Universidad de los Andes

Semantic Web
Entrega de Proyecto – Etapa 1

Julián Camilo Mora Valbuena -202012747
Juan Esteban Rodríguez - 202011171
Andrea Isabel Vega Márquez - 202416862

1. Introducción

El proyecto que estamos desarrollando sobre la aplicación de los conceptos y tecnologías de la Web Semántica, nos ha dado la tarea de analizar un dataset específico que contiene información sobre los metadatos de un conjunto de artículos de investigación. Nuestra labor principal consistió en realizar un análisis del contenido de los documentos asociados a estos artículos, con el fin de extraer información adicional.

2. Contenido

Para llevar a cabo esta tarea, hemos empleado algunas fuentes y APIs disponibles, con el objetivo de obtener los metadatos asociados a cada artículo, como el título, autor, año de publicación, entre otros. Además, hemos procurado la descarga de los documentos en formato PDF siempre que ha sido posible.
El proceso llevado a cabo abordó aspectos como la identificación de la información disponible en las fuentes utilizadas, las relaciones entre los datos, la organización de la información desplegada y la homogeneidad de los datos obtenidos. Además, nos enfrentamos a diferentes desafíos y problemas durante este proceso.

3. Fuentes de datos

Las fuentes de datos utilizadas incluyeron un dataset con metadatos de artículos asignados a nuestro grupo y las APIs mencionadas en las consideraciones del proyecto, principalmente Semantic Scholar. Estas fuentes nos proporcionaron información sobre los artículos de investigación.

4. Tecnologías utilizadas

Las herramientas en las que nos apoyamos para llevar a cabo esta primera etapa de proyecto fueron:

Python: Desarrollo de los scripts de extracción, descarga y limpieza de datos.
Requests: Realización de solicitudes HTTP a la API de Semantic Scholar.
Pandas: Manipulación y limpieza de los datos.
JSON: Manejo de archivos JSON generados por las solicitudes a la API.
CSV: Para almacenar la información en este formato.

5. Proceso de extracción y limpieza de datos

Extracción de Datos. Utilizamos Python para interactuar con la API de Semantic Scholar y obtener información adicional de los artículos. Primero nos enfocamos en recorrer todos los datos necesarios y descargarlos en formato JSON. Cada artículo descargado resultaba en la generación de un archivo JSON individual. Al final, se descargaron alrededor de 30,000 artículos. Extrajimos títulos, autores, y otros metadatos relevantes.

Hicimos uso de varios endpoints para realizar solicitudes a la API de Semantic Scholar. Estos endpoints fueron:

1) https://api.semanticscholar.org/graph/v1/paper/search: Este endpoint fue empleado para buscar artículos científicos en la base de datos de Semantic Scholar. Se utilizó para realizar consultas específicas con el objetivo de obtener datos relevantes.

2) https://api.semanticscholar.org/v1/paper/{id}: Este endpoint se utilizó para obtener información detallada de un artículo científico particular a partir de su ID. Proporcionó acceso a metadatos importantes sobre los artículos.

3) https://api.semanticscholar.org/graph/v1/paper/batch?fields=title,isOpenAccess,openAccessPdf: Utilizado para obtener información de múltiples artículos científicos simultáneamente, incluyendo detalles como el título y la URL del PDF de open acces.

Una vez que hemos descargado los metadatos asociados a todos los artículos científicos presentes en la base de datos dispuesta para el proyecto, procedemos a iterar sobre los artículos de referencia de cada uno de ellos. Esta acción nos permite ampliar el conjunto de artículos en nuestro dataset. Dado que un artículo puede tener un gran número de referencias, limitamos el número de referencias a 5, sobre las cuales iteramos por cada uno de ellos. Esto resulta en un aumento del número de artículos en nuestra base de datos en un factor de 5. Posteriormente, este número se verá reducido, ya que no todos cuentan con archivos PDF.

Esta limitación se implementó para evitar una carga excesiva de datos y facilitar el proceso de iteración y descarga de nuevos artículos. Además, este proceso de análisis de referencias pudo haber contribuido a la aparición de datos duplicados en la base de datos. Esto se debe a que algunos de los artículos descargados como parte de las referencias podrían hacer referencia a otros artículos que ya se encontraban en la base de datos. Este problema fue abordado y resuelto mediante la limpieza de datos.

Descarga de Artículos. Desarrollamos un programa en Python que nos permitió descargar los artículos en formato PDF a partir de los IDs obtenidos de Semantic Scholar. Creamos una carpeta para almacenar los PDF descargados y guardamos las rutas de estos archivos en una lista.

Limpieza de Datos. En cuanto a la limpieza de datos, se eliminaron columnas que contenían valores nulos o no aportaban información relevante. Además, se corrigieron problemas de formato en los datos, como la conversión de fechas de tipo FLOAT a INT; además de la eliminación de registros con fechas fuera del rango esperado (1800-2025). También se eliminaron columnas duplicadas y registros con valores nulos. Estas acciones resultaron en la *reducción del conjunto de datos de alrededor de 26,000 a 5,400 registros*.

6. Retos y problemas del proceso

Tamaño de los datos. Surgieron problemas con la gestión de un único archivo JSON que contenía todos los metadatos, ya que su tamaño alcanzaba los 15 GB, lo que dificultaba su manipulación y procesamiento debido a las limitaciones de la memoria RAM. Para abordar este problema, se implementó un nuevo enfoque que iteraba sobre cada archivo JSON individualmente, permitiendo realizar las operaciones necesarias sin cargar el conjunto de datos completo en memoria. Afortunadamente se logró conseguir una máquina virtual con 128 de RAM, factor determinante para conseguir la concentración de los artículos en un CSV. Esto facilitó la gestión de los datos y permitió continuar con las tareas de extracción y análisis.

Descarga de PDFs. Enfrentamos dificultades con la descarga de los PDFs de los artículos. A pesar de disponer de una gran cantidad de IDs de artículos, solo fue posible descargar aquellos que tenían open access y una URL que dirigiera al PDF correspondiente.

Para evitar problemas con la cantidad de solicitudes o ser detectados como bots, se implementó un retraso de un segundo entre cada petición realizada. Esto aseguró que no se superara la tasa permitida de solicitudes. Además, se estableció un tiempo de espera de 5 segundos durante la descarga de los PDFs. Esto ya que, en caso de no poder descargar un PDF dentro de este límite de tiempo, el programa continuara con el siguiente artículo sin quedar trabado. Durante la descarga de los PDFs, especialmente cuando los nombres de los artículos contenían caracteres como barras diagonales, ocasionaba una confusión, reconociendo como ruta de archivo en lugar de nombre de archivo. Este problema se solucionó mediante la corrección de los nombres de los archivos.

Al ejecutar todas las operaciones para todos los artículos simultáneamente, se incrementaba el riesgo de errores y la pérdida de progreso en caso de fallos. Se implementó un enfoque más segmentado para evitar este problema en el futuro.