ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA DE SISTEMAS

ANÁLISIS DE SIMILITUD EN REPRESENTACIÓN DE LA INFORMACIÓN EN LÍNEAS CELULARES

COMPONENTE: ÁRBOL DE DISTANCIA CORRESPONDIENTE A LA REPRESENTACIÓN DE LÍNEAS CELULARES BASADA EN MINERÍA DE TEXTO

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN CIENCIAS DE LA COMPUTACIÓN

HENRY DAVID GUANOLUISA HERRERA

henry.guanoluisa@epn.edu.ec

DIRECTOR: Dr. IVAN MARCELO CARRERA IZURIETA, PhD.

ivan.carrera@epn.edu.ec

DMQ, agosto 2023

CERTIFICACIONES

Yo, **HENRY DAVID GUANOLUISA HERRERA** declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

_____

HENRY DAVID GUANOLUISA HERRERA

Certifico que el presente trabajo de integración curricular fue desarrollado por HENRY DAVID GUANOLUISA HERRERA, bajo mi supervisión.

_____

Dr. IVAN MARCELO CARRERA IZURIETA, PhD

DIRECTOR DE PROYECTO

Certificamos que revisamos el presente trabajo de integración curricular.

_____	_____
NOMBRE_REVISOR1	**NOMBRE_REVISOR2**
REVISOR1 DEL TRABAJO DE INTEGRACIÓN CURRICULAR	**REVISOR2 DEL TRABAJO DE INTEGRACIÓN CURRICULAR**

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el producto resultante del mismo, es público y estará a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

HENRY DAVID GUANOLUISA HERRERA

Dr. IVAN MARCELO CARRERA IZURIETA

DEDICATORIA

Quiero dedicar mi título profesional con todo mi amor y buenos deseos a Dios, quien me ha brindado fuerza, aliento, refugio y sabiduría en cada paso de este camino.

Este título se lo dedico a mi familia, a mis padres Mayra Herrera y Kleber Guanoluisa, que me han visto crecer y han forjado en mí la persona que soy ahora, gracias porque con ustedes he visto lo que debo y lo que no debo hacer.

A mis hermanos, Katherine, Kleber, Omar y Santiago, dedico este hito con profundo cariño, por apoyarme en todo momento, demostrando que juntos podemos superar cualquier circunstancia, ustedes siempre serán mi fuerza, mi motor.

A mi tío Jimmy Herrera quien me ha acompañado y respaldado durante todos estos años, has sido mi figura paternal y tu constante ánimo con frases como "sigue adelante, cholito" me ha motivado a persistir. Este logro es también tuyo.

Principalmente, dedico este logro, resultado de años de esfuerzo y dedicación, a mis entrañables abuelitos Enma y Ángel. Su generosidad al recibirme en su hogar y brindarme su amor y apoyo incondicional, como si fuera su propio hijo, es un tesoro que guardo profundamente. Agradezco por compartir conmigo su amor, sabiduría y energía, especialmente en los momentos más cruciales de mi camino. Sus palabras alentadoras, como "Dale mijito, tú puedes" y "descansa, te vas a enfermar", siempre han sido un faro de guía y fortaleza para mí.

Este tributo no es solo un agradecimiento, sino una promesa de que su compartir de amor y apoyo no quedará sin recompensa. Siempre llevaré su influencia en mi corazón y trabajaré incansablemente para honrar su amor y dedicación en cada paso que tome en mi futuro.

Esta dedicatoria es una amalgama de emociones y recuerdos que me han otorgado la fuerza para seguir adelante. Gracias por creer en mí y por depositar su confianza en cada uno de mis pasos.

AGRADECIMIENTO

Expreso mi más sincero agradecimiento a Dios, por brindarme fortaleza en momentos difíciles, en aquellos momentos en los que me sentía perdido y no tenía a dónde acudir. Agradezco por la salud, el coraje, la sabiduría y la fortaleza que me otorgó para alcanzar este hito.

Mi más sincero reconocimiento va dirigido a la Escuela Politécnica Nacional y a mi facultad, la Facultad de Sistemas, por darme la bienvenida como miembro de la comunidad politécnica. Estoy profundamente agradecido por los años de crecimiento académico y por formarme como profesional.

Quiero expresar un agradecimiento muy especial a mis amigos, y en particular a esos compañeros y profesores que no solo fueron mentores, sino que también se convirtieron en amigos cercanos, brindándome un constante aliento para perseverar en este camino.

Extiendo un profundo agradecimiento a mi mentor, el Ingeniero Iván Carrera, cuya orientación y conocimientos fueron fundamentales para moldear el desarrollo de este proyecto, brindándome dirección y recursos invaluables.

Quiero expresar un sincero reconocimiento a mi querida amiga y compañera sentimental, Joss, quien ha sido un pilar significativo tanto en mi vida académica como personal. Tu apoyo inquebrantable, cariño y motivación han sido fundamentales para alcanzar este hito. Estoy agradecido por las experiencias que hemos compartido, tanto los momentos positivos como los desafíos, con la certeza de que Dios siempre está a nuestro lado.

Agradezco a los Ingenieros y amigos Alexis Miranda y Jonathan Vargas, cuyo conocimiento y apoyo han sido invaluables. Nuestras conversaciones y planes futuros han fortalecido mi camino a través de este proyecto.

A mis amigos Paúl Salazar, David Cardona y Manuel Anchatuña, quienes me han demostrado el valor de la amistad, perseverancia y la determinación, les agradezco de corazón.

A todos aquellos que han sido parte de este recorrido, especialmente a mi familia y mis incansables abuelos, quienes me han inculcado valores de respeto y carácter, extiendo mi más profundo agradecimiento.

¡Gracias a todos!

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

RESUMEN

En la presente investigación, la representación y comparación de líneas celulares tiene un papel sumamente importante en el entendimiento del comportamiento e interacciones. Este estudio implementa un enfoque innovador al representar líneas celulares utilizando técnicas de minería de texto, utilizando análisis de componentes principales (PCA) y Descripción de datos basado en vectores de soporte (SVDD). Los principales objetivos fueron la extracción y procesamiento de información textual de la literatura científica, para luego transformarla en representaciones numéricas y así desarrollar una metodología de agrupación jerárquica.

Para lograr los objetivos del proyecto, se recopiló un conjunto de data de líneas celulares de fuentes como Cellosaurus y PubMed. Se utilizó Python, junto con librerías como pandas y scikit-learn, para el procesamiento, análisis y modelado de la data. La data de texto fue sometida a un preprocesamiento, que incluyó la reducción a raíz y la transformación TF-IDF, lo que arrojó como vectores de características numéricas. PCA se utilizó para la reducción dimensional de la data que al mismo tiempo preserva su variación. SVDD identificó aquellos valores no típicos y las distancias entre líneas celulares que se visualizaron mediante la construcción de un dendograma.

Los resultados revelaron una forma novedosa de representar las líneas celulares, permitiendo la identificación de grupos y similitudes entre diferentes líneas. El dendograma representó visualmente relaciones jerárquicas, mostrando información sobre conjuntos celulares.

PALABRAS CLAVE: Lineas celulares, Minería de texto, PCA, SVDD, agrupación jerárquica.

ABSTRACT

In the current research, the representation and comparison of cell lines play a crucial role in understanding the behavior and interactions of cell lines. This study implements an innovative approach to representing cell lines using text mining techniques, employing Principal Component Analysis (PCA) and Support Vector Data Description (SVDD). The main objectives were the extraction and processing of textual information from scientific literature, followed by its transformation into numerical representations, thereby developing a hierarchical clustering methodology.

To achieve this, a dataset of cell lines was collected from sources such as Cellosaurus and PubMed. Python, along with libraries like pandas and scikit-learn, was employed for data processing, analysis, and modeling. The textual data underwent preprocessing, which included stemming and TF-IDF transformation, resulting in numerical feature vectors. PCA was utilized for dimensional reduction of the data while preserving its variance. SVDD identified atypical values, and the distances between cell lines were visualized through the construction of a dendrogram.

The results revealed an innovative way to represent cell lines, allowing for the identification of groups and similarities among different lines. The dendrogram visually represented hierarchical relationships, providing insights into cell line clusters.

KEYWORDS: Cell lines, Text mining, PCA, SVDD, hierarchical clustering.

1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

La representación de la información es una tarea clave en el desarrollo de algoritmos de Aprendizaje de Máquina. Al representar entidades complejas en un contexto de Inteligencia Artificial, las descripciones deben presentarse de manera que una máquina inteligente pueda obtener nuevas conclusiones al manipular estas representaciones simbólicas. Así, se puede evaluar la representatividad de una expresión a través de medir cómo captura las características de las entidades que representa.

En el campo de aplicación de la Bioinformática y la Informática Médica, las líneas celulares son modelos utilizados para evaluar nuevos fármacos y sus posibles efectos en el organismo humano.

En el presente proyecto se busca evaluar una formar de representar la información asociada a las líneas celulares por medio de la minería de texto. Investigando cómo transformar la información textual en representaciones numéricas que puedan ser procesadas por algoritmos de aprendizaje. Al adentrarse en este desafío, se busca mejorar la comprensión de las características que distinguen las líneas celulares y su relación con las respuestas a diferentes tratamientos. Esta inspección puede ofrecer nuevas perspectivas en el campo de la Bioinformática y contribuir a una evaluación más precisa y efectiva en ámbito de los tratamientos de la investigación científica médica.

1.1 Objetivo general

Construir un árbol de distancia correspondiente a la representación de líneas celulares basada en minería de texto. En este contexto, se entiende por línea celular al conjunto de células que se utilizan para estudiar posibles aplicaciones de fármacos sobre células cancerígenas.

1.2 Objetivos específicos

1. Realizar una investigación bibliográfica científica acerca de “A Representation Method for Cellular Lines based on SVM and Text Mining”.
2. Extraer datos de las bases de datos Cellosaurus y PubMed, recopilando información relevante sobre líneas celulares y artículos científicos.
3. Obtener una representación computacional de líneas celulares a partir de la información de texto que se encuentra en la literatura científica.

4. Utilizar la representación computacional obtenida de las líneas celulares para construir un árbol de distancia que permita visualizar y comparar la similitud entre diferentes líneas celulares.

1.3 Alcance

En el presente proyecto se evaluará la representación computacional de líneas celulares a partir de minería de texto. La información relativa a las líneas celulares se recopila en una base de datos. Se establecen métricas para la representación y formación de un árbol de distancia, y posteriormente realizar una evaluación de la representatividad de los árboles de distancia.

2 MARCO TEÓRICO

2.1 Conceptos teóricos

2.1.1 Líneas celulares

En el contexto de la biología y la bioinformática, las líneas celulares, también conocidas como cultivos celulares, consisten en una agrupación de células de un único tipo, ya sea humano, animal o vegetal, que se mantienen en un entorno controlado [1]. Estas células pueden vivir, reproducirse e incluso proliferar, manifestando propiedades diferenciadoras en un ambiente de cultivo adecuado, como una placa de cultivo de tejidos. Estas células se utilizan para diversos fines de investigación porque pueden proliferar indefinidamente en las condiciones adecuadas. Las líneas celulares se utilizan en investigaciones celulares como una fuente de células uniformes para contrastar con estudios que emplean organismos intactos. Además, las líneas celulares han adquirido una importancia creciente en los campos de la biomedicina y la biotecnología, particularmente en la producción de vacunas y otros productos biológicos [2].

En bioinformática, las líneas celulares se utilizan a menudo como fuente de datos. Por ejemplo:

- Perfiles de expresión génica: los científicos podrían estudiar cómo se expresan los genes en una línea celular particular bajo diversas condiciones o tratamientos. Esto se puede analizar utilizando herramientas bioinformáticas para comprender las respuestas genéticas subyacentes.
- Secuenciación genómica: el ADN de las líneas celulares se puede secuenciar para comprender su composición genética. Esto es especialmente importante para las líneas celulares derivadas de tumores, donde se pueden identificar las mutaciones.
- Proteómica y metabolómica: las proteínas y los metabolitos producidos por una línea celular se pueden estudiar mediante diversas técnicas, y los datos se pueden analizar mediante herramientas bioinformáticas.
- Respuestas a los medicamentos: las líneas celulares se pueden tratar con varios medicamentos para estudiar sus efectos, y los datos resultantes se pueden analizar para comprender los mecanismos de los medicamentos o para identificar posibles objetivos farmacológicos. [2]

2.1.2 Representación computacional de líneas celulares basada en texto

La representación computacional de líneas celulares es de gran importancia dentro de la Bioinformática porque permiten construir modelos computacionales basados en las características de líneas celulares específicas. Estos modelos se pueden utilizar para simular procesos celulares, predecir respuestas celulares a fármacos o comprender mecanismos de enfermedades a nivel celular. También, la representación computacional de las líneas celulares permite el desarrollo de fármacos y medicina personalizada. Las líneas celulares, especialmente las derivadas de tumores se utilizan a menudo en la detección de fármacos. Las representaciones computacionales pueden ayudar a predecir cómo las diferentes líneas celulares (y, por extensión, los diferentes tumores de los pacientes) podrían responder a un fármaco en particular, lo que ayuda en el desarrollo de estrategias de tratamiento personalizadas. [2]

El método de representación computacional de líneas celulares puede basarse en el procesamiento de texto, ya que la literatura científica se considera el repositorio principal del conocimiento biomédico. Las fuentes de datos para muchas bases de datos y portales web suelen ser la literatura publicada, que a menudo se presenta de manera no estructurada. Por lo general, se requiere un esfuerzo manual para limpiar, identificar la literatura relevante y extraer y estructurar la información de acuerdo con las necesidades. Sin embargo, este esfuerzo se vuelve insostenible a medida que la información crece exponencialmente. [2]

En este trabajo se presenta un método para representar líneas celulares basado en SVDD y el procesamiento del texto de la literatura científica, donde se generó un corpus de resúmenes de artículos científicos relacionados con las líneas celulares. [2]

2.1.3 Fuentes de Datos

a. Cellosaurus

Es una fuente de conocimiento sobre líneas celulares que tiene como objetivo generar descripciones utilizadas en la investigación biomédica. Entre sus muchos usos, se destaca la identificación de líneas celulares potencialmente contaminadas o mal identificadas, contribuyendo así a mejorar la calidad de la investigación en las ciencias de la vida.

El campo de aplicación de esta fuente abarca especies de vertebrados e invertebrados, y en la actualidad proporciona información sobre más de 100.000 líneas celulares. Cada línea celular cuenta con descripciones que incluyen el nombre recomendado, sinónimos, número de acceso único, especie, comentarios estructurados, referencias cruzadas, sexo,

edad, categoría, sitos web y referencias de publicaciones. Además, esta información puede descargarse en varios formatos. [3]

b. PubMed

Es una fuente de información gratuita que patrocina la búsqueda y recuperación de literatura biomédica y de ciencias de la vida con el propósito de mejorar la salud, tanto a nivel personal como mundial.

PudMed en su base de datos contiene más de 35 millones de citas y resúmenes de la literatura, esto no incluye textos completos, aunque los enlaces al texto completo generalmente se encuentran cuando están disponibles en otras fuentes como por ejemplo el sitio web del editor o PubMed central (PMC). [4]

2.1.4 Web Scraping

El web scraping, también conocido como *screen scraping*, *web data extraction* o *web harvesting,* se refiere a un conjunto de prácticas o procesos utilizados para extraer de manera automática datos de páginas web en lugar de realizar la extracción manualmente. Esta técnica permite obtener datos relevantes a partir del código HTML de las páginas web y almacenarlos en bases de datos, archivos Excel, JSON o CSV, entre otros formatos. Los datos extraídos mediante *web scraping* pueden ser utilizados posteriormente para su análisis, procesamiento o cualquier otro propósito específico. [5]

2.1.5 Term frequency–Inverse document frequency (TF-IDF)

TF-IDF (Frecuencia de término - frecuencia de documento inverso) es comúnmente utilizado en un esquema de ponderación de términos para la representación de documento de texto como vectores con varios fines, entre ellos la de clasificación, agrupación visualización, recuperación etc. [6]

Esta técnica se encarga de eliminar los términos más comunes, elimina datos ruidosos o menos útiles, pues se concentra en extraer los términos más relevantes del corpus. [7]

2.1.6 Principal Components Analysis PCA

PCA explica la correlación entre los datos que contienen variables como columnas y observaciones como filas, el objetivo del PCA se centra en reducir grandes variables que contienen correlación en un pequeño grupo de variables, a estas variables correlacionadas se las llama componentes principales. Estas componentes principales agrupan y representan las variables originales que mantienen una correlación significativa entre sí. [8]

2.1.7 Support Vector Domain Description SVDD

Support Vector Domain Description (SVDD), el cual se utiliza para identificar valores atípicos en conjuntos de datos. El SVDD se encarga encontrar una esfera o hiperesfera con un radio mínimo (R) y un centro que abarque la mayor parte de los datos. La presencia de valores atípicos (outliers) se aborda mediante la introducción de variables de holgura, las cuales permiten la inclusión de puntos de datos que se encuentran externas de la esfera. El objetivo principal del SVDD es minimizar el volumen de la esfera, de manera que se pueda encapsular la mayor cantidad posible de puntos de datos dentro de ella, mientras se permite cierto grado de flexibilidad para la inclusión de valores atípicos. Esta técnica es especialmente utilizada para detectar patrones inusuales o anomalías en conjuntos de datos y contribuye al análisis de datos y la detección de valores atípicos en diversos campos de investigación. [9]

2.2 Herramientas de desarrollo

2.2.1 Python

Es un lenguaje de programación de alto nivel interpretado y orientado a objetos. Se destaca por su flexibilidad y facilidad de uso debido a su escritura dinámica, lo que lo hace llamativo para el desarrollo de aplicaciones, entre sus ventajas es que cuenta con estructuras de datos integradas, lo que facilita la manipulación y gestión de la información en el código.

Además, Python es precursor de la modularidad y reutilización de código esto gracias a su soporte para paquetes y módulos, lo que permite a los desarrolladores a dividir sus programas en componentes funcionales, de igual forma Python ofrece una extensa biblioteca estándar que contiene una amplia gama de funciones y herramientas para tareas, lo que acorta el proceso de desarrollo y reduce costos de mantenimiento [10].

2.2.2 Pandas

Es una poderosa biblioteca de Python especializada en la manipulación y el análisis de datos. Su principal herramienta es el DataFrame que facilita la exploración, limpieza y procesamiento de datos, permitiendo a los usuarios mejorar el enfoque de manejo de tablas, una de las ventajas más notables de pandas es su gran capacidad para manejar diversas fuentes de datos, lo que incluye diferentes extensiones como son CSV, EXCEL, SQL, PARQUET entre otros, esto facilita la importación y exportación desde y hasta diferentes fuentes de datos.

Pandas es altamente utilizado en tareas de procesamiento de información debido a su versatilidad, porque se pueden desarrollar una gran gama de operaciones como seleccionar subconjuntos específicos de tablas, crear nuevas columna derivadas de otras, calcular estadísticas, realizar remodelaciones de las estructuras de tablas, también cuenta con funcionalidades avanzadas que permiten la combinación y el manejo de datos entre tablas, así mismo proporciona herramientas para el manejo de series de datos temporales y manipulación de datos textuales, lo que amplía su aplicación en campos como el análisis financiero, científico y de datos. [11]

3 METODOLOGÍA

Como se indicó en la sección 1.1, el objetivo de este trabajo es construir un árbol de distancia correspondiente a la representación de líneas celulares basada en minería de texto. Para cumplir con este objetivo, es necesario realizar una representación computacional de las líneas celulares, y posteriormente un análisis de similitud de esta representación.

En el presente trabajo, se empleará la metodología CRISP-DM Cross-Industry Standard Process for Data Mining. Esta metodología está compuesta de los siguientes procesos: (1) entendimiento del negocio, (2) entendimiento de los datos, (3) preparación de los datos, (4) modelado, y (5) evaluación e implementación. Además, se sigue también la metodología descrita en [2], que aborda un método de representación para líneas celulares, el cual permite conocer la caracterización y representación de estas líneas basada en el procesamiento de texto de la literatura científica.

Como primer paso, es necesario consultar y extraer la información relevante de bases de datos que contengan datos sobre líneas celulares y sus características. En este caso, se ha utilizado la base de datos llamada *Cellosaurus*[1], la cual proporciona información acerca de líneas celulares, incluyendo su nombre, sinónimos, especie, entre otros detalles.

En el desarrollo de esta investigación, se empleó la plataforma Anaconda, la cual se basa en el lenguaje de programación Python y se destaca por su amplia adopción en el análisis y procesamiento de datos. A través de la combinación de Anaconda y Jupyter Notebook, se generaron las instrucciones requeridas para el procesamiento de los datos obtenidos, así como para la construcción del árbol de distancias que representa las relaciones entre las líneas celulares. Esta elección de herramientas proporcionó un entorno eficiente y versátil para el manejo de los datos y la generación de resultados pertinentes en esta investigación.

3.1 Preparación de los datos

Para la preparación de los datos se descargó la base de datos desde su sitio web. Posteriormente, se procedió a desarrollar un script que navegó a través de dicho archivo. El objetivo de este script fue construir un dataframe con la información relevante para el tratamiento de los datos.

[1] Disponible en: https://www.cellosaurus.org/

El dataframe resultante se generó con los siguientes campos:

- El campo "línea_celular" se utilizó para proporcionar una identificación clara de cada línea celular tratada en el estudio.
- El campo "sinónimos" se incluyó para mencionar otros nombres que se pueden encontrar en la literatura científica y que están asociados a cada línea celular en particular.
- El campo "Especie" se utilizó para indicar la especie a la que está relacionada cada línea celular. Es importante destacar que puede haber líneas celulares de diferentes especies, como *Mus musculus*, *Homo sapiens*, *Rattus norvegicus*, *Pan troglodytes*, *Drosophila melanogaster*, entre otras.

Cabe mencionar que una vez generado el dataframe se procedió a seleccionar el valor del campo “Especie” igual a “Homo sapiens” dado que solo realizaremos el tratamiento de información a esta especie. Este dataframe se puede observar en la Figura 1.

	linea_celular	sinonimos	Especie
2	CVCL_E548	[#15310-LN, 15310-LN, TER461, TER-461, Ter 461...	Homo sapiens
7	CVCL_E549	[#W7079, #W7079 REM, REMUS, W7079]	Homo sapiens
9	CVCL_VG99	[(L)PC6]	Homo sapiens
17	CVCL_B5B3	[0.5alpha, 0.5 alpha]	Homo sapiens
18	CVCL_E557	[00136, 136]	Homo sapiens
20	CVCL_VG31	[0162D]	Homo sapiens
21	CVCL_VG32	[0165D]	Homo sapiens
22	CVCL_ZW87	[017-PC-A, PC-A]	Homo sapiens
23	CVCL_ZW88	[017-PC-M, PC-M]	Homo sapiens
24	CVCL_ZW89	[017-PC-O, PC-O]	Homo sapiens
25	CVCL_XB90	[0225-02Sp]	Homo sapiens

Figura 1 Dataframe "cell_df"

3.2 API de PUBMED

Para establecer la conexión con la API de PubMed, se realizó el tratamiento del dataframe previamente generado (cell_df). En primer lugar, se seleccionaron los datos relevantes del dataframe, como la columna "línea_celular", "sinónimos" y "especie". Estos datos se utilizaron para construir un data_query, que posteriormente se transformó en un nuevo dataframe llamado df_queries.

El dataframe df_queries contiene la siguiente información:

- El campo "cell_line" se refiere a la línea celular que está siendo tratada en la consulta.
- El campo "query" corresponde a la consulta generada con los posibles nombres de la línea celular referenciada. Esta consulta se utilizará para buscar información adicional en la API de PubMed.
- El campo "Num_syn" indica el número de sinónimos encontrados para la línea celular referenciada. Esto proporciona información sobre la diversidad de nombres asociados a la línea celular en la literatura científica.

Este dataframe se puede observar en la Figura 2 con la información previa generada:

	cell_line	query	num_syn
0	CVCL_E548	CVCL_E548 OR #15310-LN OR 15310-LN OR TER461 O...	9
1	CVCL_E549	CVCL_E549 OR #W7079 OR #W7079 REM OR REMUS OR ...	4
2	CVCL_VG99	CVCL_VG99 OR (L)PC6	1
3	CVCL_B5B3	CVCL_B5B3 OR 0.5alpha OR 0.5 alpha	2
4	CVCL_E557	CVCL_E557 OR 00136 OR 136	2
5	CVCL_VG31	CVCL_VG31 OR 0162D	1
6	CVCL_VG32	CVCL_VG32 OR 0165D	1
7	CVCL_ZW87	CVCL_ZW87 OR 017-PC-A OR PC-A	2
8	CVCL_ZW88	CVCL_ZW88 OR 017-PC-M OR PC-M	2
9	CVCL_ZW89	CVCL_ZW89 OR 017-PC-O OR PC-O	2
10	CVCL_XB90	CVCL_XB90 OR 0225-02Sp	1

Figura 2 Dataframe "df_queries"

Una vez obtenida la información consolidada en un dataframe denominado df_queries, se procedió a la generación de credenciales para establecer la conexión con la base de datos PubMed.

Para llevar a cabo las consultas a la base de datos PubMed, se emplearon los términos de búsqueda y las líneas celulares previamente proporcionadas en el dataframe df_queries. El objetivo de estas consultas fue obtener los identificadores de los artículos científicos relevantes relacionados con las líneas celulares en cuestión. Los identificadores de los artículos encontrados fueron posteriormente guardados en un archivo denominado results.csv como se puede observar en la Figura 3.

```
Entrez.email = 'henry.guanoluisa@epn.edu.ec'

#df_queries = pd.read_csv('queries.csv')
paper_id_list = []
error_paper_id_list = []
for index, row in df_queries.iterrows():
    try:
        string_search = Entrez.read(Entrez.esearch(db="pubmed", term=row['query'] +
                                                   ' AND ("cell line" OR "cell-line" OR "cellular line")'))
        time.sleep(1)
        pubmed_ids = [int(id) for id in string_search['IdList']]
        paper_id_list.append({'cell_line': row['cell_line'], 'pubmed_ids': pubmed_ids, 'num_ids': len(pubmed_ids)})
        print("Linea Celular " + row['cell_line'])
    except Exception as e:
        print(f"Error Linea Celular {row['cell_line']}: {str(e)}")
        error_paper_id_list.append({'cell_line': row['cell_line'], 'pubmed_ids': [], 'num_ids': 0})

df_results = pd.DataFrame(paper_id_list)
df_results.to_csv('results.csv', index=False)

df_errors = pd.DataFrame(error_paper_id_list)
df_errors.to_csv('errores.csv', index=False)
```

Figura 3 Consulta base de datos PUBMED.

3.3 Histograma

A continuación, se compiló la información sobre la frecuencia de los diferentes números de identificadores de artículos de PubMed para cada línea celular. Para este propósito, se creó una lista llamada frequency_papersid, la cual está compuesta por diccionarios. Cada diccionario en esta lista registra el número de identificadores y una lista de líneas celulares relacionadas con dicho número de identificadores.

Este enfoque permitió analizar la distribución de los identificadores de artículos científicos obtenidos de PubMed para cada línea celular. Al contar la frecuencia de los diferentes números de identificadores, se obtuvo una visión general de la cantidad de estudios y publicaciones científicas relacionadas con cada línea celular en la base de datos de PubMed.

Posteriormente, se creó un archivo llamado output.csv que almacena la información obtenida en la lista frequency_paperid. Cada elemento de la lista corresponde a una fila en el archivo, y se registraron las columnas ID, num_ids, cell_lines y line_count.

La columna "ID" representa el índice de cada fila en el archivo. La columna "num_ids" indica

```
import plotly.express as px

df = pd.read_csv('output.csv')
df_subset = df.loc[:, ['num_ids', 'line_count']]

print(df_subset.head())

fig = px.bar(df_subset, x='num_ids', y='line_count')
fig.update_layout(xaxis_title='line_count', yaxis_title='Número de papers',
                  title='Frecuencia de ID')

fig.show()
```

Figura 4 Proceso de generar diagrama de frecuencias.

el número de identificadores de artículos relacionados con las líneas celulares correspondientes. La columna "cell_lines" contiene las líneas celulares asociadas a cada número de identificadores. Finalmente, la columna "line_count" muestra el recuento de líneas celulares para cada número de identificadores específico como se muestra en la Figura 4.

Finalmente, con el archivo output.csv se creó un gráfico de barras que muestra la frecuencia de los diferentes números de identificadores de artículos y muestra una figura resultante como indica la Figura 5.

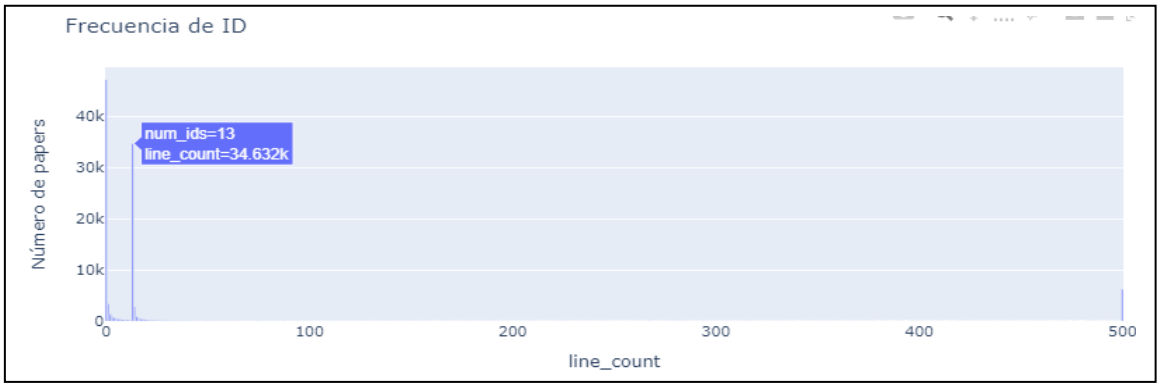

Figura 5 Frecuencia de indicadores de artículos

3.4 Scraping de PUBMED

Mediante la utilización de la API de Entrez de Biopython, se llevaron a cabo consultas a la base de datos de PubMed para obtener información detallada de artículos científicos. Se desarrolló un script que iteró sobre cada fila del DataFrame df_results, el cual contenía información sobre líneas celulares y los identificadores de PubMed correspondientes.

Durante el proceso de iteración, se recuperaron los datos detallados de cada artículo, incluyendo el ID de PubMed, el título del artículo y el resumen, en caso de estar disponible. Con esto se automatizó el proceso de obtención de información detallada de los artículos científicos relacionados con las líneas celulares de interés. Los resultados obtenidos fueron almacenados en archivos comprimidos en formato .gz, permitiendo una gestión eficiente y compacta de la información recopilada como se puede observar en la Figura 6.

```
for _, row in df_results.iterrows():
    try:
        if len(row['pubmed_ids']) != 0:
            pubmed_ids_search = Entrez.efetch(db="pubmed", id=','.join(map(str, row['pubmed_ids_13'])), rettype="xml", retmode="text")
            result_pubmed_ids_search = Entrez.read(pubmed_ids_search)
            result = []
            for pubmed_article in result_pubmed_ids_search['PubmedArticle']:
                pubmed_id = int(str(pubmed_article['MedlineCitation']['PMID']))
                title = pubmed_article['MedlineCitation']['Article']['ArticleTitle']
                article = pubmed_article['MedlineCitation']['Article']
                if 'Abstract' in article:
                    abstract = article['Abstract']['AbstractText'][0]
                    data = { "cell_line":row['cell_line'], "pubmedid": pubmed_id, "title": title, "abstract": abstract }
                    result.append(data)
                else:
                    print("La linea celular " + row['cell_line'] + " sin Abstract")
                    data = { "cell_line": row['cell_line'], "pubmedid": pubmed_id, "title": title, "abstract": None }
                    result_wo_abstract.append(data)

            df = pd.DataFrame(result)
            cell_line = row['cell_line']
            if not os.path.exists(directory):
                os.makedirs(directory)
            file_path = os.path.join(directory, '{}_data.json.gz'.format(cell_line))
            with gzip.open(file_path, 'wb') as f:
                f.write(df.to_json(orient='table').encode('utf-8'))
            print("La linea celular " + row['cell_line'] + " se ha guardado")

            df_errors = pd.DataFrame(result_wo_abstract)
            df_errors.to_csv('errores_abs.csv', index=False)

    except Exception as e:
        print(f"Error Linea Celular {row['cell_line']}: {str(e)}")
        result_errors.append({'cell_line': row['cell_line']})

df_result_errors = pd.DataFrame(result_errors)
df_result_errors.to_csv('abstract_errors.csv', index=False)
```

Figura 6 Obtención de artículos científicos.

3.5 Tabulación de datos sobre Abstracts

Una vez consolidada la información resultante en archivos .gz, se procedió a desarrollar un script para visualizar cada componente del archivo y presentarlos en forma de un DataFrame llamado df. Este DataFrame contiene los siguientes campos: índice, línea celular, identificador de PubMed, título y resumen del artículo como lo muestra la Figura 7.

	cell_line	pubmedid	title	abstract
0	CVCL_A2TJ	36710992	Investigation of Curcumin-Loaded OA400 Nanopar...	Curcumin, a compound derived from the root of ...
1	CVCL_A2TJ	36200530	Development of a novel bioengineered 3D brain-...	Primary blast injury is caused by the direct i...
2	CVCL_A2TJ	35036572	Preparation of fatty acid solutions exerts sig...	Free fatty acids are essentially involved in t...
3	CVCL_A2TJ	33969526	In vivo CRISPR screening for novel noncoding R...	CRISPR (clustered regularly interspaced short ...
4	CVCL_A2TJ	33453083	Select neurotrophins promote oligodendrocyte p...	Axonal damage and the subsequent interruption ...
5	CVCL_A2TJ	33165868	Establishment of a bladder cancer cell line ex...	Several experimental models including patient ...
6	CVCL_A2TJ	32954502	Noradrenaline protects neurons against H<sub>2...	Oxidative stress has been implicated in a vari...
7	CVCL_A2TJ	32197467	Elevated PDK1 Expression Drives PI3K/AKT/MTOR ...	Resistance to radiotherapy (IR), with conseque...
8	CVCL_A2TJ	32133675	Tomosyn regulates the small RhoA GTPase to con...	Tomosyn, a protein encoded by syntaxin-1-bindi...
9	CVCL_A2TJ	31157458	Protective roles of carbonic anhydrase 8 in Ma...	Machado-Joseph disease (MJD)/Spinocerebellar a...
10	CVCL_A2TJ	30137675	Differentiated mitochondrial function in mouse...	Mouse 3T3 fibroblasts are commonly used for in...
11	CVCL_A2TJ	29873307	Corrigendum: Potential role of the glycolytic ...	At the time of publication, our group had perf...
12	CVCL_A2TJ	29577375	Ca2+/calmodulin-dependent protein kinase II an...	Traumatic injury often results in axonal sever...

Figura 7 Dataframe df

A continuación, se creó un archivo denominado "abstracts.zip" con el propósito de comprimir la información del DataFrame df. Se agregó el contenido de dicho DataFrame en un nuevo archivo llamado "abstracts.csv" dentro del archivo comprimido tal como se puede observar en la Figura 8.

```
import zipfile
file = 'data/abstracts.zip'
with zipfile.ZipFile(file, 'w') as zf:
    zf.writestr('abstracts.csv', df.to_csv(index=False

with zipfile.ZipFile(file, 'r') as zf:
    with zf.open('abstracts.csv', 'r') as f:
        df_r = pd.read_csv(f, index_col=0)

df_r
```

Figura 8 Archivo de información Abstracts.csv

Este enfoque permitió la visualización y manipulación eficiente de la información obtenida de los artículos científicos relacionados con las líneas celulares de interés. El archivo comprimido "abstracts.zip" facilita la distribución y el almacenamiento de los datos de manera compacta y organizada.

3.6 Tratamiento de los datos

Con el archivo "abstracts.csv" generado, se inició el proceso de limpieza de datos en cada uno de los campos. Se aplicaron técnicas de procesamiento de texto para preparar los datos, lo cual incluyó las siguientes etapas:

- Eliminación de caracteres especiales: Se removieron caracteres especiales y puntuaciones que no son relevantes para el análisis de los datos.
- Conversión a minúsculas: Todos los textos fueron convertidos a minúsculas para asegurar la consistencia y evitar duplicados basados en diferencias de mayúsculas y minúsculas.
- Tokenización en palabras: Se dividió cada texto en palabras individuales para facilitar el procesamiento posterior. Se creó una lista de palabras para cada resumen.
- Eliminación de stop words: Se eliminaron las palabras comunes y poco significativas, como artículos, preposiciones y conjunciones, que no aportan información relevante para el análisis.
- Extracción de raíces de palabras: Se utilizó un stemmer para extraer la raíz de cada palabra, lo que permite reducir las palabras a su forma base y agrupar términos similares.

Al finalizar este proceso, se agregó una columna adicional al DataFrame llamada "abstract_p" que contenía los resúmenes procesados. Esta columna permitió realizar comparaciones entre los resúmenes originales y los resúmenes procesados, lo que facilitaba el análisis y la comprensión de los datos, como lo muestran las Figura 9 y Figura 10 respectivamente.

```python
from nltk.stem import PorterStemmer
# Inicializar el stemmer
stemmer = PorterStemmer()

# Definir una función para limpiar el texto
def clean_text(text):
    # Eliminar caracteres especiales y números
    text = re.sub('[^a-zA-Z]', ' ', text)

    # Convertir el texto a minúsculas
    text = text.lower()

     # Remove leading and trailing white spaces
    text = text.strip()
    # Replace multiple spaces with a single space
    text = re.sub('\s+', ' ', text)

    # Tokenizar el texto en palabras
    words = word_tokenize(text)

    # Eliminar stopwords y aplicar stemming
    stop_words = set(stopwords.words('english'))
    #words = [word for word in words if word not in stop_words]
    stemmed_words = [stemmer.stem(word) for word in words if word not in stop_words]

    # Unir las palabras nuevamente en una cadena
    cleaned_text = ' '.join(stemmed_words)

    return cleaned_text

# Aplicar la función clean_text a la columna "abstract"
df['abstract_p'] = df['abstract'].apply(clean_text)
```

Figura 9 Limpieza de archivo abstracts.csv

	Unnamed: 0	index	cell_line	pubmedid	title	abstract	abstract_p
0	0	0	CVCL_0028	33040078	Splicing factor SF3B1 promotes endometrial can...	Although endometrial cancer is the most common...	although endometri cancer common cancer femal ...
1	1	1	CVCL_0028	34476599	Sirtuin 2 promotes cell stemness and MEK/ERK s...	Sirtuin 2 (SIRT2) is functionally important in...	sirtuin sirt function import cancer progress t...
2	2	2	CVCL_0028	35401936	Role of the prorenin receptor in endometrial c...	Endometrial cancer is the most diagnosed gynec...	endometri cancer diagnos gynecolog malign desp...
3	3	3	CVCL_0028	32431202	NLRC5 promotes cell migration and invasion by ...	NOD-like receptor family caspase recruitment d...	nod like receptor famili caspas recruit domain...
4	4	4	CVCL_0028	24526410	GRP78 mediates cell growth and invasiveness in...	Recent studies have indicated that endoplasmic...	recent studi indic endoplasm reticulum stress ...
...	...	...	...	...	...	...	...
113003	113003	8	CVCL_ZZ70	28815590	Juxtanodin in retinal pigment epithelial cells...	Juxtanodin (JN, also known as ermin) was initi...	juxtanodin jn also known ermin initi identifi ...
113004	113004	9	CVCL_ZZ70	15871909	Steps of the tick-borne encephalitis virus rep...	Tick-borne encephalitis virus (TBEV) is an imp...	tick born enceph viru tbev import human pathog...
113005	113005	10	CVCL_ZZ70	32954502	Noradrenaline protects neurons against H<sub>2...	Oxidative stress has been implicated in a vari...	oxid stress implic varieti neurodegen disord a...
113006	113006	11	CVCL_ZZ70	29577375	Ca2+/calmodulin-dependent protein kinase II an...	Traumatic injury often results in axonal sever...	traumat injuri often result axon sever initi o...
113007	113007	12	CVCL_ZZ70	36200530	Development of a novel bioengineered 3D brain-...	Primary blast injury is caused by the direct i...	primari blast injuri caus direct impact overpr...

113008 rows × 7 columns

Figura 10 Dataframe de datos procesados y comparados

3.7 Term frequency–Inverse document frequency (TF-IDF)

Se utilizó TF-IDF (Term Frequency-Inverse Document Frequency) que es una técnica utilizada en el procesamiento de texto con el objetivo de evaluar la importancia relativa en un documento, se calculó las características de la columna abtract_p del dataframe, el resultado fue una matriz TF-IDF donde cada columna representa una palabra y cada fila representa un documento como lo muestra la Figura 11 y Figura 12 respectivamente.

```
# Inicializar el vectorizador TF-IDF
vectorizer = TfidfVectorizer()

# Aplicar TF-IDF a la columna "abstract_p"
tfidf_matrix = vectorizer.fit_transform(df['abstract_p'])


# Obtener las palabras (features) del vectorizador TF-IDF
features = vectorizer.get_feature_names_out()

# Convert the TF-IDF matrix to a DataFrame / Convertir la matriz TF-IDF en un DataFrame
tfidf_df = pd.DataFrame(tfidf_matrix.toarray(), columns=vectorizer.get_feature_names_out(

# Imprimir el total de palabras y los primeros 3 registros de la columna "abstract_p"
print("Total de palabras:", len(features))
print(df['abstract_p'].head(2))
tfidf df
```

Figura 11 Importancia relativa de un documento con TF-IDF

	aa	aacr	aacrjourn	aadac	ab	abc	abca	abe	aberr	aberrantli	...	znf	zno	zo	zone	zonula	zr	zro	zta	z
0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
1	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
2	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
3	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
995	0.0	0.0	0.0	0.0	0.126996	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
996	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Figura 12 Dataframe conjunto de características TF-IDF

La combinación de TF y IDF proporciona una medida de la importancia relativa de una palabra en un documento específico y en todo el conjunto de documentos.

3.8 Principal Component Analysis (PCA)

Una vez calculado el conjunto de características TF-IDF, con el fin de reducir la dimensionalidad de los datos, se utilizó PCA (Principal Component Analysis) TF-IDF para mejorar la visualización de los datos a dimensiones más compactas, y eliminar características redundantes o irrelevantes, se lo representó en 5 dimensiones principales,

el resultado fue una matriz donde cada columna del DataFrame representa una dimensión y cada fila representa un documento.

Para observar la línea celular y los resultados de la matriz generada con pca, se procedió a agregar la columna cell_line a la matriz df_pca dando como resultado final un dataframe llamado df_final.

Tanto el script como el resultado de la matriz se muestra en Figura 13 y Figura 14 respectivamente.

```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import PCA


 # Step 2: Reduce dimensions using PCA
pca = PCA(n_components=5)  # Specify the number of desired dimensior
tfidf_pca = pca.fit_transform(tfidf_matrix.toarray())



# Create a new DataFrame with the PCA results
df_pca = pd.DataFrame(data=tfidf_pca)



# Print the final DataFrame with TF-IDF and PCA results
print(df_pca)
```

Figura 13 Reducción de dimensionalidad con PCA.

	0	1	2	3	4
0	0.024581	0.002895	-0.019362	0.009844	0.005580
1	0.025005	0.019973	-0.019519	0.025590	0.019812
2	0.020746	0.013963	-0.017419	0.017501	0.018854
3	0.010465	0.026303	-0.001925	0.010304	0.006178
4	-0.029837	-0.037502	-0.040361	0.015280	-0.011181
...	...	...	...	...	...
113003	0.109730	-0.072203	0.021938	-0.191086	-0.007287
113004	-0.008012	0.032055	0.001148	0.012356	-0.011872
113005	-0.168293	0.118696	0.408429	-0.010349	0.124727
113006	-0.473479	-0.417366	-0.158590	-0.078571	-0.068002
113007	-0.112975	0.143744	0.267155	-0.239997	0.146898

[113008 rows x 5 columns]

Figura 14 Dataframe utilizando PCA

3.9 Support Vector Domain Description (SVDD)

Para obtener un conjunto de textos que se consideran representativos se implementó SVDD (Support Vector Data Description) utilizando OneClassSVM con el kernel RBF y gamma, por medio de un filtrado de índices de los valores de los inliers, estos inliers

corresponden a los textos, se procedió a calcular el centro de los vectores, mediante el cálculo de la media de los vectores TF-IDF que fueron transformados por PCA, este centro es una representación promedio de los textos, posteriormente se calculó el radio de la esfera mediante la distancia máxima entre los vectores transformados y el centro.

Finalmente se filtra el dataframe original para incluir únicamente los inliers identificados, esto con el fin de obtener un subconjunto de textos que se consideran representativos o relevantes, así como lo muestra la Figura 15.

```python
# aplicando pca
from sklearn.svm import OneClassSVM
import numpy as np

# Step 3: Apply SVDD for outlier detection and subject identification

subjects = df['cell_line'].unique()

subject_centers_pca = {}
subject_radii_pca = {}

for subject in subjects:
    subject_indices = df[df['cell_line'] == subject].index

    # Extract TF-IDF vectors for the subject's texts
    subject_tfidf_matrix_pca = tfidf_pca[subject_indices]

    # Apply SVDD (OneClassSVM) for the subject's texts
    model = OneClassSVM(kernel='rbf', gamma='scale')
    model.fit(subject_tfidf_matrix_pca)

    # Filter inliers for the subject
    inlier_indices = subject_indices[model.predict(subject_tfidf_matrix_pca) == 1]
    inliers_df = df.loc[inlier_indices]

    # Calculate the center of the vectors for the subject
    subject_center_pca = np.mean(subject_tfidf_matrix_pca, axis=0)
    subject_centers_pca[subject] = subject_center_pca

    # Calculate the radius of the sphere that describes the subject
    subject_radius_pca = np.max(np.linalg.norm(subject_tfidf_matrix_pca - subject_center_pca, axis=1)
    subject_radii_pca[subject] = subject_radius_pca

# Filter the DataFrame to include only the inliers
df_inliers = df.loc[inliers_df.index]
```

Figura 15 Obtención de conjunto de textos representativos (SVDD).

3.10 Centroides y radios

Se generaron radios y centroides asociados a cada línea celular, estos radios y centros son importantes porque de este modo se permite representar de una forma compacta y significativa cada característica y variabilidad del texto al que se encuentra asociado la línea celular, esto mediante la implementación de algoritmos de SVDD y PCA aplicados en pasos anteriores con esto, se permitió encontrar una esfera o hiperesfera de radio mínimo,

al tiempo que PCA se dedica a reducir la dimensionalidad de las líneas celulares. Todo esto con el fin de facilitar la detección de sujetos no comunes o inusuales en la data.

Esta información se puede evidencias en la figura Figura 16 y Figura 17 respectivamente.

```python
import pandas as pd
import numpy as np

# Crear una lista para almacenar los resultados
result_data = []

# Llenar la lista con los resultados
for subject in subjects:
    center = np.squeeze(subject_centers_pca[subject])
    radius = np.squeeze(subject_radii_pca[subject])
    result_data.append({'cell_line': subject, 'center': center
                        'radius': radius})

# Crear el dataframe a partir de la lista de resultados
result_df_pca = pd.DataFrame(result_data)

# Imprimir el dataframe de resultados
result_df_pca
```

Figura 16 Obtención de centroides.

	cell_line	center	radius
0	CVCL_0028	[0.00251791017518784, 0.007314898799435672, -0...	0.066173
1	CVCL_0080	[0.019450449413684838, 0.013177261231692031, -...	0.054771
2	CVCL_0081	[-0.00948665032411371, 0.027937856571742012, 0...	0.046550
3	CVCL_0082	[0.023437894045063443, 0.006965141811858809, 0...	0.038501
4	CVCL_0107	[0.005348316471685562, 0.01594712288027022, 0....	0.098690
...	...	...	...
9243	CVCL_ZZ58	[-0.007880641290901352, -0.0014381966014572116...	0.897165
9244	CVCL_ZZ59	[-0.007880641290901352, -0.0014381966014572116...	0.897165
9245	CVCL_ZZ60	[-0.007880641290901363, -0.0014381966014572116...	0.897165
9246	CVCL_ZZ61	[-0.007880641290901363, -0.0014381966014572116...	0.897165
9247	CVCL_ZZ70	[-0.0037931607355641476, -0.008175344523148489...	0.892744

Figura 17 Dataframe de obtención de centroides y radios.

3.11 Matriz de distancia

Se procedió al cálculo de la matriz de distancias entre los centros previamente determinados, la cual fue denominada "distance_matrix". Esta matriz de distancia es simétrica, por lo que solo se calculó la distancia entre los elementos una vez y luego se copiaron los valores en las posiciones simétricas. La distancia utilizada en esta matriz es la conocida como distancia euclidiana, y para su cálculo se empleó la función "euclidean"

de la biblioteca "scipy.spatial.distance". Esta matriz resultante es cuadrada y recibe el nombre de "distance_matrix", donde el valor en cada fila i y columna j representa la distancia euclidiana entre el centro del elemento i y el centro del elemento j.

Como se pueden observar el código y la matriz generada en Figura 18 la Figura 19 y respectivamente.

```python
import numpy as np
from scipy.spatial.distance import euclidean

n = len(result_data)  # Number of observations
distance_matrix = np.zeros((n, n))  # Initialize an empty distance matri

# Calculate pairwise distances using a for loop
for i in range(n):
    for j in range(i + 1, n):
        center_i = np.asarray(result_data[i]['center'])
        center_j = np.asarray(result_data[j]['center'])
        distance = euclidean(center_i.flatten(), center_j.flatten())
        distance_matrix[i, j] = distance
        distance_matrix[j, i] = distance

# Print the distance matrix
print(distance_matrix)
```

Figura 18 Código de distancia euclidiana.

```
[[0.         0.0210112  0.02695875 ... 0.02686492 0.02686492 0.03268134]
 [0.0210112  0.         0.03404552 ... 0.03398719 0.03398719 0.0359151 ]
 [0.02695875 0.03404552 0.         ... 0.03619793 0.03619793 0.04255467]
 ...
 [0.02686492 0.03398719 0.03619793 ... 0.         0.         0.01416422]
 [0.02686492 0.03398719 0.03619793 ... 0.         0.         0.01416422]
 [0.03268134 0.0359151  0.04255467 ... 0.01416422 0.01416422 0.        ]]
```

Figura 19 Matriz distancia.

Esta matriz proporciona una valiosa visión de las distancias entre los diferentes pares de elementos, lo que permite realizar comparaciones y analizar la similitud entre ellos con base en la ubicación de sus centros en el espacio de características reducidos. Esta información es esencial para el análisis y representación de las relaciones entre los componetes y sus características asociadas en el contexto de la investigación.

4 MODELO

4.1 Árbol de distancias

Se procedió a generar la representación de un árbol de distancias mediante un dendrograma utilizando la matriz de distancias "distance_matrix". Este dendrograma es una visualización gráfica que muestra jerarquías y refleja cómo las líneas celulares se agrupan o relacionan entre sí según las distancias calculadas.

Para lograr esto, se utilizaron funciones de la biblioteca "scipy.cluster.hierarchy". En primer lugar, se calculó la matriz de enlace "z" mediante la función "linkage()". Esta función tomó la matriz de distancias como entrada y aplicó el método de agrupamiento "average" para determinar las distancias entre los grupos.

A continuación, se empleó la función "dendrogram()" para trazar el dendrograma. Esta función utilizó la matriz de enlace "z" para generar una representación gráfica del dendrograma. En el eje x se muestran las líneas celulares, mientras que en el eje y se representan las distancias entre ellas.

El dendrograma obtenido brinda una valiosa visualización de cómo las líneas celulares se agrupan en función de las distancias previamente calculadas. Esta representación gráfica ofrece información relevante sobre las relaciones de similitud o cercanía entre las diferentes líneas celulares como se puede observar en la Figura 20.

Para una mejor visualización este dendograma se podrá encontrar como anexo en un archivo llamado “dendograma.png”.

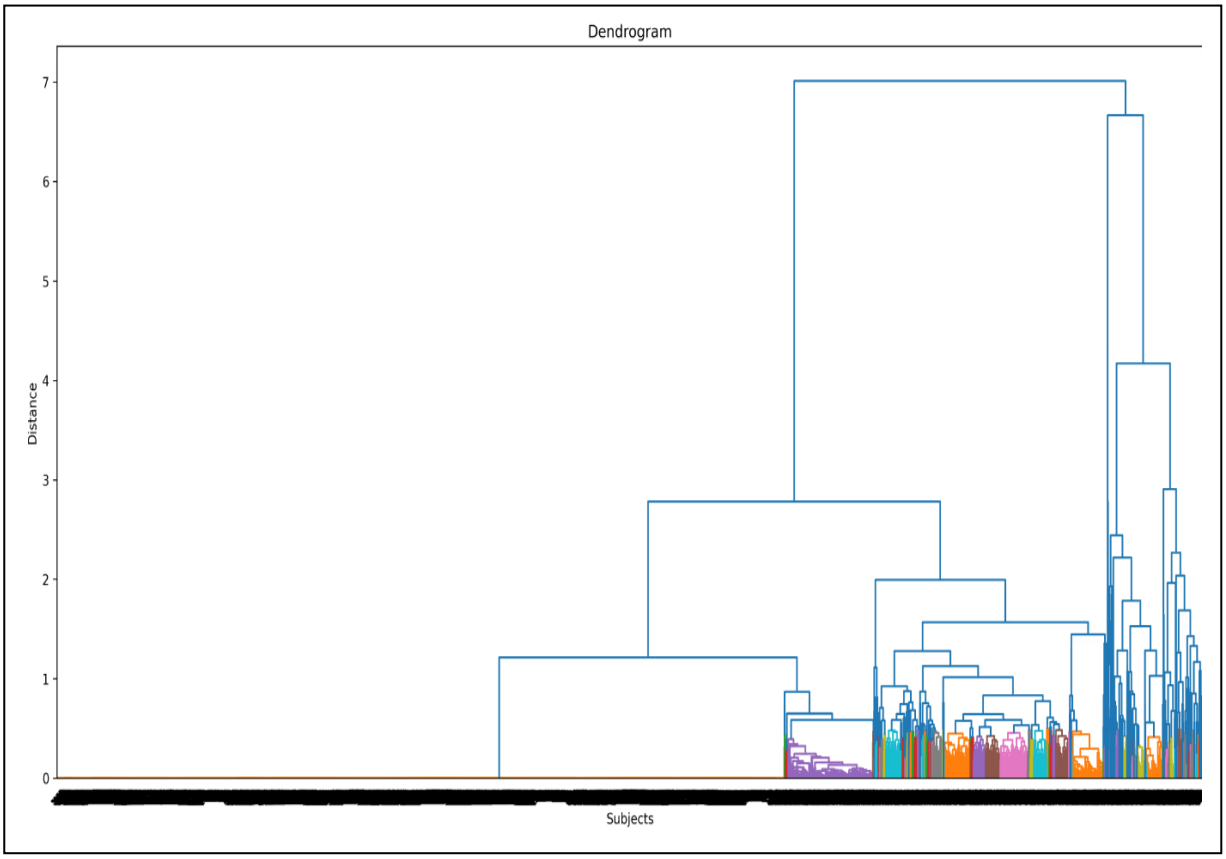

Figura 20 Árbol de distancias.

4.2 Servidor de alto procesamiento.

Para procesar toda la base de datos contenida en el archivo “abtracts.csv” se empleó un servidor de alto rendimiento equipado con grandes recursos, este servidor cuenta con 256 GB de memoria RAM, además de una tarjeta gráfica NVIDIA TESLA M1 y espacio en disco 9.28 TB, lo que permite manejar grandes cantidades de datos y ejecutar operaciones complejas, la conexión a este servidor se realizó a través de una plataforma VDI Citrix, lo que proporciona un entorno de trabajo virtual que permite acceder a los recursos y programas necesarios de forma remota como se puede observar en la Figura 21.

Una vez establecida la conexión se crearon credenciales de acceso para utilizar el entorno de Jupiter Notebook mediante un navegador web.

Cabe mencionar que todos los componentes y recursos utilizados en este proceso son propiedad de la Escuela Politécnica Nacional que proporciona a sus investigadores y tesistas las herramientas que permiten llevar a cabo investigaciones y proyectos de tesis.

Características	Recurso disponible
CPU	4 CPU con 16 núcleos C/U
Memoria RAM	256 GB
Espacio en disco	9.28 TB
Sistema Operativo	Ubuntu 22.04 2 LTS
Recursos de virtualización	Docker Engine - Community V24.0.5
Dirección IP	Conexión con IPV4 para pruebas y desarrollo
Conectividad de red externa	Conexión a Internet habilitada Descarga:91.74 Mbit/s Carga: 74.01 Mbit/s

Figura 21 Servidor de altas prestaciones para procesamiento.

5 RESULTADOS, CONCLUSIONES Y TRABAJOS FUTUROS

5.1 Resultados

Una vez que se ha ejecutado el código y obtenido el dendograma, se ha generado una representación visual de las relaciones jerárquicas entre las líneas celulares. El análisis del dendograma permite identificar grupos o patrones relevantes en el conjunto de datos, lo que contribuye a una compresión profundizada de las interacciones entre las líneas celulares.

Esta herramienta visual se convierte en una valiosa ayuda para el estudio de las relaciones entre las características asociadas a cada línea celular. La similitud entre las ramas del dendograma revela que las líneas celulares comparten características similares y cuáles están más relacionadas. Esto puede ser especialmente útil para evaluar posibles tratamientos o fármacos, ya que las líneas celulares con características similares podrían responder de manera similar a ciertos tratamientos o intervenciones.

5.2 Conclusiones

- Los resultados obtenidos tienen un impacto significativo en la mejora del análisis y la comprensión de las interacciones y similitudes entre los sujetos, basándose en sus representaciones en el espacio de características reducido.
- La interpretación del dendograma proporciona nuevas perspectivas sobre las relaciones entre las líneas celulares y su agrupación según sus similitudes en características. Estas percepciones pueden guiar futuras investigaciones y análisis detallados de las propiedades de las líneas celulares.
- La transformación de información textual en un conjunto de puntos, seguida del cálculo de radios y distancias, ha confirmado la posibilidad de representar la información de las líneas celulares en estructuras de esferas o hiperesfera. Esto ha validado la aproximación de cercanía y similitud entre diversas líneas celulares, simplificando así el análisis y la interpretación de sus interrelaciones.
- La aplicación de la matriz de distancias, junto con la visualización del dendograma, ha permitido identificar patrones y grupos de líneas celulares que comparten características similares. Esta metodología presenta un potencial significativo para avanzar en el análisis y la clasificación de líneas celulares en futuras investigaciones.

5.3 Trabajos futuros

Se podría explorar otras formas de representar la información de líneas celulares a través de diversas arquitecturas, como redes neuronales recurrentes, transformadores y word2vec. Cada una de estas técnicas ofrece un enfoque diferente para la representación de la información, lo que puede brindar diferentes perspectivas y resultados interesantes en el análisis de datos, cada una de estas arquitecturas cuenta con ventajas y desafíos, la elección de alguna dependerá de la naturaleza de los datos y los objetivos que conlleva la investigación, es importante mencionar que la exploración de diferentes formas de representación puede generar valor agregado y enriquecer el análisis y la comprensión de las características y relaciones de las líneas celulares, lo que contribuye a una mejora en la calidad de las investigaciones en los campos de las ciencias de la vida y bioinformática.

Se podría también implementar una aplicación que permita realizar una comparativa en tiempo real de las líneas celulares basada en sus características. Esta aplicación podría estar compuesta de un *frontend* y un *backend*, ofreciendo una interfaz amigable y accesible para los investigadores y profesionales.

El *frontend* podría implementar una interfaz intuitiva donde los usuarios puedan ingresar información sobre una línea celular específica, como su nombre o identificador. Con esta información, el *backend* realizaría el procesamiento del texto y cálculo de distancias utilizando las técnicas como PCA Y SVDD, tal como se ha realizado en el presente trabajo.

El *backend* procesaría la información ingresada y procedería a la búsqueda de similitud en función de sus características y devolvería resultados en tiempo real. Esto permitiría a los usuarios obtener de manera rápida la comparativa de la línea celular consultada con otras líneas celulares en la base de datos.

6 REFERENCIAS BIBLIOGRÁFICAS

[1] I. N. D. CANCER, «cancer.gov,» 06 07 2023. [En línea]. Available: https://www.cancer.gov/espanol/publicaciones/diccionarios/diccionario-cancer/def/linea-celular-de-cultivo.

[2] I. Carrera, I. Dutra y E. Tejera, «A Representation Method for Cellular Lines based on SVM and Text Mining,» de *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, Seoul, Korea (South), 2020.

[3] A. Bairoch, «The Cellosaurus, a Cell-Line Knowledge Resource,» *Journal of Biomolecular Techniques,* vol. 29, 第 2 号, pp. 25-38, 2018.

[4] C. N. d. I. B. NCBI, «PubMed,» 1996. [En línea]. [Último acceso: 06 07 2023].

[5] K. K. N. Newtown, «A Review on Web Scrapping and its Applications,» de *2019 International Conference on Computer Communication and Informatics (ICCCI - 2019), Jan. 23 – 25, 2019.*, Coimbatore, INDIA, 2019.

[6] G. I. W. claude sammut, «Springerlink,» 2011. [En línea]. Available: https://link.springer.com/referenceworkentry/10.1007/978-0-387-30164-8_832. [Último acceso: 10 07 2023].

[7] D. P. A. V. Prafulla Bafna, «Document Clustering: TF-IDF approach,» de *International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, Chennai, India, 2016.

[8] H. S. M. A. V. B. S. Shruti Sehgal, «Data analysis using principal component analysis,» de *2014 International Conference on Medical Imaging, m-Health and Emerging Communication Systems (MedCom)*, Greater Noida, India, 2014.

[9] W. z. w. Xin-dong, «Support vector domain description for speaker recognition,» de *Redes neuronales para el procesamiento de señales XI: Actas del* Taller *de la Sociedad de Procesamiento de Señales IEEE de 2001 (IEEE Cat. No.01TH8584)*, North Falmouth, MA, EE. UU., 2001.

[10] P. S. Foundation, «Python.org,» [En línea]. Available: https://www.python.org/doc/essays/blurb/. [Último acceso: 02 08 2023].

[11] L. Carvajal, Metodología de la Investgación Científica. Curso general y aplicado, 28 ed., Santiago de Cali: U.S.C., 2006, p. 139.

[12] pandas.pydata.org, «pandas.pydata.org,» [En línea]. Available: https://pandas.pydata.org/about/. [Último acceso: 02 08 2023].

7 ANEXOS

7.1 ANEXO I:

Repositorio de GitHub del proceso de Integración curricular.

El presente trabajo de integración curricular cuenta con un repositorio de GitHub que proporciona detalles sobre el proceso llevado a cabo, así como los datos utilizados en el estudio. Ha sido creado para facilitar el acceso y la revisión del material relacionado con el proyecto.

Link: https://github.com/Henry-Guanoluisa/TRABAJO-DE-INTEGRACION-CURRICULAR-LINEAS-CELULARES