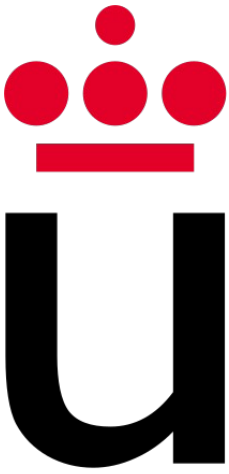

Universidad Rey Juan Carlos

MÁSTER EN DATA SCIENCE

CURSO ACADÉMICO 2017-2018

Trabajo de fin de Máster

ANÁLISIS DE SEMEJANZAS ENTRE ARTÍCULOS Y AUTORES

Autor: Carlos Sánchez Vega

Tutores: Alberto Fernández-Isabel

Isaac Martín de Diego

Agradecimientos

Por una parte quisiera agradecer el esfuerzo dedicado por mi tutor, Alberto Fernández Isabel ,que 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).

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 functiality the application allows to show, we can mention the similarity analysis between authors and papers, similarity between text by the same author or, finally, similarity anlysis 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).

Índice

Índice de ilustraciones

Capítulo 1

Introducción

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. Por ello trabaja áreas como el entendimiento por parte de una máquina del lenguaje humano, su percepción o generación. Por ejemplo, un software de traducción aplica NLP, siendo una de sus tareas entender que «Hello» es una palabra inglesa que en castellano se traduce como «Hola».

El procesamiento de lenguaje natural ha sido un campo de estudio desde la antigüedad y lo sigue siendo hoy en día[1]. De hecho cuando Alan Turing formula por allá 1950 su famoso test en “Computing Machinery and Intelligence”[2], está fijando las bases del NLP. El NLP, en mayor o menor grado de complejidad, se aplica a múltiples tareas de nuestro día a día, y desde hace bastantes años. Por ejemplo, y como ya se ha comentado, cualquier traducción de texto (por ejemplo un tuit) emplea NLP[3]. De la misma forma, un sistema que extrae información de un email y partir de esos datos sugiere apuntar una cita en la agenda. También es NLP un análisis de sentimiento sobre si una expresión es positiva, negativa o neutra («Me encanta mi nuevo teléfono» vs «Mi nuevo teléfono va lento«). Igualmente usan NLP la clasificación de texto para detectar spam en el correo electrónico o la indicación de errores gramaticales mientras se escribe un texto.

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 obviamente 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

El objetivo de éste proyecto de final de máster es crear una aplicación que permita, de manera funcional y sencilla, reflejar las semejanzas entre textos y autores de documentos. Además, durante el desarrollo del prototipo se descubrió la fuerte interrelación entre papers académicos y autores, por lo que se incluyó como funcionalidad la representación, en forma de grafo, de los papers académicos y los autores que habían escrito los mismos.

El desarrollo se realizo 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 primer capítulo de la memoria nos centraremos en la introducción, poniendo el foco en el contexto, motivación y objetivos principales.
- 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 capitulo 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 dsarrollo de la herramienta.
- Finalmente, en el capítulo 7 se mostrarán las conclusiones a las que se ha llegado con el desarrollo y se jhará incapié en las posibles líneas de trabajo futuras existentes.

Capítulo 2

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[4] y psicología cognitiva[5]. 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[6] 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 complemente 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[7] es incluir estas definiciones en una computadora y luego usarlas para crear una oración estructurada y sin ambigüedades con un significado bien definido.

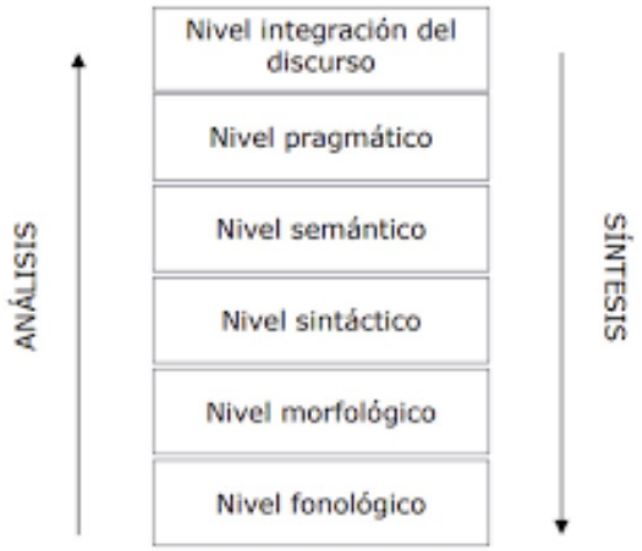

Ilustración 1: Niveles del lenguaje

2.2. APRENDIZAJE AUTOMÁTICO EN EL PROCESAMIENTO DE LENGUAJE

Los algoritmos de Procesamiento del Lenguaje Natural[8] 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. Resumenes 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[9] 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[10] 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 sentimiento

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 cuentifica 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 caracter ojetivo.

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 RI 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 eificiencia de la aplicación es mayor.

Después de analizar las opciones, se optó por escoger el *abtract*, 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. Lemming

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

```
soy, son, es => ser
gato, gata, gatos => gato
ir, voy, iré, fui => ir
```

en ingles:

```
am, are, is => be
car, cars,car's,cars' => car
```

2.4.3. Tokenización

Consiste en separar palabras del texto en entidades llamadas *tokens*, con las que trabajaremos luego. Deberemos 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[11] 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[12] es una técnica muy utilizada en Machine Learning[13]. 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.

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[14], detección de temas de documentos, traducción de documentos…

El cáculo 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 expllicaremos 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:

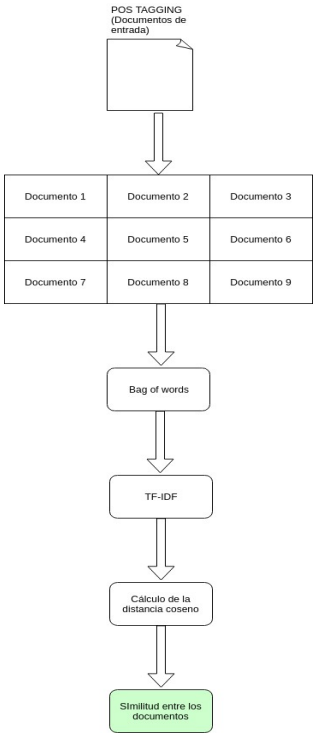

Ilustración 2: Semejanza en

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[15] 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 indénticos. 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 use de la similitud de Jacquard.

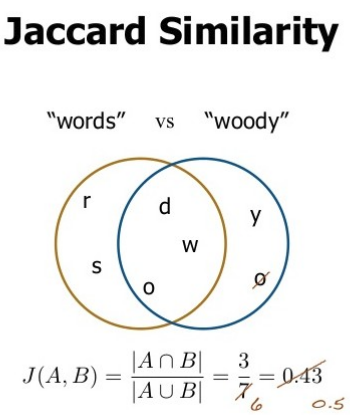

Ilustración 3: Similitud de Jaccard

Como se puede ver ern la imagen superior, “words” y “woody” están formados cada una por un conjunto de letras, de las cuales unícas 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[16], 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

Éste 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[17] 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 cientfícico de datos e ingenieros de datos.

Se trata de un lenguaje de programación multiparadigma, que 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 sera 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.

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

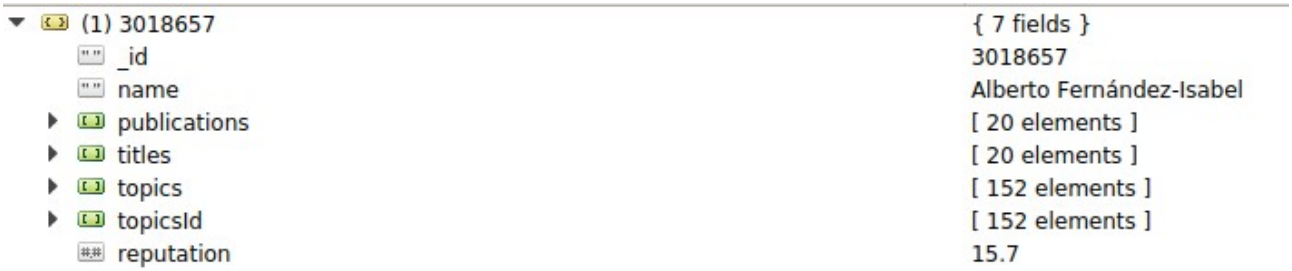

Ilustración 4: Colección de los autores

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:

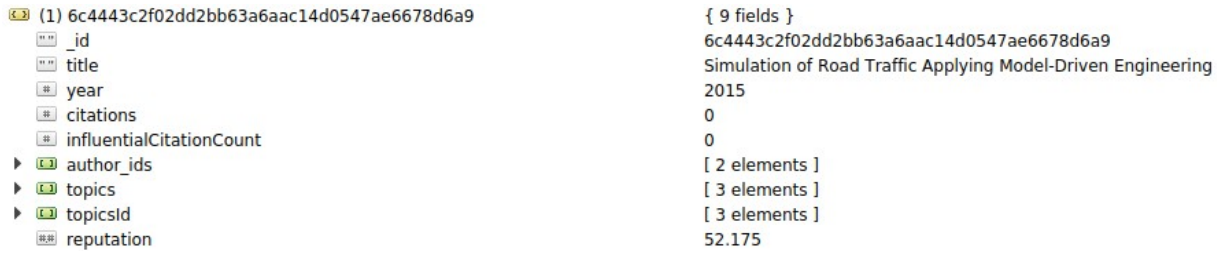

Ilustración 5: Colección de las publicaciones

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[18] es una de las etapas más importantes en problemas de reconocimiento de patrones[19], 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[20] 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 situan los n puntos de una configuración inicial en p dimensiones, esto es, suponer para cada objeto las coordenadas (x1, x2, ..., xp) en el espacio de p dimensiones.

2) Se calculan las distancias euclidianas entre los objetos de esa configuración, esto es, calcular las dij, 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)[21]: 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 *S1* 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 norma. En éste punto obtendríamos otra matriz de similitud llamado *S2.*
4. Posteriormente, t-SNE compara las matrices S1 y S2 y calcula las diferencias mínimas entre los puntos de la matriz S1 y S2.

Capítulo 3

Alcance funcional de la propuesta

En esta sección se describe la funcionalidad que es capaz de realizar la herramienta desarollada y en que etapas está organizada esta funcionalidad. Se introducen estas últimas y se reprresenta 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.

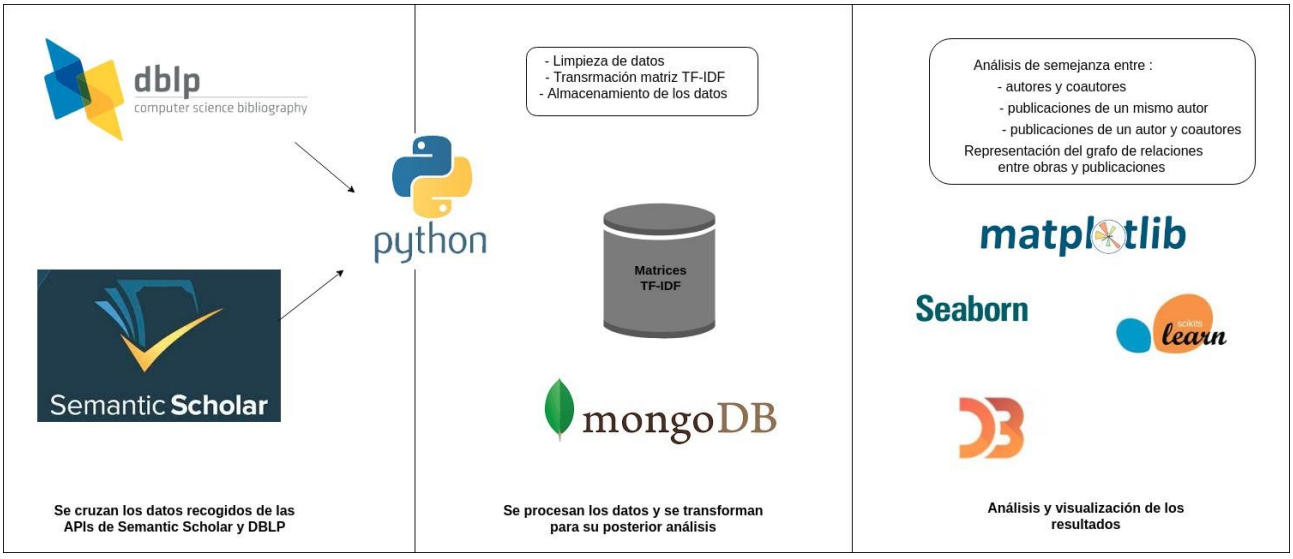

Ilustración 6: Esquema funcional de la propuesta

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 pueder ver en la imagen inferior:

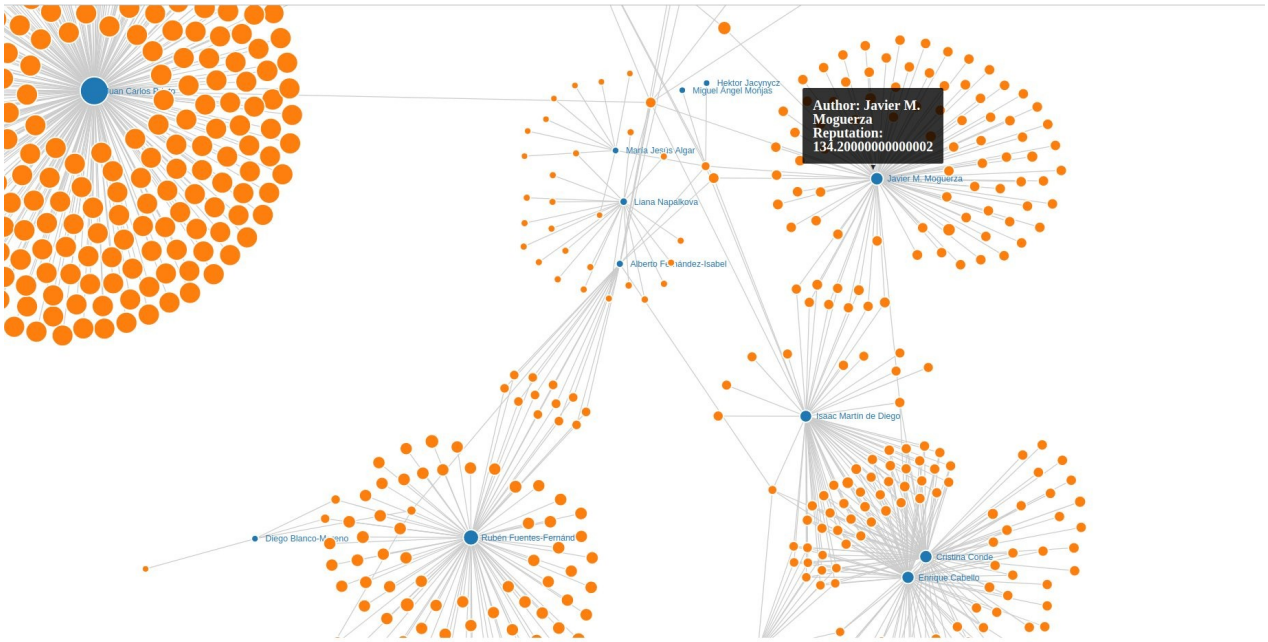

Ilustración 8: Mensaje informativo respecto sobre un autor

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.

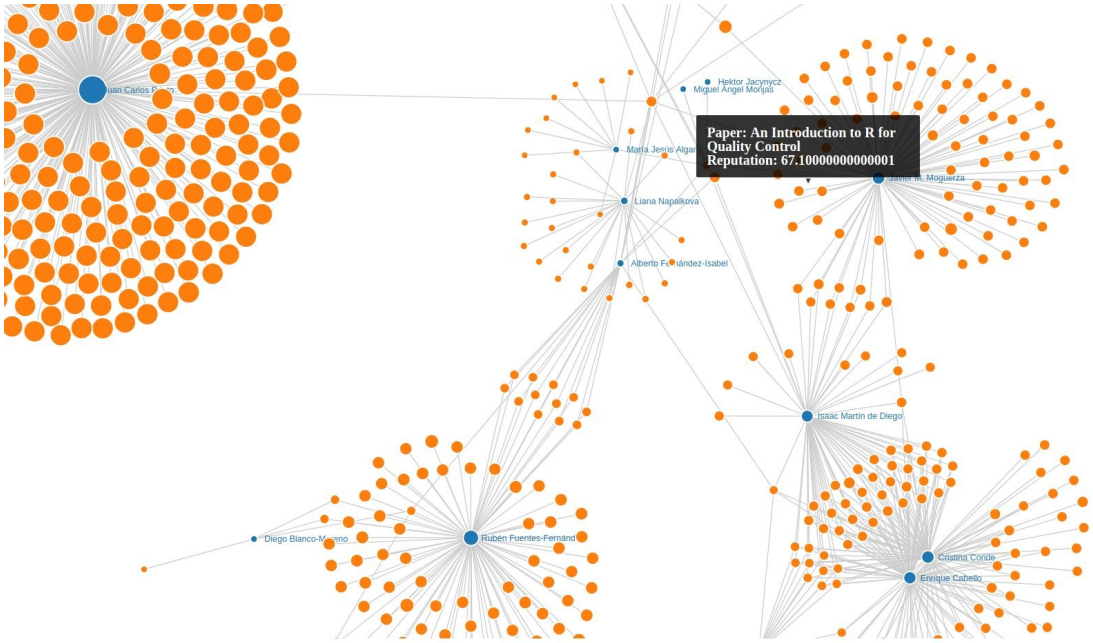

Ilustración 9: Mensaje informativo sobre un artículo

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

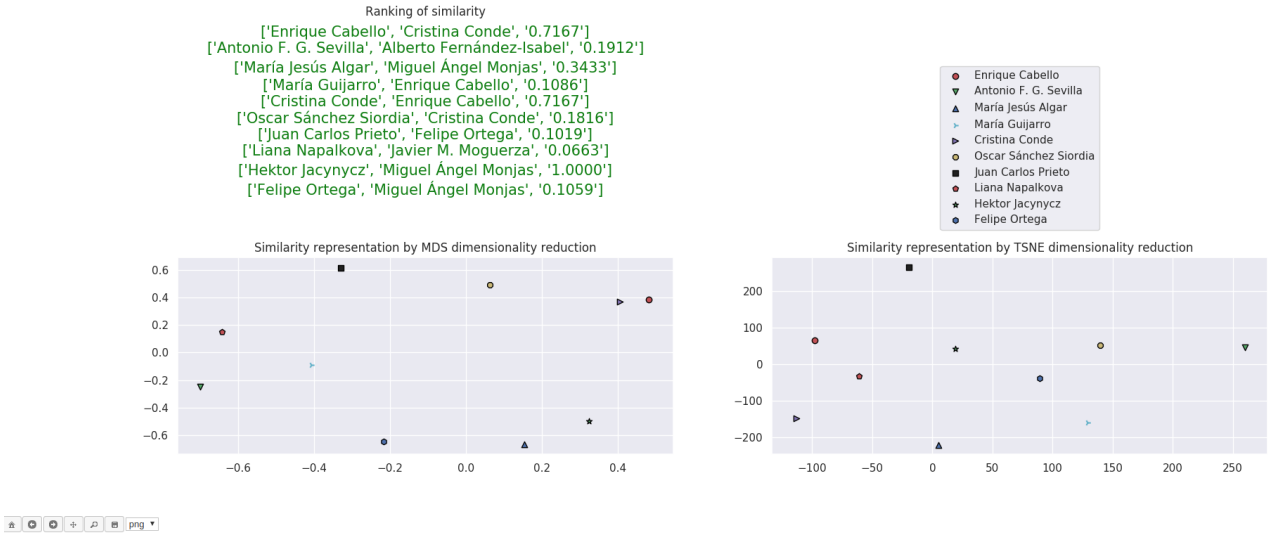

Ilustración 10: Comparador del autor con otros autores

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

Ésta 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” [22] 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).

Ilustración 12: Comparador de obras entre todos los autores

Un punto a notar en éste caso es el tiempo de ejecución, ya que puede ser más elevado que en cualquiera de las opciones anteriores porque el número de elementos a comparar es mucho mayor (se comparan todos las obras de cada autor, uno a uno respecto a los demás)

3.2.Caracteristicas relevantes de las visualizaciones de similitud

Todas las visualizaciones de similitudes se han implementado con el objetivo de una buena interactividad del usuario. Un ejemplo de ello es la botonera que aparece en la esquina inferior izquierda de la pantalla:

Ilustración 13: Botonera

A continuación explicaremos la funcionalidad de cada uno de los botones:

El botón de zoom sirve para aumentar la visualización de una zona concreta. Para ello habrá que “dibujar” el recuadro en el cuál quememos hacer incapié.

Este botón servirá para volver a recentrar la imagen una vez que se ha hecho zoom en una zona concreta de la visualización

El botón servirá para descargar la imagen en el formato deseado. Una vez seleccionado el formato de la imagen a descargar en el *combobox* de la imagen, al pulsar el botón del disco se abrirá una nueva pestaña sin los controles de la botonera, de forma que podamos descargarlo.

Los formatos soportados para descargar son: eps, jpeg, pdf, png, ps, raw, svg y tif

Los botones servirán para retornar a la visualización anterior (izquierda) o posterior (derecha). Ésta funcionalidad puede ser útil en el caso en el que se haya cambiado la visualización inicial, como cuando hacemos zoom en puntos concreto del gráfico.

Capítulo 4

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 varie en función de su reputación. A continuación mostramos las formulas por las que calculamos las reputaciones

Reputación de un autor:

$$Reputation_{publication} = Coef.Reputation_{article} * Nº\ veces\ citado + Coef.Reputation_{author} * \sum_{1}^{n} \frac{RepAuthor_n}{n}$$

siendo n el total de autores de esa publicación

$$Reputation_{author} = Coef.Reputation_{article} * Nº\ artículos + C.Citations_{author} * Nº\ Citas * Nº Citations_{influential} * NºCitas_{influential} + Coef._{Seniority} * Seniority$$

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 antiguedad que tiene ese autor como publicador.

Capítulo 5

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.

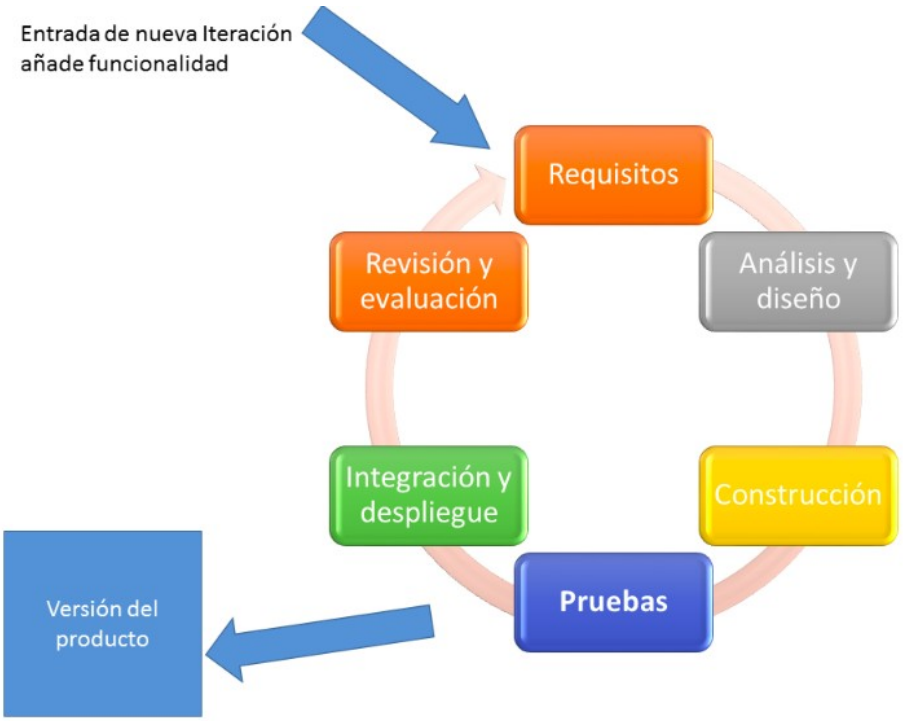

Ilustración 14: Desarrollo iterativo

Captítulo 6

Etapas de desarrollo

En este apartado se muestran las fases del desarrollo del protitipo. Se muestran las etapas en orden cronológco, comenzando por la etapa de la fase 6.0.1 , en la que se afronta la recogida, “data massaging”[25] y guardado de datos.

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 de lal prototipo, que consta de la selecciónm del algotitmo 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

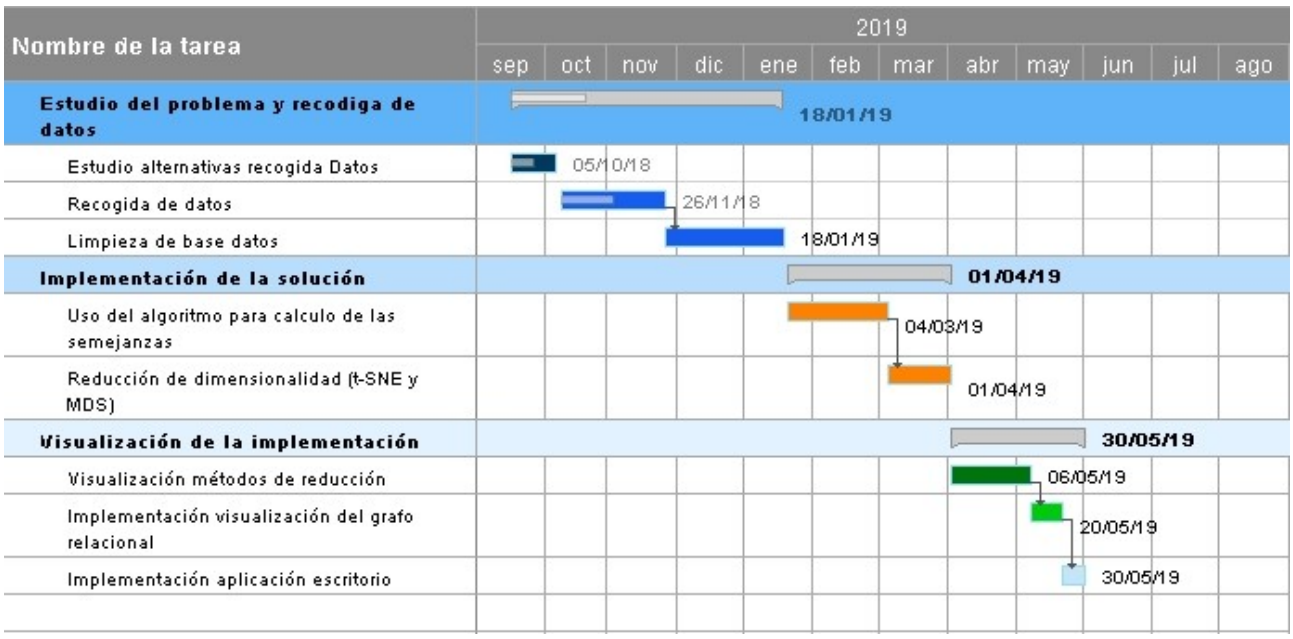

Ilustración 15: Etapas del desarrollo del prototipo

6.2.1.Estudio alternativas de recogida de datos

La primera fase del desarrollo consite en estudiar las diferentes alternativas existentes de cara a la recogida de datos.
En un primer momento, se pensó en *scrapear[26]* 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 obteber la información necesaria.

Recogida y limpieza de datos

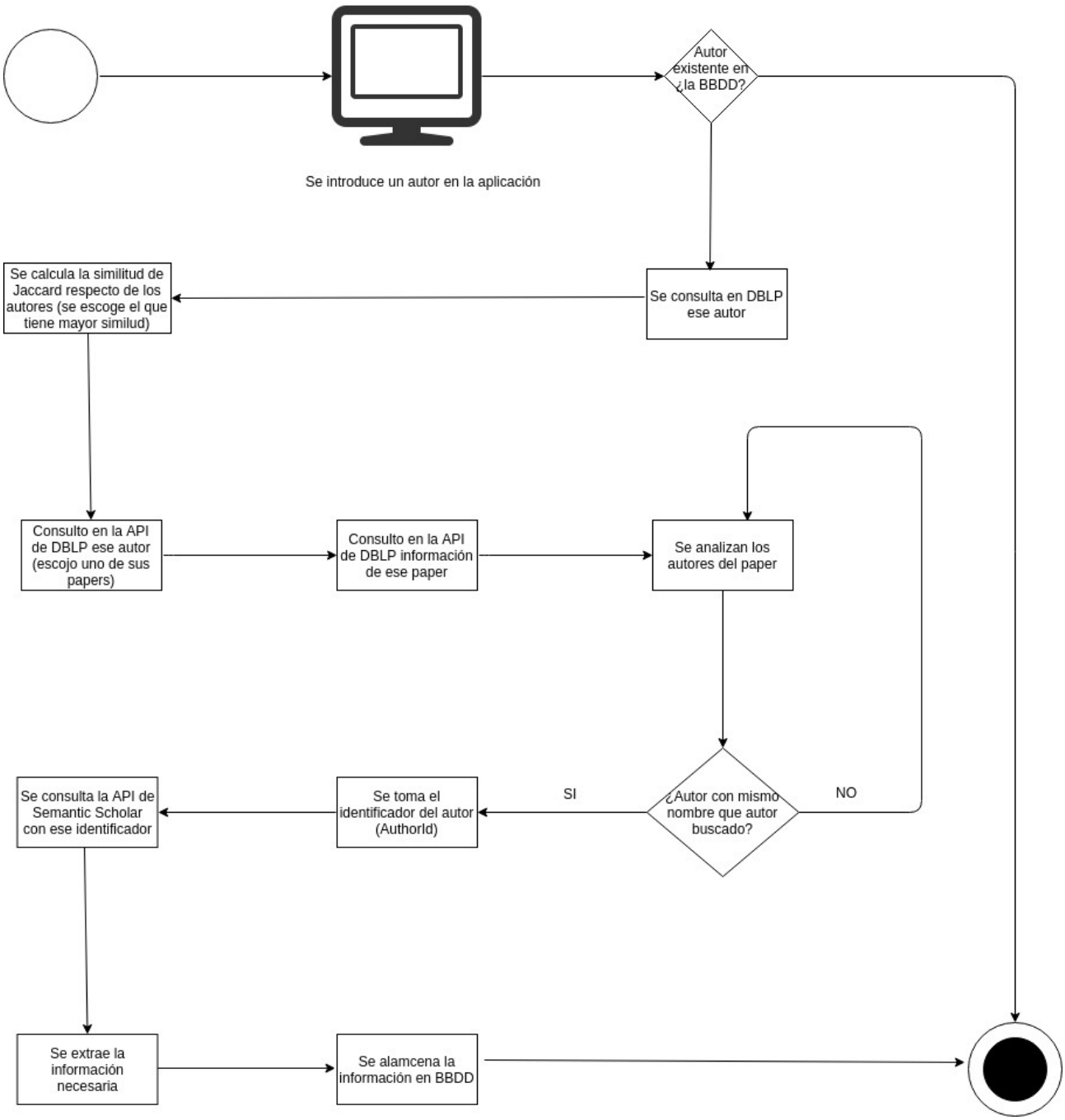

Ilustración 16: Proceso de almacenaje de información en la BBDD

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 unívucos.

Por otro lado, en lo que respecta al formato de las bísquedas 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 contulta 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 17: 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 (indentificador):


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


Ilustración 18: 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.

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 (indeitificador unívuco del paper):

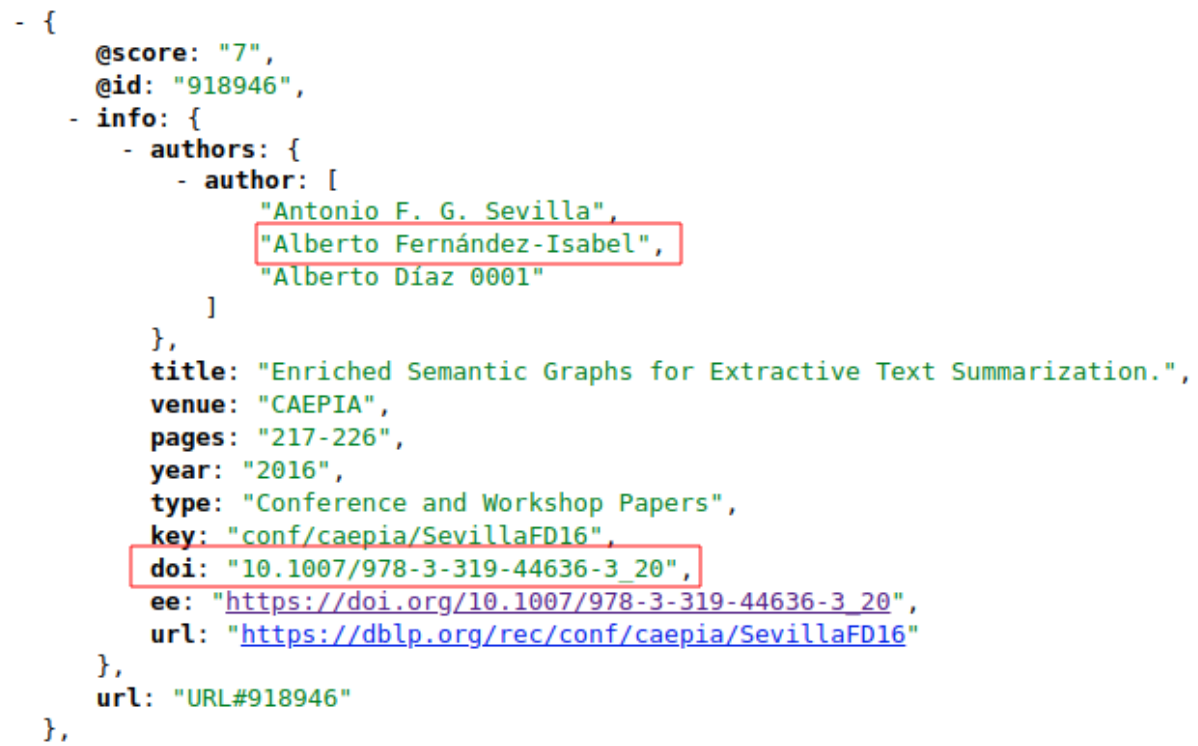

Ilustración 19: Se recoge el identificador de uno de sus artículos

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):

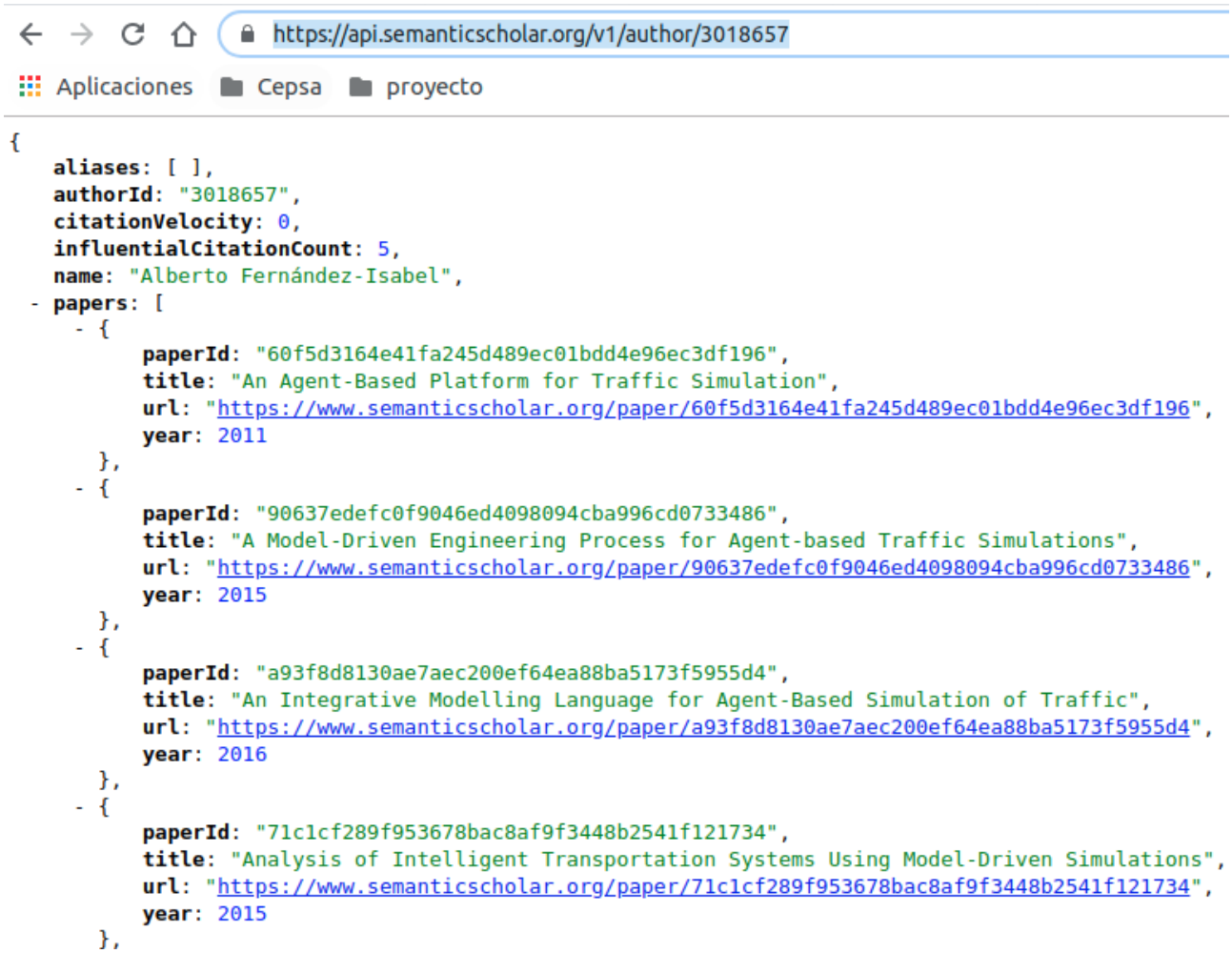

Ilustración 20: Consulta final a la API de Semantic Scholar

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.

6.2.2.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.

6.2.3.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.

6.2.4. 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.

6.2.5.Visualización de los métodos de reducción

En ésta étapa 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.

6.2.6.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.

6.2.7.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 7

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 conseguiendo 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[27] 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 unívuco 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.

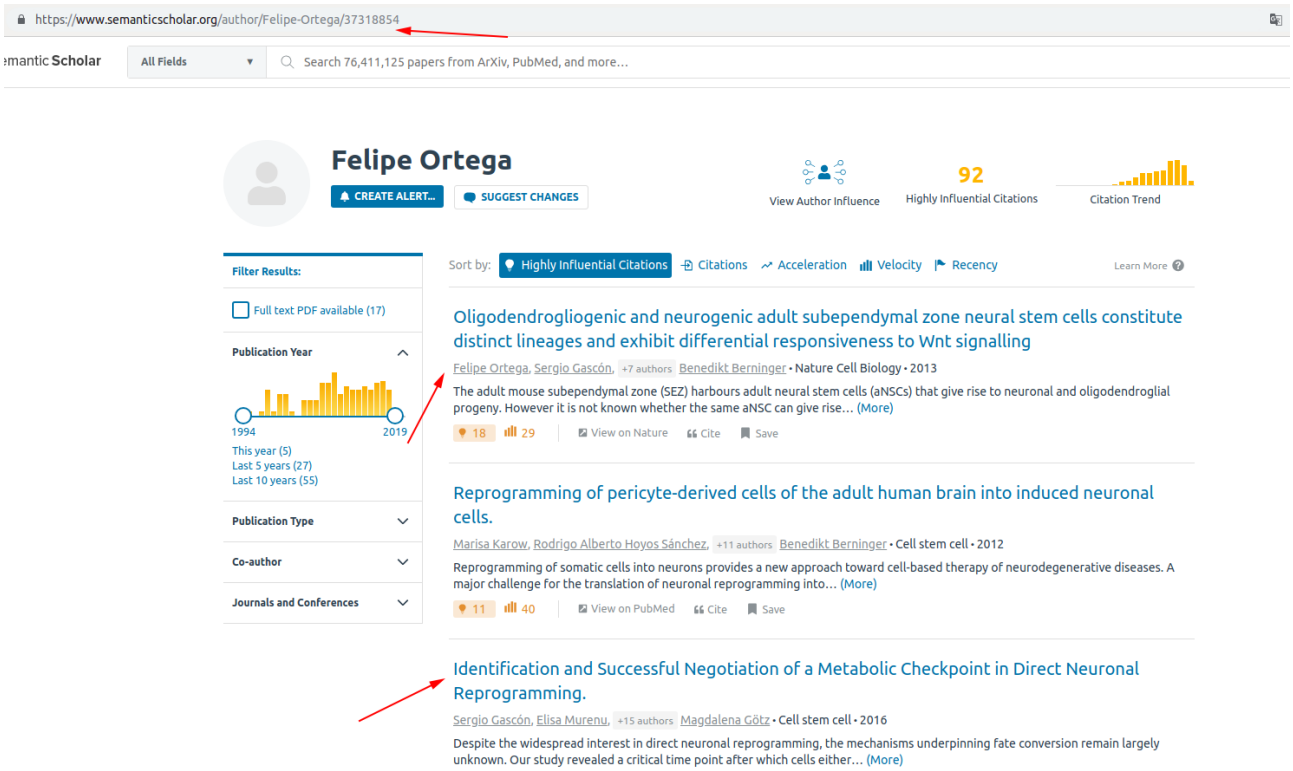

Ilustración 21: Varias personas comparten nombre en Semantic Scholar

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: Fernández-Isabel, A., Prieto, J. C., Ortega, F., de Diego, I. M., Moguerza, J. M., Mena, J., ... & Napalkova, L., A unified knowledge compiler to provide support the scientific community. Knowledge-Based Systems, 2018
2: , Computing Machinery and intelligence, October 1950
3: James Allen, Natural Language Processing, 2003
4: MA Arbib, A Lecci, The metaphorical brain: An introduction to cybernetics as artificial intelligence and brain theory, 1972
5: García García, Emilio , Primera ponencia. Teoría de la mente y ciencias cognoscitivas», 2007
6: MA Martí, JL Boix, J Llisterri, Tratamiento del lenguaje natural: tecnología de la lengua oral y escrita, 2002
7: C. MANNING, H. SCHÜTZE, Foundations of Statistical Natural Language Processing, 1999
8: Mg. Augusto Cortez Vásquez1,2, Mg. Hugo Vega Huerta1,2, Lic. Jaime Pariona Quispe, Procesamiento de lenguaje natural, 2009
9: 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
10: BW Locke, Named entity recognition: Adapting to microblogging, 2009
11: Y Zhang, R Jin, ZH Zhou, Understanding bag-of-words model: a statistical framework, 2010
12: Y Seki, Sentence Extraction by tf/idf and position weighting from Newspaper Articles, 2002
13: F Sebastiani, Machine learning in automated text categorization, 2002
14: LK Wives, Utilizando conceitos como descritores de textos para o processo de identificação de conglomerados (clustering) de documentos, 2004
15: JR Demey, L Pla, JL Vicente-Villardón, Medidas de distancia y similitud, 2011
16: T Mitchell, B Buchanan, G DeJong, Machine learning, 1990
17: R Caruana, A Niculescu-Mizil , An empirical comparison of supervised learning algorithms, 2006
18: L Van Der Maaten, E Postma, J Mach Learn Res, Dimensionality reduction: a comparative, 2009
19: JT Tou, RC Gonzalez, Pattern recognition principles, 1974
20: I Borg, P Groenen , Modern multidimensional scaling: Theory and applications, 2003
21: L Maaten, G Hinton, Visualizing data using t-SNE, 2008
22: A Fernández-Isabel, Developing an integrative Modelling Language for enhancing road traffic simulations, 2015
23: A Fernández-Isabel, R Fuentes-Fernández, A Model-Driven Engineering Process for Agent-based Traffic Simulations., 2015
24: EM Méndez Nava, G Ramón, Modelo de evaluación de metodologías para el desarrollo de software, 2006
25: K Diederichs, PA Karplus, Better models by discarding data?,
26: D Glez-Peña, A Lourenço, Web scraping technologies in an API world, 2013
27: A Islam, D Inkpen , Semantic text similarity using corpus-based word similarity and string similarity, 2008