

Universidad
Rey Juan Carlos

MÁSTER EN DATA SCIENCE

CURSO ACADÉMICO 2017-2018

Trabajo de fin de Máster

ANÁLISIS DE SIMILITUD ENTRE ARTÍCULOS Y AUTORES

Autor: Carlos Sánchez Vega

Tutores: Alberto Fernández-Isabel

Agradecimientos

Por una parte quisiera agradecer el esfuerzo dedicado por mi tutor, Alberto Fernández Isabel, quien desde el primer momento siempre se mostró dispuesto a ayudar.

En segundo lugar, en el plano personal, quisiera agradecer a mi familia el apoyo que me han dado en todo momento. Ellos me animaron a seguir cuando la meta se veía tan lejana.

¡Muchas gracias!

Resumen

El procesamiento de lenguajes naturales (PLN) es un campo de las ciencias de la computación, inteligencia artificial y lingüística que estudia las interacciones entre las computadoras y el lenguaje humano. Entre las diferentes funcionalidades que nos ofrece el procesado de texto, podemos destacar el cómputo de la semejanza de textos de manera analítica.

El objetivo del proyecto de fin de máster consiste en el desarrollo de un prototipo de herramienta para el cálculo de la semejanza, ya sea entre investigadores o textos, procedentes del buscador académico de Semantic Scholar. La finalidad es crear una aplicación fácil de usar y funcional, que permita mostrar de manera gráfica las semejanzas. Para tal cometido, se han usado diferentes técnicas de visualización, como pueden ser las herramientas de reducción de la dimensionalidad TSNE o MDS.

En cuanto al tipo de funcionalidad que permite reflejar, podemos citar el cálculo de la semejanza entre autores de papers, semejanza entre textos procedentes del mismo autor o, por último, el análisis de la semejanza entre textos de diferentes autores.

Además, se ha añadido como funcionalidad la visualización del grafo de las relaciones de un autor, los coautores de sus obras y las obras que los interrelacionan (puesto que puede que un paper de investigación esté escrito por más de un autor).

Palabras clave

PLN, TSNE, MDS, semejanza, visualización, grafo

Summary

Natural language processing (NLP) is a field from computer science, artificial intelligence and linguistics which examines the interactions between computers and human language. As far as the functionalities provided by text processing are concerned, we can mention the study of the similarity between texts in an analytic way.

The aim of the final work of the master's degree lie in the development of a tool prototype to calculate similarity, either between professors or texts, from the academic searcher Semantic Scholar. The target is to create a functional and easy to use application, which allows us to show similarities graphically. To carry out that objective, we have utilized some visualization methods, as the tools related with dimensionality reduction of TSNE or MDS .

As for the type of functionality the application allows to show, we can mention the similarity analysis between authors and papers, similarity of texts text by the same author or, finally, similarity analysis between texts of different authors.

Furthermore, we have included as a functionality the graph visualization of the relationships between author, coauthors and the papers interrelating them (a paper may have been written by more than one author).

Keywords

NLP, TSNE, MDS, similarity, visualization, graph

Índice

Capítulo 1.....	11
Introducción.....	11
1.1. Información general y contexto.....	11
1.2. Objetivos principales del proyecto.....	13
1.3. Estructura de la Memoria.....	13
Capítulo 2.....	15
Estado del arte.....	15
2.1. NIVELES DE COMPRENSIÓN DEL LENGUAJE.....	15
2.2. APRENDIZAJE AUTOMÁTICO EN EL PROCESAMIENTO DE LENGUAJE.....	16
2.3. Aplicaciones del PLN.....	17
2.3.1. Resúmenes de textos:.....	17
2.3.2. Traducción automática entre lenguas:.....	17
2.3.3. Crear chatbots.....	17
2.3.4. Generar automáticamente etiquetas de palabras clave.....	17
2.3.5. Reconocer entidades.....	18
2.3.6. Análisis de sentimientos.....	18
2.3.7. Enfoques usados en ésta propuesta:.....	18
2.4. Técnicas de NLP utilizadas: Stemming y Lematización.....	18
2.4.1. Stemming.....	19
2.4.2. Lematización.....	19
2.4.3. Tokenización.....	20
2.4.4. Bag of words (bow).....	20
2.4.5. Term frequency – Inverse document frequency (TF-IDF).....	20
2.4.6. Term frequency(TF).....	20
2.4.7. IDF.....	21
2.5. Análisis de semejanza entre documentos.....	22
2.5.1. Medidas de similitud.....	23
2.5.1.1. Similitud coseno.....	23
2.5.1.2. Similitud Jaccard.....	23
2.6. Algoritmos de aprendizaje.....	24
2.6.1. Algoritmos no supervisados.....	24
2.6.2. Algoritmos supervisados.....	25
2.7. Tecnología usada.....	25
2.7.1. NLTK.....	25
2.7.1. Python.....	26
2.7.2. D3.js.....	26
2.7.3. MongoDB.....	26
2.8. Base de datos.....	26
2.9. Métodos de reducción de dimensionalidad.....	28
2.9.1. MDS.....	28
2.9.2. t-SNE.....	29
Capítulo 3.....	31
Alcance funcional de la propuesta.....	31
3.1. Desarrollo de la interfaz de usuario.....	32
3.1.1. Grafo asociado.....	33
3.1.2. Comparador del autor con otros autores.....	35
3.1.3. Comparador de obras del mismo autor.....	36

3.1.4.Comparador de obras entre todos los autores.....	36
Capítulo 4.....	39
Cálculo de reputación.....	39
Capítulo 5.....	41
Modelo de desarrollo.....	41
Etapas de desarrollo.....	42
5.1.Estudio alternativas de recogida de datos.....	43
5.2.Limpieza de la base de datos.....	46
5.3.Uso del algoritmo para el cálculo de las semejanzas.....	46
5.4.Reducción de dimensionalidad (t-SNE y MDS).....	47
5.5.Visualización de los métodos de reducción.....	47
5.6.Implementación de visualización del grafo relacional.....	47
5.7.Implementación de aplicación de escritorio.....	47
Capítulo 6.....	49
Casos de estudio.....	49
Pruebas parciales.....	49
Pruebas de integración.....	49
6.1. Experimento de la similitud entre autores.....	50
6.2. Experimento de la similitud entre obras de un autor.....	52
Capítulo 7.....	55
Conclusiones y trabajo futuro.....	55
7.1. Conclusiones.....	55
7.2.Trabajo Futuro.....	56

Índice de ilustraciones

Ilustración 1: Niveles del lenguaje.....	17
Ilustración 2: Etapas en el cálculo de la semejanza entre elementos.....	24
Ilustración 3: Similitud de Jaccard.....	25
Ilustración 4: Colección de los autores.....	28
Ilustración 5: Colección de las publicaciones.....	28
Ilustración 6: Esquema funcional de la propuesta.....	33
Ilustración 7: Interfaz de usuario.....	33
Ilustración 8: Mensaje informativo respecto sobre un autor.....	34
Ilustración 9: Mensaje informativo sobre un artículo.....	35
Ilustración 10: Comparador del autor con otros autores.....	36
Ilustración 11: Comparador de obras del mismo autor.....	37
Ilustración 12: Comparador de obras entre todos los autores.....	38
Ilustración 13: Botonera.....	38
Ilustración 14: Desarrollo iterativo.....	44
Ilustración 15: Etapas del desarrollo del prototipo.....	44
Ilustración 16: Proceso de almacenaje en la BBDD.....	45
Ilustración 17: Consulta a la API de DBLP en búsqueda de autores iguales al buscado.....	47
Ilustración 18: url con pid del autor.....	47
Ilustración 19: Se recoge el identificador de uno de sus artículos.....	48
Ilustración 20: Consulta final a la API de Semantic Scholar.....	49
Ilustración 21: Matriz de correlación de similitud entre autores.....	54
Ilustración 22: MDS de similitud entre autores.....	55
Ilustración 23: Matriz de correlación de obras de Alberto Fernández-Isabel.....	55
Ilustración 24: MDS de similitud entre obras.....	56
Ilustración 25: Varias personas comparten nombre en Semantic Scholar.....	60

Capítulo 1

Introducción

En la época actual el uso de los recursos naturales, industriales y humanos depende del manejo eficiente de la información y conocimiento. Gracias a la llegada de los ordenadores y el desarrollo tecnológico relacionados, el procesamiento del conocimiento ha sido mayor. Sin embargo, lo que es conocimiento para nosotros, los seres humanos, no lo es para las computadoras. La computadora almacena datos e información en archivos, puede realizar acciones con ellos, pero no puede encontrar respuestas a preguntas formuladas, hacer inferencias sobre su contenido, generalizar y resumirlo, por sí solos. Sin embargo, el PLN posibilita dichas tareas, creando mecanismos eficaces computacionalmente para la comunicación entre personas y máquinas por medio de lenguajes naturales.

1.1. Información general y contexto

El procesamiento de lenguaje natural (PLN) o NLP (en inglés) es el campo que estudia la comprensión y manipulación del lenguaje natural humano, es decir tal y como nos expresamos por escrito o de viva voz, por parte de un ordenador.

Las primeras patentes de "máquinas de traducción" se solicitaron a mediados de la década de 1930. Una propuesta de Georges Artsrouni era simplemente un diccionario automático bilingüe que utilizaba cinta de papel. La otra propuesta, de Peter Troyanskii, un ruso, era más detallada. Incluía tanto el diccionario bilingüe como un método para tratar los roles gramaticales entre idiomas, basado en el esperanto.

En 1950, Alan Turing publicó su famoso artículo "Computing Machinery and Intelligence" [1] que proponía lo que hoy llamamos test de turing como criterio de inteligencia.

En 1954, la universidad de Georgetown se implicó en un proyecto que consistía la traducción completamente automática de más de sesenta frases rusas al inglés. En aquel momento se pensaba que en menos de cinco años, la traducción automática sería un problema resuelto. Sin embargo, el progreso real fue mucho más lento y no se cumplieron con las expectativas. Además, los fondos para la traducción automática se redujeron drásticamente.

En 1969 Roger Schank introdujo la teoría de la dependencia conceptual para la comprensión del lenguaje natural [2] Este modelo, parcialmente influenciado por el trabajo de Sydney Lamb, fue

ampliamente utilizado por los estudiantes de Schank en la Universidad de Yale, como Robert Wilensky, Wendy Lehnert y Janet Kolodner.

En 1970, William A. Woods introdujo la red de transición aumentada (ATN) para representar la entrada de lenguaje natural [3]. En lugar de reglas de estructura de frases, las ATN utilizaban un conjunto equivalente de autómatas de estado finito que se llamaban recursivamente. Durante los años 70, muchos programadores comenzaron a escribir "ontologías conceptuales", que estructuraban la información del mundo real en datos comprensibles por ordenador. Algunos ejemplos son Margie (Schank, 1975), SAM (Cullingford, 1978), PAM (Wilensky, 1978), TaleSpin (Meehan, 1976), QUALM (Lehnert, 1977), Politics (Carbonell, 1979) y Plot Units (Lehnert 1981). Durante este tiempo, muchos chatterbots fueron escritos incluyendo PARRY, Racter, y Jabberwacky.

Hasta la década de 1980, la mayoría de los sistemas de PNL se basaban en complejos conjuntos de reglas escritas a mano. Sin embargo, a partir de finales de la década de 1980, se produjo una revolución en la PNL [4]. con la introducción de algoritmos de aprendizaje automático para el procesamiento de idiomas. Esto se debió tanto al aumento constante de la potencia computacional resultante de la Ley de Moore, como a la disminución gradual del dominio de las teorías chomskianas de la lingüística (por ejemplo, la gramática transformacional), cuyos fundamentos teóricos se oponían al tipo de lingüística de corpus, base en la que se apoya el enfoque de aprendizaje automático del procesamiento de idiomas. Algunos de los algoritmos de aprendizaje automático más utilizados anteriormente, como los árboles de decisión, producían sistemas de reglas difíciles de aplicar, aunque no muy parecidas, a las reglas escritas a mano existentes. Sin embargo, la investigación se ha centrado cada vez más en modelos estadísticos, que toman decisiones probabilísticas basadas en la asignación de pesos reales a las características que componen los datos de entrada. Los modelos de lenguaje de caché en los que se basan muchos sistemas de reconocimiento de voz son ejemplos de estos modelos estadísticos.

Muchos de los éxitos iniciales se produjeron en el campo de la traducción automática, debido especialmente al trabajo en IBM Research, donde se desarrollaron los modelos estadísticos más complicados. Estos sistemas pudieron aprovechar los corpus textuales multilingües existentes que habían sido producidos por el Parlamento de Canadá y la Unión Europea como resultado de las leyes que exigían la traducción de todos los procedimientos gubernamentales a todos los idiomas oficiales de los sistemas de gobierno correspondientes. Sin embargo, la mayoría de los demás sistemas dependían de corpus desarrollados específicamente para las tareas implicadas.

El NLP intenta hacer comprensible el lenguaje humano para una máquina en 5 grandes áreas: la fonología, la morfología, la sintaxis, la semántica y la pragmática. Su enemigo es la ambigüedad, algo de lo que el lenguaje humano está repleto, ya sea por el uso de la ironía, el sarcasmo, los registros informales, los errores de pronunciación o escritura, los emojis, la mezcla de idiomas y tantas otras variantes que afectan al lenguaje humano escrito y hablado. Hay progresos importantes (por ejemplo, Siri, Alexa, Google Now o Cortana) pero todavía hay un camino largo por recorrer, Por tanto, que una máquina sea capaz de comprendernos y dar respuesta no es tarea fácil todavía.

1.2 Objetivos principales del proyecto

Este proyecto de final de máster tiene como objetivo crear una aplicación que permita, de manera funcional y sencilla, reflejar las semejanzas entre textos y autores de documentos académicos. Dicho análisis permitirá crear una medida de la relación de similitud entre artículos escritos por autores y coautores de un artículo o, además, establecer la similitud entre artículos de los autores.

Por otro lado, durante el desarrollo del prototipo se descubrió la fuerte interrelación entre papers académicos y autores. Por ello, se desarrolló una nueva funcionalidad que tenía como objetivo representar, en forma de grafo, la relación existente entre papers del mismo autor, o diferentes autores, así como la relación entre ellos.

Se realizarán una serie de experimentos para comprobar la relación de semejanza entre los diferentes elementos, ya sea artículos o autores, que permitirá reflejar gráficamente la relación entre los elementos.

El desarrollo se realizó de manera iterativa. Es decir, a partir de los resultados completados en las iteraciones anteriores, se fueron añadiendo nuevos objetivos/requisitos o mejorando los que ya fueron completados.

1.3. Estructura de la Memoria

- En el Capítulo 2, haremos hincapié en el estado de la arte, es decir, citaremos los ámbitos de estudio más destacados para el proyecto y comentaremos qué se ha hecho recientemente sobre el tema seleccionado.
- En el Capítulo 3, se define el diseño elegido y la estrategia de desarrollo del proyecto.. En éste apartado describiremos las diferentes decisiones tomadas, las fases de funcionamiento y los algoritmos seleccionados para el desarrollo de la aplicación.
- En el Capítulo 4, hablaremos de la implementación de la propuesta y la estrategia de desarrollo. Además se detallan las etapas por las que ha pasado el desarrollo de la propuesta.
- En el Capítulo 5 se detallan el modelo de desarrollo seleccionado para implementar el prototipo
- En el Capítulo 6 se mostrarán las diferentes etapas por las que ha pasado el desarrollo de la herramienta.
- Finalmente, en el Capítulo 7 se mostrarán las conclusiones a las que se ha llegado con el desarrollo y se hará hincapié en las posibles líneas de trabajo futuras existentes.

Capítulo 2

La lingüística, es la ciencia que estudia el lenguaje humano. Como disciplina científica que es, ha sistematizado el estudio de este objeto distinguiendo diferentes niveles de análisis. Aunque hay algunas diferencias entre los autores, existe consenso en distinguir a lo menos seis niveles focales: fonético, fonológico, morfológico, sintáctico, semántico y discursivo (conviene advertir que estos no son niveles nítidamente diferenciados unos de otros). Existen, en efecto, zonas de traslape donde se intersectan los dominios de unos y otros (hay traslape entre fonética y la fonología; morfología y sintaxis, etc).

Estado del arte

En esta sección nos centraremos en los aspectos generales del procesamiento de lenguaje natural. Se comienza hablando de su definición, seguiremos hablando de los distintos niveles de comprensión, citaremos algunas de sus funcionalidades en el mundo real y, por último, hablaremos de las dos corrientes existentes para analizar para analizar el lenguaje.

El Procesamiento del Lenguaje Natural o NLP es una disciplina que se encuentra en la unión de otras de varias ciencias, tales como las Ciencias de la Computación, la Inteligencia Artificial [5] y psicología cognitiva [6]. Su idea central es la de darle a las máquinas la capacidad de leer y comprender los idiomas que hablamos los humanos. La investigación del Procesamiento del Lenguaje Natural [7] tiene como objetivo responder a la pregunta de cómo las personas son capaces de comprender el significado de una oración oral / escrita y cómo las personas entienden lo que sucedió, cuándo y dónde sucedió; y las diferencias entre una suposición, una creencia o un hecho.

2.1. NIVELES DE COMPRENSIÓN DEL LENGUAJE

En general, en Procesamiento del Lenguaje Natural se utilizan seis niveles de comprensión con el objetivo de descubrir el significado del discurso. Estos niveles son:

- Nivel fonético: Aquí se presta atención a la fonética, la forma en que las palabras son pronunciadas. Este nivel es importante cuando procesamos la palabra hablada, no así cuando trabajamos con texto escrito.
- Nivel morfológico: Aquí nos interesa realizar un análisis morfológico del discurso; estudiar la estructura de las palabras para delimitarlas y clasificarlas.

- Nivel sintáctico: Aquí se realiza un análisis de sintaxis, el cual incluye la acción de dividir una oración en cada uno de sus componentes.
- Nivel semántico: Este nivel es un complemento del anterior, en el análisis semántico se busca entender el significado de la oración. Las palabras pueden tener múltiples significados, la idea es identificar el significado apropiado por medio del contexto de la oración.
- Nivel discursivo: El nivel discursivo examina el significado de la oración en relación a otra oración en el texto o párrafo del mismo documento.
- Nivel pragmático: Este nivel se ocupa del análisis de oraciones y cómo se usan en diferentes situaciones. Además, también cómo su significado cambia dependiendo de la situación.

Todos los niveles descritos anteriormente, son inseparables y se complementan entre sí. El objetivo de los sistemas de NLP [8] es incluir estas definiciones en una computadora y luego usarlas para crear una oración estructurada y sin ambigüedades con un significado bien definido.

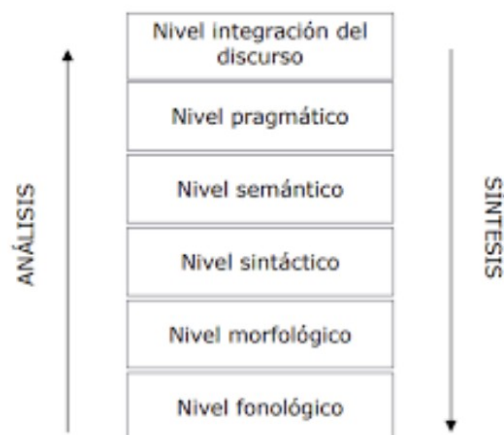


Ilustración 1: Niveles del lenguaje

2.2. APRENDIZAJE AUTOMÁTICO EN EL PROCESAMIENTO DE LENGUAJE

Los algoritmos de Procesamiento del Lenguaje Natural [9] suelen basarse en algoritmos de aprendizaje automático (machine learning). En lugar de codificar manualmente grandes conjuntos de reglas, el NLP usa aprendizaje automático para aprender estas reglas automáticamente analizando un conjunto de ejemplos y haciendo una inferencia estadística. En general, cuanto más datos analizados, más preciso será el modelo.

2.3. Aplicaciones del PLN

Éstos algoritmos pueden ser utilizados en algunas de las siguientes aplicaciones:

2.3.1. Resúmenes de textos:

El resumen automático permite generar automáticamente breves resúmenes de documentos. Esta tarea es muy importante con el creciente número de documentos que hay en la red y la necesidad de recuperar el contenido. Tradicionalmente es una tarea que se ha llevado a cabo en documentos muy estructurados, por ser más coherentes y contener frases y párrafos clave para describir las ideas principales de un texto. Actualmente también se aplica a textos cortos, no formales y no demasiado estructurados.

2.3.2. Traducción automática entre lenguas:

Las técnicas de traducción automática son aquellas que se llevan a cabo sin la intervención de un humano. Gracias al Procesamiento de Lenguaje Natural se puede llegar a romper la barrera del idioma para conseguir una correcta comunicación entre emisor y receptor.

2.3.3. Crear chatbots

Consiste en un software de Inteligencia Artificial (I.A.) diseñado para realizar una serie de tareas de manera independiente y sin la ayuda de un humano. Por ejemplo, los bots podrían hacer una reserva en un hotel o marcar una fecha en el calendario de nuestro smartphone. El modelo más habitual es el del robot virtual con la capacidad de simular una conversación con una persona, y por ello, cada vez están más presentes en el mundo digital.

2.3.4. Generar automáticamente etiquetas de palabras clave

Con NLP también podemos realizar un análisis de contenido aprovechando el algoritmo de LDA [10] para asignar palabras claves a párrafos del texto. Mediante ésta técnica es posible asignar automáticamente términos de indexación a un texto para facilitar su posterior recuperación. Las unidades léxicas extraídas representan los conceptos propios de un documento y son propuestas como candidatos descriptores para los documentos a indexar.

2.3.5. Reconocer entidades

El reconocimiento de entidades nombradas (NER [11] por sus siglas en inglés) (también conocido como extracción de entidades) es una tarea de extracción de información que busca localizar y clasificar en categorías predefinidas, como personas, organizaciones, lugares, expresiones de tiempo y cantidades, las entidades nombradas encontradas en un texto

2.3.6. Análisis de sentimientos

El análisis de sentimiento es una de las funcionalidades más usadas de NLP.. Mediante el análisis de sentimiento se intenta analizar el cuerpo de un texto para analizar un texto y la opinión expresada en él. Para ello se cuantifica dicho sentimiento por la polaridad, que podrá ser positiva, negativa o neutra. La polaridad de todo el documento. El análisis de sentimientos funciona mejor en textos con un contexto subjetivo que en aquellos que tienen un carácter objetivo.

2.3.7. Enfoques usados en ésta propuesta:

En el desarrollo de nuestra aplicación se ha centrado en la capa del nivel sintáctico, puesto que se analiza cada palabra por separado sin tener en cuenta la semántica de la frase. Una posible mejora futura consistiría en analizar los textos, además de un punto sintáctico, analizarlo semánticamente.

2.4. Técnicas de NLP utilizadas: Stemming y Lematización

Para llevar a cabo un análisis de la semejanza de documentos primero hay que realizar una extracción de features (características) destinadas a ser informativos y no redundantes. La extracción de características (features) se trata de un proceso de reducción y codificación, donde un conjunto inicial de variables sin procesar, es decir, el texto de un paper se reduce a características más manejables para su procesamiento (vectores) y que se describa con precisión el conjunto de datos original.

Para el análisis de los documentos se podrían haber optado por diferentes estrategias:

- Recoger y analizar todo el texto de los papers de cada uno de los autores: esto implicaría una mayor complejidad de la aplicación, además de un crecimiento muy notable del tiempo de ejecución. En consecuencia, la usabilidad de la aplicación se vería muy afectada.

- Recoger y analizar el *abstract* de cada uno de los papers. Al ser menor la sección de texto a analizar, la eficiencia de la aplicación es mayor.

Después de analizar las opciones, se optó por escoger el *abstract*, recogiendo los datos de la Api de Semantic Scholar. Centrándonos en la definición de *abstract*, lo podríamos describir como la sección en la cual se hace un resumen documental que representa de manera objetiva y precisa el contenido de un documento académico o científico del paper, constituyendo el contenido esencial del reporte de investigación.

En el proceso de Recuperación de Información los rasgos de los objetos a recuperar juegan un papel fundamental. La mayoría de las técnicas de recuperación de información utilizan el recuento de las frecuencias de los términos que aparecen en los documentos y las consultas. Esto implica la necesidad de normalizar dichos términos para que los recuentos puedan efectuarse de manera adecuada, tomando en consideración aquellos términos que derivan de un mismo lema o raíz. En lo que respecta a las técnicas de RI usadas en éste proyecto, podríamos citar la *tokenización*, *bag of words* y el *stemming*.

2.4.1. Stemming

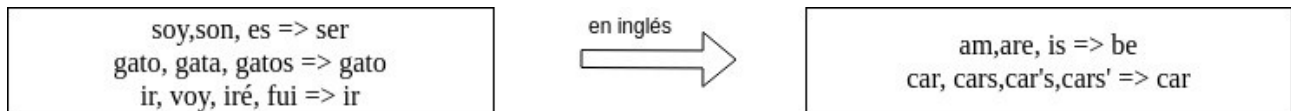
Se refiere a un crudo proceso heurístico que corta los extremos de las palabras con la esperanza de lograr este objetivo correctamente la mayor parte del tiempo, y a menudo incluye la eliminación de los afijos derivacionales. El algoritmo más común para stemming es el algoritmo de Porter.

Ejemplo en castellano: "gato", "gata", "gatos" => "gat"
 Ejemplo en ingles: "automates", "automatic" => "automat"

2.4.2. Lematización

La lematización es un proceso lingüístico que consiste en, dada una forma flexionada (es decir, en plural, en femenino, conjugada, etc), hallar el lema correspondiente.

El lema es la forma que por convenio se acepta como representante de todas las formas flexionadas de una misma palabra. La lematización generalmente se refiere a hacer las cosas correctamente con el uso de un vocabulario y un análisis morfológico de las palabras, normalmente con el objetivo de eliminar las terminaciones flexionales solamente y devolver la forma base o diccionario de una palabra, lo que se conoce como el lema



2.4.3. Tokenización

Consiste en separar palabras del texto en entidades llamadas *tokens*, con las que trabajaremos luego. Debemos pensar si utilizaremos los signos de puntuación como token, si daremos importancia o no a las mayúsculas y si unificamos palabras similares en un mismo token. En el caso de nuestro proyecto, no tendrán importancia para los análisis que estamos realizando.

2.4.4. Bag of words (bow)

Bow [12] es una manera de representar el vocabulario que utilizaremos en nuestro modelo y consiste en crear una matriz en la que cada columna es un token y se contabilizará la cantidad de veces que aparece ese token en cada oración (representadas en cada fila).

2.4.5. Term frequency – Inverse document frequency (TF-IDF).

TF-IDF [13] es una técnica muy utilizada en Machine Learning [14] para otorgar la relevancia de una palabra, en un documento de una colección, a través de una medida numérica. Esta medida numérica se utiliza para calificar la relevancia de una palabra dentro de un documento, a partir de la frecuencia que aparece en el mismo. La idea en la que se basa esta técnica es que si una palabra aparece frecuentemente en el documento, debe ser importante y se le debe dar una puntuación alta. Sin embargo, si una palabra aparece frecuentemente en otros documentos, probablemente no sea un identificador único, y por tanto, se le debe asignar una puntuación más baja.

A continuación, descompondremos la explicación en sus dos componentes: TF e IDF.

2.4.6. Term frequency(TF).

Las siglas TF provienen de la expresión en inglés *Term Frequency*, (Frecuencia de término) en español, y determina la frecuencia relativa de un término específico, una palabra o una combinación de palabras, en un documento. Este valor se compara con la frecuencia relativa de todos los demás

términos de un texto, documento o sitio web. La fórmula se compone por un logaritmo y se escribe del siguiente modo:

$$TF(i) = \frac{\log_2(Freq(i,j) + 1)}{\log_2(L)}$$

El logaritmo evita que un aumento sustancial del uso de la palabra clave específica no afecte al valor final del cálculo. Mientras la densidad de una palabra clave calcula solo el porcentaje de distribución de una palabra comparado con el número total de palabras en un texto, la frecuencia de término, TF, contempla también la proporción de todas las palabras usadas en el texto.

2.4.7. IDF

El término IDF proviene del inglés *Inverse Document Frequency* y significa Frecuencia inversa de documento. Esta segunda parte de la fórmula completa el análisis de evaluación de los términos y actúa como el *corrector del TF*. La Frecuencia Inversa de Documento es muy importante ya que incluye en el cálculo la frecuencia de documento de términos específicos: compara el número de todos los documentos disponibles con el número de documentos que contienen el término. Y para finalizar, el logaritmo se encarga de comprimir los resultados:

$$IDF_t = \log \left(1 + \frac{N_D}{f_t} \right)$$

En resumen: el IDF determina la relevancia de un texto con respecto a una palabra clave específica. Las fórmulas anteriores calculan la relevancia de un documento en comparación con los demás documentos que contienen esa misma palabra clave. Para obtener resultados útiles, la fórmula debe calcularse para todas las palabras relevantes de un texto. Cuanto mayor sea la base de datos usada para el cálculo del valor TF*IDF, más preciso será el resultado.

Respecto a la utilización de TF-IDF, podríamos citar por ejemplo el posicionamiento SEO: TF-IDF nos da una medida de originalidad de contenido y ayuda a los buscadores a determinar que hace que una página sea especial, que sea única. La densidad de palabras clave es una métrica muy simple y fácilmente manipulable.

Matemáticamente, el TF-IDF es el producto de la frecuencia de la palabra clave en la página (TF) y la frecuencia estimada media de la palabra clave en un grupo grande de documentos (IDF)

Dado que TfxIDF compara una palabra clave individual respecto a un grupo de documentos es una estimación bastante fiel de la importancia de una palabra en una página, rebaja el peso de las palabras y frases no importantes e incrementa el valor de las palabras únicas, raras y significativas.

2.5. Análisis de semejanza entre documentos

Las medidas de análisis de semejanza de textos juegan un papel determinante en investigación y tareas como recuperación de información (RI), clustering de documentos [15], detección de temas de documentos, traducción de documentos...

El cálculo de la semejanza entre palabras es una parte fundamental del cálculo de la semejanza de documentos, que es usado como una fase previa para el análisis de la semejanza entre frases, párrafos y, finalmente, documentos. Podríamos decir que las palabras pueden ser similares en dos contextos: léxicamente y semánticamente. Dos palabras son similares léxicamente si comparten secuencias de caracteres. Por otro lado, se dicen que dos palabras son similares semánticamente si tienen un significado similar. A continuación explicaremos el proceso de cálculo de la semejanza entre documentos:

La tarea más habitual antes de trabajar sobre cualquier documento es extraer el texto y realizar una limpieza del mismo. Normalmente se eliminan caracteres no útiles (signos de puntuación), se realiza un etiquetado gramatical y en la cual usaremos las técnicas de RI mencionadas en la sección anterior.

Tras la limpieza, tendremos un conjunto de documentos que será la entrada de nuestro modelo bow. Para cada documento, se crea un diccionario con las palabras presentes y su frecuencia. Es decir, tendríamos un diccionario con una estructura similar:

```
{natural:45, language:21, processing: 37...}
```

Después, creamos un modelo tf-idf con el modelo bow. El modelo tf-idf contiene por cada documento, un diccionario de palabra y valor tf-idf. Para obtener las palabras más relevantes de cada documento, a las que llamamos keywords o palabras clave, nos quedaremos con las que tengan mayor valor tf-idf. Por ejemplo, las keywords son (ordenadas por relevancia):

```
natural, reinforcement, technique, financial, language, method,  
statistical, processing, machine, semantic, vessel, bovary,  
important, information, part,
```

El siguiente paso, sería calcular la distancia coseno de los vectores que representan los documentos. El proceso es reflejado en la siguiente imagen:

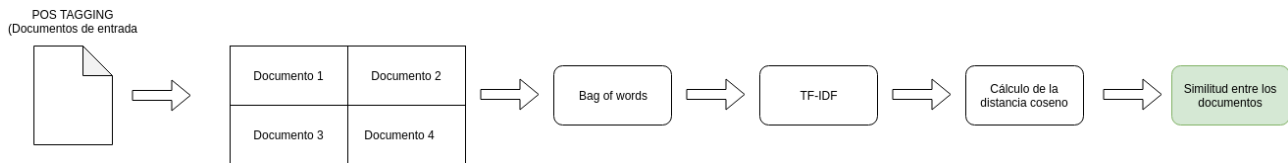


Ilustración 2: Etapas en el cálculo de la semejanza entre elementos

2.5.1. Medidas de similitud

Las medidas de similitud que hemos usado en la implementación de nuestro prototipo son las siguientes:

2.5.1.1. Similitud coseno

Nuestro autor o paper vendrá representado de manera algebraica por un vector. La semejanza entre varios autores o papers se calculará por el valor del coseno del ángulo comprendido entre dichos vectores. Esta función trigonométrica proporciona un valor igual a 1 si el ángulo comprendido es cero, es decir si ambos vectores apuntan a un mismo lugar. De esta forma, el valor de esta métrica se encuentra entre -1 y 1, es decir en el intervalo cerrado $[-1,1]$.

2.5.1.2. Similitud Jaccard

La similitud Jaccard [16] es una medida de similitud que se basa en el número de palabras comunes en todo los textos. Cuantos más terminos sean comunes, más similitud habrá entre ambos textos. Toma el valor entre 0 y 1. Si el resultado es 1, ambos textos son idénticos. Si no hay ninguna palabra en común, es resultado es 0. Nosotros lo usaremos en nuestro proyecto para encontrar el autor con mayor semejanza con el autor introducido por el usuario (puede que el usuario no introduzca exactamente el nombre de un autor por faltas ortográficas, por ejemplo). En nuestro caso, se compararán caracter a caracter ambos nombres.

En nuestro prototipo, hemos usado el coeficiente de Jaccard para encontrar el autor que guarde mayor semejanza, en la base de datos de Semantic Scholar, con el autor introducido por el usuario. Es decir, cuando el usuario introduce el nombre de un autor, puede incurrir en errores ortográficos o que el autor firme con un nombre diferente sus papers académicos. Por ello, cada vez que el usuario introduce un autor en nuestra aplicación, se lanzará una petición a la API de DBLP, que devolverá

una lista con todos los autores existentes en su base de datos, con un nombre similar al buscado. A partir de dicha lista, se devolverá el nombre que tiene mayor coeficiente de Jaccard con el autor introducido por el usuario.

A continuación mostraremos un ejemplo de uso de la similitud de Jaccard.

Jaccard Similarity

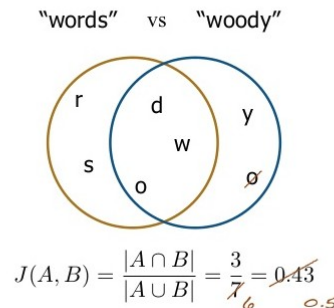


Ilustración 3: Similitud de Jaccard

Como se puede ver en la imagen superior, "words" y "woody" están formados cada una por un conjunto de letras, de las cuales únicas hay seis ("w", "o", "d", "r" y "s"). Por otro lado, ambas palabras comparten tres letras ("d", "w" y "o"). Finalmente, aplicando la similitud de Jaccard al ejemplo, podemos ver que la similitud es de "0.5"

2.6. Algoritmos de aprendizaje

A la hora de hablar de un sistema basado en Machine Learning [17], nos referimos a aquel sistema capaz de aprender de manera autónoma. Es decir, sistemas que utilizan algoritmos capaces de extraer modelos o utilizar unos datos de entrenamiento para posteriormente predecir unos resultados. Entre los diferentes tipos de algoritmos podemos encontrar:

2.6.1 Algoritmos no supervisados

Este tipo de algoritmos son utilizados cuando disponemos básicamente de un conjunto de datos de entrada, pero desconocemos el resultado de la salida. El objetivo de estos consiste en encontrar la estructura y distribución de estos datos para encontrar un modelo.

Las aplicaciones más comunes de estos algoritmos se pueden observar a la hora de realizar agrupaciones y en la detección de anomalías. Los datos similares pueden agruparse y formar

clusters que contendrán las características comunes de los datos. Un algoritmo de éste tipo podría aplicarse, por ejemplo, sobre un conjunto de datos para hacer clustering de documentos.

Podríamos definir como clustering el agrupamiento de documentos similares, en función de la importancia y relevancia (clustering jerárquico)

2.6.2. Algoritmos supervisados

El algoritmo de aprendizaje supervisado [18] se utiliza cuando conocemos el conjunto de datos de entrada y además conocemos el resultado para estos datos. Podemos utilizar éste tipo de algoritmos para predecir los resultados cuando introducimos nuevos datos.

El primer paso consiste en entrenar el algoritmo con los datos conocidos, se construirá el modelo . Después se puede aplicar este modelo a una serie nueva de datos y predecir el resultado.

Un claro ejemplo es al clasificar correo entrante entre Spam o no. Entre las diversas características que queremos entrenar deberemos incluir si es correo basura o no con un 1 o un 0. Otro ejemplo son al predecir valores numéricos por ejemplo precio de vivienda a partir de sus características (metros cuadrados, nº de habitaciones, incluye calefacción, distancia del centro, etc.) y deberemos incluir el precio que averiguamos en nuestro set de datos.

2.7. Tecnología usada

2.7.1. NLTK

Es la librería líder para el procesamiento del lenguaje natural. Proporciona interfaces fáciles de usar a más de 50 corpus y recursos léxicos, junto con un conjunto de bibliotecas de procesamiento de texto para la clasificación, tokenización, el etiquetado, el análisis y el razonamiento semántico.

2.7.1. Python

Python es un lenguaje de programación interpretado cuya filosofía hace incapié en una sintaxis que favorezca un código legible. Es un lenguaje de programación muy usado por la comunidad de científico de datos e ingenieros de datos.

Se trata de un lenguaje de programación multiparadigma, que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, de tipado fuerte y multiplataforma.

2.7.2. D3.js

D3.js (o simplemente D3 por las siglas de Data-Driven Documents) es una biblioteca (informática) de JavaScript para producir, a partir de datos, infogramas dinámicos e interactivos en navegadores web.

2.7.3. MongoDB

MongoDB es una base de datos NoSQL, donde a diferencia de SQL que maneja una estructura llave-valor, hace hincapié en el valor de las llaves donde las llaves son llamadas colecciones y tienen una estructura tipo JSON llamados documentos. Para quienes dominen javascript, lenguaje sobre el cual se basa esta solución, le será más fácil su manipulación.

2.8. Base de datos

Este tipo de bases de datos presentan una serie de problemas cuando se utilizan como sistema de almacenamiento para la cantidad de datos que se manejan en los sistemas Big Data.

En las bases de datos documentales el concepto principal es el de “documento”. Un documento es la unidad principal de almacenamiento de este tipo de base de datos, y toda la información que aquí se almacena, se hace en formato de documento.

En nuestro caso, la información requerida para almacenar durante el desarrollo de nuestro proyecto fue modificándose a lo largo del tiempo, por lo que necesitábamos una base de datos versátil, que permitiera una modificación dinámica y fácil de la misma. Es por ello que hemos optado por usar una base de datos NoSQL como MongoDB.

En cuanto al diseño de la base de datos, se ha decidido crear dos colecciones: una para autores y otra para publicaciones.

▼ (1) 3018657	{ 7 fields }
_id	3018657
name	Alberto Fernández-Isabel
▶ publications	[20 elements]
▶ titles	[20 elements]
▶ topics	[152 elements]
▶ topicsId	[152 elements]
reputation	15.7

Ilustración 4: Colección de los autores

Por un lado, la colección de autores tiene el siguiente diseño:

A continuación describimos cada uno de los campos anteriores:

- El campo “_id” guarda un identificador único del autor.
- “name”: nombre del autor en la base de datos de Semantic Scholar
- “publications”: una lista de los ids de las publicaciones en las que ha participado el autor.
- “titles”: lista de los nombres de las publicaciones
- “topics”: los topics forman parte del *abstract* de cada publicación, y corresponden con conceptos que resumen el ámbito del documento. Éste campo contendrá el conjunto de topics de todos sus documentos.
- “topicsId”: identificador numérico correspondiente a cada elemento del campo “topics”
- “reputation”: valor numérico correspondiente a la influencia del autor respecto a los demás autores de Semantic Scholar (explicaremos el cálculo de la reputación en el capítulo 4)

Por otro lado, la colección de publicaciones tiene el siguiente diseño:

(1) 6c4443c2f02dd2bb63a6aac14d0547ae6678d6a9	{ 9 fields }
_id	6c4443c2f02dd2bb63a6aac14d0547ae6678d6a9
title	Simulation of Road Traffic Applying Model-Driven Engineering
year	2015
citations	0
influentialCitationCount	0
▶ author_ids	[2 elements]
▶ topics	[3 elements]
▶ topicsId	[3 elements]
reputation	52.175

Ilustración 5: Colección de las publicaciones

A continuación se describen cada uno de los campos de la colección:

- “_id”: identifiacador de la publicación
- “title”: título de la publicación
- “year”: año de publicación
- “citations”: número de menciones en otros artículos
- “influencialCitationCount”: de las menciones, número de las que fueron influyentes
- “author_ids”: lista de los indentificadores de los autores que han escuroto el artículo.
- “topics”: lista de conceptos, que forman parte del abstract, y que resumen de la publicación.
- “topicsId”: lista de los identificadores que forman parte del campo “topics”.
- “reputation”: valor numérico correspondiente a la influencia de la publicación respecto a las demás publicaciones de Semantic Scholar.

2.9. Métodos de reducción de dimensionalidad

La reducción de dimensión [19] es una de las etapas más importantes en problemas de reconocimiento de patrones [20], pues permite revelar la estructura intrínseca de los datos y extraer la información más relevante del problema en estudio, mejorando el desempeño tanto en tareas de visualización como de clasificación.

Los ordenadores si que pueden procesar grandes cantidades de datos multidimensionales. Pero los humanos a veces necesitamos “ver” y entender los datos. Cuando estamos trabajando en un espacio multidimensional no podemos imaginarnos nuestro dataset. Es por ello que usamos algún método de reducción de dimensionalidad (a 2 o 3 dimensiones) para que podamos visualizar los datos.

2.9.1. MDS

Es una representación visual de las distancias o diferencias entre conjuntos de objetos. Los "objetos" pueden ser colores, caras, coordenadas de mapas, persuasión política o cualquier tipo de estímulo real o conceptual. Los objetos que son más similares (o tienen distancias más cortas) están más cerca al representarlos visualmente, que los objetos que son menos similares (o tienen distancias más largas). Además de interpretar las diferencias como distancias en un gráfico, MDS [21] también se usa como técnica de reducción de dimensionalidad para conjuntos de datos de alta dimensión, que será para lo que lo usaremos nosotros.

El proceso de reducción de dimensionalidad mediante MDS consta de los siguientes pasos:

- 1) Se sitúan los n puntos de una configuración inicial en p dimensiones, esto es, suponer para cada objeto las coordenadas (x_1, x_2, \dots, x_p) en el espacio de p dimensiones.
- 2) Se calculan las distancias euclidianas entre los objetos de esa configuración, esto es, calcular las d_{ij} , que son las distancias entre el objeto i y el objeto j .
- 3) Se hace una regresión lineal utilizando el método de los mínimos cuadrados
- 4) Se mide la bondad de ajuste entre las distancias de la configuración y las disparidades.

Se repetirán los pasos del 2 al 4 hasta que al parecer la medida de ajuste entre las disparidades y las distancias de configuración no puedan seguir reduciéndose. El resultado final del análisis es entonces las coordenadas de los n objetos en las p dimensiones.

2.9.2. t-SNE

t-SNE (t-Distribución Estocástica de puntos más cercanos o, en inglés, t-Distributed Stochastic Neighbor Embedding) [22]: minimiza la divergencia entre dos distribuciones. Una distribución que mide la similitud entre pares de objetos de entradas y una medida de distribución entre parejas similares de los correspondientes puntos de análisis, en un espacio de representación implementado de menor dimensión (en nuestro caso en 2 o 3 dimensiones.)

Suponiendo que disponemos de un conjunto de datos correspondiente a las características de nuestros abstract de alrededor de 50 dimensiones, es imposible visualizar esa matriz por el ojo humano. Por ello, el objetivo es convertir ese conjunto de datos 50 dimensiones en algo que sea representable y que se pueda ver (2 dimensiones en nuestro caso). Los pasos que se siguen son los siguientes:

1. Partiendo de la matriz de características original. se mide la similitud de cada punto respecto a los demás, formando una matriz S_1 para cada punto. Los puntos de datos similares tendrán mayor valor de similitud y los diferentes puntos de datos tendrán menor valor.
2. Posteriormente se convierte esa distancia de similitud a probabilidad (probabilidad conjunta) de acuerdo a la distribución normal.
3. Mediante t-SNE, ordenamos todos los puntos de datos aleatoriamente en la dimensión inferior requerida (supongamos que 2) y se calculan las distancias de similitud en esa dimensión más baja y se le asigna una distribución de probabilidad t en lugar de una distribución normal. En éste punto obtendríamos otra matriz de similitud llamado S_2 .
4. Posteriormente, t-SNE compara las matrices S_1 y S_2 y calcula las diferencias mínimas entre los puntos de la matriz S_1 y S_2 .

Capítulo 3

Cuando se habla de similitud de texto, se suele tener una noción ligeramente diferente de lo que significa la similitud de texto. En esencia, el objetivo del cálculo de la semejanza entre textos es calcular cuan cerca están las dos partes, tanto semántica, como léxicamente. Aunque los métodos para la similitud léxica se utilizan a menudo para lograr la similitud semántica (hasta cierto punto), lograr la verdadera similitud semántica es a menudo mucho más complicado.

Sin embargo, se puede diseñar sistemas basados en NLP, como éste prototipo, que hace más práctico y eficiente el reconocimiento y comprensión del lenguaje humano y las relaciones de los elementos del lenguaje.

Alcance funcional de la propuesta

En esta sección se describe la funcionalidad que es capaz de realizar la herramienta desarrollada y en que etapas está organizada esta funcionalidad. Se introducen estas últimas y se representa un diagrama asociado a las mismas.

El objetivo de éste proyecto de final de máster es el desarrollo de un prototipo para mostrar visualmente el grado de semejanza entre publicaciones o autores .

Para ello, se obtienen datos de diferentes fuentes (API de DBLP y Semantic Scholar) y se cruzan para recoger los datos en la base de datos. El prototipo permite los siguientes tipos de similitudes:

- Análisis de similitud entre autores y coautores
- Análisis de similitud entre las publicaciones de un mismo autores
- Análisis de similitud entre las publicaciones de autores y coautores de una obra.

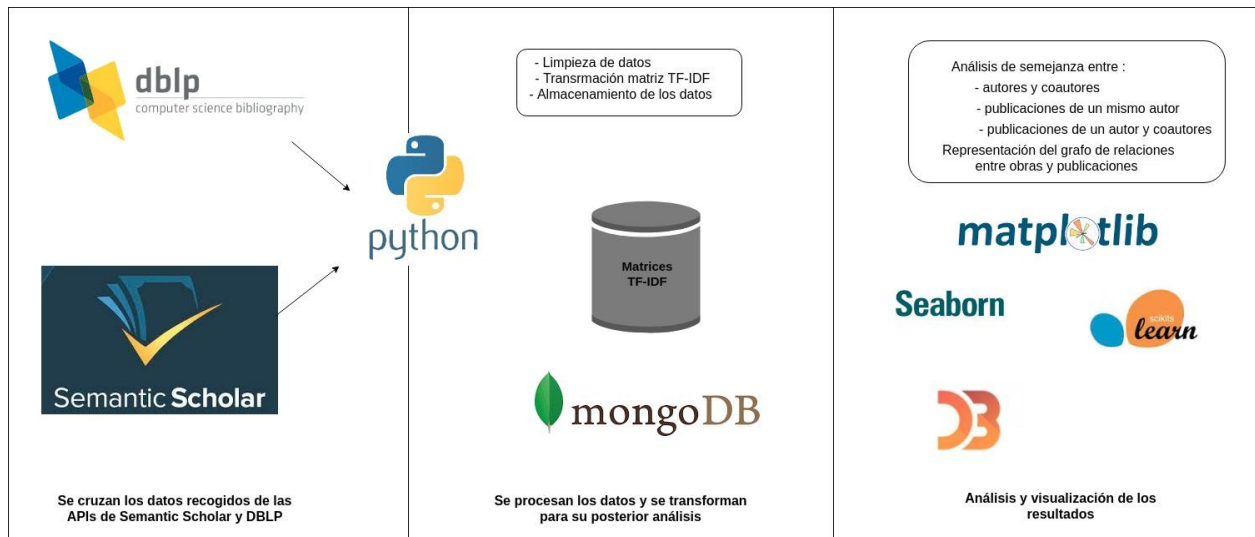


Ilustración 6: Esquema funcional de la propuesta

3.1.Desarrollo de la interfaz de usuario

En ésta parte del desarrollo, se diseñó una interfaz de usuario intuitiva y simple, con el objetivo de minimizar los posibles errores que se pudieran encontrar y simplificar su uso. Para ello se ha usado la librería de python “tkinter”, la cual se integra perfectamente con los demás integrantes del desarrollo del prototipo.



Ilustración 7: Interfaz de usuario

Como se puede ver en la imagen superior, la aplicación tiene un componente *radioButton* para discriminar la funcionalidad a ejecutar.

3.1.1. Grafo asociado

La representación del grafo asociado tiene como objetivo representar las relaciones entre autores, coautores y las obras en las que participan. El grosor de los nodos depende de la reputación que tiene dicho autor respecto a los demás miembros de la red de Semantic Scholar. Los grafos son interactivos, porque los nodos que lo componen se pueden estirar, mover o contraer. Además, todos los nodos incorporarán mensajes de información cuando se sitúa el cursor encima de sus nodos.

Por un lado, los nodos de los autores serán de color azul y estarán enlazados con las publicaciones en las que han participado. Entre las funcionalidades que incorporan, podemos citar los mensajes de información que se muestran cuando se deja el cursor encima de uno de dichos nodos, tal y como se puede ver en la imagen inferior:

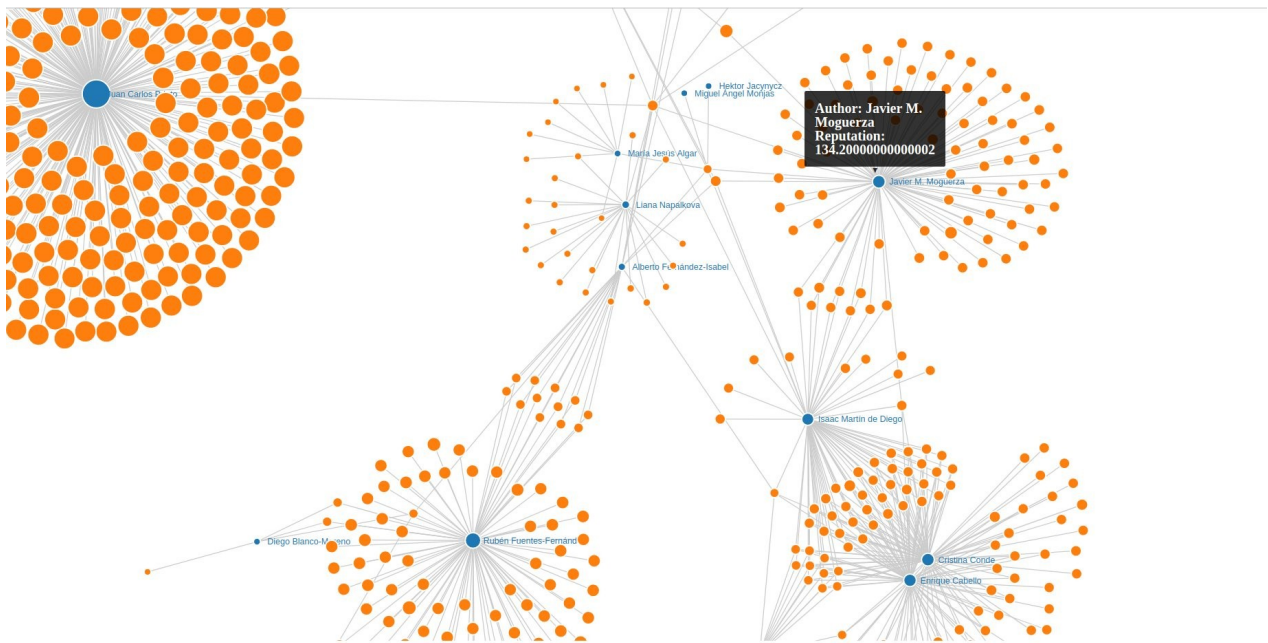


Ilustración 8: Mensaje informativo respecto sobre un autor

Como se puede ver en la imagen superior, correspondiente al profesor universitario Javier M. Moguerza, en el mensaje informativo se mostrará el nombre del autor, así como la reputación asociada.

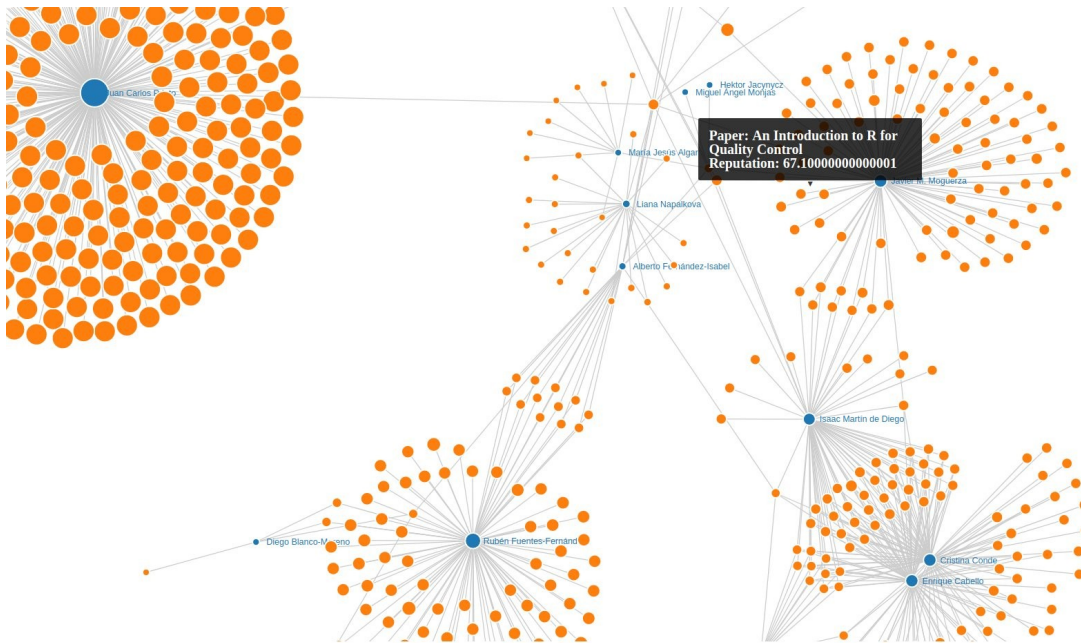


Ilustración 9: Mensaje informativo sobre un artículo

Por otro lado, los nodos de las publicaciones son naranjas y estarán unidos por aristas con los autores y coautores que los han escrito. Tal y como ocurría con los nodos de los autores, los nodos de las publicaciones incorporarán también la mensajes informativos que se mostrarán al mantener el cursor en uno de los nodos, tal y como se muestra en la imagen inferior.

Además, al hacer click en cualquiera de los nodos (autor o publicación) se abrirá en un navegador web el enlace correspondiente en la web de Semantic Scholar.

Como se puede ver, la representación mediante grafos permite distinguir rápidamente nodos relevantes y acceder a su perfil de Semantic Scholar.

En cuanto al desarrollo de la visualización, mediante consultas a la base de datos obtenemos la información necesaria para crear un fichero html, que con la ayuda de la librería de D3.js, nos ayudará a representar el grafo.

3.1.2.Comparador del autor con otros autores

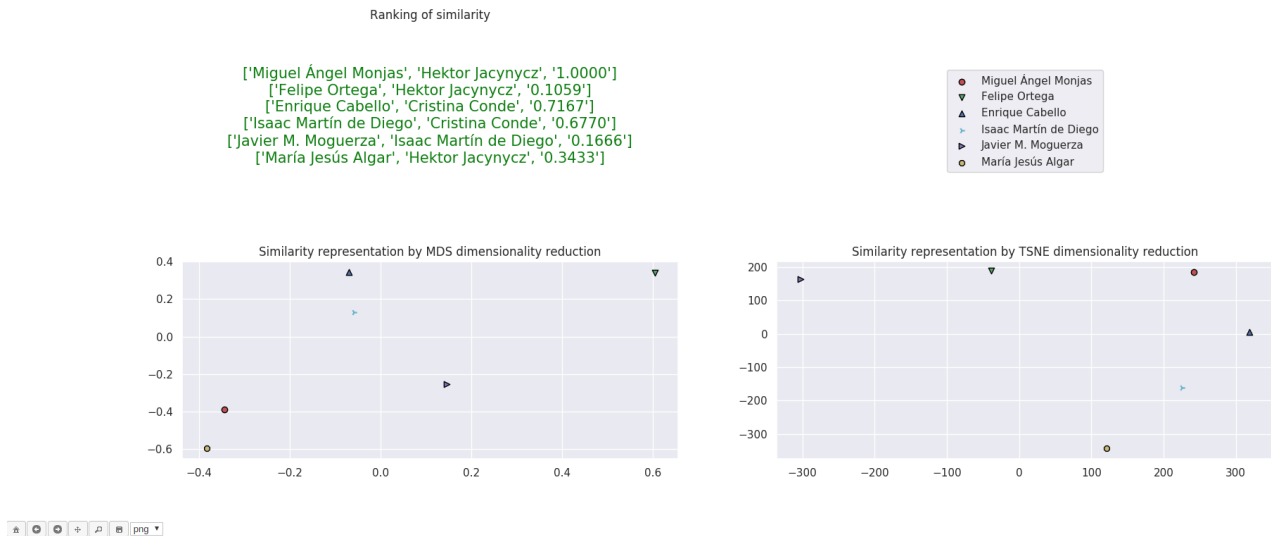


Ilustración 10: Comparador del autor con otros autores

Éste gráfico tiene como objetivo mostrar la semejanza entre autores relacionados con el autor. Para ello, hemos dividido la pantalla de visualización en varias partes:

En la esquina superior izquierda, se mostrará un ranking de semejanzas entre autores, entendiéndose la semejanza entre autores relacionados con el autor buscado. Concretamente, podríamos esquematizar de un autor con otro de la siguiente manera:

[Autor1 (semejanza buscada) . Autor2 (autor con mayor semejanza):grado de semejanza]

Un punto a notar es que no se da la propiedad “conmutativa” en el análisis de la semejanza entre autores. Por ejemplo, si el autor que tiene mayor semejanza con el *autor1* es el *autor2*, no tiene por qué ocurrir que el autor que tiene mayor semejanza con el *autor2* sea el *autor1*.

En el caso de la visualización mostrada más arriba, podemos ver que la autora con el que Isaac Martín de Diego guarda mayor semejanza es Cristina Conde. Sin embargo, Cristina Conde no es la autora que guarda mayor semejanza respecto a Isaac Martín de Diego.

Por otro lado, es la esquina inferior izquierda y derecha podemos ver dos representaciones mediante métodos de reducción de dimensionalidad. A la izquierda, se puede ver la representación mediante reducción por MDS. Además, a la derecha, podemos ver la representación mediante t-SNE. La leyenda, para ambos esquemas, se encuentra en la esquina superior derecha de la pantalla.

3.1.3.Comparador de obras del mismo autor

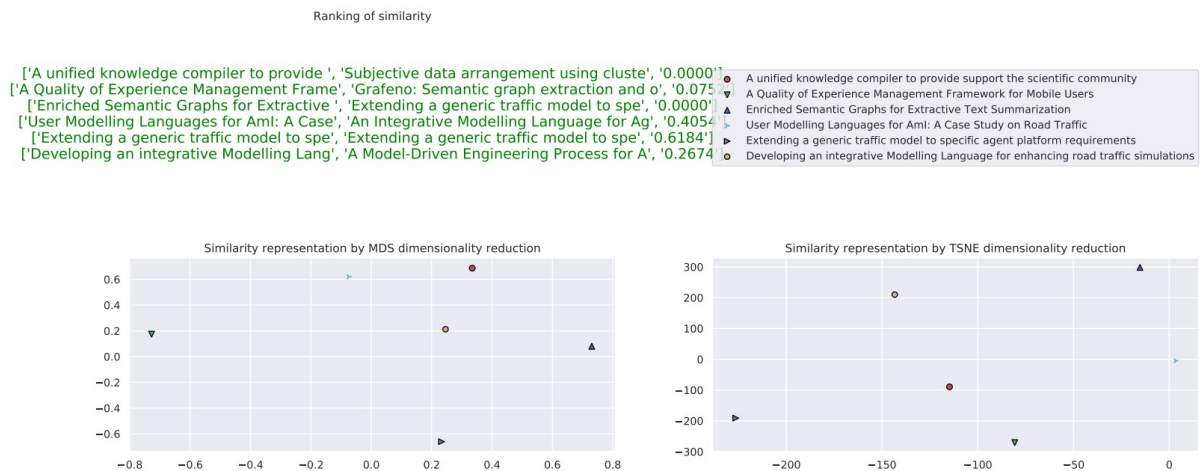


Ilustración 11: Comparador de obras del mismo autor

Esta representación tiene como objetivo mostrar la similitud entre obras de un mismo autor. En el caso del ejemplo de la parte superior, las semejanzas relativas a obras de Alberto Fernández-Isabel.

La representación sigue la misma estructura que las demás representaciones. Es decir, en la izquierda estarán los elementos de los cuales se quiere sacar su similitud y a la derecha los elementos con los que guarda mayor semejanza. Es decir:

[Obra1 (semejanza buscada) . Obra2 (obra con mayor semejanza):grado de semejanza]

En éste caso podemos ver que, por ejemplo, la obra que mantiene mayor similitud con el artículo de “Developing an integrative Modelling Language for enhancing road traffic simulations” [23] es “A Model-Driven Engineering Process for Agent-based Traffic Simulations”[23]. Por motivos de visibilidad, se han limitado el número de caracteres a mostrar.

3.1.4.Comparador de obras entre todos los autores

En ésta visualización mostraremos la semejanza entre los artículos de todos los autores relacionados con el autor buscado. Es decir, buscará las relaciones de similitud existente entre obras del autor y obras de los coautores de sus libros (puesto que el autor buscado posee obras escritas por más de una persona).

Capítulo 4

El cálculo de la reputación en nuestro prototipo se basa en la relevancia de los autores o textos, cálculo basado en el número de citas de una publicación.

Para todas las publicaciones, Semantic Scholar identifica las citas en las que la publicación citada tiene un impacto significativo, lo que facilita la comprensión de cómo las publicaciones se basan y relacionan entre sí. Las citas influyentes se determinan utilizando un modelo de aprendizaje por máquina que analiza una serie de factores, incluyendo el número de citas de una publicación y el contexto relacionado con cada una.

Cálculo de reputación

En nuestro prototipo hemos cuantificado la reputación de los autores y publicaciones, con el objetivo de medir su influencia respecto a los demás miembros de la comunidad de Semantic Scholar. Por otro lado, dicho cálculo nos será de utilidad a la hora de representar el grafo de las relaciones entre autores, coautores y sus publicaciones, de tal manera que el tamaño de los nodos varíe en función de su reputación. A continuación mostramos las formulas por las que calculamos las reputaciones

Reputación de un autor:

$$Reputation_{publication} = Coef . Reputation_{article} * N^o \text{ veces citado} + Coef . Reputation_{author} * \sum_1^n \frac{RepAuthor_n}{n}$$

siendo n el total de autores de esa publicación

$$Reputation_{author} = Coef . Reputation_{article} * N^o \text{ artículos} + C . Citations_{author} * N^o \text{ Citas} * N^o Citations_{influential} * N^o \text{ Citas}_{influential} + Coef . Seniority * Seniority$$

En cuanto a los coeficientes, son siempre valores desde 0 a 1. El objetivo de los coeficientes es dar peso a cada uno de los sumandos de la ecuación, de tal forma que podemos establecer la importancia que tienen respecto al valor total de la reputación.

Respecto al valor del *seniority*, es el valor numérico correspondiente a la diferencia de años en la que se hizo la primera publicación. Es decir, es la antigüedad que tiene ese autor como publicador.

Captítulo 5

El desarrollo iterativo y creciente (o incremental) consiste en un conjunto de tareas agrupadas en pequeñas etapas repetitivas (iteraciones). Es uno de los más utilizados en los últimos tiempos ya que, como se relaciona con novedosas estrategias de desarrollo de software y una programación extrema, es empleado en metodologías diversas. Una de las ventajas del desarrollo es el alineamiento con los objetivos del cliente, puesto que se se priorizan los objetivos y requerimientos del desarrollo, en función del valor que ofrece el cliente.

Para apoyar el desarrollo de proyectos por medio de este modelo se han creado frameworks (entornos de trabajo), de los cuales los dos más famosos son el Rational Unified Process y el Dynamic Systems Development Method. El desarrollo incremental e iterativo es también una parte esencial de un tipo de programación conocido como Extreme Programming y los demás frameworks de desarrollo rápido de software.

Modelo de desarrollo

El modelo de desarrollo de la herramienta es iterativo incremental [24]. La idea principal detrás de mejoramiento iterativo es desarrollar un sistema de programas de manera incremental, permitiéndole al desarrollador sacar ventaja de lo que se ha aprendido a lo largo del desarrollo anterior, incrementando, versiones entregables del sistema.

Los pasos claves en el proceso son comenzar con una implementación simple de los requerimientos del sistema, e iterativamente mejorar la secuencia evolutiva de versiones hasta que el sistema completo este implementado. En cada iteración, se realizan cambios en el diseño y se agregan nuevas funcionalidades y capacidades al sistema.

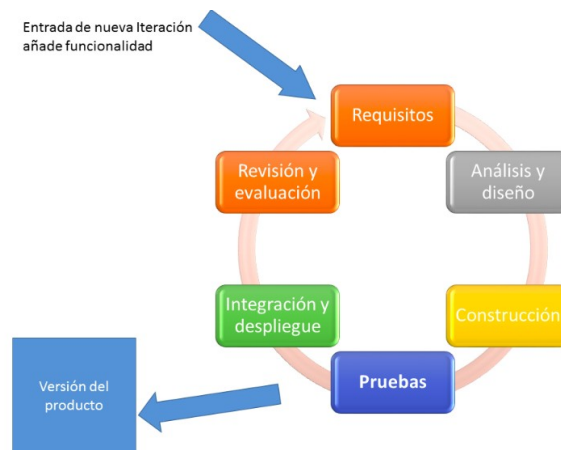


Ilustración 13: Desarrollo iterativo

Etapas de desarrollo

En esta sección se describen las etapas de desarrollo en orden cronológico, siendo la primera etapa el apartado 6.1 en la que se aborda la recogida, tratamiento y almacenaje de los datos. Después, en el apartado 6.2 se muestra la segunda etapa, en la que se muestra el desarrollo del prototipo, que consta de la selección del algoritmo para calcular las semejanzas, así como el análisis e implementación de los métodos de reducción de dimensionalidad. A continuación, en el apartado 6.3, se centra en la implementación de las diferentes visualizaciones y el desarrollo de la aplicación de escritorio, como interfaz de usuario para el prototipo.

Nombre de la tarea	2019											
	sep	oct	nov	dic	ene	feb	mar	abr	may	jun	jul	ago
Estudio del problema y recogida de datos	18/01/19											
Estudio alternativas recogida Datos		05/10/18										
Recogida de datos			26/11/18									
Limpieza de base datos				18/01/19								
Implementación de la solución	01/04/19											
Uso del algoritmo para calculo de las semejanzas					04/03/19							
Reducción de dimensionalidad (t-SNE y MDS)						01/04/19						
Visualización de la implementación	30/05/19											
Visualización métodos de reducción							06/05/19					
Implementación visualización del grafo relacional								20/05/19				
Implementación aplicación escritorio									30/05/19			

Ilustración 14: Etapas del desarrollo del prototipo

5.1. Estudio alternativas de recogida de datos

La primera fase del desarrollo consiste en estudiar las diferentes alternativas existentes de cara a la recogida de datos. En un primer momento, se pensó en *scrapear* [25] u obtener la información directamente del código html de la web de Semantic Scholar. No obstante, ésta opción complicaba el desarrollo puesto que había que utilizar herramientas que ralentizarían el desarrollo del proyecto porque el proceso de obtención de la información iba a ser muy lento. Finalmente se decidió cruzar información de varias búsquedas para poder obtener la información necesaria.

Recogida y limpieza de datos:

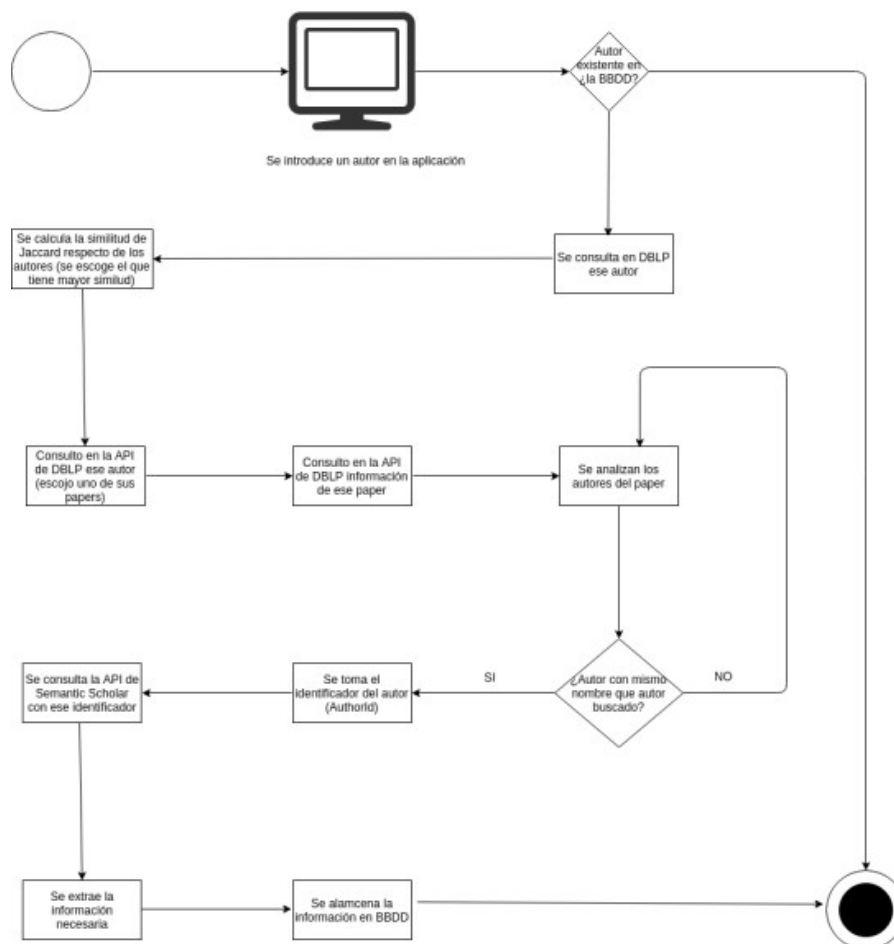


Ilustración 15: Proceso de almacenaje en BBDD

A continuación mostramos un diagrama de los pasos que se siguieron para almacenar la información necesaria en la BBDD:

La API de Semantic Scholar, que proporcionaba gran parte de la información necesaria para obtener para nuestro desarrollo. Sin embargo, las búsquedas de un autor o publicación en la API de Semantic Scholar no se puede hacer de manera textual (es decir, a partir del nombre del autor o publicación) sino a partir de un clave unívoca, que indentifica cada elemento (autor o publicación).

Concretamente, las búsquedas de las publicaciones tienen el siguiente formato:

```
http://api.semanticscholar.org/[S2PaperId | DOI | ArXivId]
```

Siendo S2PaperID, DOI y ArXivId identificadores unívocos. Por otro lado, en lo que respecta al formato de las búsquedas por autor, tienen el siguiente formato:

```
http://api.semanticscholar.org/v1/author/[S2AuthorId]
```

Siendo S2AuthorId un identificador respecto al autor. Como no se conocían los identificadores anteriormente mencionados, nos apoyamos en la api de DBLP para obtenerlos (la API de DBLP sí que permite la búsqueda textual, es decir, a partir del nombre del autor o de la publicación). En resumen, la Api de DBLP permite la consulta de los autores según el siguiente formato:

```
https://dblp.org/seach/publ/api?q= [nombre publicacion]&formato=json
```

Podríamos resumir el proceso de obtención de información del autor en los siguientes pasos (mostramos el proceso de búsqueda de información del profesor del máster en “Data Science” Alberto Fernández Isabel):

```
- hits: {
  @total: "29",
  @computed: "29",
  @sent: "29",
  @first: "0",
  - hit: [
    - {
      @score: "3",
      @id: "374993",
      - info: {
        author: "Alberto Cocaña-Fernández",
        url: "https://dblp.org/pid/159/3045"
      },
      url: "URL#374993"
    },
    - {
      @score: "3",
      @id: "553300",
      - info: {
        author: "Alberto Fernández 0002",
        - aliases: {
          alias: "Alberto Fernández Gil"
        },
        - notes: {
          - note: {
            @type: "affiliation",
            text: "University Rey Juan Carlos, Mostoles, Spain"
          }
        },
        url: "https://dblp.org/pid/132/1966"
```

Ilustración 16: Consulta a la API de DBLP en búsqueda de autores iguales al buscado

Inicialmente, realizaremos una consulta contra la API DBLP, que nos puede devolver varios resultados que se ajustan al criterio de búsqueda (puede ser que haya varios autores o publicaciones con un nombre similar).

Mediante la distancia de Jaccard compararemos el elemento buscado (Alberto Fernández Isabel) con cada uno de los elementos devueltos en la consulta a la API y, el que tenga mayor coeficiente de Jaccard, será el que guarde mayor semejanza y, por tanto, el que seleccionaremos.

Por otro lado, si nos fijamos en los autores devueltos por las consultas a la API de Semantic Scholar, vemos que uno de los valores que acompaña al autor es una url con un pid (identificador):

```
author: "Alberto Cocaña-Fernández",
url: "https://dblp.org/pid/159/3045"
```

Ilustración 17: url con pid del autor

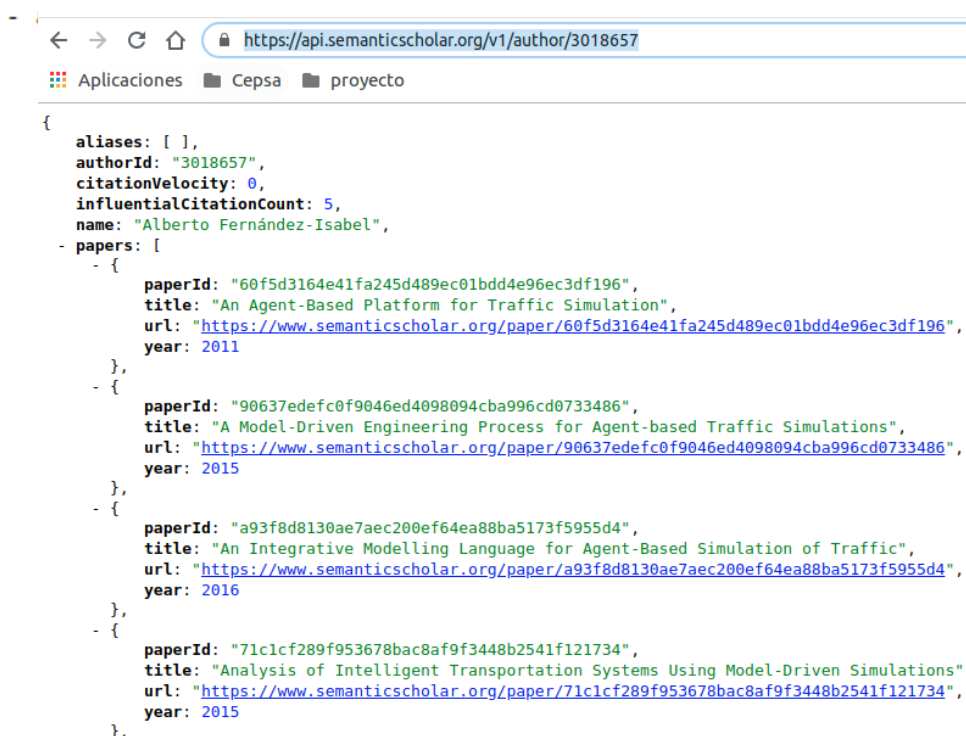
No obstante, ese identificador sólo es válido para la API de DBLP y no para Semantic Scholar, que es de donde queremos extraer información. Por dicho motivo, tendremos que seguir realizando consultas a las APIs de DBLP y Semantic Scholar.

```
- {
  @score: "7",
  @id: "918946",
  - info: {
    - authors: {
      - author: [
        "Antonio F. G. Sevilla",
        "Alberto Fernández-Isabel",
        "Alberto Díaz 0001"
      ]
    },
    title: "Enriched Semantic Graphs for Extractive Text Summarization.",
    venue: "CAEPIA",
    pages: "217-226",
    year: "2016",
    type: "Conference and Workshop Papers",
    key: "conf/caepia/SevillaFD16",
    doi: "10.1007/978-3-319-44636-3_20",
    ee: "https://doi.org/10.1007/978-3-319-44636-3_20",
    url: "https://dblp.org/rec/conf/caepia/SevillaFD16"
  },
  url: "URL#918946"
},
```

Ilustración 18: Se recoge el identificador de uno de sus artículos

A continuación, tendremos que realizar una nueva consulta a la API de DBLP, respecto a dicho autor. Ésta consulta nos devolverá información entre la que se encuentra los títulos que han sido escritos por él, de los cuales extraeré todos los títulos que tengan el atributo DOI (identificador único del paper):

Después, a partir del identificador del artículo, volveremos a realizar otra consulta respecto a ese identificador contra la API de DBLP, que nos devolverá información relevante del artículo, entre los que podemos destacar los el nombre de los autores y su identificador (authorId):



```

{
  aliases: [ ],
  authorId: "3018657",
  citationVelocity: 0,
  influentialCitationCount: 5,
  name: "Alberto Fernández-Isabel",
  papers: [
    {
      paperId: "60f5d3164e41fa245d489ec01bdd4e96ec3df196",
      title: "An Agent-Based Platform for Traffic Simulation",
      url: "https://www.semanticscholar.org/paper/60f5d3164e41fa245d489ec01bdd4e96ec3df196",
      year: 2011
    },
    {
      paperId: "90637edefc0f9046ed4098094cba996cd0733486",
      title: "A Model-Driven Engineering Process for Agent-based Traffic Simulations",
      url: "https://www.semanticscholar.org/paper/90637edefc0f9046ed4098094cba996cd0733486",
      year: 2015
    },
    {
      paperId: "a93f8d8130ae7aec200ef64ea88ba5173f5955d4",
      title: "An Integrative Modelling Language for Agent-Based Simulation of Traffic",
      url: "https://www.semanticscholar.org/paper/a93f8d8130ae7aec200ef64ea88ba5173f5955d4",
      year: 2016
    },
    {
      paperId: "71c1cf289f953678bac8af9f3448b2541f121734",
      title: "Analysis of Intelligent Transportation Systems Using Model-Driven Simulations",
      url: "https://www.semanticscholar.org/paper/71c1cf289f953678bac8af9f3448b2541f121734",
      year: 2015
    }
  ]
}

```

Ilustración 19: Consulta final a la API de Semantic Scholar

De los cuales tomaremos el identificador (authorId) del autor que guarde mayor semejanza con el autor buscado. Con dicho identificador, el paso final sería realizar una consulta a la API de Semantic Scholar y así obtener la información necesaria para las diferentes visualizaciones.

5.2.Limpieza de la base de datos

Durante el proceso de obtención de los datos de las APIS, se almacenó la información en la BBDD. No obstante, para el cálculo de la reputación de los autores y las obras, hubo que realizar cálculos con datos almacenados previamente para después almacenar dichos valores.

5.3.Uso del algoritmo para el cálculo de las semejanzas

En cuanto al algoritmo para el cálculo de las semejanzas, escogimos la semejanza coseno y usamos la librería de scikitlearn, la cual proporciona funcionalidades que nos ayudaron a calcular fácilmente la distancias cosenos de los vectores intergrantes de la matriz.

5.4.Reducción de dimensionalidad (t-SNE y MDS)

En ésta etapa tuvimos que buscar información respecto a dichos métodos de reducción de dimensionalidad. La mayor dificultad radicó en la comprensión de los dos anteriores métodos de reducción. Concretamente, la reducción mediante t-SNE necesitaba de una capacidad de abstracción mayor que la de MDS y hubo que buscar varios artículos para interiorizar las bases del método de reducción.

5.5.Visualización de los métodos de reducción

En ésta etapa tuvimos que manejar las diferentes visualizaciones y la manera de situarlas en la misma pantalla. Para las visualizaciones relacionadas con las similitudes entre autores, usamos la librería de python matplotlib, que es muy intuitiva y fácil de usar.

5.6.Implementación de visualización del grafo relacional.

En ésta sección del desarrollo, la dificultad estuvo en la implementación de los controles (botones) mediante los cuales el usuario podría interactuar la visualización del grafo, con funcionalidades como “zoom”, moverse por la visualización, guardarlo en varios formatos...

Además, tuvimos que analizar algún método para normalizar las dimensiones de los nodos, puesto que había una diferencia abismal entre algunos autores u obras, porque unos tenían mucha mayor reputación en comparación con otros y esto hacía que la visualización fuera poco clarificante.

5.7.Implementación de aplicación de escritorio

En ésta etapa construimos la ventana principal de la aplicación. Dudamos qué librería escoger, pero nos decantamos por la de tkinter. Además, durante el desarrollo de la misma nos dimos cuenta de que no se podían lanzar varias ejecuciones consecutivas de una representación, puesto que después de la primera el botón de ejecución quedaba bloqueado. Esto era debido a que el proceso que se encargaba de dicha ejecución nunca moría, así que lo solucionamos controlando los estados de las ejecuciones e incorporando hilos de ejecución, lo cual permitía varias ejecuciones consecutivas de la misma.

Capítulo 6

En este apartado se recopilan las pruebas que se hayan realizado para verificar que la implementación del prototipo funciona correctamente, es decir, que la solución ha sido validada y es eficaz y que hace lo que se supone que debería hacer.

Tiene por objetivo asegurar la corrección y fiabilidad del prototipo. La finalidad es obtener una aplicación que responda bien ante todas las situaciones posibles que se puedan dar.

Casos de estudio

Pruebas parciales

En todos los casos de uso analizados se han realizado pruebas parciales e integradas. En cuanto a las pruebas parciales, consiste en acotar pequeñas parte del desarrollo probando una funcionalidad concreta, aunque sea reducida, ya que es más fácil detectar un error en una parte pequeña del desarrollo que en el sistema global. Además las pruebas sobre un único subsistema son más extensivas, cubriendo mayor parte del espectro de circunstancias que pueden aparecer.

Entre las pruebas parciales podemos destacar el caso de uso relativo a la extracción de la información sobre el autor. Es un paso previo a todos los demás y es la base sobre la que se sostiene el desarrollo y representación posterior de las similitudes. Es un proceso complejo que requiere el tratamiento de la información devuelto por las diferentes APIS y su almacenamiento en la base de datos.

Otra prueba parcial consiste en el cálculo de la reputación de los autores relacionados con un autor. Una vez procesada la información, los datos se deben analizar para evaluar la influencia de dicho autor en relación con los demás. Ello implica cálculos, como los mostrados en el Capítulo 4. Gracias a la reputación, se puede representar la dimensión de los nodos del grafo y en definitiva, la relación entre sus elementos.

Pruebas de integración

En lo que respecta a las pruebas de integración, consiste en demostrar la funcionalidad del proyecto englobando todos los componentes que se han desarrollado, de tal forma que haya varios subsistemas funcionando conjuntamente.

Para ello, se propone una matriz de correlación en la que se muestra el grado de similitud entre cada uno de sus integrantes. Por simplicidad, se desarrollarán las explicaciones relativas a las funcionalidades de “similitud entre autores” y “similitud entre obras de un autor” y no se desarrollará la explicación de la funcionalidad “similitud de obras entre autores”.

Para nuestros experimentos nos basamos en una matriz de correlación, se establece la relación de correlación entre los autores que refleja el grado de similitud entre sus elementos, siendo el valor 1.0 (color rojo) el valor indicante de una similitud total y 0.0 el grado de similitud mínimo (color azul).

En las matrices de correlación mostradas en la parte inferior, se muestra de manera gráfica la relación de sus integrantes. Es decir, la relación de los elementos de la columna de la izquierda con los elementos de la parte superior. La relación entre cada par de elementos es “conmutativa” (llamando “relación” a la similitud entre los elementos). Es decir, la relación de correlación entre, por ejemplo, Jorge Navarro y Cristina Conde es la misma que la existente entre Cristina Conde y Jorge Navarro (en la matriz de correlación de la parte inferior). Lo cual tiene sentido, porque la similitud de un primer elemento con el segundo es la misma que la similitud entre el segundo elemento y el primero.

Del mismo modo, la relación de correlación de un elemento consigo mismo es máxima, ya que la similitud de un elemento consigo mismo es total (de ahí que la diagonal principal, indicante de la relación de los elementos consigo mismo, sea de color rojo).

6.1. Experimento de la similitud entre autores

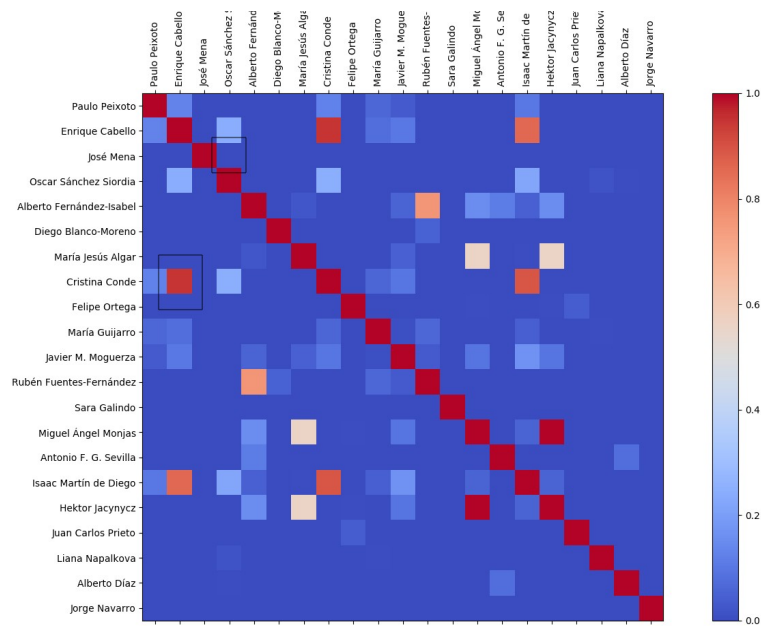


Ilustración 20: Matriz de correlación de similitud entre autores

El grado de similitud podemos verlo en nuestra funcionalidad mediante la reducción por MDS. La similitud entre los elementos dependerá de la distancia entre ellos: cuanto menor sea la distancia entre un par de elementos mayor será la similitud entre ellos y, en cambio, cuanto mayor sea la distancia entre los elementos, menor será la similitud entre ellos (los ejemplos vendrán remarcados en la matriz de correlación mediante un recuadro negro).

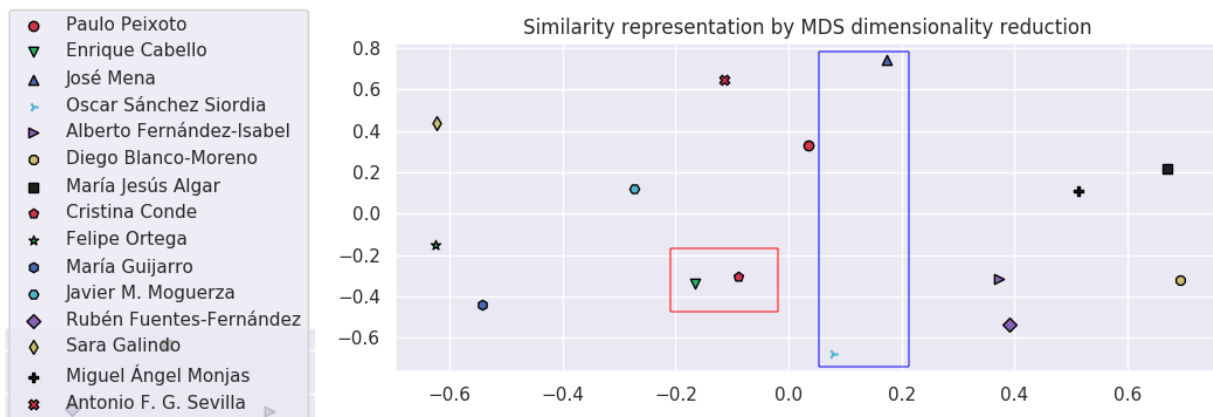


Ilustración 21: MDS de similitud entre autores

En el caso concreto de los autores podríamos mostrar, por un lado, la relación de semejanza fuerte entre Cristina Conde y Enrique Cabello (señalada en rojo en la representación mediante MDS). Por otro lado, un ejemplo de similitud debil es la existente entre José Mena y Óscar Sánchez Siorda (señalada en azul en el gráfico MDS)

Como se puede ver, la funcionalidad muestra correctamente las proporciones de las relaciones de similitud entre sus elementos.

6.2. Experimento de la similitud entre obras de un autor

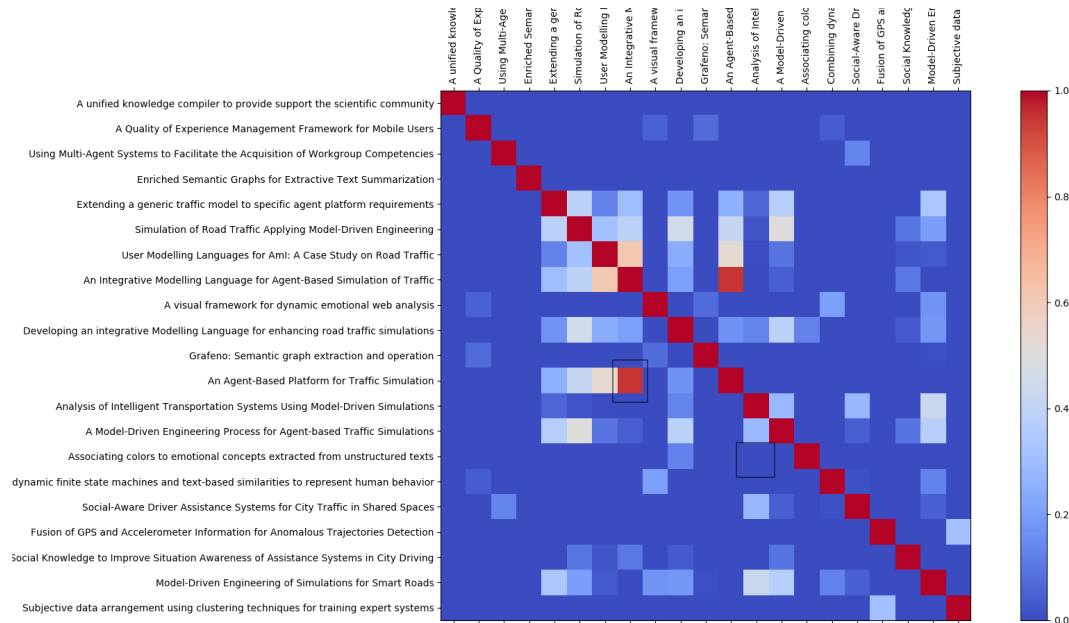


Ilustración 22: Matriz de correlación de obras de Alberto Fernández-Isabel

A continuación mostramos la matriz de correlación relativa a las relaciones existentes entre las obras de Alberto Fernández-Isabel.

En esta matriz de correlación de similitudes entre obras, podemos distinguir la fuerte relación existente, por ejemplo, entre la obra de “An Agent-Based Platform for Traffic Simulation” y “An Integrative Modelling Language for Agent-Based Simulation of Traffic”(ejemplos señalados mediante un recuadro negro en la matriz).



Ilustración 23: MDS de similitud entre obras

En la representación mediante MDS, se puede ver que la distancia entre los elementos “An Agent-Based Platform for Traffic Simulation” y “An Integrative Modelling Language for Agent-Based Simulation of Traffic”(señalado en rojo) es mucho menor que la distancia existente entre las obras de “Associating colors to emotional concepts extracted from unstructured texts” y “Analysis of Intelligent Transportation Systems Using Model-Driven Simulations” (señaladas mediante un recuadro azul)

Por ello, podemos concluir que el prototipo desarrollado muestra correctamente las relaciones de similitud entre obras

Capítulo 7

El Procesamiento del Lenguaje Natural es el futuro de la tecnología. No obstante, el comportamiento y eficacia de las distintas técnicas de NLP variará en función de la naturaleza de la tarea que tratemos de resolver, del tipo de documentos a analizar, y del coste computacional que podamos asumir. De todo lo dicho, se deduce la necesidad de continuar trabajando con el fin de dilucidar nuevas técnicas o enfoques que contribuyan a superar las deficiencias de las existentes. Sólo así podremos alcanzar la quimera que hoy es la comprensión automática del lenguaje natural.

Conclusiones y trabajo futuro

En esta capítulo se muestran las conclusiones alcanzadas en proyecto y se identifican las posibles líneas de trabajo futuro que se han ido detectando durante el desarrollo del mismo. En la sección 6.1 se recapitula el contexto de la herramienta desarrollada y se exponen las conclusiones obtenidas del desarrollo de este proyecto. En la sección 6.2 se muestran las posibles líneas de trabajo futuro más destacadas para la evolución de la herramienta.

7.1. Conclusiones

El objetivo de este prototipo ha consistido en el desarrollo de una herramienta que permita, de una manera gráfica y fácil de interpretar, reflejar las semejanzas entre artículo y autores. Para el desarrollo del proyecto, se ha usado el modelo iterativo e incremental. Partiendo de unos recomendaciones por parte del tutor, se han ido consiguiendo los objetivos planteados y mejorando la propuesta.

La fase de obtención de datos se ha realizado usando los datos de dos APIs: Semantic Scholar y DBLP. Respecto al análisis de semejanza [26] entre documentos o autores, se ha usado la distancia coseno, comparando las distancias entre las palabras integrantes de los abstract de los documentos. Además, se han usado varios métodos de reducción de dimensionalidad (t-SNE y MDS), que nos permiten tener diferentes interpretaciones de las semejanzas entre documentos y autores.

Posteriormente, se añadió como funcionalidad la representación mediante un grafo de la relación entre autores y publicaciones. Por último, se desarrolló una interfaz de usuario amigable que permitiera la ejecución de la funcionalidad deseada por el usuario.

Se puede concluir que ha conseguido desarrollar un prototipo funcional capaz de realizar los análisis de semejanzas que se pretendían de forma sencilla y manejable, con una interfaz amigable.

Esta herramienta puede continuar evolucionando en sucesivos desarrollos para mejorar los análisis que realiza y dotarlos de más precisión y profundidad.

7.2.Trabajo Futuro

En esta sección se detallan las posibles líneas de trabajo futuro para continuar la evolución de esta herramienta. Se muestra una lista de líneas de trabajo que se han detectado a lo largo del desarrollo y que pueden contribuir a la mejora de la precisión y funcionalidad de la herramienta.

Hemos encontrado varias dificultades durante el proceso de recuperación de información y el proceso del “data massaging” (proceso en el que se cruzan, limpian y preparan los datos). En nuestro caso, hemos tenido información procedente de Semantic Scholar y DBLP, cuyas APIs hemos usado para obtener la información.

Existe un identificador único de cada autor que publica en Semantic Scholar. No obstante, si no se conoce, al hacer una búsqueda por nombre en el buscador de dicha web, se devolverán todos los autores cuyo nombre coincida con el nombre buscado. Es por ello que encontramos un problema respecto a la imposibilidad para diferenciar autores que, por coincidencia, se llamen igual. En nuestro caso, durante el desarrollo del proyecto, hemos buscado información respecto al profesor Felipe Ortega, profesor del máster en Data Science de la URJC. Sin embargo, el buscador nos devuelve además publicaciones de otro autor perteneciente al ámbito de la medicina, tal y como se puede ver en la imagen inferior.

The screenshot shows the Semantic Scholar profile for Felipe Ortega. The browser address bar displays the URL: <https://www.semanticscholar.org/author/Felipe-Ortega/37318854>. The profile header includes the name "Felipe Ortega", a "CREATE ALERT" button, a "SUGGEST CHANGES" button, and statistics: "View Author Influence", "92 Highly Influential Citations", and a "Citation Trend" chart.

The "Filter Results" section on the left shows "Full text PDF available (17)", "Publication Year" (1994 to 2019), "Publication Type", "Co-author", and "Journals and Conferences". The main list of publications is sorted by "Highly Influential Citations".

Three publications are visible, with red arrows pointing to them:

- Publication 1:** "Oligodendroglial and neurogenic adult subependymal zone neural stem cells constitute distinct lineages and exhibit differential responsiveness to Wnt signalling". Authors: Felipe Ortega, Sergio Gascón, +7 authors. Published in Nature Cell Biology, 2013. It includes citation counts (18, 29) and options to view on Nature, cite, or save.
- Publication 2:** "Reprogramming of pericyte-derived cells of the adult human brain into induced neuronal cells." Authors: Marisa Karow, Rodrigo Alberto Hoyos Sánchez, +11 authors. Published in Cell stem cell, 2012. It includes citation counts (11, 40) and options to view on PubMed, cite, or save.
- Publication 3:** "Identification and Successful Negotiation of a Metabolic Checkpoint in Direct Neuronal Reprogramming." Authors: Sergio Gascón, Elisa Murenu, +15 authors. Published in Cell stem cell, 2016. It includes a brief abstract snippet.

Ilustración 24: Varias personas comparten nombre en Semantic Scholar

En conclusión, si no se puede conocer a priori el identificador del autor buscado, podríamos establecer unos criterios de búsqueda. Es decir, si conocemos el ámbito de estudio del autor (cardiología, machine learning, estadística...), podríamos crear una bolsa de términos relacionados con ese autor, de forma que podamos diferenciar o limitar la problemática anterior.

Además, se podría optar por analizar también el texto de los documentos de los autores, lo que haría que la aplicación fuera mucho más precisa en sus análisis, ya que en el caso de nuestro prototipo, solo evaluamos la semejanza entre documentos en función de su *abstract*.

Bibliografía

- 1: , Computing Machinery and intelligence, October 1950
- 2: Roger Schank, A conceptual dependency parser for natural language Proceedings of the 1969 conference on Computational linguistics, 1969
- 3: Woods, William A, Transition Network Grammars for Natural Language Analysis, 1970
- 4: James Allen, Natural Language Processing, 2003
- 5: García García, Emilio , Primera ponencia. Teoría de la mente y ciencias cognitivas», 2007
- 6: García García, Emilio , Primera ponencia. Teoría de la mente y ciencias cognitivas», 2007
- 7: MA Martí, JL Boix, J Llisterri, Tratamiento del lenguaje natural: tecnología de la lengua oral y escrita, 2002
- 8: C. MANNING, H. SCHÜTZE, Foundations of Statistical Natural Language Processing, 1999
- 9: Mg. Augusto Cortez Vásquez^{1,2}, Mg. Hugo Vega Huerta^{1,2}, Lic. Jaime Pariona Quispe, Procesamiento de lenguaje natural, 2009
- 10: KANAGASUNDARAM, A., VOGT, R., DEAN, D. B., and SRIDHARAN, S. , Plda based speaker recognition on short utterances. In The Speaker and Language Recognition Workshop, 2012
- 11: BW Locke, Named entity recognition: Adapting to microblogging, 2009
- 12: Y Zhang, R Jin, ZH Zhou, Understanding bag-of-words model: a statistical framework, 2010
- 13: Y Seki, Sentence Extraction by tf/idf and position weighting from Newspaper Articles, 2002
- 14: F Sebastiani, Machine learning in automated text categorization, 2002
- 15: LK Wives, Utilizando conceitos como descritores de textos para o processo de identificação de conglomerados (clustering) de documentos, 2004
- 16: JR Demey, L Pla, JL Vicente-Villardón, Medidas de distancia y similitud, 2011
- 17: T Mitchell, B Buchanan, G DeJong, Machine learning, 1990
- 18: R Caruana, A Niculescu-Mizil , An empirical comparison of supervised learning algorithms, 2006
- 19: L Van Der Maaten, E Postma, J Mach Learn Res, Dimensionality reduction: a comparative, 2009
- 20: JT Tou, RC Gonzalez, Pattern recognition principles, 1974
- 21: I Borg, P Groenen , Modern multidimensional scaling: Theory and applications, 2003
- 22: L Maaten, G Hinton, Visualizing data using t-SNE, 2008
- 23: A Fernández-Isabel, Developing an integrative Modelling Language for enhancing road traffic simulations, 2015
- 24: EM Méndez Nava, G Ramón, Modelo de evaluación de metodologías para el desarrollo de software, 2006
- 25: D Glez-Peña, A Lourenço, Web scraping technologies in an API world, 2013
- 26: A Islam, D Inkpen , Semantic text similarity using corpus-based word similarity and string similarity, 2008