colección etiquetada de documentos.

A través de esta funcionalidad, el sistema tiene la capacidad de obtener conocimiento y de ajustar su comportamiento frente a un cierto tipo de documentos, analizando la información que determina su calidad y viendo cómo dicha información se refleja en distintos ejemplos del tipo analizado.

Para utilizar esta funcionalidad, el sistema requiere dos elementos:

- Una serie de criterios (y subcriterios) que, potencialmente, determinan la calidad de los documentos de la colección utilizada.
- Una colección de documentos previamente evaluados y etiquetados (indicando para cada uno qué criterios se cumplen y cuáles no).

Estos criterios pueden tanto especificarse de forma manual (introduciendo los criterios y subcriterios mediante sentencias en lenguaje natural) como extraerse de forma automática. El sistema propuesto tiene la capacidad generar la representación de un criterio a partir de un documento en el que se describe el mismo. Este proceso se describe detalladamente en la sección 4.3.1.

Una vez que el sistema dispone tanto de la colección etiquetada de documentos como de los criterios que determinan su calidad, el sistema empieza con la autoconfiguración.

El primer paso es la evaluación de los documentos de la colección utilizando los criterios introducidos. Esta evaluación consiste en el proceso descrito en la sección 4.2.1 pero con una pequeña modificación: en lugar de proporcionar veredictos para cada documento, la evaluación devuelve los resultados que se utilizan para generar los veredictos. Estos resultados consisten, entre otros aspectos, en los subcriterios que se han identificado en cada documento.

Utilizando esta información, el sistema analiza la presencia de los criterios y subcriterios utilizados en los documentos de la colección. Una vez analizados y atendiendo a las etiquetas asociadas a cada documento, el sistema filtra la información utilizada para evaluar cada criterio (sección 4.3.4), manteniendo solo la relevante o la que puede ser de utilidad para evaluar otros documentos del mismo tipo.

Para finalizar, el sistema evalúa los resultados obtenidos con los criterios ya filtrados y los compara con los resultados esperados (las etiquetas de los documentos de la colección). Atendiendo a las diferencias existentes, ajusta los umbrales asociados a cada criterio tal y como se explica en la sección 4.3.5.

El resultado del uso de esta funcionalidad, por lo tanto, consistirá en el ajuste de los umbrales asociados a cada criterio y en la obtención de una colección de criterios (y subcriterios) relevantes para la evaluación de documentos de un cierto tipo.

Todos los pasos que dan lugar a esta funcionalidad se representan en forma de diagrama en la siguiente figura:

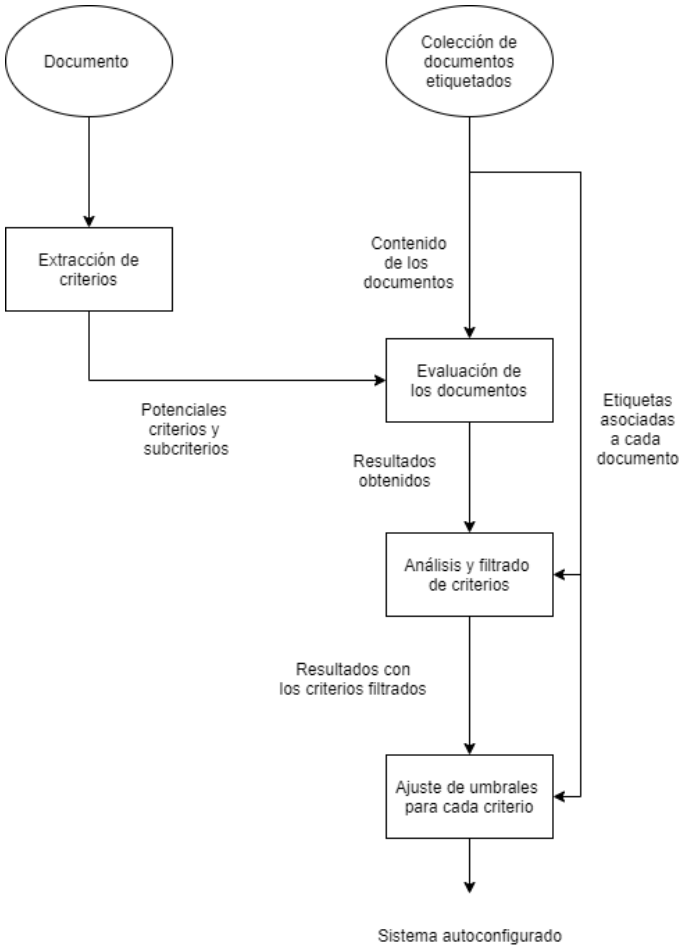

Figura 4.2: Diagrama que representa la autoconfiguración del sistema.

4.3. Descripción detallada

Tal y como se puede observar, en la sección 4.2, se proporciona una descripción de alto nivel de los principales usos del sistema propuesto. En esta descripción no se explica detalladamente el funcionamiento de la mayoría de los elementos intermedios que dan lugar a los usos del sistema para así evitar acumular información y para facilitar su entendimiento.

En este apartado, por lo contrario, se describe de forma detallada el funcionamiento de dichos elementos intermedios, ignorando los que ya se han

descrito en la sección anterior.

4.3.1. Extracción de criterios

El sistema tiene la capacidad de extraer la información asociada a un criterio a partir de un documento que contiene su descripción. Dicho de otra forma, el sistema tiene la capacidad de extraer potenciales subcriterios asociados a un cierto criterio a partir de un documento que lo describe.

Para obtener información relevante para la evaluación de un tipo de documentos, es importante que el texto utilizado para la extracción de información almacene los contenidos que determinan la completitud de dicho tipo de documentos (en base a un cierto criterio).

En el caso de que este documento almacene otra información irrelevante, el sistema puede inferir que dicha información también son subcriterios. Por lo tanto, es de especial interés evitar introducir información irrelevante en el documento utilizado, ya que esto puede generar que el sistema ofrezca peores resultados.

Por otro lado, el sistema también tiene la capacidad de analizar la información extraída y de filtrarla (sección 4.3.4), quedándose solo con la que es relevante para la evaluación de una cierta colección de documentos.

El primer paso para extraer un criterio a partir de un documento es el procesado del mismo. Este procesado consiste en las distintas técnicas descritas en la sección 4.3.2.

Cuando disponemos del texto procesado, el sistema utiliza distintas técnicas de extracción de información para obtener fragmentos del texto que potencialmente serán de utilidad para realizar evaluaciones en el futuro (subcriterios).

Concretamente, el sistema utiliza tres técnicas distintas de extracción de información. Todas estas, se centran en la extracción de fragmentos relevantes del texto en relación con los demás componentes del mismo (extracción de términos clave, de términos relacionados con gran parte de los elementos del texto, etc). Las técnicas que utiliza el sistema son las siguientes: *Rapid Automated Keyword Extraction* (RAKE), *TextRank* y extracción de *collocations*.

Una vez aplicadas, se unifica la información extraída con cada una de

ellas y se estructura de forma que el sistema pueda utilizar esta información para realizar evaluaciones.

4.3.2. Procesamiento del contenido textual

Para realizar las operaciones correspondientes sobre conocimiento textual, las distintas funcionalidades que ofrece el sistema transforman el texto entrante para así facilitar su tarea, obtener nueva información de utilidad y eliminar otra información irrelevante.

Esta transformación se aplica del mismo modo en las distintas funcionalidades que ofrece el sistema. En el caso de que una funcionalidad requiera transformaciones complementarias, estas se realizan después de aplicar las descritas a continuación.

Dado un documento o un fragmento de texto, el sistema realiza las siguientes transformaciones (en el orden especificado):

1. **Conversión a minúsculas**: Conversión de los distintos términos que componen el texto a su versión en minúsculas.

2. **Eliminación de los términos numéricos**: Supresión de los términos compuestos por caracteres numéricos. Esto incluye tanto a los términos que tan solo están compuestos por números como a los términos que alguno de sus caracteres es numérico.

3. **Eliminación de caracteres extraños**: Eliminación de los caracteres del texto que no son alfanuméricos.

4. **Lematización del contenido**: Obtención, para cada término del texto, de su versión lematizada. Esto nos permitirá utilizar una versión normalizada de cada término que no depende de su tiempo, número, etc.

5. **Eliminación de espacios en blanco redundantes**: Supresión de los espacios en blanco irrelevantes generados por las eliminaciones realizadas o que, directamente, aparecían en el texto original.

6. **Eliminación de las *stopwords***: Supresión de las palabras vacías presentes en el texto. Son aquellos términos que, aunque son indispensables en el lenguaje, no aportan información relevante a la hora de indexar o realizar operaciones sobre contenido textual.

7. **Eliminación de los términos compuestos por un único caracter**: supresión de este tipo de términos debido a la poca información

que ofrecen a la hora de realizar un análisis sobre un cierto contenido textual.

Una vez se han realizado todas estas transformaciones, el resultado consiste en un texto cuyo contenido serán los términos relevantes en su versión lematizada. Esta información será la que se utilizará en las distintas funcionalidades del sistema para llevar a cabo su labor.

4.3.3. Búsqueda de la información asociada a un criterio

Esta funcionalidad es la responsable de identificar la presencia de una cierta información dentro del contenido textual evaluado. Concretamente, es la responsable de identificar la información asociada a un criterio o subcriterio dentro de una sentencia o fragmento del texto.

De este modo, el sistema tendrá la capacidad de determinar qué información asociada a cada criterio está presente en el documento correspondiente para así poder generar un veredicto.

Tal y como se puede observar en la figura 4.1, la entrada consistirá en:

- El contenido del documento a evaluar procesado tal y como se indica en la sección 4.3.2.
- La colección de criterios que determinan la calidad del documento y su información asociada (subcriterios).

El objetivo de esta funcionalidad es buscar la información asociada a cada subcriterio del criterio evaluado dentro del contenido del documento. Una vez realiza esta función, devuelve qué subcriterios se han encontrado para que en otras operaciones del sistema se genere el veredicto correspondiente.

Es importante matizar que esta búsqueda se realiza en los distintos fragmentos que componen el texto a evaluar (títulos, sentencias, etc). Cuando se busca la información asociada a un subcriterio, no se busca en la totalidad del documento, si no que se lleva a cavo en los distintos fragmentos que componen el documento por separado.

Para realizar esta funcionalidad, el primer paso consiste en la transformación del contenido textual utilizado. Este consiste tanto en la sentencia a buscar como el texto en el que se busca. La transformación que se realiza sobre el contenido textual es la descrita en el apartado 4.3.3.1.

Una vez se ha realizado dicha transformación, cada sentencia consistirá en una lista de elementos. Estos elementos serán los que originalmente formaban la sentencia una vez se ha realizado la transformación correspondiente sobre ellos.

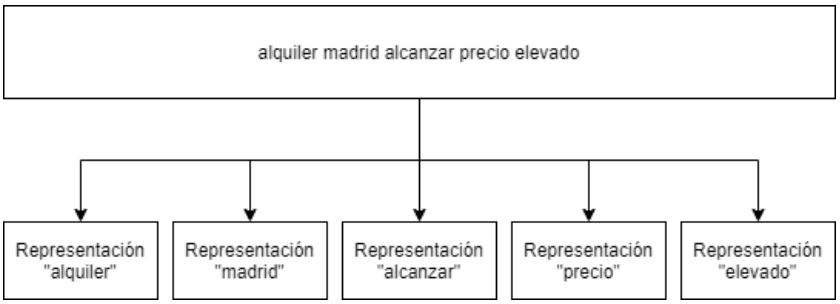

Figura 4.3: Diagrama que representa la transformación de la frase "El alquiler en Madrid ha alcanzado precios elevados" (una vez procesada).

El siguiente paso será el cálculo del índice de semejanza entre ambas sentencias. Dicho índice indica el grado en el que la información del concepto buscado (criterio o subcriterio) aparece en el texto en el que se busca. Consiste en un valor real entre 0 y 1 siendo 0 el mínimo de semejanza y 1 el máximo.

Para calcularlo, se utiliza una variante de la fórmula que aparece en la figura 3.21. Concretamente, en lugar de calcular la semejanza de forma bidireccional (ambas tienen que ser semejantes entre sí) se calcula en un único sentido (el contenido del concepto debe aparecer en el texto, independientemente de si el texto contiene más información que el concepto). Esta variante es la que aparece en la siguiente imagen:

$$sim(S_1, S_2) = \frac{1}{2}(\frac{\sum_{w_i \in S_1} \text{máx}_{w_j \in 2}\, sim(w_i, w_j)}{|S_1|})$$

Figura 4.4: Fórmula para calcular la semejanza existente entre la información buscada y el texto analizado.

Para realizar el cálculo de este modo, el sistema, en primer lugar, determina los pares de términos de cada sentencia que se asemejan más entre sí. Para cada término del concepto a buscar (criterio o subcriterio), determina el término del texto que presenta una mayor similitud con él.

Una vez se forman estos pares y se tiene el valor de semejanza asociado a cada par, el índice de semejanza entre ambas sentencias será la media aritmética de la semejanza de los términos que forman los pares.

Utilizando este valor, esta funcionalidad determinará si la información del concepto está presente en el texto. El sistema indicará que esto sucede en el caso de que el valor obtenido supere un cierto valor configurable.

La siguiente imagen representa el proceso descrito en este apartado:

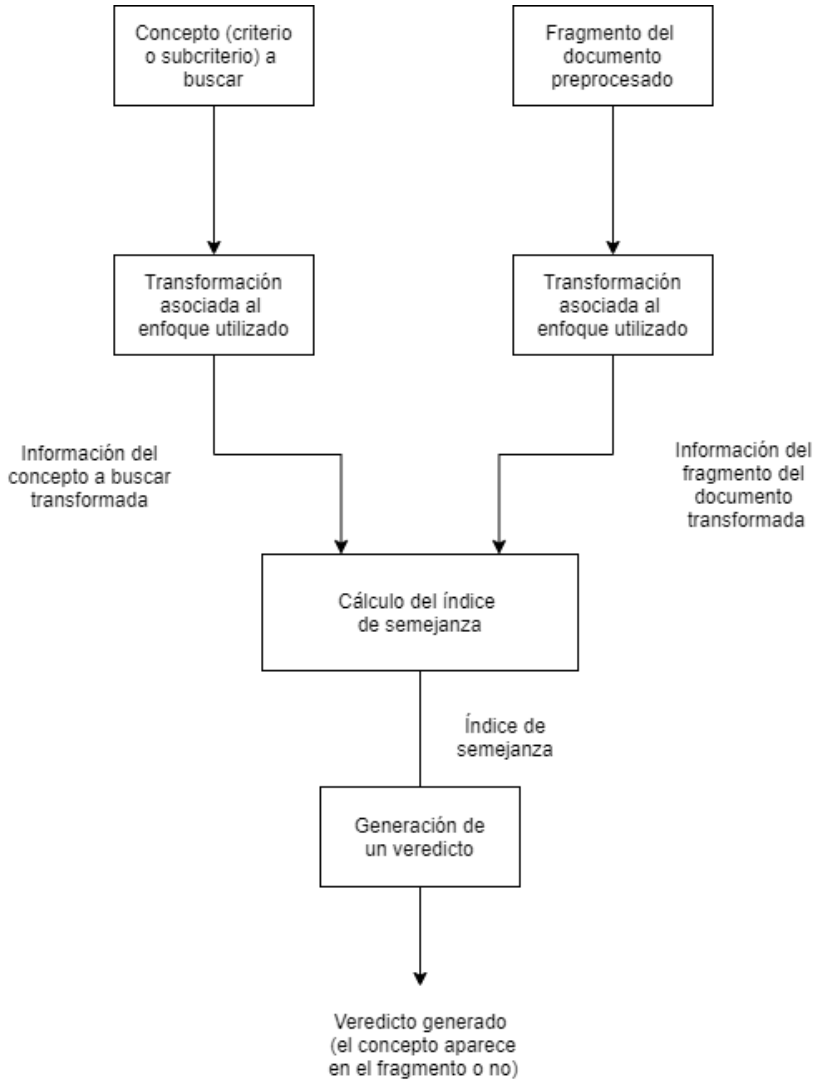

Figura 4.5: Diagrama que representa el proceso de búsqueda de información asociada a un criterio.

La cuestión en este punto es cómo calcular la semejanza existente entre dos términos. El sistema contempla el uso de *Word2Vec* para realizar esta tarea. Las transformaciones y los cálculos correspondientes se describen en la sección 4.3.3.1.

En las primeras versiones del sistema propuesto, este cálculo se realizaba utilizado la información almacenada en una base de conocimiento estructurado (concretamente, en *WordNet*). Esta opción se descartó, tal y como se verá en la sección Resultados obtenidos, debido al elevado coste computacional y a los tiempos de ejecución que implica su uso.

Aún así, tanto este enfoque como el que utiliza el sistema propuesto se describen en los siguientes apartados.

4.3.3.1. Enfoque basado en semántica distribucional

Consiste en la representación de los términos que componen ambas sentencias a través de vectores dentro de un espacio vectorial en el que las ubicaciones de los términos dependerán de su semejanza.

Dos términos que compartan información, que hagan referencia a realidades similares o que pertenezcan a campos parecidos ocuparán posiciones cercanas mientras que dos términos que no comparten ninguna información ocuparán posiciones distantes.

Word2vec nos permite utilizar un modelo pre-entrenado que tiene la capacidad de generar vectores verosímiles (siguiendo las reglas de proximidad) a partir de sentencias en lenguaje natural. Por lo tanto, para comparar dos términos representados mediante cadenas de caracteres, tan solo es necesario obtener su representación en forma de vector y comparar ambos.

Para transformar los términos en vectores, el sistema utiliza un modelo pre-entrenado con más de 15 millones de términos presentes en la lengua española. Este modelo es el resultado del trabajo de Cristian Cardellino, el cual utilizó un corpus de 2 billones de palabras ubicadas en más de 95 millones de sentencias para obtener dichos *embeddings*.

Una vez dispone de los vectores asociados a los términos, el sistema calcula su semejanza utilizando la similitud basada en el coseno (*cosine similarity*) existente entre ambos.

4.3.3.2. Enfoque basado en conocimiento estructurado

Tal y como se indica en la sección 3.3.3.2, *WordNet* es una base de conocimiento estructurado que almacena información léxica sobre gran parte de los términos contemplados en la lengua española generando un grafo que muestra las relaciones existentes entre ellos.

Utilizando este conocimiento, el sistema puede obtener los *synsets* asociados a los términos que desea comparar. También puede utilizar las distintas técnicas que ofrece *WordNet* para comparar su semejanza.

De entre todas las descritas en la sección 3.3.3.2, el sistema utilizaba *Path-Lenght Measure*. Esto se debe a que, de forma empírica, se estimó que era la que ofrecía mejores resultados frente al problema que se trata de resolver.

Por lo tanto, al utilizar esta métrica, el resultado consistía en un número real entre 0 y 1 que determinaría el grado de semejanza existente entre los dos términos a comparar, siendo 0 la mínima semejanza y 1 la máxima.

Es importante recordar que esta aproximación no forma parte del sistema final debido a su coste computacional y a los tiempos de ejecución que implica su uso.

4.3.3.3. Términos no contemplados por los enfoques anteriores

Es posible que algunos de los términos presentes en los documentos evaluados no se encuentren en el modelo pre-entrenado de *Word2vec* o en la base de conocimiento de *WordNet*. Este sería el caso, por ejemplo, de anglicismos, acrónimos, nombres propios, etc.

En este caso, ambos enfoques no tienen la capacidad de comparar la información de este tipo de términos pero, a su vez, no tiene sentido desechar la información que estos contienen.

Por esta razón, el sistema incluirá otra aproximación complementaria, la cual utilizará en el caso de que los dos enfoques descritos anteriormente no reconozcan el término analizado.

Debido a su naturaleza, la aproximación utilizada no puede necesitar información previa sobre el término utilizado. Tan solo puede utilizar la propia representación de los términos (las cadenas de caracteres de cada uno) para realizar dicha comparación.

Por esta razón, el sistema utiliza una de las técnicas descritas en la sección 3.2. Concretamente, utiliza *longest common substring*. Esto se debe a que, de forma empírica, se ha estimado que es la aproximación que ofrece mejores resultados frente al problema que se trata de resolver.

Esta métrica, al igual que los resultados de los demás enfoques, devolverá un valor entero entre 0 y 1.

4.3.4. Análisis y filtrado de criterios

Esta funcionalidad es la responsable de analizar y ajustar la información (subcriterios) asociada a cada criterio que el sistema va a utilizar para realizar evaluaciones en un futuro.

Esta funcionalidad forma parte de la autoconfiguración del sistema y se ejecuta después evaluar la colección correspondiente tal y como se indica en la sección 4.2.

Para llevar a cabo dicha funcionalidad, se utilizan:

- Los resultados de la evaluación de los documentos: entre otros aspectos, qué información o subcriterios ha identificado el sistema en cada documento para generar un veredicto.
- Los resultados esperados: las etiquetas asociadas a cada documento de la colección.

Utilizando esta información, el sistema analiza la presencia de los subcriterios en las instancias positivas y en las negativas. Para cada subcriterio, el sistema calcula el porcentaje de instancias positivas y negativas en las que aparece.

Las instancias positivas son aquellos documentos cuya etiqueta proporcionada indica que el documento cumple con el criterio correspondiente. Por lo contrario, las instancias negativas son aquellos documentos que su etiqueta indica que no lo cumple.

Una vez dispone del porcentaje de instancias positivas y negativas en las que aparece cada subcriterio, el sistema analiza los resultados obtenidos y determina qué subcriterios son relevantes para realizar evaluaciones en un futuro. Tal y como está implementado el sistema, los subcriterios que son relevantes son los que cumplen las siguientes reglas:

- **% Pos - % Neg >= 30**: Si no cumple esta regla, el subcriterio no es lo suficientemente discriminatorio. Por lo tanto, no es de utilidad para determinar si un documento es de calidad en base a un criterio.
- **% Neg <= 40**: Si no cumple con esta regla, el subcriterio aparece en numerosas ocasiones en instancias negativas. Por lo tanto, aunque en algunos casos pueda ser de utilidad para determinar la calidad de un documento, en la mayoría de los casos podrá inducir a error al sistema.
- **% Neg = 0 y % Pos >0**: Si se cumple esta regla, no es necesario que se cumplan las demás (y viceversa). Si se cumple, aunque no necesariamente ofrezca mucha información (% Pos puede tener un valor bajo), será relevante para la discriminación de instancias negativas y proporcionará información para la identificación de positivas.

siendo **% Pos** el porcentaje de instancias positivas en el que aparece un cierto subcriterio y **% Neg** el porcentaje de instancias negativas.

El resultado de esta operación es una colección de subcriterios útiles asociados a cada criterio. Habiendo eliminado la información irrelevante, el sistema ahora dispone de la información idónea para realizar evaluaciones más verosímiles.

Para finalizar, el sistema actualiza los resultados obtenidos en la evaluación realizada durante la autoconfiguración. Esta actualización consiste en la modificación de los resultados obtenidos inicialmente conservando tan solo la información (subcriterios) que se mantiene después del filtrado. Estos resultados son los que se utilizarán en el ajuste de los umbrales para cada criterio descrito en la sección 4.3.5.

4.3.5. Ajuste de umbrales para cada criterio

Esta operación es la encargada de analizar los resultados obtenidos y de obtener los umbrales asociados a cada criterio que ofrecen mejores resultados.

Tal y como se indicaba en la sección 4.2.1, estos umbrales determinan el porcentaje mínimo de subcriterios que el sistema debe identificar para determinar que un documento cumple un cierto criterio.

Cada criterio es independiente. Por lo tanto, a cada criterio se le asigna un umbral propio y dicho umbral no repercute en el de los demás criterios.

Para calcular el umbral idóneo para cada criterio, el sistema compara los resultados obtenidos sobre una colección de documentos con los resultados

esperados. Los resultados obtenidos consisten en los subcriterios encontrados en cada documento (con los cuales el sistema generaba un veredicto) mientras que los resultados esperados son las etiquetas que indican, para cada documento, qué criterios se cumplen y cuáles no.

Es importante matizar que esta operación, tal y como se puede ver en la figura 4.2, se realiza después de la operación de Análisis y filtrado de criterios (sección 4.3.4). Por lo tanto, los resultados obtenidos consistirán en los resultados actualizados generados por esta operación (aquellos en los que tan solo se contempla la información obtenida mediante la colección de criterios filtrada).

Dicho esto, para determinar los umbrales correspondientes, el sistema prueba distintos valores y analiza cuáles ofrecen mejores resultados. Para cada umbral, prueba todos los valores enteros contenidos entre 0 y 100.

Para cada valor, el sistema determina la tasa de acierto (*accuracy*) que obtendríamos en el criterio correspondiente utilizando dicho valor comparando los resultados obtenidos con los resultados esperados. Una vez calculadas las tasas de acierto asociadas a cada umbral (y para cada criterio), el sistema almacena el que ofrece un acierto mayor.

Por lo tanto, el resultado será, para cada criterio, el umbral que ofrece una mayor tasa de acierto. Una vez calculados dichos umbrales, el sistema modifica su configuración interna introduciendo dichos umbrales calculados. De este modo, el sistema tendrá la capacidad de realizar evaluaciones más verosímiles y de ofrecer mejores resultados frente a un cierto tipo de documentos.

Capítulo 5

Componentes del sistema

Contenido: Se describen los distintos componentes que interactúan entre sí para dar lugar a las funcionalidades del sistema propuesto, indicado sus principales funciones, cómo se relacionan entre sí y las tecnologías externas utilizadas para ello.

El sistema propuesto está implementado en *Python* 3.6. Por lo tanto, las tecnologías y recursos externos utilizados consisten en librerías y archivos accesibles desde *Python* 3.6.

En la siguiente imagen aparece una representación de los distintos componentes que forman el sistema y de cómo estos interactúan entre sí:

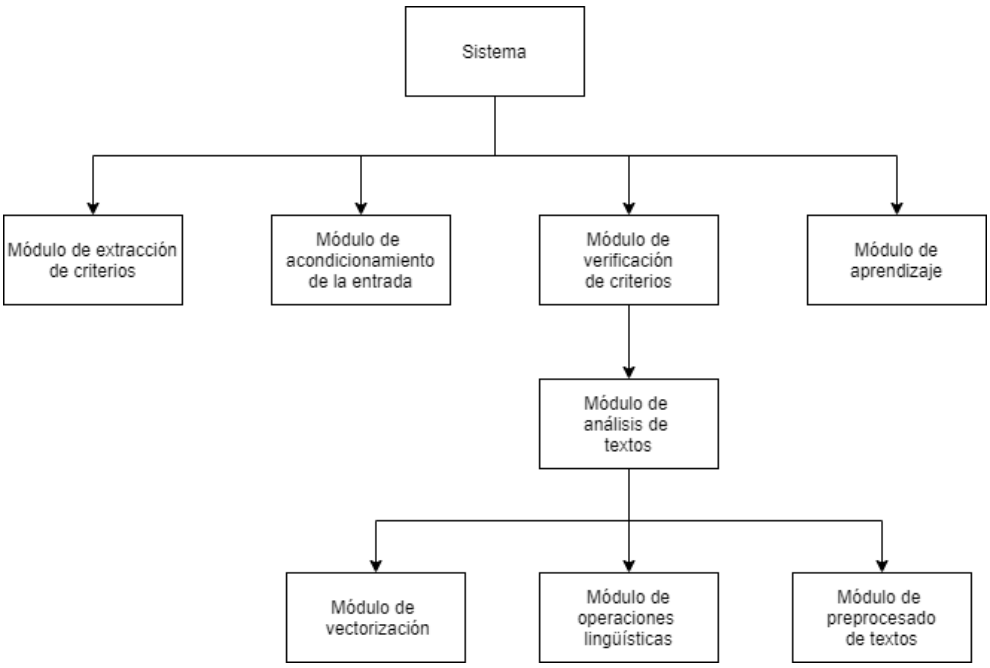

Figura 5.1: Diagrama que muestra los distintos componentes del sistema propuesto y cómo estos se relacionan entre sí.

El componente principal del sistema propuesto es el módulo que recibe el nombre de "Sistema". Este es el punto de acceso a las distintas funcionalidades que ofrece nuestra aproximación y es el encargado de interactuar con los demás componentes del mismo para utilizar sus capacidades y para, conjuntamente, realizar su labor como sistema: evaluar documentos en base a su completitud a partir de unos criterios definidos.

A través de este componente, cualquier usuario puede:

- Modificar la configuración del sistema.
- Extraer nueva información (criterios y subcriterios) de utilidad para realizar evaluaciones.
- Evaluar, en base a unos criterios, la completitud de un documento o de una colección de ellos.
- Autoconfigurar el sistema utilizando una colección de documentos previamente evaluada y etiquetada.

Para ello, este módulo interactúa directamente con los siguientes componentes del sistema:

- **Módulo de verificación de criterios**: para el análisis del cumplimiento de un cierto criterio en el documento correspondiente. Para

cumplir con su funcionalidad, este módulo interactúa con otros componentes como el módulo de análisis de textos.

- **Módulo de acondicionamiento de la entrada**: para ajustar y modificar el formato de la colección de documentos proporcionada a la hora de ejecutar funcionalidades como la autoconfiguración del sistema o la evaluación de múltiples documentos.
- **Módulo de extracción de criterios**: para la extracción de nueva información que potencialmente es de interés para realizar evaluaciones sobre un cierto tipo de documentos.
- **Módulo de aprendizaje**: para el aprendizaje y el ajuste del sistema. Para mejorar su capacidad a la hora de evaluar un cierto tipo de documentos.

5.1. Módulo de análisis de textos

Es el responsable de analizar si la información almacenada en una sentencia en lenguaje natural aparece, parcial o totalmente, en otra frase aplicando el proceso descrito en la sección 4.3.3.

Por lo tanto, este componente es el que se utiliza para determinar si la información asociada a un criterio o subcriterio está presente en una cierta sentencia de un texto.

Para lograr su objetivo, utiliza la logística almacenada en los módulos de preprocesado de textos, de operaciones lingüísticas y de vectorización de contenido textual.

5.1.1. Módulo de preprocesado de textos

Es el encargado de realizar todas las operaciones relacionadas con el procesamiento del contenido textual utilizado por parte del sistema. Este preprocesado consiste en el descrito en la sección 4.3.2.

Este procesamiento es fundamental para el correcto funcionamiento de muchas de las funcionalidades que ofrece el sistema como la extracción de criterios o la búsqueda de la información asociada a un criterio, por ejemplo.

Este módulo se almacena en el fichero *sentence_processing.py*. Para lograr su objetivo, utiliza las siguientes librerías:

- NLTK 3.4.4

- Spacy 2.1.4

5.1.2. Módulo de operaciones lingüísticas

Es el componente responsable de las operaciones de caracter lingüístico sobre contenido textual por parte del sistema. Estas operaciones consisten, por ejemplo, en el análisis sintáctico de sentencias, la lematización de sus componentes, la eliminación de términos irrelevantes (*stopwords*, determinantes, etc), la detección del lenguaje del contenido, etc.

Utilizando estas operaciones (junto a otras de otros módulos) el sistema tiene la capacidad de analizar el contenido de un cierto texto para identificar la aparición de la información correspondiente.

La lógica asociada a este módulo se almacena en el fichero *linguistic_ model.py* y utiliza los siguientes recursos:

- Spacy 2.1.4
- NLTK 3.4.4

5.1.3. Módulo de vectorización

Su principal tarea es la conversión de términos en lenguaje natural (cadenas de caracteres) en vectores dentro de un espacio vectorial creado mediante métodos estadísticos o basados en corpus (sección 3.4).

Por lo tanto, dicho espacio vectorial contendrá representaciones para distintos términos en lenguaje natural cuya ubicación en el mismo determinará su significado. Si dos términos tienen significados parecidos o suelen utilizarse en contextos similares, se ubicarán en posiciones próximas mientras que dos términos que comparten poca información en común se ubicarán en posiciones distantes entre sí.

Estas representaciones en forma de vectores se utilizan por parte de otros componentes del sistema (por ejemplo, el módulo de análisis de textos) para determinar la semejanza entre el contenido de dos sentencias comparando los distintos términos que contienen.

Para realizar esta vectorización, este módulo utiliza un modelo pre-entrenado vía *Word2Vec* que incluye representaciones para 15 millones términos incluidos en la lengua española. Este modelo se ha generado gracias al trabajo de Cristian Cardellino, el cual utilizó un corpus de 2 billones de palabras ubicadas en más de 95 millones de sentencias.

Utilizando este modelo, el módulo tiene la capacidad tanto de obtener el vector asociado a un término como de obtener la cadena de caracteres asociada a un cierto vector.

La implementación de este módulo se almacena en el fichero *words_model.py* y, para cumplir su objetivo, la lógica incluida en *Python* 3.6 y en la librería *Gensim 0.9.4*.

5.2. Módulo de verificación de criterios

Es el encargado de identificar qué información asociada a criterios y subcriterios aparece en un cierto documento. Utilizando esta información y atendiendo a la configuración del sistema, generará veredictos sobre la completitud de un cierto documento para cada criterio que se evalúe.

Por lo tanto, este componente es el que utiliza el sistema para realizar cualquier tarea relacionada con la evaluación de un documento. Puede incluir tanto la generación de un veredicto (en la funcionalidad de la evaluación de un documento) como la devolución de información relacionada con la evaluación (la información que se utiliza en tareas como la autoconfiguración del sistema).

Para ello, aplica las distintas operaciones almacenadas en el módulo de análisis de textos sobre las sentencias que componen el documento a evaluar. Una vez realizadas dichas operaciones, procesa los resultados obtenidos y genera una respuesta adecuada para otros componentes del sistema.

Se ubica en el fichero *criteria_checker.py* y está implementado en su mayoría mediante lógica en *Python* 3.6 nativo y llamadas a las funciones del módulo de análisis de textos.

5.3. Módulo de acondicionamiento de la entrada

Es el responsable de ajustar y adecuar las colecciones de documentos utilizadas por parte del sistema en funcionalidades como la autoconfiguración del sistema o la evaluación de colecciones con múltiples documentos.

Este ajuste no consiste en el preprocesamiento del contenido textual utilizado (la cual es responsabilidad del módulo de preprocesado de textos a través del módulo de análisis de textos). Dicho ajuste consiste, entre otros

aspectos, en la modificación del formato de las colecciones entrantes generando una estructura uniforme que sea comprensible para los demás componentes del sistema.

El acondicionamiento realizado por este componente consiste principalmente en:

- Eliminación de documentos no válidos (textos que no estén en español o documentos cuya longitud no sea mayor que un cierto número de términos).
- Extracción de forma uniforme de los contenidos asociados a cada documento de la colección a partir de su identificador (el nombre del documento).
- Extracción de las evaluaciones asociadas a cada criterio para cada documento de la colección a partir de su identificador.

5.4. Módulo de extracción de criterios

Es el encargado de extraer la información asociada a un criterio a partir de un documento en el que se describe el mismo, aplicando el proceso descrito en la sección 4.3.1.

Por lo tanto, este componente extrae y procesa nueva información mediante la cual el sistema tendrá la capacidad de realizar evaluaciones sobre un cierto tipo de documentos.

La lógica asociada a este módulo se ubica en el fichero *criteria_extractor_module.py* y utiliza las funcionalidades que ofrecen las siguientes librerías:

- NLTK 3.4.4
- Spacy 2.1.4
- Pandas 0.24.2
- Multi-rake 0.0.1
- Numpy 1.18.1

5.5. Módulo de aprendizaje del sistema

Engloba todas las funcionalidades del sistema relacionadas con la autoconfiguración del mismo descritas en la figura 4.2.2. Esto implica:

- Análisis y filtrado de la información asociada a los criterios utilizados (apartado 4.3.4).
- Obtención de los umbrales idóneos para cada criterio utilizado (4.3.5).

Cuando se utiliza esta funcionalidad, el sistema realiza las evaluaciones correspondientes utilizando el módulo de verificación de criterios. Utilizando los resultados obtenidos de esta operación, aplica las tareas que ofrece este componente. Una vez este módulo elimina la información irrelevante y calcula los umbrales idóneos para cada criterio, el sistema modifica su configuración para mejorar su capacidad de evaluación de documentos.

Este módulo está implementado en *Python* 3.6 nativo y no utiliza ninguna librería o recurso externo para realizar su tarea.

Capítulo 6

Resultados obtenidos

Contenido: En esta sección se indican y se analizan los resultados obtenidos al utilizar el sistema propuesto frente a dos colecciones de documentos diferentes. Además, se compara el rendimiento del sistema utilizando tanto *WordNet* como *Word2Vec* (en la comparación de términos) sobre una tercera colección.

Una vez implementado el sistema que se describe en las secciones anteriores, es el momento de evaluar su comportamiento frente a varias colecciones de documentos distintas.

De este modo, será posible tanto analizar la utilidad del sistema propuesto como determinar el impacto de las funcionalidades que ofrece.

Los documentos con los que se han realizado las pruebas que se describen en este apartado han sido proporcionados por profesionales especialistas en las materias tratadas.

A lo largo del desarrollo del proyecto, se han utilizado distintas colecciones de documentos con la finalidad de evaluar el funcionamiento de las versiones intermedias del sistema, de determinar qué técnicas de PLN eran más efectivas, de ajustar distintos parámetros a lo largo de la implementación, etc.

Para analizar los resultados obtenidos, tan solo haremos referencia a tres de ellas:

- Dos para evaluar el comportamiento del sistema propuesto.
- Una para comparar el funcionamiento del sistema utilizando tanto *WordNet* como *Word2Vec* a la hora de comparar los significados de los términos de las sentencias evaluadas.

Las tres engloban gran parte de los documentos utilizados en las colecciones empleadas a lo largo del proyecto. Reduciendo el número de colecciones utilizadas, será posible acotar el análisis del funcionamiento del sistema para así evaluar profundamente los resultados alcanzados.

Al comparar los resultados obtenidos vía *WordNet* y *Word2Vec*, podremos determinar qué enfoque ofrece mejores resultados. Además, como *WordNet* fue la aproximación utilizada al inicio del proyecto (hasta que se cambió por *Word2Vec*), también podremos analizar la evolución del sistema propuesto a lo largo del proyecto.

En último lugar, las tres colecciones utilizadas incluyen documentos con características distintas. Sobre cada una de ellas, el sistema se evalúa utilizando dos configuraciones diferentes: la predeterminada y la ajustada a cada colección.

6.1. Metodología de las pruebas

Para cada colección utilizada, se probará el sistema utilizando dos configuraciones distintas:

- La configuración predeterminada que este ofrece (sin aplicar autoconfiguración).
- La configuración resultante del ajuste del sistema a la colección correspondiente (mediante la funcionalidad de autoconfiguración).

En primer lugar, la configuración predeterminada es aquella que contiene el sistema en el momento de su inicialización. En esta, al no haberse utilizado la funcionalidad de autoconfiguración, en el sistema no se ha aplicado ningún tipo de ajuste que le permita realizar evaluaciones más verosímiles frente a una cierta colección de documentos.

Aún no tiene información sobre los umbrales idóneos asociados a un cierto tipo de documentos ni sobre qué información (criterios y subcriterios) es relevante para realizar evaluaciones.

En este caso, el sistema utiliza unos criterios (proporcionados por el usuario o extraídos automáticamente) y los umbrales asociados tomarán un valor fijo preestablecido para todos los criterios.

Por otro lado, las pruebas en las que se ajusta el comportamiento del sistema a una colección de documentos consistirán en:

1. El ajuste del sistema utilizando un fragmento de la colección de documentos correspondiente.
2. La evaluación de los documentos del fragmento restante (una vez el sistema está ajustado) y el análisis de los resultados obtenidos.

Para realizar este proceso, se utiliza *cross validation* (CV) estratificada con 4 iteraciones. En cada una de estas iteraciones, se fragmenta la colección correspondiente en dos grupos disjuntos de documentos:

- Uno de los grupos contendrá el 75 % de los documentos de la colección. Estos se utilizarán para ajustar el sistema (mediante la funcionalidad de autoconfiguración del mismo).
- El otro contendrá el 25 % de los documentos y se utilizarán para evaluar el comportamiento del sistema. El sistema los evaluará (una vez está ajustado) y se compararán los resultados esperados con los producidos por el sistema.

Gracias a la aplicación de CV estratificada, los resultados obtenidos serán más verosímiles. Esto se debe a que:

- El fragmento de la colección de documentos utilizado para la autoconfiguración del sistema será distinto al utilizado para el análisis de los resultados obtenidos.
- Los resultados producidos por el sistema no serán fruto de la autoconfiguración utilizando un fragmento concreto de la colección. Estos se obtienen utilizando distintas distribuciones de la colección (una en cada iteración) y uniendo los resultados obtenidos.
- En cada iteración del proceso se mantendrá la proporción de instancias de cada tipo (que cumplen un criterio concreto y que no lo cumplen) ya que la CV aplicada es estratificada.

6.2. Colecciones utilizadas

Tal y como se indica en el apartado anterior, se utilizan tres colecciones de documentos distintas:

- Dos colecciones para evaluar el comportamiento del sistema definitivo.
- Una para comparar el funcionamiento del sistema utilizando *WordNet* y *Word2Vec* a la hora de comparar los términos de las sentencias evaluadas.

Cada una de ellas incluye documentos que responden a una necesidad y utilidad específica. Todos los documentos de una misma colección se evalúan en base a los mismos criterios.

Para realizar estas pruebas, se han especificado los criterios que determinan la calidad de los documentos de cada colección. Estos se han extraído a partir de fuentes proporcionadas por profesionales especialistas en las distintas materias tratadas en los documentos.

Por otro lado, para evaluar la calidad de los veredictos generados por el sistema, todos los documentos de cada colección se han etiquetado indicando si cumplen con los criterios especificados. Este etiquetado también se ha realizado a través de profesionales especialistas en la materias tratadas. De este modo, será posible comparar los resultados esperables con los resultados generados por el sistema.

A continuación se describen las tres colecciones de documentos utilizadas para evaluar el sistema propuesto.

6.2.1. Colección I

Los documentos de esta colección presentan las siguientes características:

- Contienen información sobre la aplicación de políticas en múltiples situaciones. Consisten en el ajuste de estas políticas en base a distintos contextos y razones particulares.
- Su contenido se adecúa a la información presente en ciertas publicaciones oficiales. La presencia del contenido de estas publicaciones es lo que determina que realicen su función correctamente.
- Por lo tanto, para que la calidad de estos documentos sea la adecuada (en base a su completitud), deben incluir la información presente en las publicaciones asociadas.
- Son documentos, en su mayoría, extensos. Tienden a contener entre 5000 y 7000 términos cada uno de ellos. Por lo tanto, el volumen de información que almacenan es elevado.

Para los documentos de esta colección se han especificado (por parte de especialistas) tres criterios de calidad distintos: dos extraídos a partir de publicaciones oficiales (mediante la funcionalidad de extracción de criterios) y uno introducido de forma manual.

Esta colección contiene una totalidad de 93 documentos en los que se evalúan tres criterios. Por lo tanto, se evalúan una totalidad de 279 criterios (93 veces cada uno de los tres analizados).

Viendo las etiquetas asociadas a cada criterio de cada documento, la distribución presente en la colección es la siguiente: el 62.72 % de los criterios evaluados se cumplen (el documento correspondiente cumple con él) mientras que el 37,28 % no.

6.2.2. Colección II

Los documentos que componen la colección descrita en este apartado:

- Consisten en la aplicación de las prácticas y conceptos presentes en distintas políticas. No consisten en el ajuste de una cierta política al problema correspondiente sino que se caracterizan por ser la aplicación de la misma a problemas concretos.
- Pueden incluir información, por ejemplo, para informar a los participantes de un proceso sobre las responsabilidades que toman en él, sobre el uso de los datos que intervienen, etc.
- Su calidad no la determina la presencia de la información almacenada en distintas publicaciones oficiales. Esto se debe a que, debido su funcionalidad, no incluyen esta información de forma implícita o explícita.
- La calidad de estos documentos la determina la información presente en distintas indicaciones o manuales especificados por profesionales especialistas en la materia.
- Su longitud, en comparación a los documentos de la primera colección, es breve. La mayoría de los documentos de esta colección contienen entre 500 y 1000 términos, de modo que el volumen de información que contienen es menor.

En este caso, la calidad de los documentos de esta colección la determina un único criterio. Este se ha extraído (de forma automática) a partir de un documento con distintas indicaciones en lenguaje natural que describen los aspectos que determinan la calidad de este tipo de documentos. Dicho manual ha sido especificado por parte de profesionales especialistas en la materia tratada.

Esta colección incluye una totalidad de 58 documentos en los cuales se evalúa un único criterio. Por lo tanto, el cumplimiento de un cierto criterio

se determina 58 veces.

Viendo las etiquetas asociadas a cada criterio de cada documento, la distribución de esta colección es la siguiente: el 51.72 % de los criterios se cumplen mientras que el 48,27 % no.

6.2.3. Colección III

Tal y como se indica anteriormente, esta colección se utiliza para comparar los resultados obtenidos utilizando *Word2Vec* y *WordNet* para realizar el cálculo de la similitud existente entre los términos de distintas sentencias.

Esta colección consiste en una versión reducida de la colección I. Esto se debe a que, cuando se realizaron las pruebas descritas, tan solo se disponía de este fragmento de la colección total.

Por lo tanto, contiene documentos con las características descritas en dicha colección: consisten en el ajuste de estas políticas en base a distintos contextos y razones particulares y su calidad la determina el contenido de varias publicaciones oficiales.

Concretamente, la calidad de los documentos la determinan los mismos tres criterios que aparecían en la primera colección: uno especificado de forma manual y otros dos extraídos de forma automática a partir de dos publicaciones distintas.

Es importante matizar que la información extraída de las dos publicaciones utilizadas, en estas pruebas, es menor. Concretamente, es la que se obtiene utilizando tan solo la extracción de *collocations* (sección 4.3.1).

Esto se debe a dos razones:

- En el momento en el que se realizaron las pruebas descritas, la extracción de información por parte del sistema tan solo contemplaba esta aproximación.
- Debido al tiempo de ejecución que implica el uso de *WordNet* (tal y como se verá en los apartados posteriores), realizar las pruebas utilizando el número de subcriterios que tiene la capacidad de extraer el sistema definitivo no es viable con los recursos de los que disponemos.

La colección descrita está compuesta por un total de 38 documentos proporcionados por expertos en la materia tratada. Para cada documento, tal y como se indica anteriormente, se evalúan tres criterios distintos. Por lo tanto,

se evalúan un total de 114 criterios diferentes.

La distribución respecto a las etiquetas asociadas a cada criterio de cada documento es la siguiente: 67.55 % de los criterios se cumplen mientras que el 32.45 % no.

6.3. Resultados

En este apartado se indican los resultados obtenidos frente a las tres colecciones definidas en el apartado anterior. Tal y como se indica previamente, se evaluará el comportamiento del sistema utilizando tanto la configuración predeterminada como la configuración resultante del ajuste del mismo con cada una de las colecciones.

Para realizar este análisis, el sistema produce las evaluaciones correspondientes de cada documento y se comparan con los resultados esperados (las etiquetas asociadas a cada documento por parte de los especialistas). Utilizando el resultado de esta comparación, se evaluarán las siguientes métricas:

- **Tasa de acierto (*accuracy*)**: para determinar el porcentaje de instancias en las que el sistema tiene el comportamiento deseado.
- ***Precision***: para determinar la certeza con la que el sistema identifica documentos que cumplen los criterios de calidad.
- ***Recall***: para identificar el porcentaje de instancias de calidad que el sistema identifica.
- ***Specificity***: para identificar el porcentaje de instancias que no cumplen los criterios de calidad que el sistema tiene la capacidad de identificar.
- ***F1 score* de las instancias positivas**: para medir el comportamiento general del sistema a la hora de reconocer las instancias de calidad.
- ***F1 score* de las instancias negativas**: para medir el comportamiento general del sistema a la hora de reconocer las instancias que no cumplen los criterios de calidad.

Además, a la hora de comparar las aproximaciones de *WordNet* y *Word2Vec* (los resultados de la colección III), también se evaluará una métrica más: el tiempo de ejecución. De este modo, será posible comparar el coste computacional que implican ambos enfoques.

6.3.1. Resultados con la colección I

Tal y como se indicaba en los apartados anteriores, se han realizado pruebas sobre una colección de 93 documentos en los que se evalúan 3 criterios (en cada uno). Por lo tanto, el sistema evalúa un total de 279 criterios. Los resultados obtenidos utilizando el sistema tanto con la configuración predeterminada como con el sistema ajustado a la colección son los siguientes:

Métrica	**Config. predeterminada**	**Config. ajustada**
Accuracy	83.33 %	96.66 %
Precision	80.95 %	100.00 %
Recall	97.14 %	94.85 %
Specificity	57.89 %	100.00 %
F1 score pos.	88.31 %	97.36 %
F1 score neg.	70.96 %	95.47 %

Los resultados obtenidos, sobre todo con la configuración ajustada a la colección, demuestran que el sistema tiene la capacidad de calificar correctamente documentos de este tipo.

Respecto a los resultados con la configuración predeterminada, aunque son correctos, no muestran las suficientes garantías para evaluar con seguridad documentos de esta índole. Aunque la tasa de acierto sea elevada (83.33 %), analizando la especificidad obtenida (57.89 %) podemos observar que no identifica las instancias que no cumplen los criterios tal y como cabría esperar.

Tan solo identifica 6 de cada 10 instancias negativas y, la detección de estas, debido a la naturaleza del problema tratado, es especialmente relevante. Si el sistema determina que un documento cumple un criterio y este no lo cumple, las consecuencias de este hecho pueden ser altamente perjudiciales.

Además, los buenos resultados obtenidos respecto a la detección de instancias positivas pueden deberse a la falta de capacidad del sistema a la hora de identificar las negativas (al determinar que la mayoría de los documentos cumplen los criterios, el acierto en los positivos aumenta mientras que en los negativos disminuye). Esto se puede observar analizando la *precision* (80.95 %) y la *specificity* (57.89 %) obtenida.

Este desajuste se resuelve al aplicar la autoconfiguración del sistema. Una vez el sistema ha filtrado la información irrelevante para realizar las evaluaciones y ha obtenido los umbrales idóneos, este adquiere la capacidad de identificar correctamente ambos tipos de instancias.

Por lo tanto, una vez el sistema se ha autoconfigurado, este se puede utilizar para evaluar con ciertas garantías este tipo de documentos.

Las principales razones de los resultados obtenidos son las siguientes:

- El umbral predeterminado que utiliza el sistema no se parece al adecuado para evaluar este tipo de documentos. Una vez se aplica la autoconfiguración y se obtienen los umbrales idóneos, este factor desaparece.
- La información extraída de las publicaciones incluye tanto información de utilidad como información irrelevante. Cuando se utiliza la configuración predeterminada, también se utiliza la información irrelevante (empeorando los resultados obtenidos) mientras que cuando se realiza la autoconfiguración y se elimina dicha información irrelevante, los resultados mejoran considerablemente.
- La información utilizada aparece de forma implícita o explícita en los documentos y, una vez filtrada mediante la autoconfiguración del sistema, es discriminante (lo que nos permite alcanzar una tasa de acierto del 96.66 % una vez se ha autoconfigurado).

Es importante matizar que el tamaño de la colección utilizada no es el idóneo para realizar este tipo de pruebas. Debido a la dificultad de obtener documentos de esta índole y de etiquetarlos en base al cumplimiento de cada uno de los tres criterios evaluados, no ha sido posible obtener más.

Aún así, durante el desarrollo del sistema, se han tomado todas las medidas posibles para tratar que los resultados obtenidos sean similares a los que obtendríamos frente a colecciones con más documentos. Por ejemplo, en cada colección se han introducido documentos de distinta índole, se ha analizado el comportamiento del sistema para evitar que se sobreajuste a las distintas colecciones, etc.

6.3.2. Resultados con la colección II

Esta colección está compuesta por 58 documentos en los cuales se evalúa un único criterio. Los resultados obtenidos utilizando el sistema con las configuraciones descritas anteriormente son los siguientes:

Métrica	**Config. predeterminada**	**Config. ajustada**
Accuracy	89.47 %	93.10 %
Precision	87.50 %	93.33 %
Recall	93.33 %	93.33 %
Specificity	85.71 %	92.85 %
F1 score pos.	90.32 %	93.33 %
F1 score neg.	88.88 %	92.85 %

Tal y como se puede observar, el comportamiento del sistema utilizando ambas configuraciones, en este caso, es adecuado para evaluar documentos de esta índole. La tasa de aciertos, en ambos casos, es cercana al 90 % y, viendo las *f1 scores* alcanzadas, el sistema tiene la capacidad tanto de identificar instancias positivas (que cumplen los criterios correspondientes) como negativas.

La mayoría de las ejecuciones realizadas con la configuración predeterminada que ofrece el sistema (las que se han realizado a lo largo del desarrollo del sistema) no ofrecían estos resultados. Los resultados obtenidos dependen directamente del umbral asociado a cada criterio que se evalúa y en el caso de que este no fuese el adecuado (o similar al idóneo), los resultados obtenidos empeorarían considerablemente.

En este caso concreto, como el umbral predeterminado que se utiliza en el sistema se ha establecido observando la autoconfiguración alcanzada con colecciones como esta, el valor que toma es parecido al idóneo. Por lo tanto, aunque esta configuración (la predeterminada) no es la que ofrece los mejores resultados posibles, sigue dando lugar a un comportamiento adecuado para realizar evaluaciones.

En la configuración predeterminada, los resultados obtenidos son buenos tanto para identificar instancias positivas como negativas. Aún así, se puede observar que identifica mejor las instancias positivas que las negativas (la *f1 score* de la primera es aproximadamente dos puntos mayor que la de la segunda).

Este desajuste se resuelve al ajustar el comportamiento del sistema a la colección. Las métricas alcanzadas mejoran (alrededor de cuatro puntos cada una) y el sistema responde de forma similar frente a instancias positivas y negativas.

Las principales razones de los resultados obtenidos son las siguientes:

- La información almacenada en el documento con las indicaciones que determinan la calidad de este tipo de documentos aparece de forma

implícita o explícita en los documentos evaluados.

- Además, esta información es discriminante. En la mayoría de los casos, dicha información aparece en los documentos que, según la etiqueta asignada por los especialistas, cumplen con los criterios de calidad mientras que no aparece en los que no los cumplen.

Al utilizar esta misma colección pero con otra fuente para extraer los criterios, los resultados no alcanzaban valores tan elevados. A continuación se muestran los resultados obtenidos al utilizar como criterios distintos artículos legislativos relacionados con la materia tratada:

Métrica	**Config. predeterminada**	**Config. ajustada**
Accuracy	53.44 %	79.31 %
Precision	57.89 %	95.00 %
Recall	36.66 %	63.33 %
Specificity	71.42 %	96.42 %
F1 score pos.	44.89 %	76.00 %
F1 score neg.	59.70 %	81.81 %

Por lo tanto, como se puede observar, los resultados alcanzados dependen enormemente de que los criterios utilizados (y su fuente) sean los adecuados.

Tal y como sucedía en la colección I, es importante matizar que el número de documentos que componen esta colección no es el idóneo. La razón de su tamaño es la dificultad de su obtención y su etiquetado por parte de especialistas.

Aún así, se han tomado las precauciones adecuadas para intentar que los resultados obtenidos sean lo más parecidos posible a los que obtendríamos utilizando una colección mayor (y con las mismas características).

6.3.3. Resultados con la colección III

En este apartado se enfrentan los resultados obtenidos utilizando tanto *Word2Vec* como *WordNet* para comparar distintos términos que componen las sentencias evaluadas.

En ambas aproximaciones enfrentadas, el único aspecto que difiere es cómo se realiza esta comparación. Las demás funcionalidades (extracción de información, autoconfiguración, etc) se realizan exactamente del mismo modo en ambas versiones.

Aún así, es importante matizar que las pruebas descritas se realizaron en una fase intermedia del proyecto. Esto implica que:

- Tanto la versión de *WordNet* como la de *Word2Vec* utilizada para realizar las pruebas eran versiones intermedias. Por lo tanto, no incluían gran parte de las mejoras presentes en el sistema definitivo.
- La colección que se utiliza para realizar estas pruebas es la que se disponía en ese momento (la colección III).

En el momento en el que se observaron los resultados obtenidos con ambas aproximaciones se descartó el enfoque basado en conocimiento estructurado y se empezaron a implementar las mejoras correspondientes sobre la versión de *Word2Vec* (hasta dar lugar al sistema propuesto).

Como estas modificaciones no se realizaron sobre la aproximación de *WordNet*, no es posible comparar el rendimiento de este enfoque frente al sistema definitivo propuesto.

Dicho esto, veamos los resultados obtenidos utilizando la configuración predeterminada en ambas aproximaciones:

Métrica	**Aprox. de *WordNet***	**Aprox. de *Word2Vec***
Tiempo de ejecución	1313.39 secs	64.17 secs
Accuracy	72.22 %	75.92 %
Precision	80.89 %	86.41 %
Recall	84.70 %	82.35 %
Specificity	26.08 %	52.17 %
F1 score pos.	82.75 %	84.33 %
F1 score neg.	28.28 %	48.14 %

El principal factor diferencial entre ambas aproximaciones es el tiempo de ejecución. Tal y como se puede observar, el tiempo de ejecución de la versión de *Word2Vec* es aproximadamente 20 veces menor que el de *WordNet*.

Esta diferencia se debe, principalmente, al coste computacional que implica el acceso a la información de *WordNet* (a los *systets* asociados a los términos) y la comparación de estos (a través de operaciones como el cálculo de la *path-length measure*).

Tan solo analizando este factor, podemos afirmar que el uso de *WordNet* para resolver esta tarea, al menos de forma exclusiva, no es viable. Su tiempo de ejecución es demasiado elevado (unos 11.5 segundos por criterio evaluado) y los resultados obtenidos tampoco son los adecuados.

Además, los resultados obtenidos utilizando *WordNet* son peores que los que ofrece la versión de *Word2Vec*, de modo que no hay ningún aspecto que incline la balanza a su favor.

Respecto a los propios resultados obtenidos, el comportamiento en ambas aproximaciones no es el adecuado. Ninguna de las dos identifica correctamente las instancias positivas y las negativas.

Por otro lado, al utilizar la configuración ajustada a la colección se obtuvieron los siguientes resultados:

Métrica	**Aprox. de *WordNet***	**Aprox. de *Word2Vec***
Tiempo de ejecución	942.35 secs	62.05 secs
Accuracy	82.40 %	83.33 %
Precision	81.73 %	87.64 %
Recall	100.00 %	91.76 %
Specificity	17.39 %	52.17 %
F1 score pos.	89.94 %	89.65 %
F1 score neg.	29.05 %	63.15 %

Tal y como se puede observar, la diferencia existente entre los tiempos de ejecución, aunque se ha reducido, sigue siendo notable. Su reducción se debe principalmente a que, a través de la autoconfiguración del sistema, se reduce la información que se busca.

Al ajustar el comportamiento del sistema, se filtra la información irrelevante. Entonces, el sistema busca menos información y, por lo tanto, el tiempo de ejecución requerido disminuye.

Además, al obtener los umbrales "idóneos" para cada criterio, el sistema se limita a buscar los subcriterios suficientes para alcanzar estos umbrales. Por lo tanto, la cantidad de información evaluada, en este sentido, también disminuye.

Respecto a los resultados obtenidos, podemos destacar que la aproximación de *Word2Vec* ofrece mejores resultados que la de *WordNet*.

Su principal diferencia radica en la capacidad de instancias negativas. Mientras que la aproximación de *WordNet* identifica correctamente el 17.39 % de las instancias negativas que evalúa, la versión de *Word2Vec* identifica el 52.17 %.

Por otro lado, a priori, ambas aproximaciones identifican de forma correcta gran parte de las instancias positivas (*f1-scores* alrededor del 90 %). Aún así, su funcionamiento frente a las negativas deja mucho que desear (*f1-score* de 29.05 % y 63.15 %) y es probable que los buenos resultados obtenidos frente a las instancias positivas se deban a la falta de capacidad de identificar las negativas.

Por lo tanto, ninguna de las dos aproximaciones sería de utilidad ni proporcionaría la fiabilidad requerida a la hora de realizar la tarea propuesta en este proyecto.

Aún así, todas estas carencias se resolvieron introduciendo distintas mejoras sobre la versión de *Word2Vec*, dando lugar al sistema propuesto y obteniendo los resultados descritos sobre las colecciones I y II.

Capítulo 7

Conclusiones y trabajo futuro

Contenido: Esta sección incluye distintas conclusiones que se pueden extraer sobre el trabajo realizado una vez se ha implementado el sistema propuesto y se han analizado los resultados obtenidos. También se indican distintas modificaciones sobre el sistema propuesto que mejorarían su comportamiento.

7.1. Conclusiones

Una vez ha finalizado la especificación y la implementación del sistema propuesto, evaluando los resultados alcanzados, podemos afirmar que el sistema resultante cumple con los objetivos marcados al inicio del proyecto.

En primer lugar, el sistema tiene la capacidad de evaluar distintos documentos de forma verosímil y precisa. Esto se debe, principalmente, a la funcionalidad de autoconfiguración que ofrece.

Gracias a ella, el sistema obtiene la capacidad de analizar una colección de documentos previamente etiquetada y modificar su configuración para así calificar documentos correctamente.

Además, el sistema propuesto es escalable y se puede utilizar en distintos dominios. En función de los criterios introducidos (ya sea manualmente o de forma automática) y de la colección a través de la cual se realice la autoconfiguración, el sistema podrá evaluar distintos tipos de documentos.

Su utilidad no se centra en un dominio o tipo de documentos en concreto. Se puede utilizar sobre todo tipo de documentos cuya completitud se puede determinar evaluando la presencia de forma implícita o explícita de una serie

de criterios.

Los resultados alcanzados sobre un cierto tipo de documentos dependerán de la calidad de los criterios utilizados (o de las fuentes de las que se extraen) y de la colección utilizada para realizar la autoconfiguración. Si los criterios utilizados no se adecúan a la colección evaluada, el sistema no será de utilidad.

Además, el sistema ofrece la funcionalidad de extraer automáticamente representaciones de criterios a partir de documentos en los que se describe el mismo. En el caso de que extraiga información irrelevante de ellos, esta información se filtra durante el proceso de autoconfiguración.

De este modo, el sistema propuesto libera a los usuarios de la labor de especificar manualmente los criterios que determinan la calidad de un cierto tipo de documentos, ahorrando esfuerzo temporal y humano.

Por otro lado, a lo largo del desarrollo de este proyecto, se ha probado la utilidad de los enfoques basados en conocimiento estructurado (concretamente, en *WordNet*). Debido al tiempo de ejecución que implica su uso, no parece una opción viable para implementar sistemas como el propuesto.

Este factor puede desaparecer utilizando una fuente de conocimiento estructurado que requiera menos consultas o, por otro lado, el tiempo de ejecución de las operaciones sobre su contenido sea menor.

Para finalizar, es importante matizar que el tamaño de las dos colecciones de documentos utilizadas para evaluar el comportamiento del sistema no es el idóneo. Mediante el uso de colecciones más extensas sería posible tanto realizar evaluaciones con una mayor precisión como obtener resultados más representativos de su funcionamiento.

Aún así, ambas colecciones se han construido tratando que su contenido muestre la variedad existente en el tipo de documentos evaluado. Gracias a esta medida, junto a otras como el uso de CV estratificada o el constante contacto con especialistas de las materias tratadas, las pruebas realizadas ofrecen una cierta idea de cómo un sistema como el propuesto funcionaría frente a colecciones más representativas y extensas.

7.2. Trabajo futuro

Aunque el sistema implementado cumple con los objetivos marcados al inicio del proyecto, en el contexto de este trabajo existen una infinidad de mejoras y nuevas ideas que se pueden aplicar.

Con estas mejoras se dotaría al sistema de una comprensión de contenido textual mayor, lo que le permitiría realizar evaluaciones de forma más precisa y verosímil.

A continuación se exponen algunas de estas posibles mejoras, indicando tanto la razón de su necesidad como las tecnologías o recursos que serían necesarios para su inserción.

7.2.1. Uso de múltiples fuentes de conocimiento

Para determinar la semejanza existente entre dos términos, el sistema propuesto utiliza representaciones en forma de vector generadas por un modelo pre-entrenado utilizando *Word2Vec*.

Los principales problemas de esta aproximación son:

- Tan solo asocia un significado (un vector) a cada término. No contempla la polisemia de forma directa.
- El significado de un término se obtiene, en la mayoría de los casos, a partir de su uso más común.

Por lo tanto, los resultados que ofrece el sistema pueden empeorar cuando se evalúan términos cuyo significado real es distinto al más utilizado (o cuando toma un significado concreto respecto a una cierta materia).

Esto podría resolverse permitiendo que el sistema acceda a distintas fuentes de conocimiento para obtener múltiples representaciones y significados para cada término. Si dispone de varias representaciones, el sistema podrá seleccionar la que más se adecúe al contexto en el que aparece el término para luego realizar la comparación de forma más precisa.

De este modo, cada fuente de conocimiento utilizada tendría una finalidad concreta. Por ejemplo, si queremos evaluar documentos cuya materia principal es el automovilismo, podríamos utilizar dos fuentes de conocimiento:

- Una cuyas representaciones respondan a significados de propósito general.

- Otra cuyas representaciones contengan el significado concreto dentro del automovilismo.

Utilizando ambas fuentes de conocimiento, el sistema tendría acceso a varios significados asociados a un mismo término. Por ejemplo, en la primera fuente de conocimiento, el término "caballo" hace referencia a un animal mientras que, en la segunda, hace referencia a una medida de potencia para vehículos.

Las fuentes de conocimiento utilizadas no tendrían por que ser de un mismo tipo, podrían ser tanto estructuradas como desestructuradas. Por ejemplo, el sistema podría consultar tanto la información almacenada en *WordNet* como la generada por el modelo pre-entrenado de *Word2Vec* utilizado actualmente.

Ahora bien, el uso de múltiples fuentes de conocimiento implica realizar las modificaciones pertinentes sobre el sistema propuesto. A grandes rasgos, implicaría la implementación del acceso a las distintas fuentes, la modificación del proceso de comparación de términos para que se contemplen las distintas opciones, etc.

7.2.2. Desambiguación de términos

En el caso de que se contemplen distintos significados para un mismo término, el propio sistema debe ser capaz de identificar qué significado concreto toman los distintos términos que contiene un texto en función del contexto en el que aparecen.

Este proceso recibe el nombre de desambiguación de un término. En el caso de que el sistema lo incluya, podrá realizar evaluaciones de forma más precisa.

La forma más efectiva de implementar la desambiguación depende de la representación de los términos analizados y de la fuente de conocimiento de la que se ha extraído.

Una de las soluciones con mayor presencia en la industria es el uso del algoritmo de Lesk (Lesk, 1896). Este infiere el significado de un término analizando la información que comparten el contexto en el que aparece y la definición de dicho término.

Por lo tanto, esta aproximación requiere el uso de fuentes de conocimiento en las que se especifique la definición de los términos que contempla. Este

es el caso, por ejemplo, de *WordNet*.

Por otro lado, también existen soluciones que resuelven esta tarea utilizando enfoques basados en semántica distribucional (como *Word2Vec*). Este es el caso de *Sense2Vec* (Trask et al., 2015), el cual consiste en una red neuronal entrenada para realizar esta tarea.

7.2.3. Incrementar la información utilizada para comparar sentencias

En la literatura actual, existen distintas aproximaciones para obtener la similitud existente entre dos sentencias a partir de la semejanza de los términos que las componen.

La utilizada en el sistema propuesto la determina atendiendo a las semejanzas existentes entre los términos que las componen. Todos los términos, mientras sean relevantes, intervienen de igual forma en el cálculo.

En otras aproximaciones, como la de (Mihalcea y Strapparava, 2006), a cada término se le asigna un peso distinto en el cómputo. Este peso depende de su *idf* (*Inverse Document Frequency*).

De este modo, los términos más específicos y que tiendan a aparecen tendrán más peso en la comparación mientras que otros conceptos genéricos (o que tiendan a aparecer en la mayoría de textos) no influirán tanto en el resultado.

Incluir esta diferenciación de pesos en el sistema propuesto, a priori, proporcionaría comparaciones de contenido textual más verosímiles y precisas. Dicha comparación se centraría en la información especifica y discriminatoria, obviando la información trivial o que aparece en la mayoría de los textos.

Por lo tanto, esta es una mejora de especial interés. Para su inserción, sería necesario obtener un documento en el que, para cada término contemplado por el sistema, se especifiquen distintas métricas que determinen su especificidad (*tf*, *idf*, etc). Utilizando esta información, sería posible asignar distintos pesos a cada término.

En el caso de que no dispusiésemos de un documento como el descrito, también sería posible utilizar otras métricas como el tamaño (en caracteres) de cada término, su *PoS*, etc.

Capítulo 8

Introduction

Contenido: This section defines the motivation for the work carried out together with the objectives set at the beginning of it. It also includes an overview of the document that makes it easier to understand.

8.1. Motivation

Among all the tasks that can be performed by applying Natural Language Processing (NLP) and Machine Learning (ML) techniques on a set of documents, the analysis of the quality of a document is presented as one of the most promising challenges and most useful within the industry today.

This is due to multiple factors. One of them is the constant creation of documentation present in the industry. With the implementation, for example, of new laws such as the General Data Protection Regulation (GDPR, April 2016), the need to produce documentation that adapts to the new context imposed by the corresponding law appears, either by updating obsolete documents, or creating new ones.

The creation of all this documentation also implies the need to carry out multiple reviews and analysis on the documents produced. Through these reviews, its quality and validity can be determined against the new established context, thus avoiding the use of the document in the event that it includes aspects that are not adequately adjusted.

Currently, to determine if a document complies with the restrictions imposed by a certain context (whether it is a law, a quality standard, a series of recommendations, etc.), it is necessary for those responsible for the text to analyze each one manually and identify the aspects that determine if a

document is valid or appropriate to the specific context.

This analysis implies a great economic and human effort. Therefore, it would be of special interest to experiment with different techniques that allow us to implement approaches that reduce the effort required to perform this task.

This context has led to *ECIX Group* determining the need for such a system. Thanks to their collaboration in the project, this work proposes a system that, through different natural language processing techniques, has the ability to perform these revisions automatically.

In this way, a system of this type would automatatize this process, considerably releasing professionals in the corresponding field from their workload.

8.2. Objectives

The main objective of this work is to build a system that has the ability to qualify different documents based on their completeness, using different natural language processing techniques.

This system will be built taking advantage of the resources on the evaluation of documents present on the Internet and those offered by *ECIX Group*. In addition, it will apply different Natural Language Processing, Information Extraction and Machine Learning techniques used in current literature.

In order for the system to be scalable and also be used for different types of documents, the system must be adaptable to different domains. This implies that its use should not be restricted to a single subject, or type of document.

To do this, the system will use different criteria (depending on the domain in which it is used) to carry out this evaluation, which can be specified manually or can be extracted automatically.

The tasks covered in this work could be classified into the following fields of computing:

- **Natural language processing**: The system will apply different techniques to analyze the textual content present in the evaluated documents and to carry out the corresponding transformations on it.
- **Information extraction**: The system will have the ability to analyze

textual knowledge in which the criteria that determine the quality of a certain type of document are specified in order to extract the relevant information for future evaluations .

- **Comparison of textual knowledge**: In order to search for the information that determines the quality of a document to be evaluated (the criteria), the system must compare fragments of textual content, determining what knowledge they share, how they differ, etc.
- **Machine Learning**: The system will have the ability to adjust its behavior in order to more plausibly qualify different types of documents. This adjustment will be made through the analysis of the evidence previously evaluated manually.

8.3. General vision of the document

The document is divided into several sections that are indicated below:

1. Análisis de la tarea: the task to be solved is defined and analyzed, indicating what factors affect it, its presence in the current literature and other related tasks of interest.
2. Fundamentos: the foundations that give rise to the proposed system are exposed. It includes explanations about these, about their presence in the current literature, and about the different techniques or resources that they encompass, and that are especially useful for solving the proposed problem.
3. Funcionamiento del sistema propuesto: the different operations that give rise to the proposed system and its main uses are described, both at a high level, and with detail.
4. Componentes del sistema. The different components that interact with each other are described to give rise to the functionalities of the proposed system, indicating their main functions, how they are related to each other and the external technologies used for it.
5. Resultados obtenidos: In this section, the results obtained using the proposed system against two different document collections are indicated and analyzed. In addition, the results obtained by using *WordNet* and *Word2Vec* are compared to compare terms using a third collection.
6. Conclusiones y trabajo futuro: includes different conclusions that can be drawn from the work done, once the proposed system has been implemented and the results obtained have been analyzed. Different

modifications are also indicated on the proposed system that would improve its behaviour.

7. Introduction: Current section. It defines the motivation for the work carried out together with the objectives set at the beginning of it. It also includes an overview of the document that makes it easier to understand.

8. Conclusions and future work: English translated version of section "Conclusiones y trabajo futuro".

9. **Appendix**: Acceso e instalación del sistema. The repository in which the proposed system is stored is indicated. In addition, it includes information on its installation and use.

Capítulo 9

Conclusions and future work

Contenido: includes different conclusions that can be drawn from the work done, once the proposed system has been implemented and the results obtained have been analyzed. Different modifications are also indicated on the proposed system that would improve its behaviour.

9.1. Conclusion

Once the specification and implementation of the proposed system has been completed, evaluating the results achieved, we can affirm that the resulting system meets the objectives set at the beginning of the project.

First, the system has the ability to plausibly and accurately evaluate different documents. This is mainly due to the auto-configuration functionality it offers.

Thanks to it, the system obtains the ability to analyze a previously tagged collection of documents, and modify their settings in order to correctly qualify documents.

Furthermore, the proposed system is scalable and can be used in different domains. Depending on the criteria entered (either manually or automatically) and the collection through which the self-configuration is carried out, the system will be able to evaluate different types of documents.

Its usefulness is not focused on a specific domain or type of documents. It can be used on all types of documents whose completeness can be determined by evaluating the presence, implicitly or explicitly, of a series of criteria.

The results achieved on a certain type of document will depend on the quality of the criteria used (or the sources from which they are extracted) and the collection used to carry out the self-configuration. If the criteria used do not match the evaluated collection, the system will not be useful.

In addition, the system offers the functionality of automatically extracting representations of criteria from documents in which they are described. In the event that you extract irrelevant information from them, this information is filtered during the auto-configuration process.

In this way, the proposed system frees users from the labor of manually specifying the criteria that determine the quality of a certain type of documents, saving temporary and human effort.

On the other hand, throughout the development of this project, the usefulness of approaches based on structured knowledge has been tested (specifically, in *WordNet*). Due to the runtime that its use implies, it does not seem a viable option to implement systems like the one proposed.

This factor may disappear using a structured knowledge source that requires fewer queries or, on the other hand, the execution time of operations on its content is less.

Finally, it is important to clarify that the size of the two document collections used to evaluate the behavior of the system is not ideal. By using larger collections, it would be possible to carry out evaluations with greater precision and to obtain more representative results of their operation.

Even so, both collections have been built trying to show the existing variety that appers in these kind of documents. Thanks to this measure, along with others such as the use of stratified CV or constant contact with specialists in the subjects covered, the tests carried out offer a certain idea of how a system, such as the one proposed, would work against more representative, and extensive collections.

9.2. Future work

Although the implemented system meets the objectives set at the beginning of the project, in the context of this work there are an infinite number of improvements and new ideas that can be applied.

These improvements would provide the system with a greater understanding of textual content, allowing it to make evaluations more accurately and plausibly.

Below are some of these possible improvements, indicating both the reason for their need and the technologies or resources that would be necessary for their insertion.

9.2.1. Multiple knowledge sources usage

To determine the similarity between two terms, the proposed system uses vector representations generated by a pre-trained model using *Word2Vec*.

The main problems of this approach are:

- Just associate a meaning (a vector) to each term. It does not contemplate polysemy directly.
- The meaning of a term is obtained, in most cases, from its most common use.

Therefore, the results offered by the system may worsen when evaluating terms whose real meaning is different from the most used (or when it takes a specific meaning regarding a certain subject).

This could be solved by allowing the system to access different sources of knowledge to obtain multiple representations and meanings for each term. If you have several representations, the system will be able to select the one that best suits the context in which the term appears, and then make the comparison more precisely.

In this way, each source of knowledge used would have a specific purpose. For example, if we want to evaluate documents whose main subject is motorsports, we could use two sources of knowledge:

- One whose representations respond to general purpose meanings.
- Another whose representations contain the concrete meaning within motorsports.

Using both sources of knowledge, the system would have access to various meanings associated with the same term. For example, in the first source of knowledge, the term ”horse” refers to an animal while, in the second, it refers to a power measurement for vehicles.

The sources of knowledge used would not have to be of the same type, they could be both structured and unstructured. For example, the system could query both the information stored in *WordNet* and that generated by the pre-trained model of *Word2Vec* currently used.

However, the use of multiple sources of knowledge implies making the pertinent modifications on the proposed system. Broadly speaking, it would imply the implementation of access to the different sources, the modification of the term comparison process so that the different options are considered, etc.

9.2.2. Word sense disambiguation

In the event that different meanings are contemplated for the same term, the system itself must be able to identify what specific meaning the different terms contained in a text take depending on the context in which they appear.

This process is called disambiguation of a term. In the event that the system includes it, you will be able to carry out evaluations more precisely.

The most effective way to implement disambiguation depends on the representation of the analyzed terms and the source of knowledge from which it has been extracted.

One of the solutions with the largest presence in the industry is the use of the Lesk algorithm (Lesk, 1896). It infers the meaning of a term by analyzing the information shared by the context in which it appears and the definition of that term.

Therefore, this approach requires the use of sources of knowledge in which the definition of the terms it contemplates is specified. This is the case, for example, of *WordNet*.

On the other hand, there are also solutions that solve this task using approaches based on distributional semantics (such as *Word2Vec*). This is the case of *Sense2Vec* (Trask et al., 2015), which consists of a neural network trained to perform this task.

9.2.3. Increasing the information used to compare sentences

In the current literature, there are different approaches to obtain the similarity between two sentences from the similarity of the terms that make them up.

The one used in the proposed system is determined based on the similarities between the terms that compose them. All the terms, as long as they are relevant, intervene in the same way in the calculation.

In other approaches, such as that of (Mihalcea and Strapparava, 2006), each term is assigned a different weight in the computation. This weight depends on your *idf* (*Inverse Document Frequency*).

In this way, the more specific terms that tend to appear will have more weight in the comparison while other generic concepts (or that tend to appear in most texts) will not influence the result as much.

Including this differentiation of weights in the proposed system, a priori, would provide more plausible and accurate comparisons of textual content. Such a comparison would focus on specific and discriminatory information, avoiding the trivial information or that appears in most texts.

Therefore, this is an improvement of special interest. To insert it, it would be necessary to obtain a document in which, for each term contemplated by the system, different metrics are specified that determine its specificity (*tf*, *idf*, etc). Using this information, it would be possible to assign different weights to each term.

In the event that we did not have a document like the one described, it would also be possible to use other accessible metrics with the technology and resources present in the current system. Some examples are the size (in characters) of each term, its *PoS*, etc.

Apéndice A

Acceso e instalación del sistema

Contenido: Se indica el repositorio en el que se almacena el sistema propuesto. Además, incluye información de utilidad sobre su instalación y uso.

El sistema que se describe en este trabajo se puede descargar a través de este enlace (https://github.com/bselles/Sistema-para-la-calificacion-de-documentos-en-base-a-su-completitud).

Está implementado en *Python* 3.6 sobre *anaconda 3*. Por lo tanto, para utilizar el sistema, es necesario instalar ambas plataformas y lanzar el sistema sobre ellas.

En el caso de *anaconda* 3, no es estrictamente necesario su uso, también se puede utilizar cualquier plataforma que permita instalar las dependencias descritas abajo. Aún así, el sistema se ha probado utilizando esta plataforma de modo que se recomienda su uso.

Además, para utilizar las distintas funcionalidades que ofrece, el sistema emplea una serie de librerías *Python* que es necesario descargar e instalar. A continuación se indican los paquetes necesarios junto a los comandos que pueden usarse para descargar e instalar estas dependencias:

- **nltk 3.4.4**: Instalación vía pip usando "pip install nltk". También es necesario instalar el **complemento de *stopwords* de *nltk*** con el siguiente comando: nltk.download('stopwords')
- **pandas 0.24.2**: Instalación vía pip usando "pip install pandas".
- **spacy 2.1.4**: Instalación vía pip usando "pip install spacy". Además, también es necesario:

  - **Descargar el modelo de spacy en español**: python -m spacy download es_core_news_sm.
  - **Complemento para la detección de lenguajes vía spacy**: pip install spacy-langdetect

- **gensim 3.8.1**: Instalación vía pip mediante el siguiente comando: pip install gensim.

- **multi-rake 0.0.1**: Vía el instalador de *anaconda*. Comando: conda install -c conda-forge multi_rake.

Es importante matizar que el sistema se ha probado en *Ubuntu*. Por lo tanto, es posible que algunas dependencias que en *linux* aparecen de serie no estén en otro sistema operativo. En ese caso, será necesario descargarlas e instalarlas. En los propios errores de *Python* aparecerían las supuestas dependencias.

Una vez se han instalado las dependencias descritas, tan solo es necesario crear una instancia de la clase "system" (del fichero "system.py") para así poder utilizar las distintas funcionalidades que ofrece el sistema. En el repositorio de *Github* en el que se ubica el sistema aparece más información sobre las dependencias del sistema y sobre su uso.

Bibliografía

[1] AdventuresInMachineLearning. Word2vec word embedding tutorial in python and tensorflow. https://adventuresinmachinelearning.com/word2vec-tutorial-tensorflow/, last access: 17/07/20.

[2] K. Agarwal. Rake: Rapid automatic keyword extraction algorithm. https://medium.com/datadriveninvestor/rake-rapid-automatic-keyword-extraction-algorithm-f4ec17b2886c, last access: 22/06/20.

[3] E. Agirre and A. Soroa. Personalizing pagerank for word sense disambiguation. , Proceedings of the 12th conference of the European chapter of the Association for Computational Linguistics (EACL-2009), Athens, Greece, 2009.

[4] E. Agirre, A. Soroa, E. Alfonseca, K. Hall, J. Kravalova, and M. Pasca. A study on similarity and relatedness using distributional and wordnet-based approaches. , Proceedings of annual meeting of the North American Chapter of the Association of Computational, 2009.

[5] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. , Journal of Machine Learning Research 3, 2003.

[6] F. Bond and R. Foster. Linking and extending an open multilingual wordnet. , Nanyang Technological University, Singapur, 2013.

[7] E. G. Caldarola, A. Picariello, and A. M. Rinaldi. Big graph-based data visualization experiences the wordnet case study. , 2015.

[8] C. Cardellino. Spanish billion words corpus and embeddings. https://crscardellino.github.io/SBWCE/, last access: 22/06/20.

[9] CodeLingo. Keyword extraction using rake. https://codelingo.wordpress.com/2017/05/26/keyword-extraction-using-rake/, last access: 22/06/20.

[10] V. Cozza, M. Petrocchi, and A. Spognardi. A matter of words: Nlp for quality evaluation of wikipedia medical articles. , IIT CNR, Pisa, Italy and DTU Lingby, Denmark, 2016.

[11] O. Ferrández, R. Muñoz, R. Muñoz, P. Martínez-Barco, and M. Palomar. Un sistema basado en conocimiento para el reconocimiento de implicación textual. , Universidad de Alicante, Alicante, España, 2006.

[12] O. Ferschke. The quality of content in open online collaboration platforms: Approaches to nlp-supported information quality management in wikipedia. , From the Computing Department Darmstadt Technical University, 2014.

[13] E. Gabrilovich and S. Markovich. Computing semantic relatedness using wikipedia-based explicit semantic analyisis. , IJCAI Internation Joint Conference on Artificial Intellifence, 2007.

[14] A. Gonzalez-Agirre. Computational models for semantic textual similarity. , Universidad del País Vasco, Donostia, País Vasco, España, 2017.

[15] G. Génova, J. M. Fuentes, J. Llorens, O. Hurtado, and V. Moreno. A framework to measure and improve the quality of textual requirements. , Universidad Carlos III, Madrid, España, 2011.

[16] M. H. Haggag, M. M. A. Elfattah, and A. M. Ahmed. Different models and approaches of textual entailment recognition. , 2016.

[17] G. Hirst and D. S. Onge. Lexical chains as representations of context for the detection and corretion of malapropisms. , MIT Press, 1998.

[18] T. Hughes and D. Ramage. Lexical semantic relatedness with random graph walks. , Stanford University, Stanford, California, Estados Unidos, 2007.

[19] InsightsBot. Word2vec explained easily. http://www.insightsbot.com/word2vec-explained-easily/, last access: 17/07/20.

[20] J. Jiang and D. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. , University of Waterloo, Ontario, Canada and McMaster University, Ontario, Canada, 1997.

[21] C. Leacock and M. Chodorow. Combining local context and wordnet similarity for word sense identification. , MIT Press, 1998.

[22] M. Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. , Bell Communications Research, Morristown, New Jersey, Estados Unidos, 1986.

[23] C. Lin, R. Socher, C. Manning, and A. Ng. Parsing natural scenes and natural language with recursive neural networks. , Proceedings of ICML, 2011.

[24] D. Lin. Using syntactic dependency as local context to resolve word sense ambiguity. , Association for Computational Linguistics, 1997.

[25] K. Lund and C. Burgess. Producing high-dimensional semantic space from lexical co-ocurrence. , Behavior Research Methods Instruments and Computers, 1996.

[26] G. Majumder, P. Pakray, A. Gelbukh, and D. Pinto. Semantic textual similarity methods, tools and applications: A survey. , 2016.

[27] R. Mihalcea and C. Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. , Proceedings of the American Association for Artificial Intelligence (AAAI 2006), 2006.

[28] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionaly. , Google Inc, Mountain View, Santa Clara, California, 2013.

[29] R. Milhacea and P. Tarau. Textrank: Bringing order into texts. , Department of Computer Science, University of North Texas, Texas, United States, 2004.

[30] G. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. Introduction to wordnet: an on-line lexical database. , Internation Journal of Lexicography, 1990.

[31] D. Milne and I. Witten. Learning to link with wikipedia. , University of Waikato, Hamilton, New Zeland, 2008.

[32] MonkeyLearn. Keyword extraction. https://monkeylearn.com/keyword-extraction/, last access: 22/06/20.

[33] NLTK. Collocations overview using nltk. http://www.nltk.org/howto/collocations.html, last access: 22/06/20.

[34] L. Page. Method for node ranking in a linked database. , Leland Stanford Junior University, Stanford, California, United States, 1998.

[35] P. Pakray, A. Gelbukh, and S. Bandyopadhyay. Aplicación de la implicación textual en un sistema de la validación de respuestas automáticas. , Research in Computing Science, An open access research journal on Computer science and computer engineering., 2013.

[36] J. B. Pollack. Recursive distributed representations. , 1990.

[37] R. Rada, H. Mili, E.Bicknell, and M. Blettner. Development and application of a metric on semantic nets. , IEEE Transactions on Systems, Man, and Cybernetics, 1989.

[38] S. Reese, G. Boleda, M. Cuadros, L. Padró, and G. Rigau. Wikicorpus: A word-sense disambiguated multilingual wikipedia corpus. , In Proceedings of 7th Language Resources and Evaluation Conference (LREC'10), La Valleta, Malta, 2010.

[39] P. Resnik. Using information content to evaluate semantic similarity in taxonomy. , Proceedings of the 14th IJCAI International Joint Conference on Artificial Intelligence, 1995.

[40] S. Rose, D. Engel, N. Cramer, and W. Cowley. Automatic keyword extraction from individual documents. , From journal of Text Mining:Applications and Theory, 2010.

[41] SemEval. Semeval 2020 web site. http://alt.qcri.org/semeval2020/, last access: 22/06/20.

[42] R. Socher, C. Manning, and A. Ng. Learning continuous phrase representations and syntactic parsing with recursive neural networks. , Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop, 2010.

[43] M. Strube and S. Ponzetto. Wikirelate! computing semantic relatedness using wikipedia. , EML Research gGmbH, Heidelberg, Alemania, 2006.

[44] D. Ten. Keyword and sentence extraction with textrank (pytextrank). https://xang1234.github.io/textrank/, last access: 22/06/20.

[45] A. Trask, P. Michalak, and J. Liu. sense2vec - a fast and accurate method for word sense disambiguation in neural word embeddings. , Digital Reasoning Systems, Inc. Nashville, TN, USA, 2015.

[46] R. Y. Wang and D. M. Strong. Beyond accuracy: What data quality means to data consumers. , Journal of Management Information Systems, Vol. 12, No. 4, 1996.

[47] Wikipedia. History of wikipedia. https://en.wikipedia.org/wiki/History_of_Wikipedia, last access: 22/06/20.

[48] Z. Wu and M. Palmer. Verb semantics and lexical selection. , Association for Computational Linguistics, 1994.

[49] E. Yeh, D. Ramage, C. Manning, E. Airre, and A. Soroa. Wikiwalk: random walks on wikipedia from semantic relatedness. , Stanford University, Stanford, California, Estados unidos y Universidad del País Vasco, Donostia, País Vasco, España, 2008.