Universidad Politécnica

de Madrid

Escuela Técnica Superior de

Ingenieros Informáticos

Grado en Ingeniería Informática

Trabajo Fin de Grado

Uso de Modelos del Lenguaje para Búsquedas Semánticas en Textos Científicos

Autor: Miguel Arne Loma-Osorio Andrés

Cotutores: Álvaro García Barragán, Víctor Robles Forcada

Madrid, junio, 2024

Este Trabajo Fin de Grado se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Trabajo Fin de Grado
Grado en Ingeniería Informática

Título: Usos de Modelos del Lenguaje para Búsquedas Semánticas en Textos Científicos
Junio, 2024

Autor: Miguel Arne Loma-Osorio Andrés

Cotutores:
Álvaro García Barragán, Víctor Robles Forcada
Arquitectura y Tecnología de Sistemas Informáticos
ETSI Informáticos
Universidad Politécnica de Madrid

Resumen

A diferencia de los buscadores tradicionales, los buscadores semánticos nos permiten obtener información más relevante y de mayor utilidad a partir de una consulta. Esto se debe a que no se limitan a buscar coincidencias en palabras clave, si no que tratan de entender el significado y el contexto detrás de la consulta. Para ello, en la búsqueda semántica se utilizan técnicas del PLN (Procesamiento del Lenguaje Natural) y, de forma más específica, de la CLN (Comprensión del Lenguaje Natural), como pueden ser el análisis sintáctico y semántico, o la contextualización de la consulta, entre otras.

Este trabajo se centra en el uso y adaptación de diferentes modelos del lenguaje para el desarrollo de un buscador semántico especializado en búsquedas en textos científicos. Para ello, el sistema utiliza distintos modelos Sentence Transfromers para la generación de embeddings de los textos científicos. Posteriormente, estos embeddings se almacenan e indexan de forma eficiente en una base de datos semántica, de forma que cuando el sistema recibe una consulta de un usuario, se pueden recuperar de forma eficiente los documentos más relevantes. Asimismo, se ha desarrollado una aplicación web que permite introducir todo tipo de consultas y filtrar los resultados según parámetros específicos, como el título o el autor del documento.

Por último, para probar la efectividad y precisión del buscador semántico, se ha llevado a cabo una evaluación del sistema utilizando artículos de investigación biomédica. Esta evaluación se ha realizado con la ayuda de Álvaro García Barragán, científico informático e investigador especializado en "machine learning", y Andrea Álvarez Pérez, ingeniera biotecnológica, investigadora y experta en reposicionamiento de medicamentos. Los resultados obtenidos en la evaluación han sido positivos y garantizan el buen funcionamiento y la efectividad del sistema a la hora de responder consultas de carácter científico.

Palabras clave: Embeddings, Sentence Transformers Models, Búsqueda Semántica, PLN

Abstract

Unlike traditional search engines, semantic search engines allow us to obtain more relevant and useful information from a query. This is because they are not limited to searching for keyword matches but try to understand the meaning and context behind the query. For this purpose, semantic search uses techniques from NLP (Natural Language Processing) and, more specifically, from NLU (Natural Language Understanding), such as syntactic and semantic analysis, or query contextualization, among others.

This work focuses on the use and adaptation of different language models for the development of a semantic search engine specialized in scientific text searches. To this end, the system uses different Sentence Transfromers models for the generation of embeddings from scientific texts. Subsequently, these embeddings are stored and indexed efficiently in a semantic database, so that when the system receives a query from a user, the most relevant documents can be retrieved efficiently. A web application has also been developed, which allows, among other functionalities, to enter all types of queries and filter the results according to specific parameters, such as the title or author of the document.

Finally, to test the effectiveness and accuracy of the semantic search engine, an evaluation of the system has been carried out using biomedical research articles. This evaluation has been conducted with the help of Álvaro García Barragán, computer scientist and researcher specialized in machine learning, and Andrea Álvarez Pérez, biotechnological engineer, researcher, and expert in drug repositioning. The results obtained in the evaluation guarantee the good performance and effectiveness of the system in answering scientific queries.

Keywords: Embeddings, Sentence Transformers Models, Semantic Search, NLP

Tabla de contenidos

Lista de figuras

1 Introducción

1.1 Contexto

En la era digital actual, la búsqueda de información en la web se ha convertido en una actividad cotidiana e indispensable para millones de personas en todo el mundo. Actualmente, los motores de búsqueda desempeñan un papel crucial al permitir a los usuarios acceder a grandes cantidades de datos e información de todo tipo en unos segundos. Sin embargo, a pesar de todos los avances recientes en la tecnología de búsqueda, los motores de búsqueda tradicionales, basados en la coincidencia de palabras clave, presentan limitaciones significativas en términos de precisión y relevancia de los resultados. Esto se debe a que los documentos que le interesan al usuario no siempre contienen las palabras clave incluidas en la consulta, ya que podría contener sinónimos o palabras de la misma familia semántica. Asimismo, al no tener en cuenta el contexto de la consulta, pueden tener dificultades con palabras ambiguas, como es el caso de las palabras homónimas, que son aquellas que tiene el mismo sonido, pero significado diferente [1].

Como respuesta a las limitaciones de los buscadores tradicionales, surge la búsqueda semántica, la cual se centra en la comprensión del significado, el contexto, y la intención detrás de la consulta con el objetivo de mostrar aquellos resultados de mayor interés para el usuario. Para ello, los buscadores semánticos utilizan el Procesamiento del Lenguaje Natural (PLN). En esta disciplina, la arquitectura de redes neuronales Transformer [2] permite a los computadores interpretar, manipular y comprender el lenguaje humano. De esta forma, la búsqueda semántica muestra como resultado de una consulta información más precisa y relevante para los usuarios.

Por lo tanto, se puede observar que la adaptación de este enfoque constituye una alternativa robusta a la búsqueda tradicional y puede jugar un rol fundamental en todo tipo de campos. De forma más específica, cabe destacar su importancia en el mundo científico, debido, entre otras razones, al vocabulario técnico poblado de ontologías utilizado en las distintas ramas de la ciencia [3] [4].

1.2 Objetivos

El principal objetivo de este TFG (Trabajo de Fin de Grado) es desarrollar una aplicación que permita a los usuarios realizar consultas de cualquier índole y, como respuesta, se muestren aquellos documentos que tienen mayor relevancia

para responder a esa consulta. Para poder lograr esta meta, se plantean los siguientes objetivos secundarios:

- Formación en Manejo de Datos en Python
- Integración de Modelos de Lenguaje para Búsqueda Semántica
- Manejo de Bases de Datos Semánticas
- Desarrollo de aplicación web para el Buscador Semántico

1.3 Estructura del documento

Con el objetivo de presentar de forma clara y ordenada la información contenida en este documento, se ha dividido en las siguientes secciones: en la siguiente sección, se detalla el estado del arte (Sección 2 en la búsqueda semántica, destacando las técnicas y funcionalidades que implementan algunos de los buscadores semánticos científicos más utilizados en la actualidad. A continuación, se introduce una sección donde se explican los materiales y métodos (Sección 3) utilizados para el desarrollo de la aplicación desde un punto de vista teórico. Después se describe el diseño y la implementación de la aplicación del buscador semántico (Sección 4), detallando los distintos módulos en los que se divide y cómo se conectan entre ellos, así como los métodos de evaluación utilizados. Posteriormente, se presentan y se analizan los resultados obtenidos en la evaluación (Sección 5). Por último, se incluyen las conclusiones extraídas de la investigación y desarrollo del proyecto (Sección 6), seguido por la proyección a futuro (Sección 7).

2 Estado del arte

En la actualidad, existen múltiples buscadores semánticos especializados en distintos campos de la ciencia. Esto se debe a la gran utilidad que proporciona la búsqueda semántica en este ámbito, ya que permite a los investigadores obtener de forma rápida los textos científicos (artículos, libros, actas de conferencia…) más relevantes, precisos, y actualizados para poder llevar a cabo su labor de investigación con mayor facilidad. Por ejemplo, en el campo de la medicina, supone un gran avance para la farmacovigilancia, que es la ciencia y las actividades relativas a la identificación, evaluación, y prevención de los riesgos asociados a medicamentos o vacunas que ya han sido comercializados [5].

2.1 Semantic Scholar

Semantic Scholar [6] es un motor de búsqueda diseñado para su uso en el campo de la literatura científica. Fue desarrollado por el *Allen Institute for Artificial Intelligence* en 2015, con el objetivo de proporcionar un método superior y más efectivo para la búsqueda y descubrimiento de conocimiento científico mediante la aplicación de técnicas y algoritmos de PLN y "machine Learning" [7].

En la actualidad, Semantic Scholar ofrece más de 218 millones de artículos científicos que cubren todos los campos de la ciencia. No obstante, se trata de una herramienta orientada para su uso en las ciencias biomédicas y en las ingenierías.

En lo relativo a su funcionalidad, al realizar una consulta, Semantic Scholar muestra todos los artículos relacionados con la consulta ordenados por relevancia, junto a su abstract. Asimismo, ofrece el TLDR (Too Long, Didnt Read) para cada uno de ellos, que se trata de un resumen breve de 2 o 3 líneas del artículo generado por IA (Inteligencia Artificial). Asimismo, además de la información elemental, como son el autor, la fecha o el campo, muestra el número de citas por cada artículo, que es una información muy útil a la hora de buscar artículos de alta calidad, y también permite filtrar los resultados según estos parámetros.

2.2 Consensus

Consensus [8] constituye otra de las herramientas más utilizadas hoy en día para la búsqueda semántica en textos científicos. Este buscador incorpora modelos del lenguaje avanzados para la búsqueda de información, utilizando una combinación de búsqueda de palabras clave y búsqueda de vectores para localizar los mejores resultados. Asimismo, utiliza GPT4 para generar un resumen para los artículos devueltos para una consulta.

En la actualidad, Consensus trabaja con más de 200 millones de artículos científicos, extraídos de las bases de datos de Semantic Scholar, que son actualizados mensualmente.

En lo relativo a su funcionalidad, al realizar una consulta Consensus muestra los resultados más relevantes con un breve resumen de 1 o 2 líneas, generado por GPT4 para cada uno de ellos, y una respuesta a la consulta, generada a partir de la información contenida en los diez artículos más relevantes. Además, Consensus ofrece una innovadora funcionalidad (Consensus Meter) al responder directamente a las preguntas de si/no, indicando porcentajes para sí, no, y posiblemente.

3 Materiales y Métodos

3.1 Datos

Para la recopilación de artículos científicos utilizados en el buscador semántico, se ha utilizado un archivo RIS, que es un formato de etiquetas de datos utilizado para realizar referencias bibliográficas. Este archivo se llama "Articles.ris" y fue elaborado por Andrea Álvarez Pérez, mediante la aplicación de consultas en Rayyan, que es una herramienta que permite filtrar, clasificar e importar los estudios relevantes de un área específica [9].

El archivo "Articles.ris" hace referencia a 5797 textos científicos pertenecientes al campo de la biomedicina, que han sido obtenidos de las bases de datos PubMed, Scopus, Web of Science, ScienceDirect y Dimensions.ai. De forma más específica, son artículos que tratan el reposicionamiento de medicamentos ("drug repurposing") y de los distintos métodos computacionales utilizados en esta área.

3.2 Embeddings

Los embeddings (incrustaciones de vectores en español) son representaciones numéricas en forma de vector que encapsulan el significado de una palabra, frase o documento, de forma que puedan ser procesados por modelos de "machine learning". Son una herramienta esencial y muy utilizada en muchas tareas del PLN como son la clasificación y generación de texto, sistemas de búsqueda y de recomendaciones, y análisis de sentimientos, entre otras.

Para aprender los embeddings, las tecnologías más recientes utilizan modelos de "machine learning" que han sido entrenados con enormes cantidades de datos, con el objetivo de aprender las sutilezas y peculiaridades del lenguaje y contenido con el que fueron entrenados. De forma más específica, son generados por las capas internas de modelos de lenguaje que han sido entrenados para aprender a predecir palabras según su contexto. Los vectores obtenidos de este proceso tienen ciertas características que resultan de utilidad en diversas áreas del PLN. En primer lugar, los embeddings constituyen de vectores densos, que son aquellos que tienen la mayoría o todos sus elementos con valores distintos de cero. Su uso supone una mejor captura de las relaciones semánticas entre palabras u oraciones, y una mayor eficiencia computacional. Por otro lado, estos vectores son similares al representar palabras que provienen de un contexto parecido, por lo cual suponen una herramienta de gran utilidad en la búsqueda semántica [10] [11].

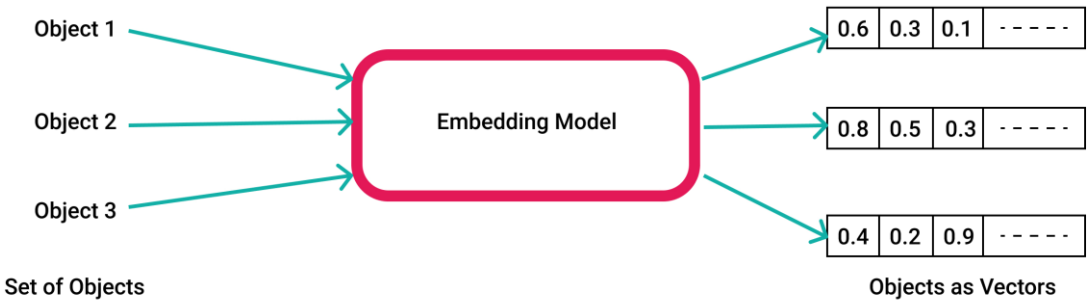

Figura 3.1: Generación de embeddings [12]

La distancia entre estos vectores indica como de relacionados están, siendo menor para aquellos elementos más relacionados y viceversa. Dependiendo del modelo utilizado para generar los embeddings, estos tendrán diferente número de elementos (dimensión). Una mayor dimensión de los embeddings implica una mejor captura de las relaciones semánticas de las palabra u oraciones, pero implica un mayor uso de tiempo y de recursos computacionales para su cálculo.

Para poder representar y visualizar estos vectores, es común utilizar PCA (Análisis de Componentes Principales) para reducir la dimensionalidad a tres o dos dimensiones.

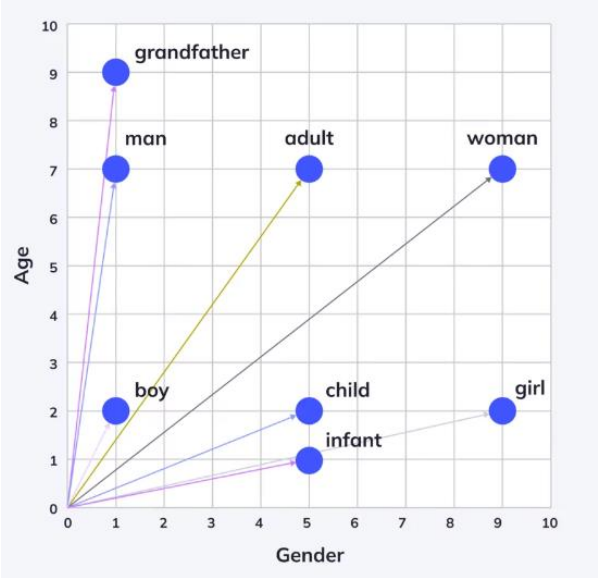

Figura 3.2: Representación en 2D de los embeddings de varias palabras con PCA [13]

Existen dos tipos principales de embeddings: estáticos y dinámicos. La principal diferencia reside en que los embeddings dinámicos o contextualizados son aquellos en que la representación de una palabra varía dependiendo del contexto en el que aparece, mientras que en los estáticos es una representación fija [14].

Las principales técnicas y algoritmos utilizados en la generación de embeddings estáticos son Word2Vec y GloVe. Word2Vec consiste en el uso de dos arquitecturas clave: CBoW (Continuous Bag of Words), que predice la palabra del centro a partir del contexto de las palabras de alrededor y Skip-gram, que funciona de forma opuesta, ya que predice las palabras de alrededor a partir de la del centro. Por otro lado, Glove (Global Vectors) se basa la co-ocurrencia de palabras en un corpus para aprender la representación de palabras [15] [16].

Por otro lado, algunos de los métodos más utilizados para la generación de embeddings dinámicos incluyen BERT, GPT2 y Sentence Transformers, los cuales todos parten de la arquitectura Transformer. Esta arquitectura modifica el patrón codificador/decodificador de las redes neuronales para implementar un mecanismo de autoatención, el cual permite al modelo procesar los datos en distinto orden y centrarse en aquellas partes de la oración con mayor relevancia [17].

3.3 Bases de datos semánticas

Las bases de datos semánticas son un tipo de base de datos que permiten almacenar y organizar la información y los datos según su significado, captando en el proceso las relaciones semánticas entre estos [18]. Surgieron en la década de los 70, en respuesta a las limitaciones de las bases de datos relacionales y, debido a su forma de estructurar los datos, permiten un almacenamiento y recuperación de la información más eficientes.

En el contexto de la búsqueda semántica, estas bases de datos actúan como bases de datos vectoriales que almacenan los embeddings de grandes dimensiones, que han sido calculados a partir de la información relevante de los textos o documentos que se pretenden utilizar. Algunas de las ventajas que proporcionan son la eficiencia de la recuperación de datos, debido a la incorporación de técnicas de indexado como la cuantización o el clustering. Asimismo, utilizan algoritmos especializados que han sido diseñados para vectores, como el método de los k vecinos más cercanos (k-nearest neighbors) [19]. Finalmente, cabe destacar que garantizan una escalabilidad eficaz del sistema que las implementa.

3.4 Modelos Sentence Transformers

Los Sentence Transformers son modelos que pueden generar embeddings a partir de oraciones y textos, encapsulando su significado de forma rápida y con alta precisión. Este nuevo enfoque surge en 2019, con el objetivo de mejorar la efectividad de la generación de sentence embeddings de BERT (Bidirectional Encoder Representations from Transformers), que daba lugar a malos resultados, con una media inferior a los embeddings de GloVe. Asimismo, tenía como objetivo mejorar de forma significativa el tiempo de computación de BERT, llegando a tardar 5 segundos aproximadamente para la computación de 10000 oraciones, frente a las 65 horas que empleaba BERT [20].

En lo relativo al funcionamiento de estos modelos, se pasa una oración o texto a un modelo Transformer preentrenedo, como puede ser BERT, que se encarga de calcular los word embeddings de cada uno de los tokens que componen el texto u oración. A continuación, se aplica la capa de pooling, existiendo distintas opciones, como son la capa de mean-pooling (calcula la media de los word embeddings) o la capa de max-pooling (calcula el máximo valor de los word embeddings). Una vez aplicada esta capa, se obtiene el sentence embedding que representa el significado de la oración que se ha pasado al modelo [21].

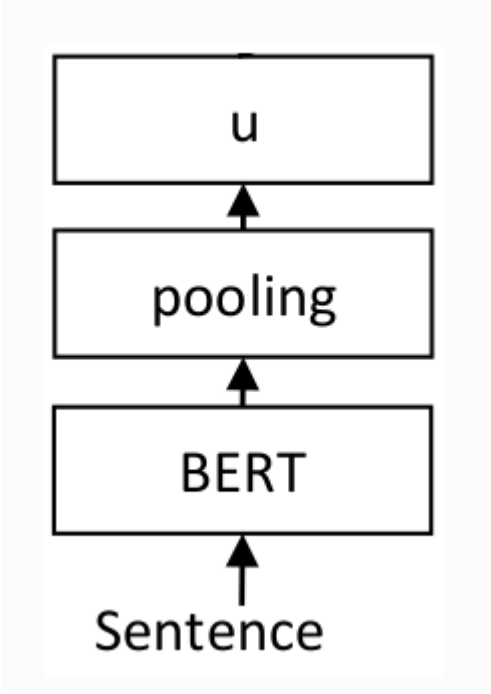

Figura 3.3: Arquitectura básica de Sentence Transformers utilizando BERT [22]

Respecto al entrenamiento de estos modelos, es necesario ir pasando pares de frases al modelo indicando la similitud entre ambas. No obstante, existen distintos métodos para preparar los datos de entrenamiento antes de alimentarlos al modelo. Un método podría ser indicar mediante una etiqueta el grado de similitud (entero o decimal) entre las dos frases. Por otro lado, se puede ir entrenando el modelo suministrándole pares de oraciones o párrafos similares, sin necesidad de utilizar ninguna etiqueta. Por ejemplo, se pueden utilizar oraciones adyacentes o párrafos de un mismo texto, ya que tienen más probabilidad de ser similares que si provienen de distinta fuente [23] [24].

Para comprobar la efectividad y la precisión del modelo a preentrenar y para ir ajustándolo (fine-tuning), se utiliza una función de pérdida que evalúa la calidad de los embeddings generados. Como se puede observar en la siguiente figura, en los modelos Sentence Transformers, se utilizan diferentes funciones de pérdida en base al método de preparación de los datos de entrenamiento que se ha elegido [23].

Datasetstructures to train your SentenceTransformers model

Datasetstructure	Example datasets(repo id in Hugging Face Hub)	Loss functions(imported from sentence_transformers)
Pair of sentences and a label indicating how similar they are	snli	ContrastiveLoss; SoftmaxLoss; CosineSimilarityLoss
Pair of positive (similar) sentences without a label	embedding-data/flickr30k_captions_quintets; embedding-data/coco_captions_quintets	MultipleNegativesRankingLoss; MegaBatchMarginLoss
Single sentence with an integer label	trec; yahoo_answers_topics	BatchHardTripletLoss; BatchAllTripletLoss; BatchHardSoftMarginTripletLoss; BatchSemiHardTripletLoss
Triplet (anchor, positive, negative) sentences	sembedding-data/QQP_triplets	TripletLoss

Figura 3.4: Función de pérdida según la estructura de datos de entrenamiento [23]

3.5 Arquitectura RAG

La arquitectura RAG (Retrieval Augmented Generation, Generación Mejorada por Recuperación en español) es un enfoque que permite a los LLMs (Large Language Models, Modelos de Lenguaje de Gran tamaño en español) acceder a información externa (más allá de con los datos que ha sido entrenada) antes de generar una respuesta, reduciendo en el proceso las alucinaciones de los LLMs [25].

Los LLMs son modelos de aprendizaje profundo que han sido preentrenados con enormes cantidades de datos y que realizan numerosas tareas, como resumir un texto, responder preguntas o traducir a otro idioma, entre otras. No obstante, existen dos problemas principales a los que se enfrentan estos modelos. El primero es la ausencia de una fuente a la hora de responder a una consulta, de forma que el usuario pueda consultarla para asegurarse de su veracidad y obtener más información en relación con esta. En segundo lugar, se encuentra la desactualización de la respuesta generada por el modelo, ya que puede haber sido entrenado hace tiempo con datos que han quedado obsoletos con el tiempo por el descubrimiento de nueva información [26].

RAG soluciona estos problemas al permitir acceder a información externa de internet o almacenada en una base de datos. De esta forma, la información a la que accede el modelo puede mantenerse actualizada sin necesidad de volver a entrenarlo, así como especificar las fuentes de donde proviene esta [25].

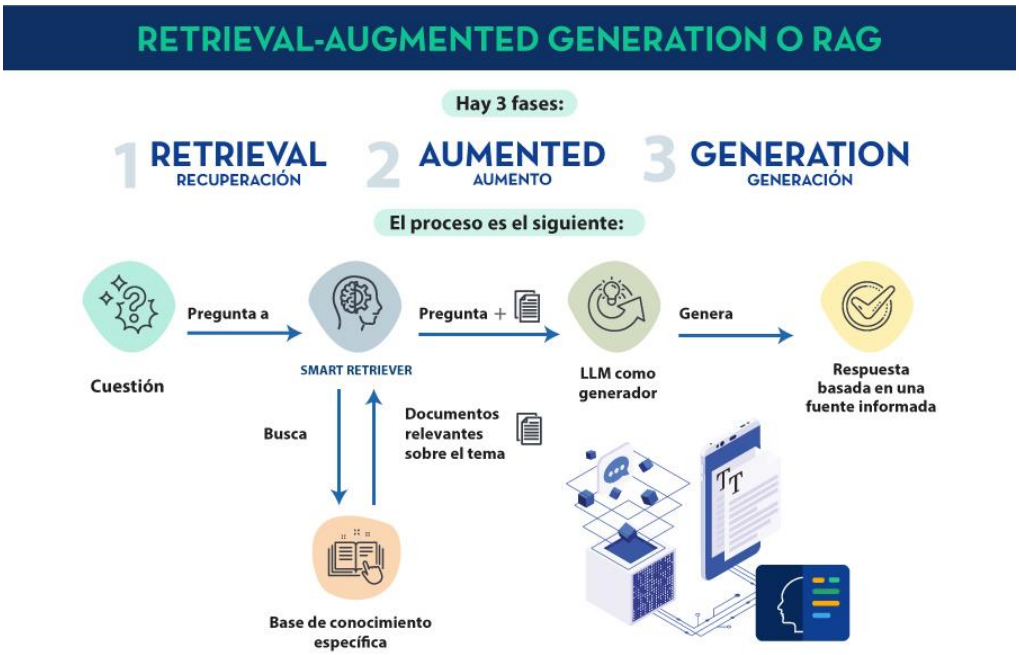

Figura 3.5: Fases del modelo RAG [25]

Por lo tanto, debido a las ventajas que proporcionan y a los problemas de los LLMs que solucionan, RAG constituye una tecnología muy beneficiosa para el campo de la IA generativa. Este suceso se puede observar en muchas empresas tecnológicas multinacionales que están adoptando este enfoque para sus LLMs. Por ejemplo, IBM está usando actualmente la arquitectura RAG para asegurarse que sus chatbots internos acceden a información que puede ser verificada y confiada [27].

A continuación, se detallan las distintas fases que componen a un modelo RAG, especificando las técnicas utilizadas y el flujo de la información para cada una de ellas.

3.5.1 Preparación de datos

Esta fase hace referencia a la inserción de datos e información en la base de datos, la cual será accedida posteriormente por el LLM para poder responder a las consultas del usuario. Aunque esta fase no forma parte directamente del modelo RAG, resulta un proceso fundamental en este ya que permite añadir y actualizar la información del modelo sin necesidad de entrenarlo de nuevo.

En primer lugar, se extrae la información de las fuentes de datos sin procesar. Para ello, existen distintas técnicas y algoritmos dependiendo de la naturaleza de la fuente de la que queremos obtener los datos. Por ejemplo, puede tratarse de información contenida un archivo pdf, para lo cual se utilizan extractores de datos, como la API de Adobe PDF extract. Por otro lado, si se trata de una imagen o un documento escaneado, se puede utilizar el Reconocimiento Óptico de Caracteres (ROC), que es el proceso por el cual se convierte una imagen de texto en un formato de texto que pueden leer las máquinas [28]. En la actualidad, existen algunas herramientas que nos permiten realizar esta tarea, como son los servicios Amazon Textract y Amazon Rekognition que ofrece AWS. Por otra parte, si se quiere extraer información de una página web, se utiliza comúnmente el “web scraping”, que es el proceso de extracción de contenido y datos de una página web de forma automática o manual.

Una vez realizada la extracción de información, mediante el uso de algunas de las técnicas mencionadas u otro tipo de métodos de recolección de datos, se realiza el chunking, que consiste en la división del texto en fragmentos más pequeños (chunks) para facilitar su procesamiento. Está técnica bien implementada permitirá al LLM recuperar partes de información con mayor precisión, ya que están más relacionadas con la consulta, y con un menor coste computacional [29].

En la actualidad, existen distintas estrategias de “chunking” dependiendo del tamaño de los fragmentos en los que se divide el texto (“chunk size”), es decir, el número de tokens en cada fragmento, y la superposición de estos fragmentos (“chunk overlap”). A continuación, se especifican algunos de los métodos de “chunking” más relevantes:

División de tamaño fijo (Fixed-size chunking): Este es el método más directo y simple de “chunking”, ya que tan solo cosiste en especificar el tamaño y la superposición de los fragmentos. No obstante, al no tener en cuenta ni el contenido de los fragmentos ni la estructura del texto, este método da lugar en muchas ocasiones a fragmentos sin sentido e incluso con palabras cortadas sin terminar. No obstante, la división de tamaño fijo tiene un coste computacional bajo y no requiero el uso de ninguna librería de PLN [30].

```
One of the most important things I didn't understand about the world when
I was a child is the degree to which the returns for performance are
superlinear.
```

Figura 3.6: Ejemplo de división de tamaño fijo con un tamaño de fragmentos de 25 tokens y sin superposición

División recursiva por caracteres (Recursive chunking): En este método se recorre de forma recursiva el texto y lo va dividiendo en fragmentos cada vez más pequeños hasta que se obtienen del tamaño que se busca. El funcionamiento de este método consiste en la especificación de una lista de delimitadores (comúnmente ["\n\n", "\n", " ", ""]) y un tamaño para los fragmentos. A continuación, se va separando el texto según el primer delimitador de la lista (normalmente en párrafos), y, en caso de que alguno de los fragmentos obtenidos supere el tamaño especificado, se divide este según el segundo parámetro de lista. De esta forma, procede el algoritmo de forma recursiva. A diferencia de la división de tamaño fijo, este método si tiene en cuenta la estructura del documento, por lo que nos proporciona fragmentos que si mantienen el significado [30] [31].

```
One of the most important things I didn't understand about the world when
I was a child is the degree to which the returns for performance are
superlinear.

Teachers and coaches implicitly told us the returns were linear. "You get
out," I heard a thousand times, "what you put in." They meant well, but
this is rarely true. If your product is only half as good as your
competitor's, you don't get half as many customers. You get no customers,
and you go out of business.
```

Figura 3.7: Ejemplo de división recursiva por caracteres con tamaño de fragmentos de 160 tokens

Fragmentación semántica (Semantic chunking): este método consiste en la división del texto en distintos fragmentos en función de la compresión semántica. Este enfoque supone una gran ventaja sobre las otras alternativas al garantizar que los fragmentos conserven la coherencia semántica. No obstante, supone un coste computacional considerablemente superior en comparación [30] [31].

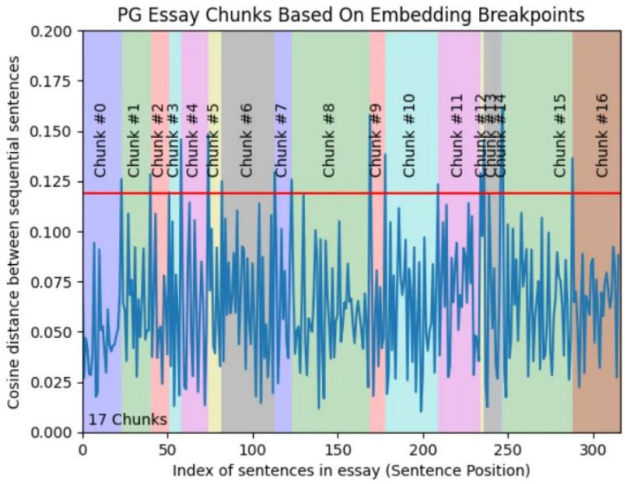

Figura 3.8: Ejemplo de similitud de frases dentro de cada fragmento al usar fragmentación semántica [32]

Finalmente, se calculan los embeddings de los distintos fragmentos utilizando un modelo, de esta forma el LLM podrá entender y procesar la información. Una vez generados los embeddings, se almacenan en una base de datos de vectores.

3.5.2 Recuperación de información relevante

Cuando el usuario hace una consulta, se emplea un modelo (el mismo que se ha empleado en la fase de preparación de datos) para calcular el embedding de esta. A continuación, se pueden emplear distintos métodos para obtener los datos más relevantes de la base de datos vectorial con el fin de enviarlos al LLM. Por ejemplo, uno de los métodos más comunes consiste en calcular la similitud entre la consulta y los textos o documentos indexados mediante la función similitud coseno.

Una vez recuperada la información con mayor relevancia, el modelo RAG la agrega a la consulta del usuario y se envía al LLM para que genere la respuesta.

3.5.3 Generación de la respuesta del LLM

El LLM recibe la consulta aumentada con la información relevante y genera una respuesta en lenguaje natural, basada no solo en sus datos de entrenamiento genéricos, sino en la información contenida en los documentos que han sido procesados e indexados en la base de datos. Al utilizar un modelo RAG se consiguen obtener respuestas con mejor precisión y mayor sentido y contexto, ya que se ha utilizado la información complementaria que se ha obtenido en la fase de recuperación.

4 Desarrollo del Buscador Semántico

Para la implementación del buscador semántico para textos científicos se ha desarrollado un proyecto en lenguaje de programación Python (versión 3.10.11). Respecto al entorno de desarrollo, se ha utilizado Visual Studio Code, debido a la rapidez, facilidad de uso, y el terminal integrado que incluye, entre otras razones. Asimismo, se ha utilizado Github Desktop para poder ir subiendo a Github las distintas versiones y actualizaciones del proyecto a lo largo de su realización. Se puede encontrar el repositorio del proyecto en el siguiente enlace: https://github.com/Miguel28021/Semantic-Searcher.git

El sistema tomará como input un archivo de tipo RIS que referencia los documentos que utilizará el buscador, y como output los documentos más relevantes que se muestran como respuesta de una consulta del usuario. El principal enfoque adaptado para el funcionamiento de este sistema es la división en tres módulos diferentes. El primer módulo se encarga de la extracción y procesamiento de información del archivo RIS para la creación del Corpus, que es una colección estructurada de los documentos que utilizará el sistema. El segundo modulo consiste en la generación de embeddings a partir del Corpus, mediante el uso de modelos de lenguaje preentrenados, y el almacenamiento de estos en una base de datos semántica. Finalmente, el último módulo consiste en una aplicación web que actúa como buscador semántico científico y que, a partir de una consulta, busca en la base de datos aquellos documentos más relevantes y los muestra al usuario.

A continuación, se muestra un diagrama en el que se pueden observar los distintos componentes del sistema y como interactúan entre ellos:

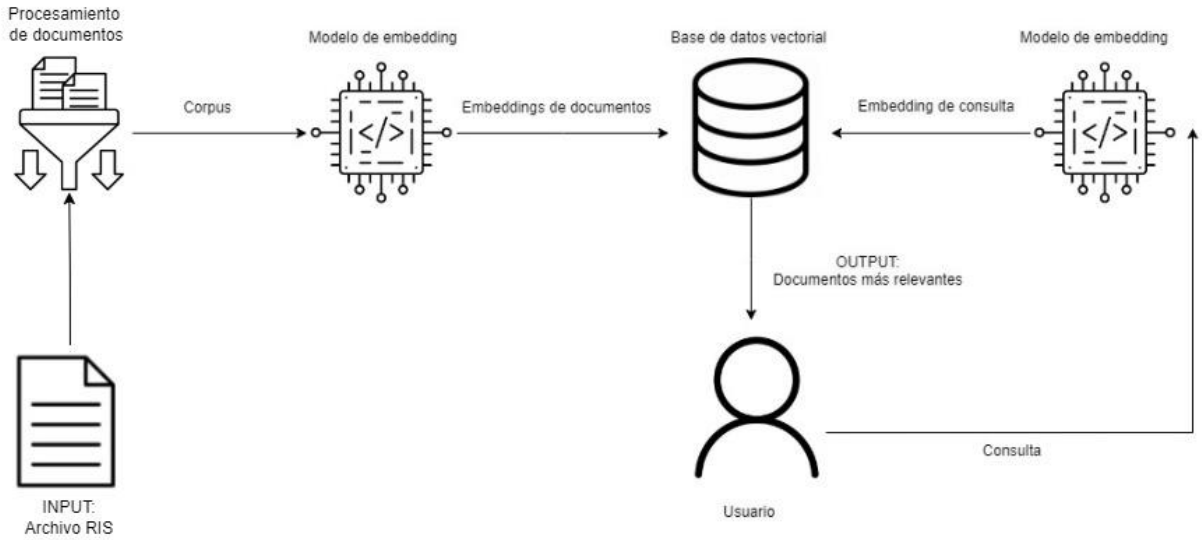

Figura 4.1: Diagrama de sistema del buscador semántico

Entre las principales tecnologías utilizadas para la implementación del sistema está el framework Sentence Transformers, de donde se han seleccionado y obtenido los modelos de lenguaje para realizar los sentence embeddings. Respecto a la base de datos, se ha utilizado Chroma DB, que es la base de datos vectorial donde se almacenan los embeddings y los metadatos de los documentos, y que ofrece herramientas que facilitan la recuperación de información. Por último, se ha utilizado el framework Streamlit para la realización de la aplicación web, ya que se centra en aplicaciones de IA y ofrece múltiples herramientas (widgets) útiles para la representación de datos.

4.1 Procesamiento de textos científicos

Como se ha mencionado anteriormente, se utilizará el archivo "Articles.ris" para obtener la información de los 5797 artículos científicos que este recopila. Entre los datos que se almacenan en este archivo para cada artículo están el tipo de publicación (artículo de revista, libro, sección de libro, acta de conferencia ...), el título, autor, fecha, y abstract, entre otra información. Además, en el formato RIS, cada dato va asociado a una etiqueta específica. En el caso de la información que más nos interesa extraer, el tipo de publicación va marcado con la etiqueta "TY", el título con "TI", el año de publicación con "Y1", autores con "AU" y el abstract con "AB".

```
TY  - JOUR
AN  - rayyan-664554228
TI  - A fingerprints based molecular property prediction method using the BERT model
Y1  - 2022
Y2  - 10
Y3  - 21
T2  - JOURNAL OF CHEMINFORMATICS
SN  - 1758-2946
VL  - 14
IS  - 1
AU  - Wen, Naifeng
AU  - Liu, Guanqun
AU  - Zhang, Jie
AU  - Zhang, Rubo
AU  - Fu, Yating
AU  - Han, Xu
CY  - ["Dalian Minzu Univ, Sch Mech & Elect Engn, Dalian, Peoples R China" ...
AB  - Molecular property prediction (MPP) is vital in drug discovery and drug reposition...
N1  - RAYYAN-INCLUSION: {"Andrea"=>"Excluded"} | RAYYAN-EXCLUSION-REASONS: No DR/biological activity
DO  - 10.1186/s13321-022-00650-3
ER  -
```

Figura 4.2: Ejemplo de referencia a un artículo en fichero "Articles.ris"

Cabe destacar que, debido a que los documentos proceden del campo de la biomedicina, el buscador semántico estará especializado en responder consultas de esta naturaleza. No obstante, la aplicación también permite seleccionar un archivo RIS propio, de forma que el buscador utilice los artículos referenciados en este para responder a las consultas.

En relación con la extracción de información, para cargar y obtener la información de los archivos con formato RIS se utiliza la librería de Python rispy. Para la elaboración del corpus solo se utilizan aquellos archivos referenciados que incluyen un abstract, ya que es la única información que utilizará el modelo para generar los embeddings. Por lo tanto, de los 5797 artículos científicos, solo se utilizan y se almacenan en la base de datos 2152 artículos. Por otro lado, el resto de información que se extrae para cada artículo, que son el título, tipo de publicación, año, autores, y la url, se almacenarán en los metadatos de cada artículo en la base de datos.

4.2 Generación y almacenamiento de embeddings

Una vez obtenido el corpus, que constituye el conjunto de documentos que se usarán en el buscador semántico, se utilizará un modelo para calcular los embeddings a partir de los abstracts de cada documento. Para poder cargar y utilizar estos modelos, se utiliza la librería Sentence Transformers.

Respecto a los distintos modelos de embeddings disponibles, existen unos determinados factores que los diferencian y que se han tenido en consideración a la hora de su elección. Entre ellos, se encuentran el origen de los datos de entrenamiento, la dimensión de los embeddings que genera, la función de pooling utilizada e incluso el modelo Transformer que utiliza internamente el modelo.

En cuanto a la selección de modelos para la aplicación, se han utilizado 3 modelos diferentes. Los modelos seleccionados han sido entrenados con datos provenientes del campo de la biomedicina, debido al origen de los documentos que se usan y al enfoque pretendido para el buscador. De esta forma, a la hora de generar los sentence embeddings, estos captarán mejor el significado y la terminología empleada en los textos, así como se podrá calcular con mayor precisión la similitud entre distintos vectores. Además, se han elegido modelos con distinta dimensión de embeddings, con el objetivo de investigar el impacto de este parámetro en la calidad de los embeddings y en la relevancia de los resultados.

A continuación, se muestran los nombres y características importantes de los modelos seleccionados, los cuales han sido obtenidos desde el repositorio de modelos de HuggingFace. Asimismo, se muestra una representación visual de los embeddings generados por cada uno para diez abstracts extraídos del archivo “Articles.ris”. Para ello, los embeddings han sido reescalados a dos dimensiones mediante la aplicación de PCA.

- BioS-MiniLM [33]: modelo Sentence Transformers que genera embeddings de 384 dimensiones. Además, utiliza un modelo BERT como Transformer, una capa de mean-pooling y una función de pérdida "MultipleNegativesRankingLoss".

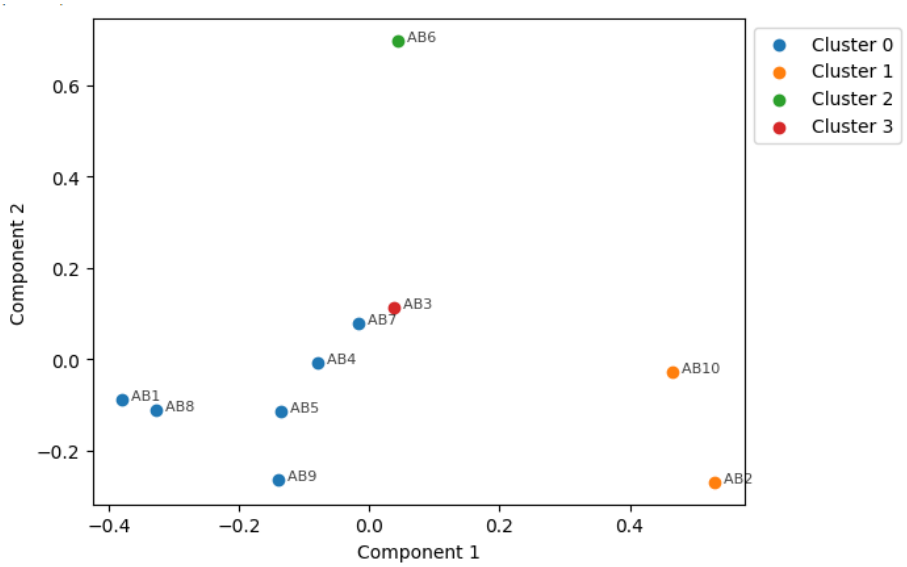

Figura 4.3: Representación de abstracts utilizando BioS-MiniLM

- BioLORD-2023 [34]: modelo Sentence Transformers que genera embeddings de 768 dimensiones. Asimismo, utiliza una capa de mean-pooling y ha sido ajustado("fine-tuned") en el campo de la biomedicina con un "dataset" propio.

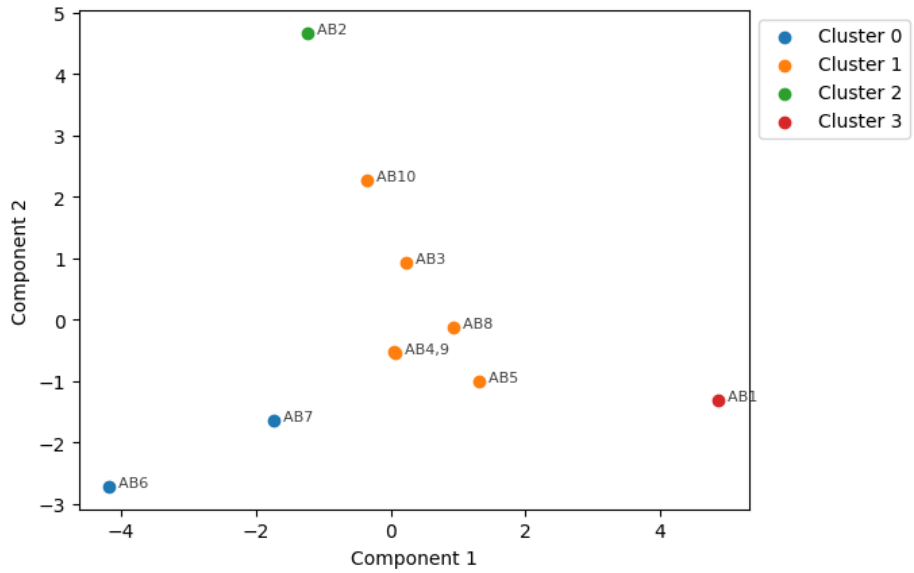

Figura 4.4: Representación de abstracts utilizando BioLORD-2023

- S-BioELECTRA [35]: modelo Sentence Transformers que genera embeddings de 768 dimensiones. Además, utiliza un modelo propio (ElectraModel) como Transformer, una capa de mean-pooling y una función de pérdida "MultipleNegativesRankingLoss", al igual que el modelo BioS-MiniLM.
-

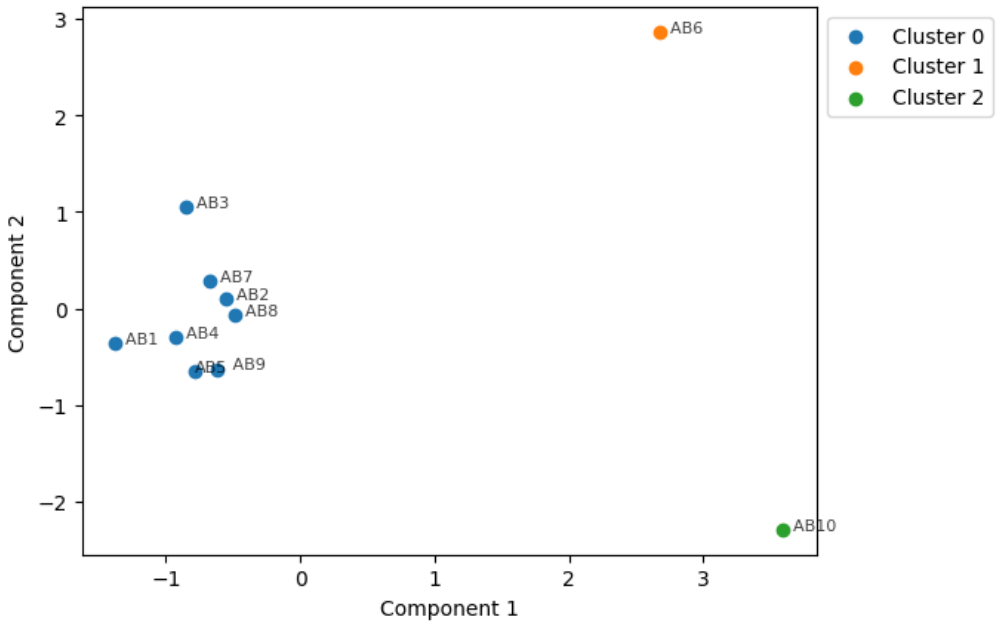

Figura 4.5: Representación de abstracts utilizando S-BioELECTRA

Una vez generados los embeddings, estos se almacenan y se indexan de manera eficiente en una base de datos vectorial ChromaDB. Asimismo, se almacenan los metadatos correspondientes a cada artículo, que se utilizarán posteriormente para la implementación de filtros y para la representación de resultados en la aplicación web.

4.3 Aplicación web

Para la implementación de la aplicación web se ha utilizado el framework de Python Streamlit. A continuación, se muestra las funcionalidades que ofrece y como actúa internamente el sistema en respuesta.

En primer lugar, la aplicación permite decidir si el buscador semántico va a utilizar el archivo "articles.ris" o si va a utilizar un archivo propio con formato RIS. No obstante, cabe destacar que la aplicación ofrecerá mejores resultados si el archivo propio elegido referencia artículos del campo de la biomedicina en vez de artículos científicos genéricos. Esto se debe a que, como ha sido

mencionado con anterioridad, los modelos utilizados han sido entrenados/ajustados con datos del campo de la biomedicina.

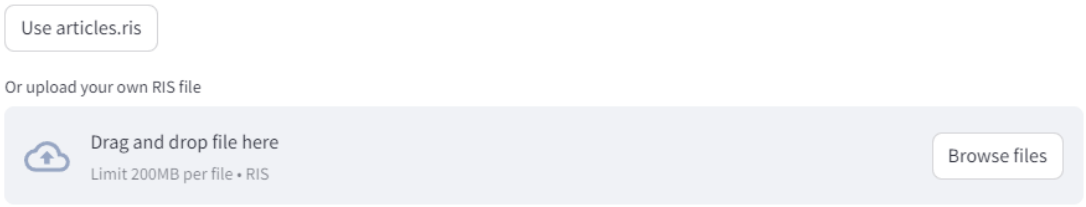

Figura 4.6: Selección de archivo RIS en la aplicación

A continuación, la aplicación permite seleccionar cuál de los tres modelos disponibles se va a utilizar para la generación de embeddings de los documentos y consultas.

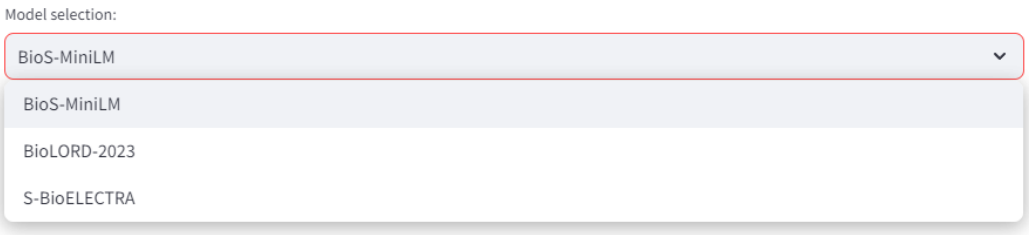

Figura 4.7: Selección de modelo en la aplicación

Tras el proceso de carga del modelo y generación de embeddings, se presenta la pantalla principal de la aplicación web, que es la que se introduce el motor de búsqueda en el que el usuario realizará las consultas. Para obtener los artículos más relevantes, el sistema pasa la consulta por el modelo que se ha utilizado para generar los embeddings del documento, y luego se calcula la similitud entre la consulta y los documentos almacenados en ChromaDB mediante la aplicación de la norma L2 al cuadrado.

La ecuación de la norma L2 al cuadrado (o distancia euclidiana al cuadrado) viene dada por la siguiente fórmula:

$$d = \sum(Ai - Bi)^2$$

(4.1)

En la ecuación (4.1), d representa la distancia entre dos vectores A y B, donde Ai y Bi representan el elemento en la posición i del vector.

What are the most advanced computational methods used for drug repurposing?

Distance	Type	Title	Year	Authors	UI
0.3223	JOUR	An Analytical Review of Computational Drug Repurposing.	2,021	Sadeghi SSKeyvanpour MR	ht
0.3643	JOUR	Computational Drug Repurposing: Classification of the Research Opportunities and C	2,020	Sadeghi, Seyedeh ShaghayeghKeyvanpour, Mohammad Reza	ht
0.3736	CHAP	Chapter 8 Using Artificial Intelligence for Drug Repurposing	2,022	Bender, A.	ht
0.4007	JOUR	Computational Methods for Drug Repurposing.	2,022	Rapicavoli RVAlaimo SFerro APulvirenti A	ht
0.4364	JOUR	ACID: a free tool for drug repurposing using consensus inverse docking strategy.	2,019	Wang FWu FXLi CZJia CYSu SWHao GFYang GF	ht
0.4413	CHAP	Drug Repurposing for COVID-19 using Computational Methods	2,023	Prakash, OmKhan, Feroz	ht
0.4523	JOUR	The Drug Repurposing Encyclopedia (DRE): a web server for systematic drug repurpo	2,023	Li, XuexinPan, LuSanchez-Burgos, LauraHühn, DanielaFernandez-Capetillo, Oscar	ht
0.4527	JOUR	A data-driven methodology towards evaluating the potential of drug repurposing hyp	2,021	Prieto Santamaría LUgarte Carro EDíaz Uzquiano MMenasalvas Ruiz EPérez Gallardo '	ht
0.4685	JOUR	Drug Repurposing Using Biological Networks	2,021	Somolinos, Francisco JavierLeón, CarlosGuerrero-Aspizua, Sara	ht
0.4736	JOUR	Implementation of a Pipeline Using Disease-Disease Associations for Computational	2,019	Balasundaram PKanagavelu RJames NMaiti SVeerappapillai SKaruppaswamy R	ht
0.4821	JOUR	Drug repurposing improvement using a novel data integration framework based on t	2,021	Lakizadeh, AmirHassan Mir-Ashrafi, Sayed Mohammad	ht
0.4878	CHAP	Chapter 9 - Strategies for drug repurposing	2,023	Vema, AparnaKalle, Arunasree M.Nagaraju, Ganji PurnachandraAmouda, Venkatesan	ht
0.5054	CHAP	Chapter 11 - Drug Repurposing From Transcriptome Data: Methods and Applications	2,019	Toro-Domínguez, DanielAlarcón-Riquelme, Marta E.Carmona-Sáez, PedroRoy, Kunal	ht
0.5092	CHAP	Chapter 24 - In Silico Databases and Tools for Drug Repurposing	2,019	Serçinoğlu, OnurSarica, Pemra OzbekRoy, Kunal	ht
0.5103	JOUR	NerLTR-DTA: drug-target binding affinity prediction based on neighbor relationship a	2,022	Ru XYe XSakurai TZou Q	ht
0.5107	JOUR	Drug Repurposing: a Shortcut to New Biological Entities.	2,022	Rao NPoojari TPoojary CSande RSawant S	ht

Figura 4.8: Motor de búsqueda semántica en la aplicación

Para la representación de los documentos que se muestran como resultado, se ha utilizado un dataframe. Entre la información que se muestra para cada documento están la distancia entre consulta y documento, tipo de publicación, título, año, autores, url y abstract.

Asimismo, la aplicación incorpora una serie de filtros que permiten filtrar los documentos que se muestran ante una consulta según tipo de publicación (artículo de revista, libro, capítulo de libro o acta de conferencia), fecha de publicación, título y autor.

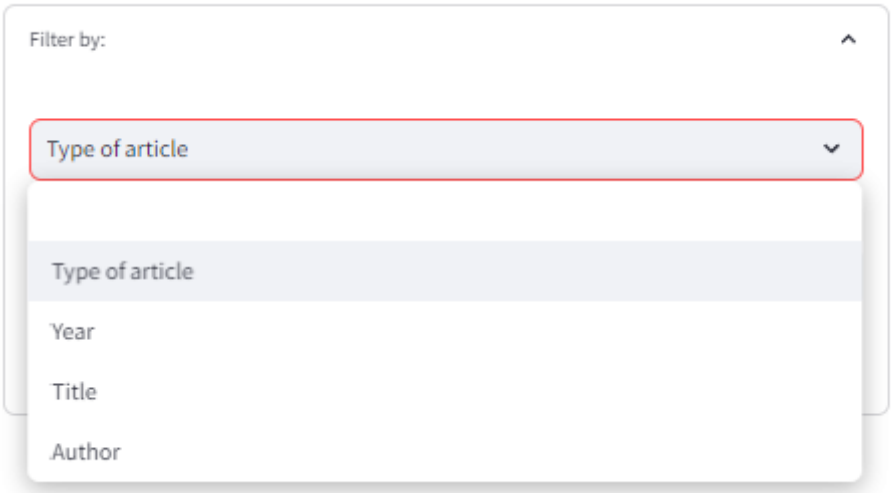

Figura 4.9: Selección de tipo de filtro en la aplicación

4.4 Métodos de evaluación

Para comprobar el correcto funcionamiento del sistema y certificar la calidad de los resultados, se ha llevado a cabo un proceso de evaluación. Para ello, se han seleccionado 4 consultas para cada uno de los 3 modelos seleccionados (BioS-MiniLM, BioLORD-2023 y S-BioELECTRA), con el fin de observar la relevancia de los mejores documentos que devuelve el sistema para cada modelo según la consulta. Para ello, se utiliza la métrica de evaluación "Precision at X" (P@X), utilizada en Búsqueda y Recuperación de Información, que permite evaluar la precisión de un sistema de recuperación.

La métrica de evaluación "Precision at X" viene dada por la siguiente fórmula:

$$P@X = (\text{N.º de documentos relevantes entre mejores X}) \ / \ X \qquad (4.2)$$

En la ecuación (4.2), X representa el número de documentos más relevantes que se utilizan para la evaluación. De forma más específica, esta métrica calcula la proporción de documentos relevantes entre los mejores X documentos devueltos.

Para la evaluación del buscador semántico se han utilizado P@1 y P@5. Asimismo, los evaluadores han realizado comentarios remarcando irregularidades o destacando características de los documentos devueltos.

Las consultas han sido elaboradas con el fin de cubrir distintos aspectos y tópicos del "drug repurposing" y son las siguientes:

- What are the most advanced **computational methods** used for drug repurposing? (¿Cuáles son los métodos computacionales más avanzados utilizados para reposicionamiento de medicamentos?)

- What role does **molecular modeling and simulation** play in drug Discovery? (¿Qué papel juega la modelización y simulación molecular en el reposicionamiento de medicamentos?)

- What are the current advancements in **machine learning** techniques applied to drug repurposing? (¿Cuáles son los avances actuales en técnicas de "machine learning" utilizadas en el reposicionamiento de medicamentos?)

- Show me some **reviews** on natural product-based drug discovery (Muéstrame algunas "reviews" sobre el descubrimiento de fármacos basados en productos naturales).

5 Evaluación del Sistema

En este apartado se especifican y se comentan los resultados obtenidos para la evaluación llevada a cabo, como ha sido mencionado anteriormente, con ayuda de Álvaro García Barragán, científico informático e investigador especializado en "machine learning", pero no experto en reposicionamiento de medicamentos, y Andrea Álvarez Pérez, ingeniera biotecnológica, investigadora y experta en reposicionamiento de medicamentos.

Query 1: What are the most advanced **computational methods** used for drug repurposing? (**Consulta 1**: ¿Cuáles son los **métodos computacionales** más avanzados utilizados para reposicionamiento de medicamentos?)

	Álvaro		**Andrea**	
Modelo	**P@1**	**P@5**	**P@1**	**P@5**
BioS-MiniLM	1	1	1	0.60
BioLORD-2023	1	1	1	0.60
S-BioELECTRA	1	1	0	0.60

Como se puede observar en las puntuaciones, los resultados obtenidos para esta consulta son muy positivos para los tres modelos. No obstante, el primer documento devuelto por el modelo S-BioELECTRA ofrece recursos y herramientas útiles, pero no corresponde a una búsqueda de metodologías.

Por otro lado, cabe destacar que el primer documento devuelto al usar BioS-MiniLM coincide con el segundo devuelto al usar S-BioELECTRA.

Query 2: What role does **molecular modeling and simulation** play in drug Discovery? (**Consulta 2**: ¿Qué papel juega la **modelización y simulación** molecular en el reposicionamiento de medicamentos?)

	Álvaro		**Andrea**	
Modelo	**P@1**	**P@5**	**P@1**	**P@5**
BioS-MiniLM	1	0.80	1	0.40
BioLORD-2023	1	1	1	0.40
S-BioELECTRA	1	1	0	0.40

Como se puede observar en las puntuaciones, los resultados obtenidos para esta consulta son positivos para los tres modelos. Sin embargo, en el caso del modelo S-BioElectra, el primer documento se centra en "virtual screening" y "ligand based- ", pero se define como una review sobre métodos computacionales de AI, por lo que, aunque esté relacionado con la consulta, carece de verdadera relevancia para responderla.

En este caso no se dan ninguna coincidencia entre los artículos devuelto al usar cada modelo.

Query 3: What are the current advancements in **machine learning** techniques applied to drug repurposing? **(Consulta 3**: ¿Cuáles son los avances actuales en técnicas de "**machine learning"** utilizadas en el reposicionamiento de medicamentos?)

	Álvaro		**Andrea**	
Modelo	**P@1**	**P@5**	**P@1**	**P@5**
BioS-MiniLM	1	0.80	1	0.60
BioLORD-2023	1	1	1	0.80
S-BioELECTRA	0	0.80	0	0.40

Se obtienen resultados muy positivos con los modelos S-BioELECTRA y BioS-MiniLM. No obstante, para el S-BioELECTRA, ambos evaluadores han coincidido en la falta de relevancia del primer resultado devuelto, ya que el artículo no trata el tema de "machine learning",

Por otro lado, al igual que con la segunda consulta, no hay coincidencias entre los documentos devueltos por cada modelo.

Query 4: Show me some **reviews** on natural product-based drug Discovery (**Consulta 4**: Muéstrame algunas **reviews** sobre el descubrimiento de fármacos basados en productos naturales)

	Álvaro		**Andrea**	
Modelo	**P@1**	**P@5**	**P@1**	**P@5**
BioS-MiniLM	1	0.20	0	0.40
BioLORD-2023	1	0.40	0	0.20
S-BioELECTRA	1	0.60	1	0.20

Como se puede observar, para la última consulta no se obtienen tan buenos resultados como para las otras consultas. El principal problema con el primer documento devuelto por los modelos BioS-MiniLM y BioLORD-2023, y que también lo devuelve S-BioELECTRA como tercera opción, es que, por muy completo que este, se trata de una metodología, y no de una review.

Los peores resultados obtenidos para esta consulta se deben probablemente a la ausencia en la base de datos de documentos y, más específicamente, de reviews, que tratan el descubrimiento de fármacos basados en productos naturales.

Para concluir la evaluación, cabe destacar que con los tres modelos se han obtenido buenos resultados que han sido relativamente similares, pese a las diferencias en sus arquitecturas. Los modelos BioS-MiniLM y BioLORD-2023 han obtenido unos valores promedios de P@5 = 0.6 y P@5 = 0.675, respectivamente, mientras que para el modelo S-BioELECTRA se ha obtenido un promedio de P@5 = 0.625, pese haber sido el modelo con el que peor P@1 promedio se ha obtenido.

6 Conclusiones

A continuación, se exponen las distintas conclusiones que han sido extraídas una vez finalizado el TFG:

1. Se han cumplido todos los objetivos propuestos en la fase inicial del proyecto. En primer lugar, se ha cumplido la integración de modelos de lenguajes para búsqueda semántica, ya que el sistema utiliza tres modelos diferentes para la generación de embeddings. En segundo lugar, se ha llevado a cabo el manejo de bases de datos semánticas mediante la implementación de una base de datos ChromaDB, entendiendo como almacenar y recuperar la información, como indexa internamente los vectores, y la función de similitud que utiliza. Además, se ha desarrollado una aplicación web para el sistema que permite realizar búsquedas semánticas y filtrar la información que se muestra en pantalla. Por último, se ha cumplido la formación en manejo de datos en Python, así como en manejo de librerías y herramientas PLN.

2. Los resultados obtenidos por el buscador semántico han sido muy buenos, como se ha podido ver en la evaluación, garantizando la efectividad del buscador semántico a la hora de responder consultas del campo de la biomedicina.

3. El proceso de investigación me ha permitido descubrir la importancia de los embeddings para la búsqueda semántica y otras aplicaciones del PLN. Asimismo, he podido observar la revolución que supuso la introducción de los modelos Sentence Transformer en el campo, mejorando a un gran nivel la precisión y el tiempo de computación a la hora de generar los sentence embeddings.

7 Líneas futuras

Respecto a la proyección a futuro del proyecto, hay algunas mejoras que se pueden introducir en el sistema:

1. La implementación de un LLM para completar la arquitectura RAG, de forma que, además de la presentación de los documentos más relevantes ante una consulta, se muestre una respuesta directa del LLM a la consulta, la cual ha sido precisamente generada a partir de los cinco documentos más relevantes.

2. La implementación de la técnica de cuantización binaria para los embeddings, con el objetivo de reducir el tamaño de almacenamiento, mejorar la escalabilidad de la aplicación, y disminuir el tiempo de generación de emebeddings, que es una de las principales limitaciones del sistema.

3. La introducción de un front-end más atractivo y con más funcionalidades, ya que, aunque el enfoque adaptado es simple y práctico, no resulta muy llamativo a simple vista. Asimismo, se puede implementar funcionalidades adicionales para mejorar la experiencia del usuario, como puede ser sugerencias de autocompletado o la búsqueda en múltiples lenguas (actualmente solo funciona de manera adecuada al realizar consultas en inglés).

4. Analizar el efecto de calcular las distancias entre embeddings haciendo uso de otras funciones de similitud más allá de la norma L2 al cuadrado, como puede ser la distancia coseno.

Bibliografía

[1] DeepJudge (2023) *Keyword search vs Semantic Search: What's the difference and why it matters, LinkedIn*. Available at: https://www.linkedin.com/pulse/keyword-search-vs-semantic-whats-difference-why-matters-deepjudge

[2] Vaswani, A. *et al.* (2023a) *Attention is all you need, arXiv.org*. Available at: https://arxiv.org/abs/1706.03762

[3] Davis, D. (2022) *What does the future hold for Semantic Search in science? Copyright Clearance Center*. Available at: https://www.copyright.com/blog/what-does-the-future-hold-for-semantic-search-in-science/

[4] Lowry, E. (2021) *5 applications of semantic search in scientific research, LinkedIn*. Available at: https://www.linkedin.com/pulse/5-applications-semantic-search-scientific-research-elise-lowry

[5] Collantes, L. (2020) *Farmacovigilancia, Cinfasalud*. Available at: https://cinfasalud.cinfa.com/p/farmacovigilancia/

[6] Fricke, S. (2018) 'Semantic scholar', *Journal of the Medical Library Association*, 106(1). doi:10.5195/jmla.2018.280.

[7] *Semantic scholar: AI-Powered Research Tool for scientific literature* (2024) *Deepgram*. Available at: https://deepgram.com/ai-apps/semantic-scholar

[8] *Consensus AI-Powered Academic Search Engine* (2024) *Consensus*. Available at: https://consensus.app/

[9] *Programa integral de formación* (no date) *Rayyan - Revisiones de literatura | Biblioteca Universidad el Bosque*. Available at: https://biblioteca.unbosque.edu.co/programa-integral-de-formacion-pif-para-la-investigacion/nivel-avanzado/rayyan-revisiones-de#:~:text=Permite%20crear%2C%20gestionar%20y%20compartir,y%20clasificar%20los%20estudios%20relevantes

[10] Espíndola, G. (2023) *¿Qué son los embeddings y cómo se utilizan en la inteligencia artificial con python?, Medium*. Available at: https://gustavo-espindola.medium.com/qu%C3%A9-son-los-embeddings-y-c%C3%B3mo-se-utilizan-en-la-inteligencia-artificial-con-python-45b751ed86a5

[11] Sanjeewa, Y. (2023) *Dense vectors in natural language processing, Medium*. Available at: https://medium.com/@yasindusanjeewa8/dense-vectors-in-natural-language-processing-06818dff5cd7

[12] Tripathi, R. (2023) *What are vector embeddings, Pinecone*. Available at: https://www.pinecone.io/learn/vector-embeddings/

[13] Team, S. (2024) *5 types of word embeddings and example NLP applications, Swimm*. Available at: https://swimm.io/learn/large-language-models/5-types-of-word-embeddings-and-example-nlp-applications

[14] H., A. (2019) *Clasificación de Textos state-of-art con embeddings contextuales, Medium*. Available at: https://afhuertas.medium.com/clasificaci%C3%B3n-de-textos-state-of-art-con-embeddings-contextuales-58cb29767669

[15] Mei, T. (2020) *From static embedding to contextualized embedding, Medium*. Available at: https://ted-mei.medium.com/from-static-embedding-to-contextualized-embedding-fe604886b2bc

[16] *Word2vec : text : tensorflow* (2024) *TensorFlow*. Available at: https://www.tensorflow.org/text/tutorials/word2vec

[17] *What are transformers? - transformers in Artificial Intelligence explained - AWS* (no date) *aws*. Available at: https://aws.amazon.com/what-is/transformers-in-artificial-intelligence/

[18] Suszterova, S. (2022) *What is a Semantic Data Model? GoodData*. Available at: https://www.gooddata.com/blog/what-a-semantic-data-model/

[19] Barrera, T. (2023) *Vector databases explained: The backbone of modern semantic search engines, Airbyte*. Available at: https://airbyte.com/data-engineering-resources/vector-databases

[20] Reimers, N. and Gurevych, I. (2019) 'Sentence-bert: Sentence embeddings using Siamese Bert-Networks', *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* [Preprint]. doi:10.18653/v1/d19-1410

[21] *Sentence transformers: Meanings in disguise* () *Pinecone*. Available at: https://www.pinecone.io/learn/series/nlp/sentence-embeddings/

[22] *Training overview* (2024) *Training Overview - Sentence Transformers documentation*. Available at: https://sbert.net/docs/sentence_transformer/training_overview.html

[23] Espejel, O. (2022) *Train and fine-tune sentence transformers models, Hugging Face – The AI community building the future.* Available at: https://huggingface.co/blog/how-to-train-sentence-transformers

[24] Gallego, J. (2022) *DeBERTa (from the ground up) + 2 approaches, Kaggle*. Available at: https://www.kaggle.com/code/javigallego/deberta-from-the-ground-up-2-approaches

[25] *RAG - Retrieval Augmented Generation: The key that unlocks the door to precision language models* (2024) *Rag - retrieval augmented generation: The key that unlocks the door to precision language models*. Available at: https://datos.gob.es/en/blog/rag-retrieval-augmented-generation-key-unlocks-door-precision-language-models

[26] (2023) *What is Rag? - retrieval-augmented generation explained - AWS*. Available at: https://aws.amazon.com/what-is/retrieval-augmented-generation/

[27] Martineau, K. (2024) *What is retrieval-augmented generation (rag)?, IBM Research*. Available at: https://research.ibm.com/blog/retrieval-augmented-generation-RAG

[28] (2023) *What is OCR? - optical character recognition explained - AWS*. Available at: https://aws.amazon.com/what-is/ocr/

[29] Espíndola, G. (2023) *División Y Superposición de chunks: Comprendiendo El Proceso, Medium*. Available at: https://gustavo-espindola.medium.com/divisi%C3%B3n-y-superposici%C3%B3n-de-chunks-comprendiendo-el-proceso-112816192d08

[30] Schwaber-Cohen, R. (2023) *Chunking Strategies for LLM Applications, Pinecone*. Available at: https://www.pinecone.io/learn/chunking-strategies/

[31] Navez, L. (2024) *A guide to chunking strategies for retrieval augmented generation (RAG), Sagacify*. Available at: https://www.sagacify.com/news/a-guide-to-chunking-strategies-for-retrieval-augmented-generation-rag

[32] Singal, A. k (2024) *Unleashing the power of semantic chunking: A journey with llamaindex, Medium*. Available at: https://ai.gopubby.com/unleashing-the-power-of-semantic-chunking-a-journey-with-llamaindex-767e3499ca73

[33] *MENADSA/BIOS-MiniLM · hugging face* (2023) *menadsa/BioS-MiniLM · Hugging Face*. Available at: https://huggingface.co/menadsa/BioS-MiniLM

[34] *FremyCompany/Biolord-2023 · hugging face* (2024) *FremyCompany/BioLORD-2023 · Hugging Face*. Available at: https://huggingface.co/FremyCompany/BioLORD-2023

[35] *MENADSA/S-BioELECTRA · hugging face* (2023) *menadsa/S-BioELECTRA · Hugging Face*. Available at: https://huggingface.co/menadsa/S-BioELECTRA

Lista de acrónimos

PLN Procesamiento del Lenguaje Natural

CLN Comprensión del Lenguaje Natural

TFG Trabajo de Fin de Grado

IA Inteligencia Artificial

GPT Transformador Generativo Preentrenado (Generative Pre-trained Transformer)

PCA Análisis de Componentes Principales (Principal Component Analysis)

CBoW Continuous Bag of Words

GloVe Global Vectors

BERT Representación de Codificador Bidireccional de Transformadores (Bidirectional Encoder Representations from Transformers)

RAG Generación Mejorada por Recuperación (Retrieval Augmented Generation)

LLM Large Language Model (Modelo de Lenguaje de Gran tamaño)

IBM International Business Machines Corporation

ROC Reconocimiento Óptico de Caracteres

AWS Amazon Web Services

Este documento esta firmado por

	**Firmante**	CN=tfgm.fi.upm.es, OU=CCFI, O=ETS Ingenieros Informaticos - UPM, C=ES
	Fecha/Hora	Mon Jun 03 23:11:25 CEST 2024
	Emisor del Certificado	EMAILADDRESS=camanager@etsiinf.upm.es, CN=CA ETS Ingenieros Informaticos, O=ETS Ingenieros Informaticos - UPM, C=ES
	Numero de Serie	561
	Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)