Pontificia Universidad Javeriana Cali
Facultad de Ingeniería y Ciencias.
Ingeniería de Sistemas y Computación.
Proyecto de Grado.

Sistema de Generación Automática de Resúmenes Académicos: Una Aplicación de Aprendizaje Automático

Cristian Camilo Tabares Pérez
Juan José Fernández Aristizábal

Director: Dr. Diego Luis Linares Ospina
Directora: Dra. Gloria Inés Álvarez Vargas

12 de febrero de 2025

Pontificia Universidad JAVERIANA Cali

Facultad de Ingeniería y Ciencias

Acta de Correcciones al Proyecto de Grado
Ingeniería de Sistemas y Computación

Fecha: 12 de febrero del 2025

Autores: Cristian Camilo Tabares Pérez y Juan José Fernández Aristizábal

Nombre del Proyecto de Grado: Sistema de Generación Automática de Resúmenes Académicos: Una Aplicación de Aprendizaje Automático

Director: Diego Luis Linares Ospina

Como indica el artículo 2.27 de las Directrices de Trabajo de Grado, he verificado que los estudiantes indicados arriba han implementado todas las correcciones que los Jurados del Proyecto de Grado definieron que se efectuaran, como consta en el Acta de Calificación correspondiente.

Firma de Director(a) del Proyecto de Grado

Santiago de Cali, 12 de febrero de 2025.

Señores
Pontificia Universidad Javeriana Cali.
Dr. Gerardo Mauricio Sarria M.
Director Carrera de Ingeniería de Sistemas y Computación.
Cali.

Cordial Saludo.

Por medio de la presente me permito informarle que los estudiantes de Ingeniería de Sistemas y Computación Cristian Camilo Tabares Pérez (cod: 8954215) y Juan José Fernández Aristizábal (cod: 8953371) culminaron satisfactoriamente bajo nuestra dirección el proyecto de grado titulado "Sistema de Generación Automática de Resúmenes Académicos: Una Aplicación de Aprendizaje Automático".

Atentamente,

Dr. Diego Luis Linares Ospina

Dra. Gloria Inés Álvarez Vargas

Santiago de Cali, 12 de febrero de 2025.

Señores
Pontificia Universidad Javeriana Cali.
Dr. Gerardo Mauricio Sarria M.
Director Carrera de Ingeniería de Sistemas y Computación.
Cali.

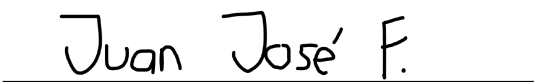
Cordial Saludo.

Nos permitimos presentar a su consideración el proyecto de grado titulado "Sistema de Generación Automática de Resúmenes Académicos: Una Aplicación de Aprendizaje Automático" con el fin de cumplir con los requisitos exigidos por la Universidad para optar al título de Ingeniero de Sistemas y Computación.

Al firmar aquí, damos fe que entendemos y conocemos las directrices para la presentación de trabajos de grado de la Facultad de Ingeniería aprobadas el 26 de Noviembre de 2009, donde se establecen los plazos y normas para el desarrollo del anteproyecto y del trabajo de grado.

Atentamente,

Cristian Camilo Tabares Pérez
Código: 8954215



Juan José Fernández Aristizábal
Código: 8953371

Abstract

This research project focused on the development of an automatic academic summarization system based on machine learning techniques, aiming to evaluate its ability to synthesize information from lengthy academic texts. The central challenge was to design a model capable of capturing and condensing the main ideas of the documents.

Our approach involved the implementation and comparison of various methods, including extractive models such as Luhn and K-Means, as well as the abstractive Seq2Seq model. The initial stages of development were dedicated to preprocessing data from academic articles on arXiv and exploring preliminary configurations to identify suitable techniques and parameters.

The experiments explored specific combinations of hyperparameters, such as batch size, number of epochs, and embedding dimensions, adjusting variables like EPOCHS, BATCH_SIZE, and LATENT_DIM. Additionally, the models were evaluated using ROUGE metrics, which measure precision, recall, and F1-score to assess the quality of the generated summaries, as well as BERTScore, which employs semantic representations to evaluate the similarity between generated summaries and reference texts.

In conclusion, this project highlights the potential of integrating extractive and abstractive approaches in summarization systems, emphasizing the importance of systematic design and the possibility of future optimizations to enhance their performance in academic contexts.

Keywords: Unsupervised learning, deep learning, extractive approach, abstractive approach, evaluation metrics, machine learning, text summarization

Resumen

Este proyecto de investigación se centró en el desarrollo de un sistema de generación automática de resúmenes académicos basado en técnicas de aprendizaje automático, con el objetivo de evaluar su capacidad para sintetizar información de textos extensos en el ámbito académico. El problema central radicaba en diseñar un modelo que pudiera capturar y condensar las ideas principales de los documentos .

Nuestro enfoque implicó la implementación y comparación de varios métodos, incluyendo modelos extractivos como Luhn y K-Means, y el modelo abstractivo Seq2Seq. Las fases iniciales del desarrollo estuvieron enfocadas en el preprocesamiento de datos provenientes de artículos académicos de arXiv, así como en la exploración de configuraciones preliminares para identificar técnicas y parámetros adecuados.

Los experimentos exploraron combinaciones específicas de hiperparámetros, como el tamaño del batch, la cantidad de épocas y las dimensiones de los embeddings. cada una ajustando variables como EPOCHS, BATCH_SIZE y LATENT_DIM. Además, los modelos se evaluaron empleando métricas ROUGE, que miden precisión, recuperación y F1-Score para validar la calidad de los resúmenes generados, y BERTScore, que utiliza representaciones semánticas para evaluar la similitud entre los resúmenes generados y las referencias.

En conclusión, el proyecto destaca el potencial de integrar enfoques extractivos y abstractivos en sistemas de generación de resúmenes, subrayando la importancia de un diseño sistemático y la posibilidad de futuras optimizaciones para mejorar su rendimiento en contextos académicos.

Palabras Clave: Aprendizaje no supervisado, aprendizaje profundo, enfoque extractivo, enfoque abstractivo, métricas de evaluación, aprendizaje automático, resumen de textos

Índice general

1. Análisis Inicial del Problema **15**
1.1. Planteamiento del Problema . . . 15
1.1.1. Formulación . . . 16
1.1.2. Sistematización . . . 16
1.2. Objetivos . . . 17
1.2.1. Objetivo General . . . 17
1.2.2. Objetivos Específicos . . . 17
1.3. Justificación . . . 17
1.4. Delimitaciones y Alcances . . . 18

2. Fundamentos Teóricos y Contexto del Aprendizaje Automático **19**
2.1. Áreas Temáticas . . . 19
2.2. Marco Teórico . . . 19
2.2.1. Términos y Conceptos . . . 19
2.2.2. Importancia del Preprocesamiento en Modelos de IA . . . 20
2.2.3. Modelos Estadísticos Para Generación de Resúmenes . . . 20
2.2.4. Modelo de Luhn . . . 21
2.2.5. Modelos de Aprendizaje no Supervisado Para Generación de Resúmenes . . . 22
2.2.6. Modelo K-Means . . . 23
2.2.7. Modelo TextRank . . . 25
2.2.8. Modelos de Deep Learning Para Generación de Resúmenes . . . 25
2.2.9. Modelo Seq2Seq . . . 26
2.2.10. Métricas de Evaluación para Generación de Resúmenes . . . 29
2.2.11. Lo Más Actual en Generación de Resúmenes . . . 31
2.3. Trabajos Relacionados . . . 32
2.3.1. "A Hybrid Approach for Automatic Text Summarization and Translation based On Luhn, Pegasus, and Textrank Algorithms" . . . 32
2.3.2. "Automatic Text Review Summarization of Digital Library System Application using TextRank Algorithm and TF-IDF" . . . 32
2.3.3. "Advancing Abstractive Bangla Text Summarization: A Deep Learning Approach Using Seq2seq Encoder-Decoder Model and T5 Transformer" . . . 33

3. Preparación y Preprocesamiento de los Datos **35**
3.1. Dataset . . . 35
3.2. Especificación y entendimiento de los datos . . . 35
3.3. Preprocesamiento . . . 36
3.3.1. Preprocesamiento para Seq2Seq . . . 36

3.3.2. Preprocesamiento para Luhn y K-Means . . . 39

4. Desarrollo e Implementación 43
4.1. Modelo de Luhn . . . 43
4.1.1. Hiperparámetros y Variables Configurables: . . . 44
4.1.2. Cálculo de TF-IDF: . . . 44
4.1.3. Selección de Palabras Clave: . . . 45
4.1.4. Aplicación del Modelo de Luhn: . . . 45
4.2. Modelo K-Means . . . 45
4.2.1. Hiperparámetros y Variables Configurables: . . . 46
4.2.2. Representación Vectorial de Oraciones . . . 47
4.2.3. Algoritmo de Agrupamiento K-means . . . 47
4.2.4. Selección de Oraciones Representativas . . . 47
4.2.5. Construcción del resumen . . . 47
4.3. Modelo Seq2Seq . . . 48
4.4. Creación del modelo Seq2Seq . . . 48
4.4.1. Arquitectura encoder-decoder . . . 49
4.4.2. Parámetros de entrenamiento . . . 49
4.5. Construcción del modelo . . . 50
4.5.1. Manejo de los pesos del Modelo . . . 50
4.5.2. Entrenamiento . . . 51
4.5.3. Estrategia de entrenamiento . . . 51
4.5.4. Exploración de hiperparámetros . . . 51
4.5.5. Arquitecturas y gráficas experimentos . . . 53
4.5.6. Etapa de decodificación . . . 58

5. Análisis de Resultados 61
5.1. Resultados . . . 61
5.1.1. Modelo Seq2Seq Experimento 1 . . . 61
5.1.2. Modelo Seq2Seq Experimento 2 . . . 62
5.1.3. Modelo Seq2Seq Experimento 3 . . . 63
5.1.4. Luhn . . . 64
5.1.5. K-Means . . . 65
5.2. Evaluación y análisis . . . 67
5.2.1. ROUGE-1 . . . 67
5.2.2. ROUGE-2 . . . 68
5.2.3. ROUGE-L . . . 70
5.2.4. BERTScore . . . 71

6. Conclusiones y Proyecciones Futuras 73
6.1. Conclusiones . . . 73
6.2. Proyecciones Futuras . . . 74

Bibliografía 75

Introducción

En el ámbito de la información, la capacidad de resumir de manera efectiva es esencial. Los resúmenes académicos facilitan la comprensión rápida de documentos extensos, permitiendo a los lectores enfocarse en los puntos clave sin necesidad de realizar lecturas exhaustivas. Sin embargo, la creciente cantidad de información disponible y la complejidad inherente del lenguaje académico hacen que esta tarea sea cada vez más desafiante.

En el contexto de la inteligencia artificial (IA), el procesamiento del lenguaje natural (NLP) ha avanzado significativamente en la última década. Modelos como BERT (Bidirectional Encoder Representations from Transformers) y GPT (Generative Pre-trained Transformer) han revolucionado el campo. Según Devlin et al., "BERT está diseñado para preentrenar representaciones bidireccionales profundas condicionando conjuntamente en ambos contextos, izquierdo y derecho, en todas las capas"[1] . Esto ha permitido una mayor precisión en tareas como la clasificación de texto, la traducción y la generación de resúmenes.

Por otro lado, los algoritmos para generación de resúmenes también han evolucionado. Subramanian et al. explican: "Presentamos un método para producir resúmenes abstractivos de documentos largos que superan varios miles de palabras a través de la abstracción neural"[2]. Este enfoque abstractivo representa un avance significativo en comparación con los métodos extractivos tradicionales.

Este proyecto tiene como objetivo explorar tecnologías que emplean técnicas de aprendizaje automático para sintetizar información y generar resúmenes. Al concentrar los datos más relevantes en resúmenes iniciales, se busca facilitar el acceso a información clave y proporcionar una base eficiente que los académicos puedan personalizar según sus necesidades específicas.

Capítulo 1

Análisis Inicial del Problema

1.1. Planteamiento del Problema

Los resúmenes de textos académicos son una herramienta fundamental para la comunicación en el ámbito académico, proporcionando una vía concisa y precisa para transmitir las ideas principales de un texto. En palabras de Wang Guan, Ivan Smetannikov y Man Tianxing (2021) [3], "la sobrecarga de información es un desafío importante en la era de la información. Los resúmenes actúan como puertas de entrada, permitiendo a los usuarios comprender los puntos clave de manera rápida y eficiente".

En el contexto actual, la producción de conocimiento científico y tecnológico se ha vuelto exponencial. La cantidad masiva de textos académicos publicados anualmente presenta un desafío creciente para que los académicos se mantengan actualizados con las últimas investigaciones. Los resúmenes se presentan como una herramienta ágil y sencilla para acceder a esta información, permitiendo a los académicos familiarizarse con los últimos avances en sus respectivos campos. Más allá de simplificar el acceso a la información, los resúmenes también facilitan a los académicos la tarea de seleccionar lo más relevante para sus necesidades. Al identificar los temas y resultados más cruciales de un texto, los resúmenes se convierten en aliados que ahorran tiempo y esfuerzo en la lectura exhaustiva de textos académicos.

Es por esto, que diversas soluciones han surgido para abordar la creciente demanda de resúmenes efectivos. Además de las metodologías manuales, los generadores automáticos de resúmenes, como Copilot, Chat GPT y Google Gemini, han destacado como respuestas tecnológicas avanzadas. Estas herramientas emplean algoritmos de procesamiento del lenguaje natural para extraer y resumir información clave de manera automatizada. Su enfoque eficiente es particularmente valioso, aunque a veces pueden carecer de la perspicacia humana en términos de relevancia y matices.

Por otro lado, la opción manual persiste, donde los académicos dedican tiempo a la elaboración manual de resúmenes. Aunque este enfoque garantiza una comprensión profunda, se enfrenta al desafío del consumo significativo de tiempo, especialmente en un entorno académico que demanda eficiencia.

Ahora bien, mientras que los generadores automáticos de resúmenes, como Copilot, Chat GPT y Google Gemini, ofrecen eficiencia y automatización, no están exentos de desafíos. En primer lugar, la generalización de estos algoritmos puede llevar a resúmenes que carecen de la especialización

necesaria para ciertos campos académicos. Además, algunas de estas herramientas avanzadas pueden tener funcionalidades premium, restringiendo el acceso completo a sus capacidades detrás de modelos de suscripción o pagos adicionales. Adicionalmente, la capacidad de estos generadores automáticos para comprender completamente la intención y el contexto específico de ciertos textos académicos puede presentar limitaciones.

Finalmente, la Inteligencia Artificial (IA) se posiciona como un componente esencial para afrontar los desafíos que plantea la síntesis de información académica. Se han desarrollado diversas técnicas, cada una contribuyendo con una perspectiva única y una capacidad distintiva para potenciar la generación de resúmenes.

Desde los enfoques iniciales basados en estadísticas, que evaluaban la frecuencia de términos y la similitud con el título, hasta los Métodos de Aprendizaje Automático Clásico, como el clasificador bayesiano ingenuo, que utiliza características como la longitud de la oración y la presencia de palabras clave para discernir la importancia. Además, los Métodos Basados en Redes Neuronales, como el modelo Transformer, con representantes notables como BERT y UniLM, han revolucionado la sintetización automática al introducir la capacidad de contextualizar la información bidireccionalmente y capturar relaciones complejas entre palabras y frases. Estas diversas aproximaciones ofrecen un conjunto variado de herramientas para mejorar la generación de resúmenes en el ámbito académico.

1.1.1. Formulación

¿Cómo implementar un modelo de generación de resúmenes de textos académicos mediante la aplicación de técnicas de aprendizaje automático?

1.1.2. Sistematización

- ¿Cómo realizar un correcto preprocesamiento y limpieza adecuada de datos enfocado a textos académicos?
- ¿Cómo se llevará a cabo el proceso de entrenamiento de los modelos de aprendizaje automático diseñados para realizar resúmenes de textos académicos?
- ¿Cómo optimizar de manera integral los modelos seleccionados, con el objetivo de obtener un mejor desempeño a la hora de resumir textos académicos?
- ¿Cómo realizar una evaluación de los modelos seleccionados en el sistema de generación de resúmenes, que permita analizar su capacidad y rendimiento?

1.2. Objetivos

1.2.1. Objetivo General

Implementar un modelo de generación de resúmenes para textos académicos que utilice técnicas de aprendizaje automático donde se sintetice la información contenida en los documentos, sin perder la intención original del autor.

1.2.2. Objetivos Específicos

1. Realizar un riguroso preprocesamiento y limpieza de los datos de textos académicos para garantizar la calidad y preparación óptima de los datos destinados al modelo de generación de resúmenes.

2. Entrenar los modelos de aprendizaje automático seleccionados para la generación de resúmenes en textos académicos. Para ello, se revisará la literatura académica, se analizarán los datos disponibles y se realizarán experimentos con diferentes modelos. Adicionalmente, estos modelos serán entrenados utilizando técnicas de aprendizaje supervisado, semisupervisado y sin supervisión.

3. Optimizar de manera integral los modelos seleccionados, optimizando parámetros, hiperparámetros y estrategias de entrenamiento para maximizar la calidad en la generación de resúmenes en textos académicos.

4. Evaluar la capacidad de los modelos de aprendizaje automático seleccionados para generar resúmenes a partir de textos académicos. Para ello, se emplearán diversas métricas como la coherencia, la consistencia, entre otras.

1.3. Justificación

Este proyecto se enfoca en el desarrollo de un sistema automático de resumen de textos destinado al ámbito educativo, específicamente dirigido a aquellos involucrados en la creación de artículos. La contribución central de este sistema radica en proporcionar a los autores un punto de partida fundamental para la generación de resúmenes, permitiéndoles optimizar tiempo y esfuerzo de manera significativa.

En el contexto de la redacción de artículos, tarea esencial en el ámbito educativo, los profesores, investigadores y estudiantes a menudo se enfrentan a un desafío que demanda tiempo y dedicación. Este sistema de resumen automático se presenta como una herramienta valiosa al ofrecer a los autores un resumen del contenido original. Este resumen no solo brinda una visión general del texto, sino que también destaca los puntos cruciales abordados, sirviendo como un punto de partida sólido y significativo para la creación del resumen final.

La ventaja de contar con un punto de partida es innegable; en lugar de iniciar el proceso de resumen desde cero, los autores pueden aprovechar un resumen inicial. Esto no solo agiliza el proceso, sino que también proporciona una base sobre la cual los autores pueden realizar modificaciones y ajustes según sus necesidades específicas. Esta eficiencia se traduce en una mejora significativa en la calidad y rapidez de la creación de resúmenes, lo que posiciona a este proyecto como una herramienta útil en el ámbito académico.

1.4. Delimitaciones y Alcances

1. El modelo que se va a desarrollar será un prototipo funcional que genere resúmenes de textos académicos, aplicando técnicas de aprendizaje automático.

2. Se construirá un dataset propio recolectando textos académicos de diversas fuentes abiertas como Google Scholar, arXiv, entre otras. Este enfoque nos brindará la diversidad y representatividad necesarias en los datos de entrenamiento, permitiéndonos abordar una amplia variedad de disciplinas y estilos académicos.

3. El idioma principal que se manejará en los resúmenes de los textos sera el inglés. Esto se debe a que el conjunto de datos de entrenamiento está en inglés.

4. La calidad de los resúmenes generados dependerá en una parte de la calidad del dataset, dado que se toman los textos de una base de datos abierta en donde no existe ningún tipo de control sobre la calidad de los textos que se publican.

5. La calidad del modelo Seq2Seq puede estar condicionada por los recursos tecnológicos disponibles, ya que este tipo de modelos, al ser computacionalmente intensivos, requieren hardware de alto rendimiento, como GPUs o TPUs, para entrenar y ejecutar de manera eficiente. La falta de acceso a estos recursos puede afectar tanto la velocidad de procesamiento como la precisión en la generación de resúmenes.

Capítulo 2

Fundamentos Teóricos y Contexto del Aprendizaje Automático

2.1. Áreas Temáticas

Este proyecto de grado se desarrolla dentro del campo de la **Inteligencia Artificial**, con un enfoque particular en el **Aprendizaje Automático (Machine Learning)** y, más específicamente, en el **Aprendizaje Profundo (Deep Learning)**. Dentro de esta extensa subdisciplina, el proyecto se centra en implementar y analizar un modelo secuencial como **Seq2Seq (Sequence to Sequence)**. Además, se incorporan y analizan **modelos estadísticos** que, aunque no están directamente relacionados con el aprendizaje automático, permiten explorar una técnica relevante dentro de la generación automática de resúmenes, así como comparar el rendimiento entre estos enfoques. Por último, estas áreas, además de ser subdisciplinas clave dentro de la Inteligencia Artificial, forman parte fundamental de las **Ciencias de la Computación**.

2.2. Marco Teórico

2.2.1. Términos y Conceptos

- **Aprendizaje Automático:** El aprendizaje automático es el área de estudio que ayuda al computador a aprender automáticamente. El aprendizaje automático se desarrolla a partir del área del reconocimiento de patrones y la inteligencia artificial en particular, el aprendizaje automático es un subcampo de las ciencias de la computación. El aprendizaje automático está asociado a la estadística computacional y especializado en la realización de predicciones. [4]
- **Resumen automático:** El resumen automático de textos es una técnica de compresión de información que utiliza un computador para convertir textos o colecciones de textos en resúmenes breves [5]. Existen dos tipos principales de resúmenes automáticos: los extractivos, que seleccionan directamente frases o fragmentos del texto original, y los abstractivos, que generan nuevas frases basadas en la comprensión del contenido.
- **Deep Learning:** El aprendizaje profundo es una de las principales tecnologías empleadas en el aprendizaje automático. Se refiere a algoritmos que pueden imitar las operaciones del cerebro humano utilizando redes neuronales artificiales. Las redes contienen decenas o incluso cientos de «capas» de neuronas, cada una de las cuales recibe y procesa información de la capa anterior. [6]

- **Técnicas Estadísticas:** Las técnicas estadísticas se basan en la frecuencia de los términos para determinar su importancia. Las frases con términos más importantes se extraen con mayor prioridad. [7]

2.2.2. Importancia del Preprocesamiento en Modelos de IA

El preprocesamiento de datos es una etapa importante en el desarrollo de modelos de inteligencia artificial enfocados en tareas de procesamiento de lenguaje natural. Para que el modelo entienda el texto de manera efectiva, es necesario limpiar y preparar los datos en un formato que el modelo pueda interpretar fácilmente. Esto incluye:

- Normalización y Limpieza: Se eliminan caracteres especiales, se corrigen errores de formato y se reduce el texto a una forma base (e.g., convertir mayúsculas en minúsculas).
- Eliminación de Ruido: Eliminar palabras irrelevantes (stopwords), signos de puntuación y palabras que no aportan significado al modelo.
- Tokenización: Dividir el texto en palabras o frases (tokens) que el modelo usará para el análisis.

2.2.3. Modelos Estadísticos Para Generación de Resúmenes

Cuando se habla de modelos estadísticos, se puede definir como una representación matemática que busca describir, explicar y predecir fenómenos reales a partir de un conjunto de datos, incorporando de forma inherente al mundo real mediante el uso de probabilidades y estadísticas. Estos modelos constituyen una herramienta fundamental para analizar relaciones entre variables, hacer inferencias sobre diversas aplicaciones a partir de muestras y realizar predicciones basadas en patrones observados. Ahora bien, dadas las diversas aplicaciones que tienen estos modelos, este proyecto se centra específicamente en la generación de resúmenes automáticos de textos académicos.

En el ámbito de la generación de resúmenes automáticos, surgieron dos enfoques principales que se han perfeccionado con el paso de los años. Precisamente, los modelos estadísticos fundamentan uno de estos enfoques, conocidos como resúmenes extractivos. Los resúmenes extractivos se destacan principalmente por enfocarse en identificar y seleccionar las frases o fragmentos más relevantes directamente del texto original. Para lograr esto, han surgido diversos modelos que analizan la frecuencia de las palabras, la posición de las frases, la similitud semántica y otras métricas cuantitativas para determinar qué partes del texto contienen la información más significativa.

En este proyecto de grado, dentro de los modelos estadísticos, se explora el modelo estadístico de Luhn. Este modelo, junto a otros del mismo enfoque, se utilizarán como referencia para realizar una comparación entre el resumen extractivo versus el resumen abstractivo. A continuación, en la figura 2.1 se presenta la estructura general seguida para el modelo seleccionado y, en términos generales, por la mayoría de los modelos estadísticos:

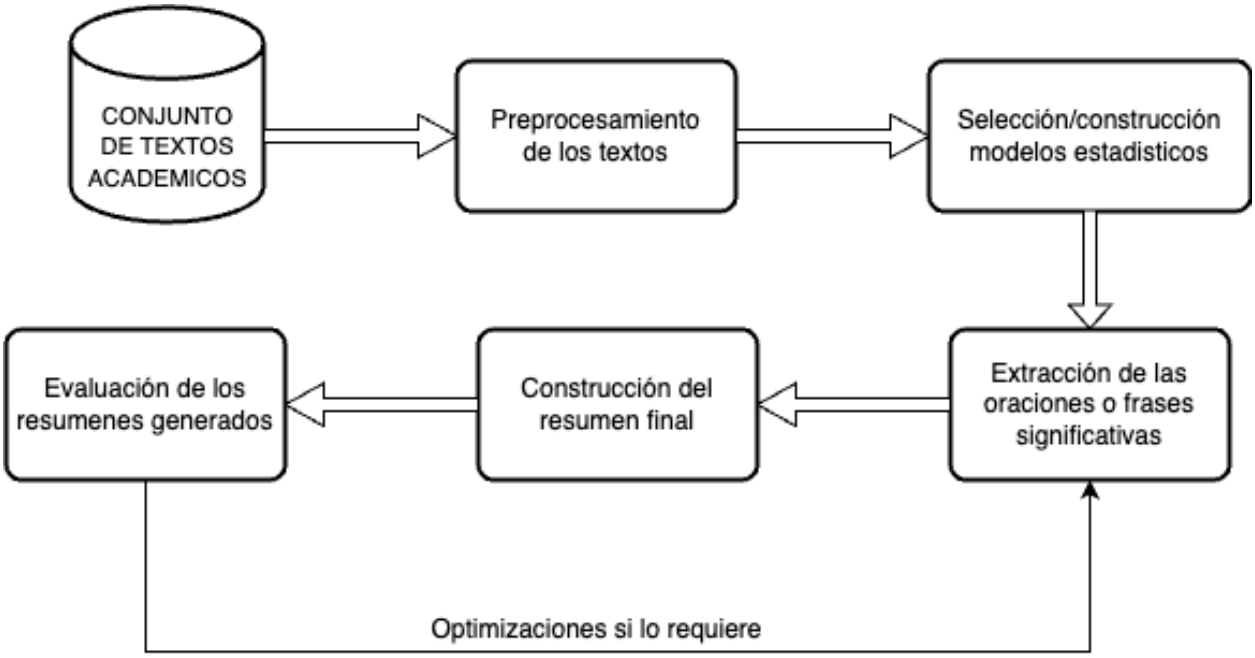


Figura 2.1: Estructura General Modelos Estadísticos
Gráfica propia de los autores

2.2.4. Modelo de Luhn

El método heurístico de Luhn, desarrollado por Hans Peter Luhn en 1958, es uno de los primeros enfoques extractivos para la generación de resúmenes de textos. El algoritmo de Luhn es un método que está basado en TF-IDF, en donde se seleccionan únicamente aquellas palabras que tienen una gran importancia según la alta frecuencia. Luhn contempla que aquellas palabras que aparezcan al principio del documento tienen un mayor peso. De acuerdo con K. Srividya et al. [8], este método puede ser aplicado en dos etapas diferentes:

Durante la primera etapa, se intenta decidir qué frases son las más importantes para el significado del texto. Luhn plantea que esto se puede lograr realizando primero una evaluación de la frecuencia de las palabras en el texto y posteriormente encontrando palabras clave que tienen una gran importancia, pero que sean símbolos no esenciales, como por ejemplo (“the”, “and”, “for”, etc.) también conocidas como “stopwords”.

Durante la segunda etapa, se incluye la búsqueda de los tokens más relevantes y la obtención de un conjunto de palabras que pueden ser más relevantes. Durante esta etapa ocurren tres subetapas importantes:

- Se inicia convirtiendo el contenido de la oración en una expresión matemática o un vector, facilitando el proceso para el modelo. En esta misma etapa, se eliminan todos aquellos tokens que no son esenciales y que no afectan directamente el contenido del texto.
- Como segundo paso, se realiza un conteo de las palabras valiosas que quedan, adicional a esto, aquellas palabras que suelen aparecer al principio del documento tienen un mayor peso y significado. Durante este paso, se usa una técnica para calcular la puntuación de las oraciones para evaluar la prioridad. Esta puntuación esta dada por la siguiente formula:

$$\text{Puntaje} = \frac{(\text{ndps})^2}{(\text{cdps})} \tag{2.1}$$

  Donde ndps es el número de palabras significativas y cdps es la cantidad de palabras significativas.
- Por ultimo, una vez seleccionadas aquellas frases que son mas relevantes y que contienen mayor significado, se juntan para formar el resumen final.

2.2.5. Modelos de Aprendizaje no Supervisado Para Generación de Resúmenes

Los modelos de aprendizaje no supervisado son una rama del aprendizaje automático en la que los algoritmos trabajan con datos no etiquetados. Esto significa que, en el aprendizaje no supervisado, tenemos la variable de entrada (P), pero no hay una variable de salida [9]. En otras palabras, los conjuntos de datos no proporcionan explícitamente una salida objetivo o etiquetas predefinidas, por lo tanto, en estos modelos el objetivo principal es descubrir patrones, estructuras o relaciones subyacentes en los conjuntos de datos.

Al igual que los modelos estadísticos, los modelos de aprendizaje no supervisado se han consolidado como herramientas fundamentales en la generación de resúmenes automáticos, particularmente en el enfoque extractivo. Estos modelos destacan por su capacidad para identificar y seleccionar las frases u oraciones más relevantes del texto original, permitiendo así construir un resumen representativo sin modificar el contenido original.

En este proyecto, para complementar el enfoque extractivo iniciado con el modelo estadístico de Luhn, se exploran los modelos K-Means y TextRank. K-Means, junto con Luhn, constituyen los dos modelos principales que serán evaluados dentro del enfoque extractivo, en contraste con el enfoque abstractivo que se presentará más adelante. Por último, el modelo TextRank desempeña un papel fundamental al formar parte de la estructura básica utilizada en la construcción del modelo abstractivo. A continuación, en la figura 2.2 se presenta la estructura general seguida por los dos modelos seleccionados y, en términos generales, por la mayoría de los modelos pertenecientes al enfoque no supervisado:

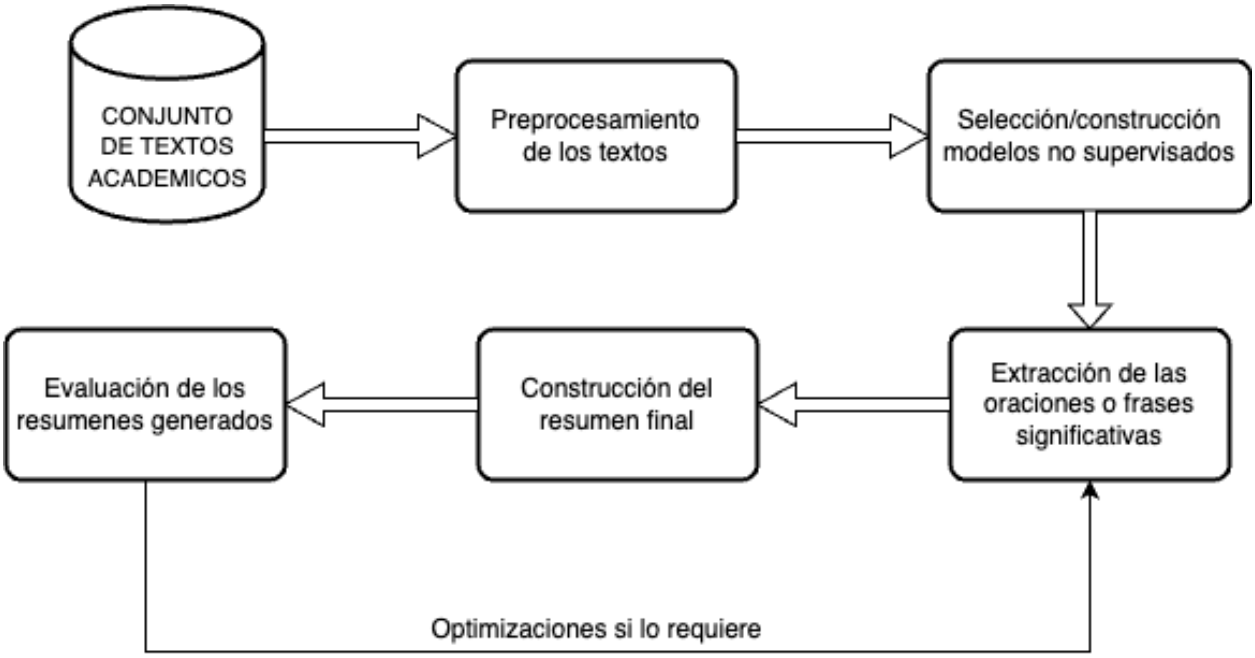


Figura 2.2: Estructura General Modelos No Supervisados
Gráfica propia de los autores

2.2.6. Modelo K-Means

El modelo K-Means es un algoritmo de aprendizaje no supervisado que se utiliza para la agrupación o "clustering" de datos. El objetivo principal de este modelo es tomar el conjunto de datos de entrada y dividirlo en k grupos o clústeres, de tal manera que cada elemento perteneciente a cada clúster sea lo más similar entre sí, mientras que los clústeres sean lo más diferentes posibles entre ellos.

En este contexto, el método de K-Means ha demostrado ser particularmente útil en la generación de resúmenes automáticos bajo el enfoque extractivo. Este método permite identificar y seleccionar las oraciones o fragmentos de texto más representativos del documento original, construyendo así un resumen final sin necesidad de modificar las palabras o frases originales. A continuación, se detalla cómo se puede relacionar K-Means con la tarea de generación de resúmenes:

1 En los textos académicos, las oraciones y los párrafos suelen estar organizados en torno a temas principales o subtemas. El clustering permite agrupar estas unidades de texto según su similitud semántica:

- **Similitud Semántica:** La posibilidad de captar la semántica, se logra a través de la incrustación de palabras ("Word Embedding"), ya que permite crear una representación

vectorial para cada palabra de una longitud determinada, donde el vector estaría formado por ceros, excepto el elemento que representa las palabras. Aquellas palabras que tienen un significado similar ocupan una posición espacial mas cercana. Esto se puede ver representado de la siguiente forma: [10]

$$sin(X,\ Y) = cos(\theta) = \frac{(X \cdot Y)}{\|X\|\|Y\|} \tag{2.2}$$

- **Representaciones Claves:** Aquellas oraciones que estén mas cercanas al centroide de un clúster, serán las mas representativas del tema principal de ese clúster.

2 La función de K-Means se puede ver de la siguiente forma:

- **Identificación de Temas Principales:** Al dividir todas las oraciones en k clústeres, el algoritmo identifica las áreas temáticas principales del texto.
- **Selección Extractiva:** Una vez conformados los clústeres, las oraciones mas cercanas a los centroides de cada clúster se seleccionan como las mas representativas. Estas oraciones tienen a resumir el contenido de su grupo temático con una mayor precisión.

Por último, el modelo se puede ver representado algorítmicamente de la siguiente forma [10]:
Sean M = {*m1*, *m2*, *m3*, ...} la colección de datos y V = {*v1*, *v2*, *v3*, ...} los centroides.

1 Seleccionar los centros del clúster c mediante k-means++.

2 Determinar la diferencia entre los puntos de datos individuales y los centros de los grupos o clústeres.

3 Asignar cada punto de datos al centroide del clúster más cercano, es decir, al centroide que se encuentra a la menor distancia de dicho punto.

4 Calcular el nuevo centro de los clusteres aplicando:

$$v_i = \frac{(1)}{(c_i)} \sum_{j=1}^{c_i} m_i \tag{2.3}$$

5 Calcular de nuevo la distancia entre cada punto de datos y los centros de clúster recién obtenidos.

6 Si no hay ningún punto de datos reasignado, repetir el paso n.°3.

El resumen final estará compuesto por la selección de los representantes más valiosos. Estos pueden obtenerse de diversas formas, como por ejemplo seleccionar aquel clúster que contenga la mayor cantidad de oraciones; esto podría indicarnos que las oraciones más valiosas pertenecen a ese clúster o, por el contrario, seleccionar únicamente n cantidad de oraciones representativas de cada clúster y construir un resumen más variado con lo mejor de cada uno.

2.2.7. Modelo TextRank

El modelo TextRank es un algoritmo de aprendizaje no supervisado basado en grafos, el cual esta inspirado en el algoritmo PageRank de Google encargado de rankear las páginas web. El objetivo principal de este modelo es determinar la importancia relativa de los modelos de un texto (palabras, frases u oraciones) basándose en sus relaciones con otros elementos.

TextRank se ha convertido en un modelo esencial en el enfoque extractivo porque aborda de manera eficiente y efectiva el problema de identificar las partes más relevantes de un texto. Su diseño aprovecha la estructura interna del texto para seleccionar las oraciones más representativas, lo que lo hace particularmente valioso en tareas de resumen donde el objetivo es preservar el contenido original.

El flujo de trabajo comienza con la creación de un grafo en el que cada nodo representa una oración del texto original que queremos resumir. Luego, conectamos cada oración de este grafo con otras oraciones similares [8].

Definición del Grafo: Para esta tarea, se puede definir un grafo de la forma $G(V,\ E,\ W)$, donde cada oración en el texto es un vértice (V), las aristas (E) y los pesos (W) se representan con el valor de la similitud entre las oraciones presentes en el texto. Por lo tanto, la matriz de similitud se debe proporcionar como argumento para la creación del grafo [8]. En este trabajo, para el cálculo de los pesos en el grafo se utiliza la similitud del coseno que está dada por la siguiente ecuación:

$$\text{Similitud del coseno} = \cos(\theta) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\|\|\vec{B}\|} \tag{2.4}$$

Donde los vectores $\vec{A}$ y $\vec{B}$ son la representación vectorial de las frases u oraciones obtenidas tras un proceso de transformación con ayuda de un embedding como Word2Vec o TF-IDF.

El resultado final de este proceso es un grafo denso que representa el texto completo. A partir de este grafo, se aplica PageRank para calcular la importancia de cada vértice. Por último, se seleccionan las oraciones más representativas y se construye el resumen final.

2.2.8. Modelos de Deep Learning Para Generación de Resúmenes

El Deep Learning o aprendizaje profundo es una rama del aprendizaje automático caracterizada por el uso de redes neuronales profundas capaces de captar patrones complejos en grandes volúmenes de datos. En el contexto del procesamiento de lenguaje natural (NLP), los modelos de deep learning han facilitado avances importantes, especialmente en la generación de texto y resúmenes automáticos, con arquitecturas populares como Seq2Seq, Transformers, y modelos preentrenados

como BERT y GPT.

Los modelos tradicionales de redes neuronales recurrentes (RNN) y sus variantes como LSTM (Long Short-Term Memory) y GRU (Gated Recurrent Units) fueron inicialmente muy populares para la generación de texto. Sin embargo, con el tiempo, los Transformers han revolucionado el campo al permitir un procesamiento paralelo y mejor manejo de dependencias de largo alcance, lo que es especialmente valioso en tareas como la generación de resúmenes. [11]

El Deep Learning conforma el segundo enfoque más importante en la generación de resúmenes automáticos, llamado abstractivo. Este enfoque va más allá de simplemente seleccionar partes del texto original (como en el enfoque extractivo). Este enfoque genera nuevos enunciados y reformula el contenido para producir un resumen que capture las ideas principales de una manera más concisa y natural.

En este proyecto, se explora el modelo Seq2Seq (Sequence-to-Sequence) dentro del ámbito de los modelos de deep learning. Este enfoque se centra en la generación abstractiva de resúmenes automáticos, permitiendo analizar su comportamiento y compararlo con los resultados obtenidos mediante modelos extractivos.

2.2.9. Modelo Seq2Seq

El modelo Seq2Seq (Sequence-to-Sequence) es una arquitectura de redes neuronales diseñada para mapear una secuencia de entrada a una secuencia de salida, independiente de la longitud de la secuencia. Este modelo ha sido fundamental en muchas tareas de procesamiento de lenguaje natural (NLP), como la traducción automática, la generación de texto y lo que evalúa este proyecto, la generación de resúmenes automáticos.

Este modelo se basa en una arquitectura de coder-encoder (codificador-decodificador) que transforma una secuencia de entrada en una secuencia de salida. A continuación, en la figura 2.3 se puede observar una arquitectura del modelo Seq2Seq:

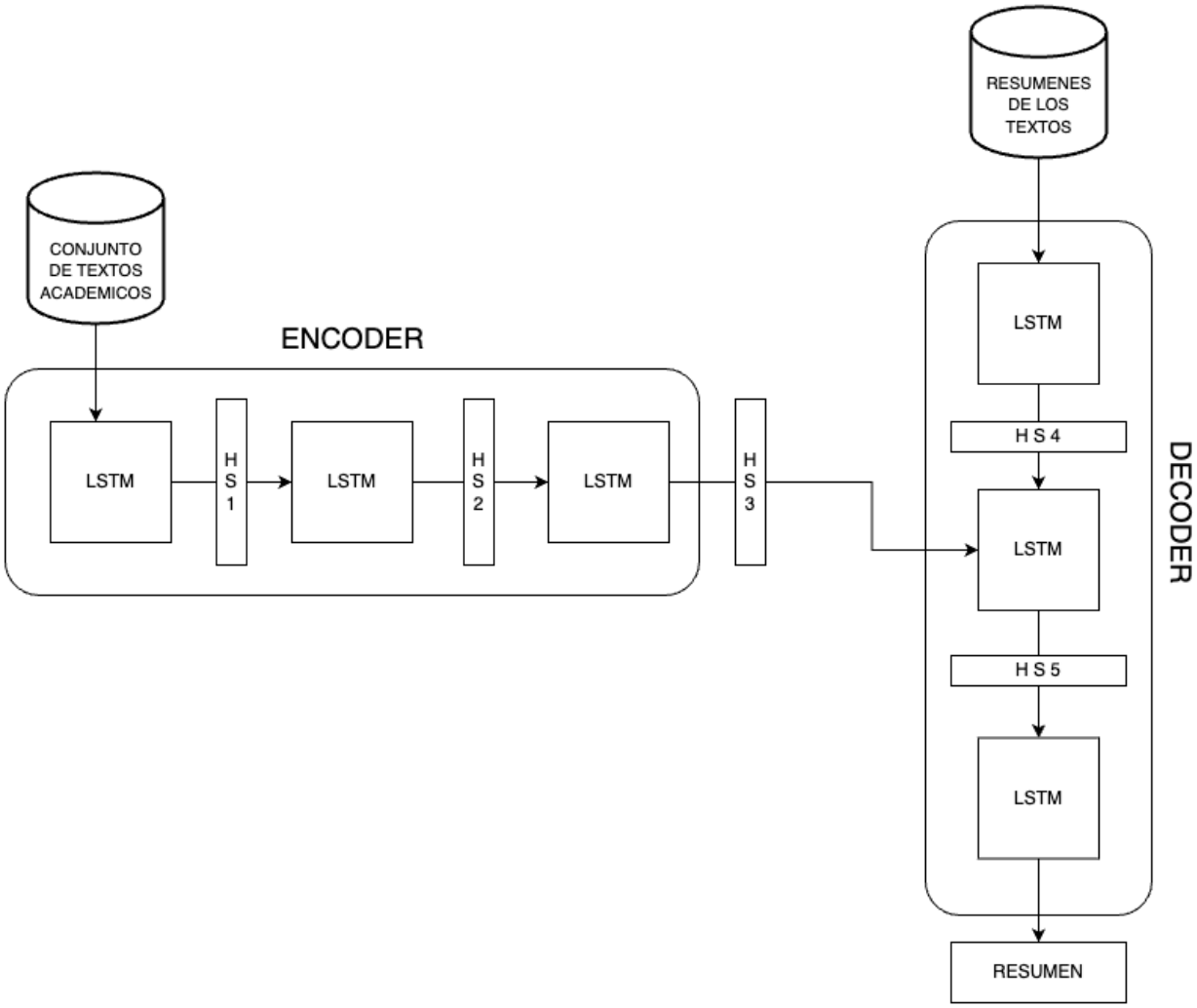


Figura 2.3: Estructura General Modelo Seq2Seq
Gráfica propia de los autores

Partes principales de una arquitectura Seq2Seq:

1. **Codificador (Encoder):**
   - El codificador toma la secuencia de entrada y la convierte en una representación interna (vector de contexto).
   - Generalmente esta compuesto por una red neuronal recurrente (RNN), como una LSTM (Long Short-Term Memory) o GRU (Gated Recurrent Unit).
   - La tarea del codificador es procesar la secuencia de entrada de manera que se comprime toda la información relevante en un vector de contexto de tamaño fijo.

- El codificador procesa la secuencia de entrada de manera secuencial, manteniendo un estado oculto que se actualiza con cada nuevo elemento de la secuencia.

2. **Vector de Contexto:**
   - El vector de contexto es una representación comprimida de la secuencia de entrada, que captura la información más relevante para la tarea de generación de salida.
   - Este vector se pasa al decodificador para que genere la secuencia de salida.
3. **Decodificador (Decoder):**
   - El decodificador toma el vector de contexto generado por el codificador y lo utiliza para generar la secuencia de salida.
   - Al igual que el codificador, el decodificador suele ser una RNN, LSTM o GRU, pero en lugar de procesar la entrada, genera la salida de manera secuencial, palabra por palabra.
   - El decodificador se entrena para predecir la siguiente palabra en la secuencia, dado el vector de contexto y las palabras previas generadas.

En los modelos Seq2Seq, los hiperparámetros representan configuraciones clave que influyen significativamente en el rendimiento y la eficiencia del modelo. A diferencia de los parámetros del modelo, los hiperparámetros no se aprenden durante el proceso de entrenamiento, sino que deben definirse previamente. Entre los diversos hiperparámetros ajustables, los más relevantes a considerar al construir un modelo Seq2Seq incluyen:

1. **Hiperparámetros Relacionados con la Arquitectura:**
   - **Número de Capas:**
     - Especifica cuántas capas tiene cada componente (encoder y decoder). Más capas pueden capturar patrones más complejos, pero pueden requerir más datos para evitar el sobreajuste.
   - **Dimensión del Embedding:**
     - Representa el tamaño de las representaciones vectoriales para las palabras. Un tamaño adecuado captura mejor las relaciones semánticas.
2. **Hiperparámetros Relacionados con el Entrenamiento:**
   - **Tasa de Aprendizaje (Learning Rate):**
     - Controla el tamaño de los pasos que da el modelo al ajustar los pesos durante el entrenamiento. Una tasa demasiado alta puede hacer que el modelo no converja, mientras que una muy baja puede ralentizar el aprendizaje.
   - **Tamaño del Lote (Batch Size):**

  - Define cuántos ejemplos se procesan en paralelo durante una iteración de entrenamiento. Los lotes más grandes pueden estabilizar el entrenamiento, pero requieren más memoria.
- **Número de Épocas (Epochs):**
  - Determina cuántas veces el modelo pasa por el conjunto de datos completo. Más épocas pueden mejorar el rendimiento, pero también aumentar el riesgo de sobreajuste.
- **Dropout:**
  - Proporción de unidades que se desactivan aleatoriamente durante el entrenamiento para evitar el sobreajuste.

2.2.10. Métricas de Evaluación para Generación de Resúmenes

2.2.10.1. ROUGE

Las métricas ROUGE (Recall-Oriented Understudy for Gisting Evaluation) son un conjunto de herramientas ampliamente utilizadas para evaluar la calidad de textos generados automáticamente, como resúmenes, traducciones o respuestas generadas en sistemas de procesamiento de lenguaje natural. Estas métricas comparan el texto generado con uno de referencia, evaluando la superposición de palabras, n-gramas y estructuras textuales para determinar su relevancia y precisión. Según Lin, ROUGE proporciona un marco de evaluación orientado al recall para evaluar automáticamente resúmenes generados por máquina. [12]

Principales variantes de ROUGE:

- **ROUGE-1 (unigramas)**: Esta métrica mide la coincidencia de palabras individuales entre el texto generado y el texto de referencia. Es útil para evaluar si los elementos básicos del contenido han sido capturados. Por ejemplo, si las palabras clave más importantes están presentes en el texto generado, el valor de ROUGE-1 será mayor.
  - **Interpretación del recall**: Proporción de palabras individuales del texto de referencia que también aparecen en el texto generado.
  - **Interpretación de la precisión**: Proporción de palabras generadas que son relevantes respecto al texto de referencia.
- **ROUGE-2 (bigramas)**: Evalúa las coincidencias de pares consecutivos de palabras (bigramas). Esta métrica es más estricta que ROUGE-1, ya que mide la capacidad del modelo para captar relaciones locales o contextuales entre términos.
  - **Recall**: Indica cuántos bigramas relevantes del texto de referencia fueron incluidos en el texto generado.
  - **Precisión**: Mide cuántos bigramas generados son válidos en comparación con el texto de referencia.

- **ROUGE-L (subsecuencia común más larga, LCS)**: Basado en la subsecuencia común más larga, esta métrica evalúa similitudes estructurales entre el texto generado y el de referencia. ROUGE-L es útil para medir cómo el texto preserva la coherencia y el flujo lógico de las ideas.

  - **Recall**: Mide cuánta de la estructura lógica del texto de referencia fue capturada.
  - **Precisión**: Evalúa qué tan relevante es la estructura generada en comparación con el texto de referencia.
  - Un mayor ROUGE-L indica que el texto generado mantiene mejor el orden lógico y la relación entre ideas.

2.2.10.2. BERTScore

La métrica BERTScore es una técnica avanzada para evaluar la calidad de texto generado, como en tareas de generación de resúmenes, traducciones automáticas o generación de texto. Esta métrica fue introducida para abordar las limitaciones de métricas más tradicionales como BLEU, ROUGE y METEOR, que se basan exclusivamente en coincidencias léxicas exactas y no capturan adecuadamente la similitud semántica.

Dada una oración de referencia $x =< x_1, x_2, ..., x_k >$ y una oración candidata $\widehat{x} =< x_1, x_2, ..., x_l >$, se utilizan representaciones contextuales (Embeddings) para representar los tokens y calcular las correspondencias utilizando la similitud de coseno, opcionalmente ponderada con puntajes de frecuencia inversa de documentos (IDF) [13]. La Figura 2.4 ilustra el cálculo.

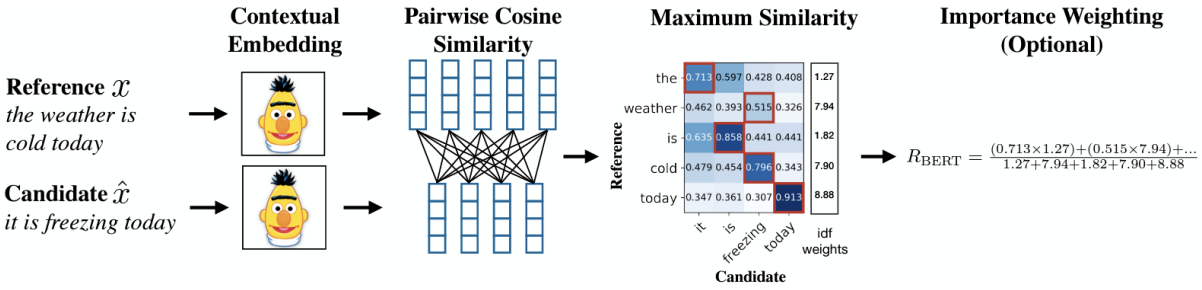


Figura 2.4: Ilustración del cálculo de la métrica de recall R_{BERT} [13].

La importancia del uso de representaciones contextuales como BERT, es porque permite generar representaciones vectoriales diferentes para la misma palabra en diferentes contextos.

Para obtener el puntaje completo, se empareja cada token en x con un token en $\widehat{x}$ y se obtiene el *Recall* y cada token en $\widehat{x}$ con un token en x para obtener *Precision*. Se utiliza un método de emparejamiento voraz (*greedy matching*) para maximizar el puntaje de similitud de emparejamiento,

donde cada token se empareja con el token más similar en la otra oración. Luego, se combina la *Precision* y el *Recall* para calcular una medida *F1*, tal como se plantea en [13].

Adicionalmente, [13] también plantea que BERTScore permite incorporar fácilmente la ponderación de importancia. Se experimenta con puntajes de frecuencia inversa de documentos (idf) calculados a partir del corpus de prueba.

2.2.11. Lo Más Actual en Generación de Resúmenes

La generación automática de resúmenes de texto ha evolucionado significativamente gracias a los avances en inteligencia artificial y procesamiento del lenguaje natural (PNL). A continuación, se describen las tendencias más recientes y las herramientas más destacadas en este ámbito:

Avances Tecnológicos Recientes:

1. **Modelos de Lenguaje Preentrenados de Gran Escala:** Modelos como GPT-4 , BERT y T5 han transformado la capacidad de generar resúmenes de alta calidad. Estos modelos entienden contextos complejos y pueden generar resúmenes coherentes y relevantes.
2. **Mecanismos de Atención y Transformers:** Las arquitecturas de transformers, introducidas por el modelo Attention Is All You Need , son la base de los modelos modernos. Los mecanismos de atención permiten al modelo identificar las partes más importantes del texto para resumir.
3. **Resúmenes Híbridos (Extractivo + Abstractivo):** En lugar de limitarse a un enfoque, los modelos actuales combinan técnicas extractivas (selección de frases clave) y abstractivas (generación de texto nuevo), logrando un equilibrio entre precisión y creatividad.

Herramientas y Plataformas Destacadas:

1. **Copilot de Microsoft:** Integrado en Microsoft Word, permite generar resúmenes directamente en documentos, identificando puntos clave y simplificando el contenido.
2. **Gemini:** El modelo generativo de Google incluye capacidades de resumen optimizadas para información web y documentos.
3. **ChatGPT y GPT-4:** OpenAI ha optimizado sus modelos para resumir textos extensos, incluyendo documentos técnicos, con capacidades ajustables según el nivel de detalle requerido.

Tendencias futuras:

1. **Mayor personalización:** Los modelos futuros permitirán ajustar el nivel de detalle y el enfoque del resumen según las necesidades del usuario.

2. **Resúmenes Multimodales:** Integración de texto con imágenes, gráficos o tablas para generar resúmenes más completos y visuales.
3. **IA Generativa Multilingüe:** Mejora en la capacidad de generar resúmenes en múltiples idiomas, con traducciones integradas.
4. **Optimización para la Web:** Herramientas diseñadas para generar resúmenes concisos de contenido web, como artículos, blogs y publicaciones de redes sociales.

2.3. Trabajos Relacionados

2.3.1. "A Hybrid Approach for Automatic Text Summarization and Translation based On Luhn, Pegasus, and Textrank Algorithms"

En los avances recientes en la generación automática de resúmenes de texto, se ha propuesto un enfoque híbrido que combina técnicas extractivas y abstractivas para mejorar la calidad y coherencia de los resúmenes generados. Este modelo híbrido integra los algoritmos Luhn y TextRank, que son métodos extractivos tradicionales, con el modelo Pegasus, una técnica avanzada de resumen abstractivo.

Este enfoque aprovecha las fortalezas de ambos métodos: TextRank clasifica las oraciones según su importancia, Pegasus realiza un resumen abstractivo para reescribir el contenido, y el algoritmo Luhn refina el resumen final eliminando palabras redundantes y de parada.

El modelo híbrido propuesto fue evaluado frente a modelos existentes como BERT, GPT-2 y XLNet utilizando evaluación ROGUE, demostrando un rendimiento superior con calificación promedio más alta. Además, este modelo fue implementado como una aplicación web, permitiendo a los usuarios ingresar texto para resumirlo y traducirlo a varios idiomas.

La integración de estas técnicas aborda las limitaciones de usar métodos puramente extractivos o abstractivos de manera aislada, logrando resúmenes más cercanos a los generados por humanos en términos de contexto y coherencia.

Las mejoras futuras de este modelo híbrido incluyen la incorporación de técnicas de resumen no supervisadas y la ampliación de su aplicabilidad a múltiples idiomas y entradas de texto más generalizadas. Este trabajo destaca el potencial de combinar diferentes estrategias de resumen para lograr resúmenes más precisos y significativos.[8]

2.3.2. "Automatic Text Review Summarization of Digital Library System Application using TextRank Algorithm and TF-IDF"

La generación de resúmenes de texto ha sido ampliamente investigada y aplicada en diversos dominios para condensar grandes volúmenes de texto en resúmenes concisos. Existen dos enfoques

principales para la generación de resúmenes de texto: extractivo y abstractivo. El resumen extractivo selecciona oraciones, frases o palabras clave directamente del texto original, mientras que el resumen abstractivo genera nuevas oraciones que capturan la esencia del texto original.

Algoritmo TextRank: El algoritmo TextRank, inspirado en el algoritmo PageRank, es un método popular para la generación de resúmenes extractivos. Construye un gráfico donde las oraciones son nodos, y los aristas representan la similitud entre oraciones. El algoritmo clasifica las oraciones según su importancia y selecciona las mejores clasificadas para formar el resumen. TextRank ha sido utilizado de manera efectiva en diversos estudios.

Métricas de evaluación: El desempeño de los algoritmos de generación de resúmenes se evalúa comparativamente utilizando la métrica ROUGE, que mide la superposición de n-gramas entre el resumen generado y los resúmenes de referencia. Las puntuaciones ROUGE más altas indican mejor calidad y relevancia de los resúmenes.

En conclusión, el algoritmo TextRank, especialmente cuando se combina con técnicas de extracción de características como TF-IDF, ha demostrado ser un método efectivo para la generación de resúmenes extractivos en diversos dominios, proporcionando información valiosa a partir de grandes conjuntos de datos de reseñas de usuarios. [14]

2.3.3. "Advancing Abstractive Bangla Text Summarization: A Deep Learning Approach Using Seq2seq Encoder-Decoder Model and T5 Transformer"

Se han explorado varios enfoques en el ámbito de la generación de resúmenes de texto en bengalí, centrándose principalmente en métodos extractivos y abstractivos. Las técnicas extractivas, como las propuestas por Kamal Sarkar, utilizan la Frecuencia de Términos (TF) y la Frecuencia Inversa de Documentos (IDF) para priorizar palabras visualmente prominentes, aunque pueden pasar por alto palabras menos evidentes pero contextualmente importantes. Métodos híbridos, como el introducido por Islam et al., combinan puntuación de sentimientos, clasificación de palabras clave y clasificación de texto para generar resúmenes a partir del 40 % superior del documento.

En el ámbito del aprendizaje profundo, Al Munzir et al. Emplearon redes neuronales recurrentes (RNN) basadas en LSTM y GRU para la generación de resúmenes de documentos en bengalí, logrando clasificación ROUGE favorables. Modelos más avanzados, como el Transformer T5, han sido evaluados por Borah et al. y Zolotareva et al., mostrando resultados prometedores en la generación de resúmenes abstractivos en varios conjuntos de datos. Sin embargo, estos estudios a menudo destacan la necesidad de conjuntos de datos más grandes y un ajuste fino adicional para mejorar el rendimiento.

Investigaciones recientes de Hasan et al. han logrado avances significativos en la generación de

resúmenes abstractivos en bengalí utilizando modelos de aprendizaje profundo como el codificador-decodificador Seq2seq y el Transformer T5. Su estudio, evaluado en el conjunto de datos AUST NLP Research y los conjuntos de datos estándar de BNLPC, demostró el rendimiento superior del modelo Seq2seq, alcanzando una puntuación F1 promedio de 0.8551. Esta investigación destaca el potencial de los modelos de aprendizaje profundo para capturar estructuras lingüísticas complejas y generar resúmenes coherentes, allanando el camino hacia técnicas más efectivas de generación de resúmenes de texto en bengalí. [15]

Capítulo 3

Preparación y Preprocesamiento de los Datos

3.1. Dataset

Para este trabajo se utilizó el conjunto de datos *Scientific Papers*, disponible a través de TensorFlow Datasets [16]. Este conjunto de datos puede descargarse directamente mediante TensorFlow Datasets, lo que facilita su integración en proyectos de aprendizaje automático.

De este conjunto, se trabajó únicamente con los datos provenientes de arXiv, que incluyen artículos académicos de disciplinas como matemáticas, ciencias y tecnología, junto con sus resúmenes generados por los autores. En total, el subconjunto de arXiv contiene aproximadamente 215,000 documentos, estructurados en texto completo y resumen.

Este recurso está orientado a tareas de procesamiento de lenguaje natural, como la generación y evaluación de resúmenes automáticos. Trabajar únicamente con el subconjunto de arXiv permite enfocar los experimentos en temas científicos amplios y diversificados, facilitando la evaluación de modelos para la síntesis y comprensión de contenido académico.

3.2. Especificación y entendimiento de los datos

En la Tabla 3.1 se describen las características principales de las columnas del dataset utilizado, incluyendo su propósito y estructura. Este conjunto de datos contiene información relevante sobre artículos académicos provenientes de arXiv, y cada columna está diseñada para representar diferentes aspectos del contenido del artículo, como su identificación, texto completo, resúmenes, secciones, y etiquetas. A continuación, se presenta una descripción detallada de cada columna para proporcionar una comprensión más profunda de cómo se organiza la información en el dataset.

Columna	Descripción breve
`article_id`	Identificador único del artículo en arXiv.
`article_text`	Texto completo del artículo académico dividido en párrafos.
`abstract_text`	Resumen del artículo dividido en frases.
`labels`	Etiquetas asociadas al artículo (puede ser nulo).
`section_names`	Nombres de las secciones presentes en el artículo.
`sections`	Contenido de las secciones, organizadas de forma paralela a `section_names`.

Cuadro 3.1: Descripción de las columnas del dataset.

- `article_id`: Sirve como un identificador único para cada artículo dentro de arXiv. Este identificador permite rastrear el artículo correspondiente en la plataforma y es esencial para tareas de organización y referencia.
- `article_text`: Contiene el texto completo del artículo científico, dividido en varios párrafos. Este texto representa el contenido principal del artículo, incluyendo detalles técnicos, discusiones y conclusiones. La segmentación en párrafos facilita su análisis y procesamiento, especialmente para tareas de comprensión de lenguaje natural.
- `abstract_text`: Incluye el resumen del artículo, compuesto por una lista de oraciones. Este resumen condensa la información más importante del trabajo y está diseñado para ser utilizado en tareas de generación o análisis de resúmenes automáticos, proporcionando una vista rápida del contenido sin necesidad de leer el texto completo.
- `labels`: Puede contener etiquetas asociadas al tema del artículo. Estas etiquetas son útiles en tareas de clasificación o agrupamiento, aunque en algunos casos esta columna puede ser nula si no se asignaron etiquetas específicas al artículo.
- `section_names`: Lista los nombres de las secciones presentes en el artículo, como *Ïntroducción"*, *Resultados"* o *"Discusión"*. Esto permite estructurar el contenido de manera jerárquica y entender mejor la organización interna de cada artículo.
- `sections`: Incluye el contenido detallado de cada sección del artículo, organizado en paralelo con los nombres presentes en `section_names`. Cada elemento de esta columna corresponde a una sección completa del texto, lo que facilita su análisis por partes según el contexto específico.

3.3. Preprocesamiento

3.3.1. Preprocesamiento para Seq2Seq

El preprocesamiento del conjunto de datos es una fase fundamental para garantizar la calidad y la relevancia de los datos utilizados. Este proceso fue diseñado con el objetivo de transformar los

textos académicos en un formato estructurado y limpio.

En primer lugar, se realizó la carga de los datos desde distintas fuentes que contenían tanto el cuerpo de los textos como los resúmenes asociados. Los datos se organizaron en dos conjuntos principales: uno para el entrenamiento y otro para la evaluación. Las partes del conjunto de entrenamiento se combinaron en una sola estructura para disponer de un volumen más amplio de información, mientras que los datos de prueba permanecieron separados, asegurando su uso exclusivo durante las fases finales de evaluación.

Una vez cargados, los datos pasaron por una etapa de limpieza exhaustiva. Durante este proceso, se eliminaron secciones que no aportaban valor a los objetivos del modelo, como agradecimientos, referencias bibliográficas y conclusiones. Adicionalmente, se dividieron los textos en oraciones individuales para facilitar su análisis. Se llevaron a cabo transformaciones para eliminar elementos no textuales, como etiquetas específicas, fórmulas matemáticas, enlaces y caracteres no alfabéticos. Estas modificaciones garantizaron que los textos fueran claros y homogéneos, maximizando su utilidad para el modelo.

Texto original	→	**Texto procesado**
Additive models @xcite provide an important family of models for semiparametric regression or classification. See @xcite for details. Acknowledgments to our sponsors.	→	Additive models provide an important family of models for semiparametric regression or classification.

Cuadro 3.2: Limpieza de datos

A continuación, se seleccionaron los fragmentos más relevantes de cada texto utilizando el algoritmo de TextRank. Este algoritmo evalúa las conexiones entre las oraciones en función de su contenido, construyendo un modelo que prioriza las más representativas. De esta manera, se seleccionaron las oraciones clave de cada texto, reduciendo su longitud total sin comprometer la calidad ni la integridad de la información esencial.

Texto original	→	**Texto procesado**
Additive models provide flexibility for regression. They reduce the curse of dimensionality. Fully nonparametric models are less interpretable.	→	Additive models provide flexibility for regression. They reduce the curse of dimensionality.

Cuadro 3.3: Selección de fragmentos relevantes con TextRank

Para mantener la consistencia del conjunto de datos, se establecieron límites máximos de longi-

tud tanto para los textos como para los resúmenes. Aquellos que excedían estas restricciones fueron descartados, asegurando que todas las muestras utilizadas cumplieran con las dimensiones definidas. Posteriormente, los textos y resúmenes resultantes se almacenaron en un formato tabular que facilita su manipulación en las siguientes etapas.

En los resúmenes, se añadieron marcas especiales al inicio y al final para indicar claramente los límites de cada secuencia. Estas marcas son cruciales para que el modelo reconozca cuándo debe empezar y terminar de generar texto, y contribuyen significativamente al correcto aprendizaje durante el entrenamiento.

Texto original	→	**Texto procesado**
Additive models are flexible.	→	<soseq>Additive models are flexible. <eoseq>

Cuadro 3.4: Marcado especial en los resúmenes

El conjunto de datos de entrenamiento se dividió en dos subconjuntos: uno para el entrenamiento propiamente dicho y otro más pequeño para la validación. Esta separación permite evaluar el rendimiento del modelo de manera iterativa sin comprometer los datos de prueba, reservados exclusivamente para la evaluación final.

En la etapa de tokenización, las palabras de los textos y resúmenes se convirtieron en índices numéricos, representando cada término mediante un identificador único. Este proceso incluyó la selección de palabras más comunes para formar un vocabulario que descartara términos raros o poco frecuentes. Las secuencias generadas fueron normalizadas mediante un relleno con ceros para asegurar que todas tuvieran la misma longitud, independientemente de su tamaño original. También se descartaron aquellas muestras que contenían únicamente las marcas especiales, evitando que estas afectaran negativamente el modelo.

Texto original	→	**Texto procesado**
Additive models are flexible.	→	[12, 45, 78, 34, 0, 0, 0]

Cuadro 3.5: Tokenización y normalización de longitud

Finalmente, se estructuraron las secuencias procesadas para que fueran compatibles con el modelo. Los resúmenes se dividieron en dos partes: una que sirviera como entrada para el decodificador y otra que representara las salidas esperadas. Esta configuración asegura que el modelo reciba información precisa y pueda aprender a generar resúmenes coherentes basados en los textos proporcionados.

Este pipeline de preprocesamiento garantiza que los datos sean consistentes, relevantes y de alta calidad. Gracias a estas etapas, los textos y resúmenes procesados están preparados para entrenar

y evaluar el modelo.

3.3.2. Preprocesamiento para Luhn y K-Means

Se inició con la carga de un conjunto de datos. Se comenzó identificando y eliminando secciones del texto que no aportan valor al resumen, tales como agradecimientos, referencias bibliográficas y conclusiones generales. Estas secciones, aunque forman parte del documento original, no suelen contener información crítica para comprender el contenido principal. Tras este proceso, las secciones restantes fueron unificadas en un único bloque de texto.

Texto original	→	**Texto sin secciones irrelevantes**
Acknowledgements: We thank our colleagues for their support. Introduction: The study explores ...	→	*The study explores ...*

Cuadro 3.6: Eliminación de secciones irrelevantes

El texto consolidado fue dividido en oraciones individuales, lo que permite un análisis más granular. Estas oraciones fueron sometidas a una serie de transformaciones para eliminar elementos que pudieran generar ruido en el análisis. Entre estos elementos se incluyen:

- **Etiquetas especiales:** Se eliminaron etiquetas específicas utilizadas en el formato original del texto.
- **Referencias y fórmulas:** Se suprimieron menciones a ecuaciones y citas numéricas comunes en artículos académicos.
- **URLs y caracteres especiales:** Se eliminó cualquier enlace web, así como símbolos no alfabéticos, garantizando que solo se conservaran palabras relevantes.
- **Espacios redundantes:** Se unificaron los espacios en blanco para asegurar una estructura uniforme en las oraciones.

Esto garantizaría que no se encuentre con elementos que no aporten nada al resumen.

Texto original	→	**Texto procesado**
The result is defined as follows: xmath1, where ...	→	*The result is defined as follows where ...*
Visit http://example.com for details.	→	*Visit for details.*

Cuadro 3.7: Eliminación de ruido en las oraciones

Adicionalmente, se filtraron palabras de baja relevancia, como aquellas de longitud menor o igual a dos caracteres, a menos que se tratara de excepciones específicas, como pronombres o preposiciones comunes. Este filtrado ayudó a reducir el impacto de términos irrelevantes en los cálculos posteriores.

Texto procesado	→	**Texto sin palabras irrelevantes**
A key result of the study is shown in eq.	→	*key result study shown eq.*

Cuadro 3.8: Filtrado de palabras de baja relevancia

Tras la limpieza inicial, se eliminaron las palabras vacías (stopwords), que son términos de alta frecuencia pero baja relevancia semántica, como "and", "the" o "in". Este paso permite enfocar el análisis en las palabras más significativas del texto. A cada oración limpia se le generó una versión sin palabras vacías, lo que resultó en una variante adicional del texto preparada para el análisis.

Como resultado de las etapas anteriores, se generaron tres variantes de las oraciones:

- **Texto limpio:** Oraciones depuradas de caracteres especiales y referencias, pero manteniendo la estructura y palabras originales.
- **Texto sin palabras vacías:** Oraciones depuradas de palabras vacías, conservando únicamente los términos significativos.
- **Texto original sin alteraciones:** Una versión de referencia de las oraciones tal como aparecían antes de la limpieza.

Estas variantes garantizan la flexibilidad en el análisis posterior, permitiendo leer el resumen con mejor entendimiento.

En el caso de Luhn, las oraciones restantes fueron procesadas mediante tokenización, dividiéndolas en palabras individuales. Cada palabra fue lematizada para reducirla a su forma base o raíz, lo que ayuda a unificar términos derivados y reducir redundancias.

Texto original	→	**Texto procesado**
"run", "running", "ran"	→	"run"

Cuadro 3.9: Lematización

Texto original	→	**Texto procesado**
key result study shown.	→	[12, 45, 78, 34, 0, 0, 0]

Cuadro 3.10: Tokenización

Para K-Means, el último paso antes del análisis implicó la vectorización del texto, transformando las oraciones en vectores numéricos. La vectorización convierte las palabras en representaciones que preservan sus relaciones semánticas.

Texto original	→	**Texto procesado**
This paper explores machine learning methods.	→	[0.12, 0.45, 0.23, 0.78, 0.01, ...]

Cuadro 3.11: Vectorización

Capítulo 4

Desarrollo e Implementación

Esta sección destaca los elementos clave en la implementación de todos los modelos explorados como Luhn, K-Means y Seq2Seq. La fundamentación teórica sobre estos modelos se puede encontrar en la sección 2.2. Adicionalmente, en esta sección se evita lo relacionado al preprocesamiento, ya que este proceso está descrito en la sección 3.3.

4.1. Modelo de Luhn

Es importante mencionar que este modelo no requiere de entrenamiento, ya que se apoya sobre unas reglas heurísticas para evaluar la importancia de las palabras. Adicionalmente, este modelo es una construcción propia de los autores

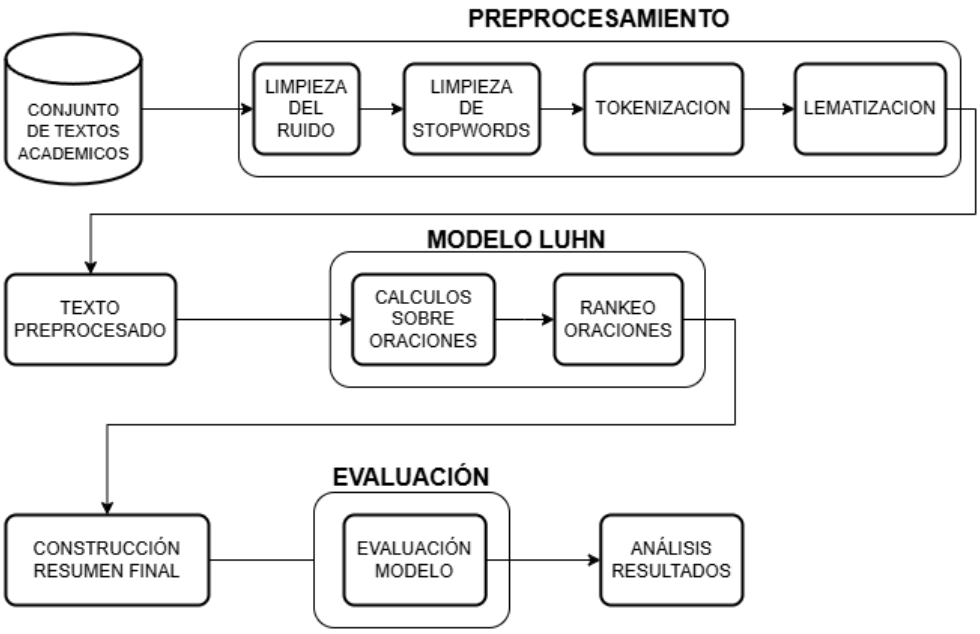


Figura 4.1: Arquitectura Modelo Luhn
Gráfica propia de los autores

La implementación del modelo de Luhn para la generación de resúmenes es una versión propia de

los autores con apoyo de librerías especializadas. Este modelo está construido bajo la arquitectura planteada en la figura 4.1. A continuación, se describen las etapas clave del desarrollo:

4.1.1. Hiperparámetros y Variables Configurables:

Aunque el modelo no utiliza hiperparámetros en el sentido estricto de los algoritmos de aprendizaje automático, se definieron variables que pueden modificar su comportamiento y resultados:

- ***percentil:*** Define qué tan selectivo es el modelo al identificar palabras clave. Un percentil alto (por ejemplo, 0.75 que es el predefinido) asegura que solo las palabras más relevantes, aquellas con puntajes TF-IDF significativamente altos, sean consideradas. Esto reduce el ruido en la selección de palabras clave, enfocándose en términos que representan conceptos clave del documento.
- ***num_oraciones_resumen:*** Especifica la cantidad de oraciones que se incluirán en el resumen generado. Este parámetro puede ajustarse según las necesidades o las características del texto original.

4.1.2. Cálculo de TF-IDF:

El modelo utiliza una función llamada *calcular_tfidf_por_palabras* para calcular los puntajes TF-IDF de cada palabra en el texto completo. Para esto:

- Se emplea *TfidfVectorizer* que es una clase en la biblioteca scikit-learn de Python utilizada para transformar un conjunto de documentos de texto en una matriz de características (matriz dispersa) basada en el modelo TF-IDF (Term Frequency - Inverse Document Frequency). Estos componentes de TF-IDF estan definidos de la siguiente manera:

  1. TF (Term Frequency): Mide la frecuencia con la que una palabra aparece en un documento. Es una medida local de la importancia de la palabra dentro de un documento específico:

  $$TF(d,t) = \frac{\text{Número de veces que aparece el termino } t \text{ en el documento } d}{\text{Número total de términos en el documento } d} \tag{4.1}$$

  2. IDF (Inverse Document Frequency): Mide la importancia global de la palabra en todo el corpus (conjunto de documentos). Cuanto más rara es una palabra en el corpus, mayor será su valor de IDF.

  $$IDF(t,D) = \log(\frac{N}{\text{Número de documentos que contienen el termino } t}) \tag{4.2}$$

  Donde N es el número de documentos. Sin embargo, hay que tener en cuenta que en esta implementación, se trabaja con un único documento a la vez.

  3. TF-IDF: Es el producto de estas dos métricas y representa la importancia de una palabra en un documento dentro del contexto de un corpus.

  $$IDF(t,d,D) = TF(d,t) \times IDF(t,D) \tag{4.3}$$

4.1.3. Selección de Palabras Clave:

Basándose en el percentil especificado, la función llamada *seleccionar_palabras_clave* identifica las palabras más relevantes. Estas palabras clave son esenciales para determinar la importancia de las oraciones en etapas posteriores.

4.1.4. Aplicación del Modelo de Luhn:

Se define una función *aplicar_metodo_luhn* para evaluar las oraciones en función de la presencia y proximidad de las palabras clave de la siguiente manera:

Cada oración se evalúa inicialmente por la cantidad de palabras clave que contiene. Cuantas más palabras clave se encuentren en una oración, mayor será su puntuación. Las palabras clave se han identificado previamente utilizando el puntaje TF-IDF, lo que asegura que solo los términos más significativos sean considerados en esta etapa.

Adicionalmente, la proximidad de las palabras clave también puede ser un factor importante. Si las palabras clave están cerca unas de otras dentro de una oración, esto indica que la oración puede concentrar un tema específico y tiene una mayor capacidad informativa.

La proximidad se evalúa considerando la distancia entre las palabras clave dentro de la oración. Cuanto más cercanas estén entre sí, mayor será el puntaje asignado a esa oración.

Por último, se juntan las oraciones más relevantes después de aplicar el modelo, y ese es el resumen final.

4.2. Modelo K-Means

Es importante mencionar que este modelo requiere de un procedimiento de agrupamiento, en lugar de un entrenamiento tradicional.

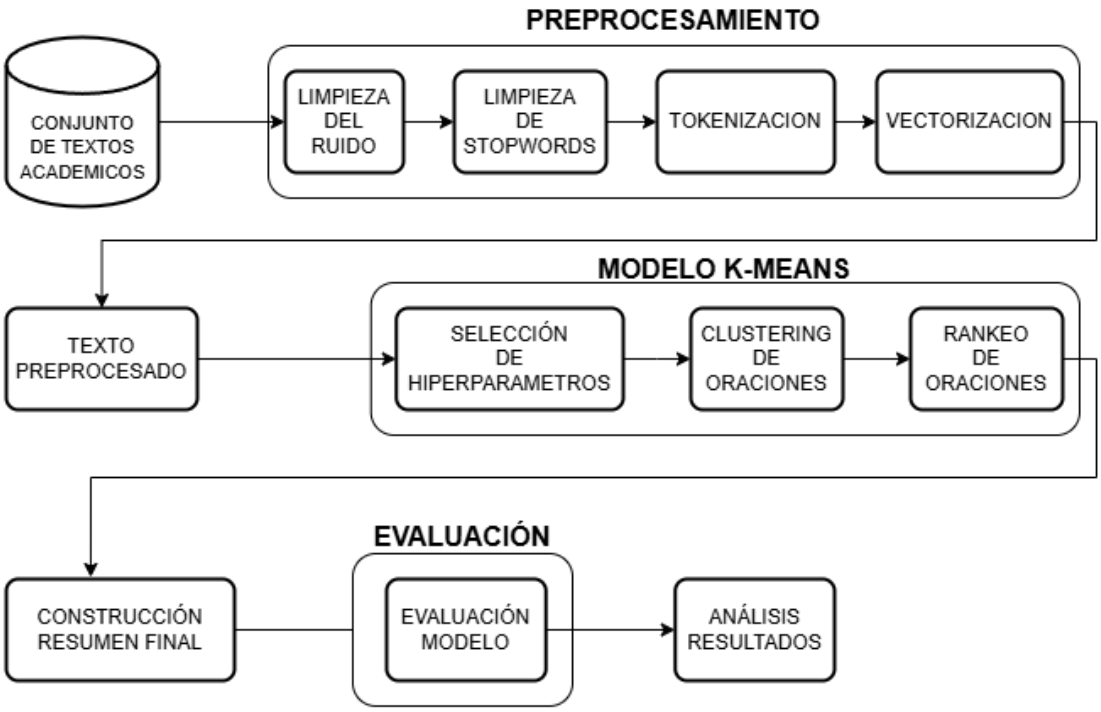


Figura 4.2: Arquitectura Modelo K-Means
Gráfica propia de los autores

La implementación del modelo de K-Means para la generación de resúmenes es una versión propia de los autores con apoyo de librerías especializadas. Este modelo está construido bajo la arquitectura planteada en la figura 4.2. A continuación, se describen las etapas clave del desarrollo:

4.2.1. Hiperparámetros y Variables Configurables:

- **Número de clusters:** Este parámetro define cuántos grupos se formarán al aplicar el modelo k-means. Cada clúster representa un tema principal identificado dentro del texto, y se selecciona una oración por cada clúster para construir el resumen. Un número bajo de clústeres genera un resumen más conciso, priorizando los temas principales, mientras que un número alto incorpora más detalles, pero puede diluir la capacidad de síntesis.
- **Inicialización de Centroides:** Determina cómo se seleccionan los centroides iniciales al comenzar el algoritmo. Por defecto se utiliza la estrategia `"k-means++"`, que distribuye los centroides de manera inteligente para reducir la probabilidad de converger a soluciones subóptimas.
- **Semilla Aleatoria:** Fija una semilla para la generación de números aleatorios utilizados en la inicialización de los centroides y otros pasos del modelo. Si se fija un valor para la semilla, el modelo producirá los mismos resultados en cada ejecución, siempre que los datos de entrada

no cambien. Si no se fija una semilla (o se usa un valor diferente), los resultados pueden variar ligeramente entre ejecuciones debido a la naturaleza aleatoria de la inicialización.

4.2.2. Representación Vectorial de Oraciones

Cada oración se convierte en un vector numérico mediante un modelo de embeddings (Word2Vec). Para representar una oración completa, se toma el promedio de los vectores de las palabras que la componen:

$$\mathbf{v}_s = \frac{1}{n} \sum_{i=1}^{n} \mathbf{v}_{w_i}$$

Donde: - v_s: El vector de la oración. - v_{w_i}: El vector de la palabra w_i en la oración, donde w_i es la palabra en la posición i de la oración. - n: El número de palabras en la oración.

4.2.3. Algoritmo de Agrupamiento K-means

Una vez obtenidos los vectores de las oraciones, se aplica el algoritmo de `k-means` para agruparlas en k clústeres. El número de clústeres, k, corresponde al número deseado de oraciones en el resumen. El algoritmo busca minimizar la distancia intra-clúster, definida como:

$$\text{mín} \sum_{j=1}^{k} \sum_{\mathbf{x} \in C_j} ||\mathbf{x} - \boldsymbol{\mu}_j||^2$$

Donde: - $\mathbf{x}$: vector de una oración. - C_j: conjunto de vectores asignados al clúster j. - $\boldsymbol{\mu}_j$: centroide del clúster j. - k: número de clústeres.

4.2.4. Selección de Oraciones Representativas

Para cada clúster, se selecciona la oración más representativa calculando la distancia euclidiana entre cada vector de oración $\mathbf{x}$ y el centroide del clúster $\boldsymbol{\mu}$:

$$d(\mathbf{x}, \boldsymbol{\mu}) = ||\mathbf{x} - \boldsymbol{\mu}||$$

La oración correspondiente al vector con la menor distancia es considerada la más representativa del clúster.

4.2.5. Construcción del resumen

Las oraciones seleccionadas de los k clústeres se ordenan según su posición original en el texto para preservar la coherencia.

4.3. Modelo Seq2Seq

A continuación, se describe la arquitectura diseñada para un modelo Seq2Seq orientado a la generación de resúmenes. En esta sección se presentan de manera general los componentes que integran el modelo final, destacando con mayor detalle el comportamiento de cada uno de ellos. Además, se presentan los diferentes experimentos realizados con diversos ajustes.

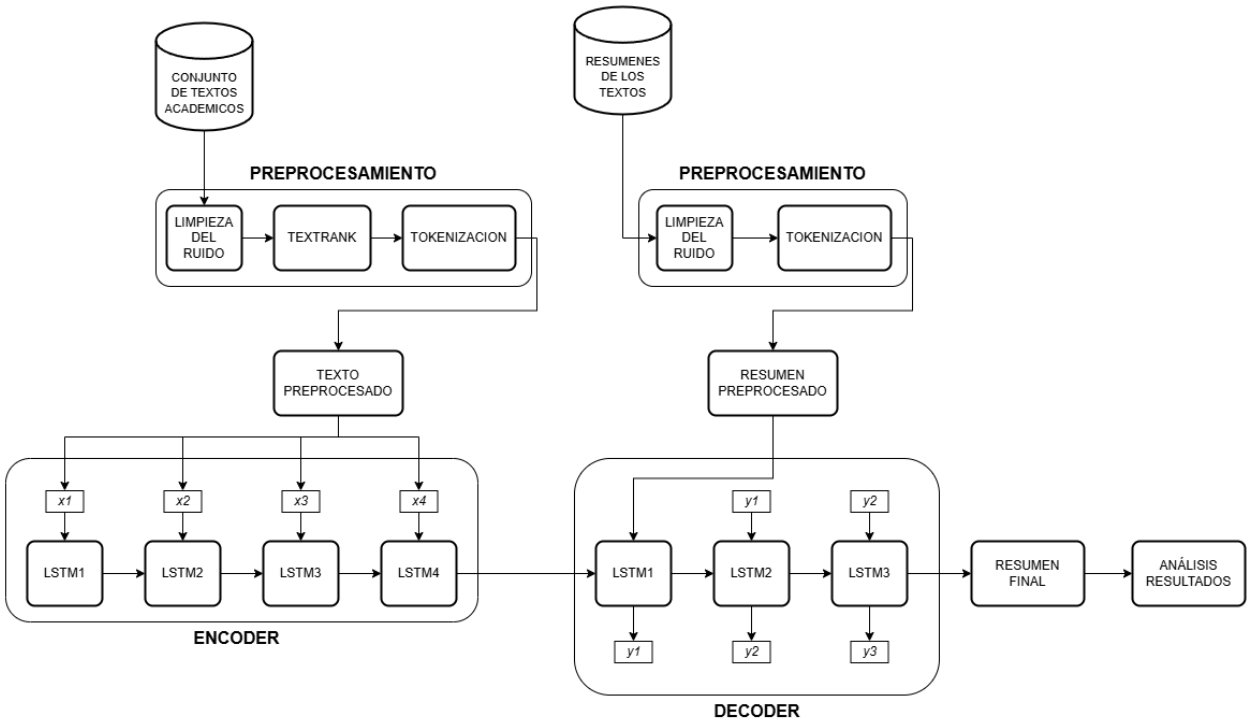


Figura 4.3: Arquitectura Modelo Seq2Seq
Gráfica propia de los autores

La implementación de este modelo Seq2Seq para la generación de resúmenes es una versión propia de los autores con apoyo de librerías especializadas. Este modelo está construido bajo la arquitectura planteada en la figura 4.3. A continuación, se describen las etapas clave del desarrollo:

4.4. Creación del modelo Seq2Seq

Como se observa en la figura 4.3, los componentes más importantes que conforman este modelo son el encoder y el decoder, que, como se mencionó en la sección 2.2.9, el encoder será el encargado de procesar la secuencia de entrada y la transforma en una representación intermedia de dimensión fija, conocida como vector de contexto, y el decoder es el encargado de recibir el vector de contexto generado por el encoder para así producir la secuencia de salida de manera iterativa.

4.4.1. Arquitectura encoder-decoder

El constructor recibe la arquitectura de la red neuronal, la cual establece el número de capas que va a tener, como también el número de nodos asignados a cada capa. La dimensión de entrada de la primera capa corresponderá a la dimensión máxima permitida para la longitud de los textos.

4.4.2. Parámetros de entrenamiento

- **Tamaño del Vocabulario en textos (*x_vocab*):** Representa el número total de palabras únicas (tokens) en el conjunto de datos (solo textos) después del preprocesamiento. Se utiliza para definir la dimensión de la capa de embedding en el codificador.
- **Tamaño del Vocabulario en resúmenes (*y_vocab*):** Al igual que como en el codificador, este vocabulario representa el número total de palabras únicas (tokens) en el conjunto de datos (solo resumenes) después del preprocesamiento. Se utiliza para definir la dimensión de la capa de embedding en el decodificador.
- **Dimensión de los Embeddings (*EMBED_DIM*):** Es la cantidad de dimensiones que se utilizarán para representar cada palabra o token en el espacio vectorial.
- **Tamaño de las Capas LSTM (*LATENT_DIM*):** Define el número de unidades ocultas en cada capa LSTM.
- **Número de Capas en el Encoder y Decoder:** Define cuantas capas LSTM se implementan en los componentes encoder y decoder. Esta cantidad es variable y dependerá de las necesidades, teniendo en cuenta que un numero grande de capas requiere una mayor capacidad de procesamiento.
- **Longitud Máxima de los textos (*MAX_TEXT_LEN*):** Representa la cantidad máxima permitida de palabras presentes en un texto, esto es particularmente útil para hacer filtros sobre textos que superen la cantidad de tokens permitidos, además de ser crucial en el calculo del vocabulario.
- **Longitud Máxima de los resumenes (*MAX_SUMMARY_LEN*):** Representa la cantidad máxima permitida de palabras a predecir por el modelo.
- **Dropout (*dropout*):** Una tasa de dropout para regularizar el modelo y prevenir el sobreajuste.
- **Función de Activación:** La función de activación introduce no linealidad en la red neuronal, permitiendo que la red aprenda relaciones complejas y no lineales en los datos. Sin funciones de activación, las redes neuronales serían equivalentes a un modelo lineal, sin importar su profundidad.

- **Criterio de Pérdida (Loss Function):** La función de pérdida mide la discrepancia entre las predicciones del modelo y las etiquetas reales. Es la métrica que el modelo intenta minimizar durante el entrenamiento.

- **Optimizador:** El optimizador ajusta los pesos de la red neuronal para minimizar la función de pérdida utilizando los gradientes calculados durante la retropropagación.

- **Tasa de Aprendizaje (Learning Rate):** Define la velocidad a la que el modelo ajusta sus pesos durante el entrenamiento.

4.5. Construcción del modelo

En este modelo, el codificador (encoder) y el decodificador (decoder), estan implementados con capas LSTM. Se definen varios hiperparámetros: *EPOCHS*, *BATCH_SIZE*, *LATENT_DIM* y *EMBED_DIM*. El encoder comienza con una capa de entrada que recibe secuencias de longitud máxima *MAX_TEXT_LEN*, seguida de una capa de embedding entrenable que convierte las palabras en vectores de dimensión *EMBED_DIM*. A continuación, se utilizan tres capas LSTM secuenciales, cada una con un tamaño de estado oculto de *LATENT_DIM* y configuradas con un 40 % de dropout y recurrent dropout para evitar el sobreajuste. Las salidas de estas capas LSTM proporcionan los estados finales que se utilizarán en el decoder. El decoder, por su parte, también comienza con una capa de embedding entrenable y luego pasa por una capa LSTM que toma como entrada los estados finales del encoder como estado inicial. Finalmente, se agrega una capa densa TimeDistributed con una función de activación softmax para predecir las probabilidades de las palabras en cada paso de la secuencia. El modelo se compila utilizando el optimizador RMSprop y la función de pérdida sparse categorical crossentropy, con la métrica de precisión. Además, se definen dos callbacks: *early_stopping*, que detiene el entrenamiento si la pérdida de validación no mejora después de dos épocas, y *reduce_lr*, que reduce la tasa de aprendizaje si la pérdida de validación no mejora.

4.5.1. Manejo de los pesos del Modelo

Se realiza un proceso de guardado *model.save_weights* y de carga de los pesos *model.load_weights*, dado que almacenar los pesos de un modelo Seq2Seq puede ser crucial por varias razones. En primer lugar, permite la continuidad del entrenamiento, ya que al guardar los pesos, se puede reanudar el proceso de entrenamiento en una sesión posterior sin perder el progreso alcanzado, incluso si se debe detener el entrenamiento por limitaciones de tiempo o recursos. Además, almacenar los pesos asegura que no se pierda el progreso del entrenamiento, ya que durante el proceso de ajuste, el modelo modifica sus pesos para minimizar el error. Si no se guardan estos pesos, todo el progreso se perdería y sería necesario comenzar desde cero. Por último, el almacenamiento de los pesos facilita la evaluación y comparación del rendimiento del modelo en diferentes etapas del entrenamiento, permitiendo observar cómo cambia el desempeño a lo largo del tiempo.

4.5.2. Entrenamiento

El proceso de entrenamiento del modelo Seq2Seq se lleva a cabo utilizando el conjunto de datos de entrada, que consta de secuencias de texto. Durante el entrenamiento, se alimentan tanto las secuencias de entrada como las de salida al modelo, que está compuesto por un encoder y un decoder, ambos implementados con capas LSTM. El modelo se entrena durante un número definido de épocas, especificado por el hiperparámetro EPOCHS, con un tamaño de lote de BATCH_SIZE. Durante cada época, el modelo ajusta sus pesos para minimizar la función de pérdida, que se calcula utilizando la sparse categorical crossentropy. El optimizador utilizado es RMSprop, que ajusta los pesos del modelo para mejorar la precisión de las predicciones.

Adicional a lo anterior, se aplica un conjunto de técnicas para mejorar la generalización y evitar el sobreajuste. Entre ellas, se encuentra el uso de dropout en las capas LSTM, lo que ayuda a prevenir que el modelo dependa demasiado de ciertas neuronas durante el entrenamiento.

Para monitorear el rendimiento del modelo durante el entrenamiento, se utilizan dos callbacks importantes. El primero es early stopping, que detiene el entrenamiento si la pérdida de validación no mejora después de un número definido de épocas, evitando así el sobreajuste. El segundo es reduce learning rate, que reduce la tasa de aprendizaje si la pérdida de validación no mejora, permitiendo una convergencia más eficiente en etapas posteriores del entrenamiento.

Este proceso se repite durante todas las épocas definidas, con el objetivo de minimizar la función de pérdida y mejorar la precisión del modelo en las tareas de predicción de resúmenes.

4.5.3. Estrategia de entrenamiento

Durante el procesamiento de los textos completos, se presentó una dificultad significativa debido al alto consumo de memoria RAM, que oscilaba entre 350 y 400 GB al trabajar con cantidades relativamente pequeñas de datos, como 100 textos completos. Esto hizo que el proceso fuera poco eficiente y difícil de manejar. Para solucionar este problema, se decidió incorporar un paso intermedio en el flujo de trabajo, que consistía en reducir la longitud de los textos a versiones más cortas utilizando el algoritmo TextRank. Además, se realizó un recorte en el dataset, limitándolo a 10,000 textos. Tras aplicar TextRank, se implementó un filtro adicional para eliminar aquellos textos que excedieran la longitud máxima permitida. Para los textos que quedaban dentro de la longitud permitida, se aplicó un proceso de padding para rellenar los tokens faltantes, garantizando que todos los textos tuvieran una longitud uniforme y adecuada para ser procesados de manera eficiente por el modelo.

4.5.4. Exploración de hiperparámetros

En las secciones 4.4.2 y 4.5 se describió una serie de hiperparámetros que pueden ser explorados. Debido a que las ejecuciones de los entrenamientos toman mucho tiempo, se decidió explorar

solamente sobre un subconjunto de estos hiperparámetros. En particular, se decidió dejar el número de capas en el encoder y decoder con valores constantes de 4 para el encoder y 3 para el decoder, la longitud máxima de los textos con un valor de 650, la longitud máxima de los resúmenes con un valor de 150, el dropout con un valor de 0.4 y la tasa de aprendizaje con un valor predefinido de 0.001. Para los hiperparámetros restantes se ha planteado el siguiente espacio de búsqueda:

Hiperparámetro	**Espacio de búsqueda**
EPOCHS	[50, 60, 100]
BATCH_SIZE	[64, 128, 256]
LATENT_DIM	[128, 256, 512]
EMBED_DIM	[50, 100, 200]

Cuadro 4.1: Espacio de búsqueda

Ya que este espacio de búsqueda sigue siendo muy grande para explorarlo por completo, se hace una elección de hiperparámetros de forma aleatoria para los experimentos 1 y 3, mientras que el experimento 2 es una ligera variación del 1. De esta manera, se generan los siguientes experimentos:

Hiperparámetro	**Valor seleccionado**
EPOCHS	100
BATCH_SIZE	128
LATENT_DIM	256
EMBED_DIM	100

Cuadro 4.2: Experimento #1

Hiperparámetro	**Valor seleccionado**
EPOCHS	50
BATCH_SIZE	128
LATENT_DIM	256
EMBED_DIM	100

Cuadro 4.3: Experimento #2

Hiperparámetro	**Valor seleccionado**
EPOCHS	60
BATCH_SIZE	128
LATENT_DIM	512
EMBED_DIM	300

Cuadro 4.4: Experimento #3

4.5.5. Arquitecturas y gráficas experimentos

- **Arquitectura compilada Experimento 1:**

```
Layer (type)                 Output Shape              Param #   Connected to
==================================================================================================
input_4 (InputLayer)         [(None, 650)]             0         []

embedding_2 (Embedding)      (None, 650, 100)          1028300   ['input_4[0][0]']

lstm_4 (LSTM)                [(None, 650, 256),        365568    ['embedding_2[0][0]']
                              (None, 256),
                              (None, 256)]

input_5 (InputLayer)         [(None, None)]            0         []

lstm_5 (LSTM)                [(None, 650, 256),        525312    ['lstm_4[0][0]']
                              (None, 256),
                              (None, 256)]

embedding_3 (Embedding)      (None, None, 100)         381000    ['input_5[0][0]']

lstm_6 (LSTM)                [(None, 650, 256),        525312    ['lstm_5[0][0]']
                              (None, 256),
                              (None, 256)]

lstm_7 (LSTM)                [(None, None, 256),       365568    ['embedding_3[0][0]',
                              (None, 256),                       'lstm_6[0][1]',
                              (None, 256)]                       'lstm_6[0][2]']

time_distributed_1 (TimeDistri  (None, None, 3810)     979170    ['lstm_7[0][0]']
buted)

==================================================================================================
Total params: 4,170,230
Trainable params: 4,170,230
Non-trainable params: 0
```

Figura 4.4: Arquitectura Compilada Experimento 1

Gráfica de perdida Experimento 1:

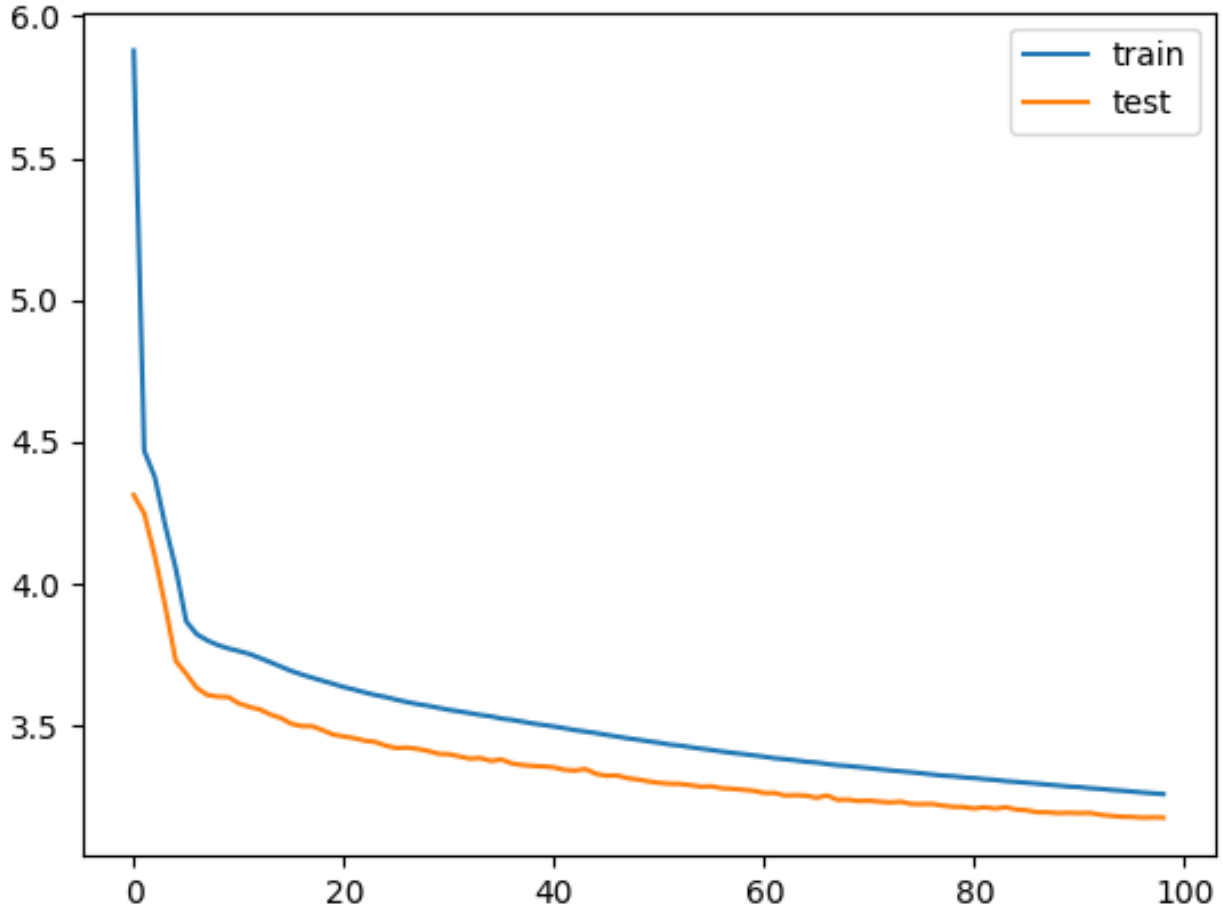


Figura 4.5: Gráfica de perdida Experimento 1

- **Arquitectura compilada Experimento 2:**

```
Layer (type)                Output Shape              Param #   Connected to
==================================================================================================
input_1 (InputLayer)        [(None, 650)]             0         []

embedding (Embedding)       (None, 650, 100)          1028300   ['input_1[0][0]']

lstm (LSTM)                 [(None, 650, 256),        365568    ['embedding[0][0]']
                             (None, 256),
                             (None, 256)]

input_2 (InputLayer)        [(None, None)]            0         []

lstm_1 (LSTM)               [(None, 650, 256),        525312    ['lstm[0][0]']
                             (None, 256),
                             (None, 256)]

embedding_1 (Embedding)     (None, None, 100)         381000    ['input_2[0][0]']

lstm_2 (LSTM)               [(None, 650, 256),        525312    ['lstm_1[0][0]']
                             (None, 256),
                             (None, 256)]

lstm_3 (LSTM)               [(None, None, 256),       365568    ['embedding_1[0][0]',
                             (None, 256),                        'lstm_2[0][1]',
                             (None, 256)]                        'lstm_2[0][2]']

time_distributed (TimeDist  (None, None, 3810)        979170    ['lstm_3[0][0]']
ributed)

==================================================================================================
Total params: 4170230 (15.91 MB)
Trainable params: 4170230 (15.91 MB)
Non-trainable params: 0 (0.00 Byte)
```

Figura 4.6: Arquitectura Compilada Experimento 2

Gráfica de perdida Experimento 2:

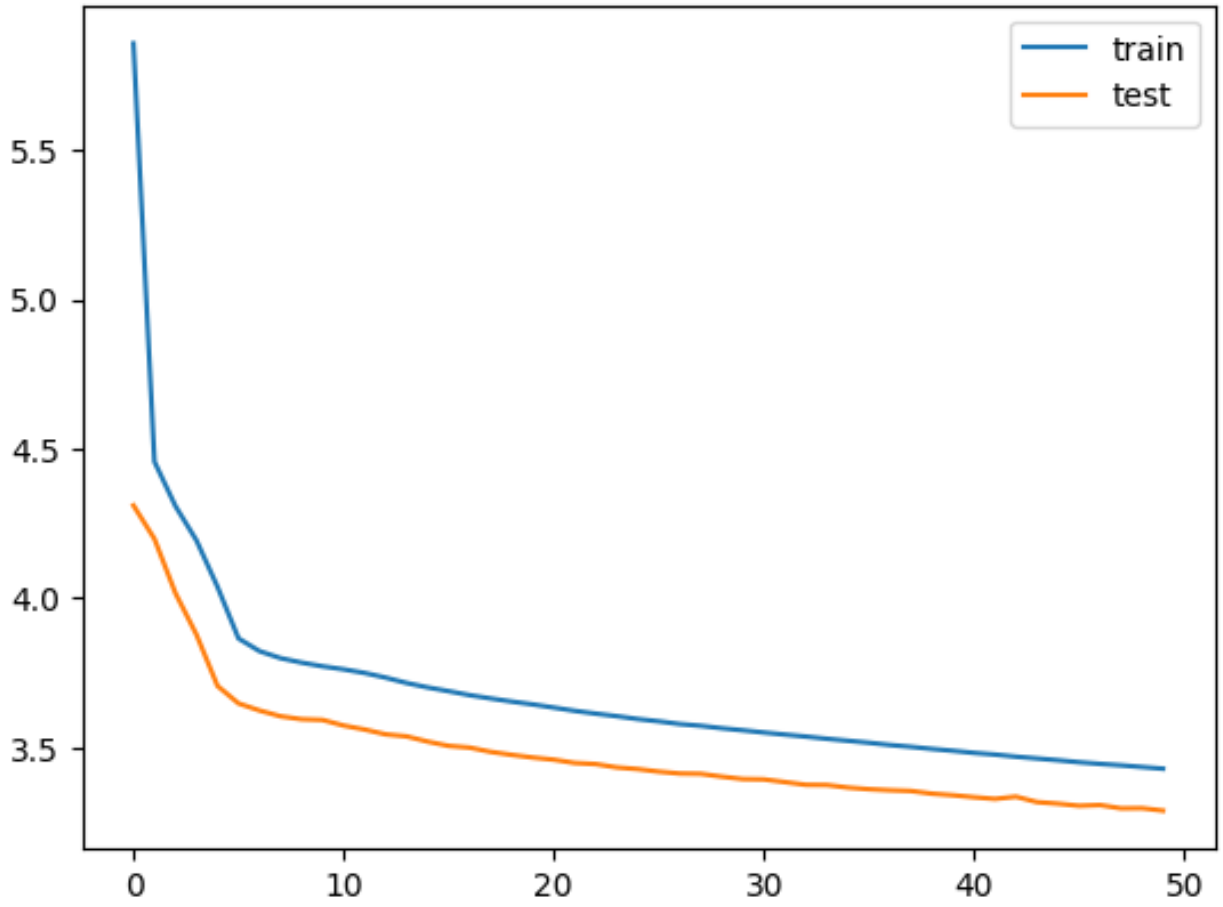


Figura 4.7: Gráfica de perdida Experimento 2

- **Arquitectura compilada Experimento 3:**

```
Layer (type)                Output Shape              Param #   Connected to
==================================================================================================
input_6 (InputLayer)        [(None, 650)]             0         []

embedding_2 (Embedding)     (None, 650, 300)          3084900   ['input_6[0][0]']

lstm_4 (LSTM)               [(None, 650, 512),        1665024   ['embedding_2[0][0]']
                             (None, 512),
                             (None, 512)]

input_7 (InputLayer)        [(None, None)]            0         []

lstm_5 (LSTM)               [(None, 650, 512),        2099200   ['lstm_4[0][0]']
                             (None, 512),
                             (None, 512)]

embedding_3 (Embedding)     (None, None, 300)         1143000   ['input_7[0][0]']

lstm_6 (LSTM)               [(None, 650, 512),        2099200   ['lstm_5[0][0]']
                             (None, 512),
                             (None, 512)]

lstm_7 (LSTM)               [(None, None, 512),       1665024   ['embedding_3[0][0]',
                             (None, 512),                        'lstm_6[0][1]',
                             (None, 512)]                        'lstm_6[0][2]']

time_distributed_1 (TimeDi  (None, None, 3810)        1954530   ['lstm_7[0][0]']
stributed)

==================================================================================================
Total params: 13710878 (52.30 MB)
Trainable params: 13710878 (52.30 MB)
Non-trainable params: 0 (0.00 Byte)
```

Figura 4.8: Arquitectura Compilada Experimento 3

Gráfica de perdida Experimento 3:

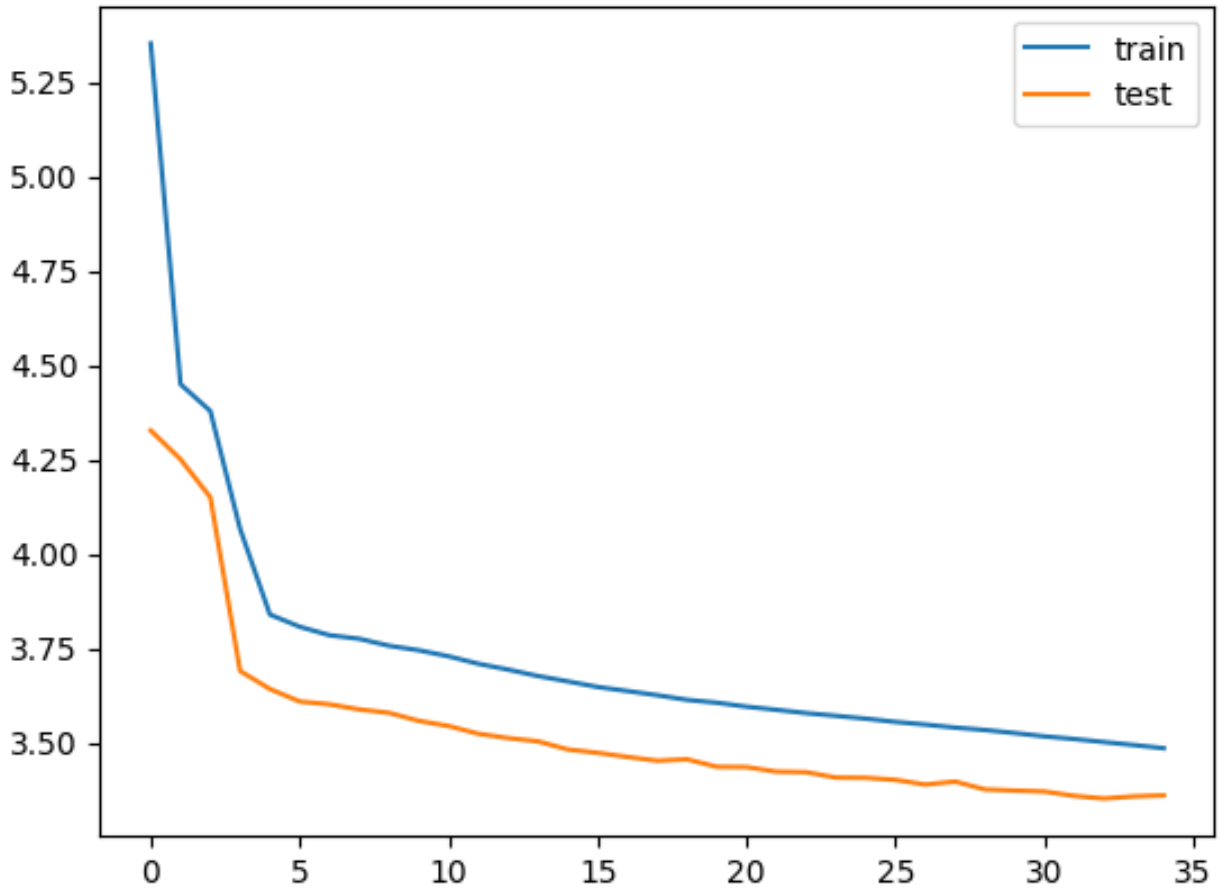


Figura 4.9: Gráfica de perdida Experimento 3

4.5.6. Etapa de decodificación

El proceso de decodificación en el modelo Seq2Seq comienza con la codificación de la secuencia de entrada, donde el modelo encoder transforma el texto en un vector de características representativo, incluyendo los estados ocultos y de celda del último paso del LSTM. Posteriormente, el modelo decoder utiliza estos estados como entrada inicial para generar el resumen palabra por palabra. Durante este proceso, se emplea un token especial de inicio de secuencia ("soseq") como punto de partida para la predicción. Las palabras subsecuentes se generan iterativamente, utilizando la salida previa como entrada en cada paso, hasta alcanzar un token especial de fin de secuencia ("eoseq") o el límite máximo de palabras permitido.

Un aspecto crucial en la decodificación es el manejo de la temperatura, un hiperparámetro que

regula la aleatoriedad en la selección de palabras. La temperatura ajusta la distribución de probabilidad generada por la capa softmax del decoder, permitiendo controlar el balance entre exploración y explotación. Valores bajos de temperatura tienden a favorecer las palabras con mayor probabilidad, generando resúmenes con menor variedad. Por otro lado, valores más altos incrementan la diversidad al amplificar las probabilidades de palabras menos frecuentes, lo que puede ser útil para evitar repeticiones o enriquecer el vocabulario del resumen.

En este modelo, la variable de temperatura se mantiene constante para todos los experimentos y tiene un valor de 1.00 que es equivalente a una temperatura moderada, además de un valor $k{=}5$ constante que representa la cantidad de palabras representativas de las cuales se puede hacer una elección de forma aleatoria.

Capítulo 5

Análisis de Resultados

5.1. Resultados

En esta sección, los resultados de las métricas, tanto de BERTScore como de ROUGE-1, ROUGE-2 y ROUGE-L, se presentan utilizando el mismo conjunto de datos para asegurar la consistencia en la comparación entre los modelos. Asimismo, se empleará un ejemplo uniforme para ilustrar el comportamiento de cada modelo.

5.1.1. Modelo Seq2Seq Experimento 1

Original summary: we study the detectability of circular polarization in a stochastic gravitational wave background from various sources such supermassive black hole binaries cosmic strings and inflation in the early universe with pulsar timing arrays we calculate generalized overlap reduction functions for the circularly polarized stochastic gravitational wave background we find that the circular polarization can not detected for an isotropic background however there is a chance to observe the circular polarization for an anisotropic gravitational wave background we also show how to separate polarized gravitational waves from unpolarized gravitational waves

Predicted summary: we consider a study and the of a model in the system in the quantum state in the of the of this and we study on the of and we find a simple results of the two and the of the system in a model in the and the system of the system the of the field and the same the of the field and and the of a energy of the energy in the field and we find the effect with the spin energy and the of the of the and the field and we also find that the of and the same the system of the two state and the of this system we find that the of and the same the system

La tabla presenta los resultados de las métricas obtenidas del modelo Seq2Seq en el Experimento 1, los cuales fueron obtenidos utilizando el mismo conjunto de datos para garantizar la consistencia en la comparación.

Métrica	Tipo	Media	Desviación Estándar
ROUGE-1	r	0.1733	0.0438
	p	0.3232	0.0726
	f	0.2213	0.0422
ROUGE-2	r	0.0268	0.0165
	p	0.0304	0.0169
	f	0.0277	0.0151
ROUGE-L	r	0.1318	0.0389
	p	0.2419	0.0528
	f	0.1673	0.0371
BERTScore	r	0.7969	0.0095
	p	0.8190	0.0089
	f	0.8077	0.0077

Cuadro 5.1: Resultados de las métricas de evaluación del Experimento 1

5.1.2. Modelo Seq2Seq Experimento 2

Original summary: we study the detectability of circular polarization in a stochastic gravitational wave background from various sources such supermassive black hole binaries cosmic strings and inflation in the early universe with pulsar timing arrays we calculate generalized overlap reduction functions for the circularly polarized stochastic gravitational wave background we find that the circular polarization can not detected for an isotropic background however there is a chance to observe the circular polarization for an anisotropic gravitational wave background we also show how to separate polarized gravitational waves from unpolarized gravitational waves

Predicted summary: the new of the system of the two field of a system is been a quantum field of a system we show the model of the model of the model we find a the number of the energy and the system of the system and the model in the model we present the new energy of the energy of the case of the model in this of the field of an energy of an field and a results we that that the results to be to the number of the model is an the model is an of the and a system of the field is a model of a two energy and the model of the model of the model we also also that the two field of the number of a two and in the energy

La tabla presenta los resultados de las métricas obtenidas del modelo Seq2Seq Experimento 2, los cuales fueron obtenidos utilizando el mismo conjunto de datos para garantizar la consistencia en la comparación.

Métrica	Tipo	Media	Desviación Estándar
ROUGE-1	r	0.1819	0.0463
	p	0.3743	0.0795
	f	0.2410	0.0488
ROUGE-2	r	0.0311	0.0197
	p	0.0360	0.0251
	f	0.0326	0.0208
ROUGE-L	r	0.1366	0.0347
	p	0.2810	0.0575
	f	0.1810	0.0359
BERTScore	r	0.7922	0.0092
	p	0.8099	0.0117
	f	0.8009	0.0089

Cuadro 5.2: Resultados de las métricas de evaluación del Experimento 2

5.1.3. Modelo Seq2Seq Experimento 3

Original summary: we study the detectability of circular polarization in a stochastic gravitational wave background from various sources such supermassive black hole binaries cosmic strings and inflation in the early universe with pulsar timing arrays we calculate generalized overlap reduction functions for the circularly polarized stochastic gravitational wave background we find that the circular polarization can not detected for an isotropic background however there is a chance to observe the circular polarization for an anisotropic gravitational wave background we also show how to separate polarized gravitational waves from unpolarized gravitational waves

Predicted summary: we study that the results to be the model of the quantum of the system of a system of the model of a two a system is be that a the number is the new the model of the system and the two a model of the model and the new a the case to the the system of the same is the the case of the same the same the model of the model of the model of the two the model is a results of the two the system of the the case to a new of the system of the case to the model to a the time of the same the the model of the system of the case to a model of the model of a model of the model to the the the model of the system is the model of the model

La tabla presenta los resultados de las métricas obtenidas del modelo Seq2Seq Experimento 3, los cuales fueron obtenidos utilizando el mismo conjunto de datos para garantizar la consistencia en la comparación.

Métrica	Tipo	Media	Desviación Estándar
ROUGE-1	r	0.1577	0.0462
	p	0.4168	0.1006
	f	0.2251	0.0571
ROUGE-2	r	0.0233	0.0146
	p	0.0377	0.0250
	f	0.0281	0.0172
ROUGE-L	r	0.1192	0.0347
	p	0.3135	0.0771
	f	0.1699	0.0435
BERTScore	r	0.7908	0.0104
	p	0.8104	0.0162
	f	0.8004	0.0114

Cuadro 5.3: Resultados de las métricas de evaluación del Experimento 3

5.1.4. Luhn

Original summary: we study the detectability of circular polarization in a stochastic gravitational wave background from various sources such supermassive black hole binaries cosmic strings and inflation in the early universe with pulsar timing arrays we calculate generalized overlap reduction functions for the circularly polarized stochastic gravitational wave background we find that the circular polarization can not detected for an isotropic background however there is a chance to observe the circular polarization for an anisotropic gravitational wave background we also show how to separate polarized gravitational waves from unpolarized gravitational waves

Predicted summary: without loss of generality we can assume then we obtain corresponding noises labelnpendaligned where labelsn quad snf equiv labelsn endaligned using the inner product endaligned we can rewrite therefore the optimal filter function can chosen using we get optimal signal to noise ratio labelsnrendaligned plugging and into we obtain snrvleft endaligned we assume all of the noise power spectral densities are the same becomes thus the compiled orfs can defined gammavequivfracgammalmvgammailmgammalmvgammailmleft endaligned this compiled orfs and describe the angular sensitivity of the four pulsars for the pure and mode of the sgwb respectively. the black solid curve the blue dashed curve the red dotted curve the dark red space dotted curve and the green long dashed curve represent respectivelytitlefigwidth using the same procedure described in the above to derive the generalized orfs for circular polarizations we can also derive the generalized orfs for the intensity where the angular integral in this case was performed in. to calculate in the complex plane we again substitute into and obtain we use the residue theorem where the residues

inside the unit circle can calculated fracizzsqrtxsinxi endaligned therefore becomes substituting to we can calculate finally substituting and into we get the generalized orf for endaligned a function of the angular separation between the two pulsars(...).

La tabla presenta los resultados de las métricas obtenidas del modelo Luhn, los cuales fueron obtenidos utilizando el mismo conjunto de datos para garantizar la consistencia en la comparación.

Métrica	Tipo	Media	Desviación Estándar
ROUGE-1	r	0.4472	0.1068
	p	0.2844	0.0987
	f	0.3371	0.0833
ROUGE-2	r	0.1535	0.0838
	p	0.0920	0.0760
	f	0.1097	0.0692
ROUGE-L	r	0.3150	0.0957
	p	0.2002	0.0833
	f	0.2372	0.0746
BERTScore	r	0.8417	0.0219
	p	0.8156	0.0276
	f	0.8283	0.0222

Cuadro 5.4: Resultados de las métricas de evaluación de Luhn

5.1.5. K-Means

Original summary: we study the detectability of circular polarization in a stochastic gravitational wave background from various sources such supermassive black hole binaries cosmic strings and inflation in the early universe with pulsar timing arrays we calculate generalized overlap reduction functions for the circularly polarized stochastic gravitational wave background we find that the circular polarization can not detected for an isotropic background however there is a chance to observe the circular polarization for an anisotropic gravitational wave background we also show how to separate polarized gravitational waves from unpolarized gravitational waves

Predicted summary: without loss of generality we can assume then we obtain corresponding noises labelnpendaligned where labelsn quad snf equiv labelsn endaligned using the inner product endaligned we can rewrite therefore the optimal filter function can chosen using we get optimal signal to noise ratio labelsnrendaligned plugging and into we obtain snrvleft endaligned we assume all of the noise power spectral densities are the same becomes thus the compiled orfs can defined gammavequivfracgammalmvgammailmgammalmvgammailmleft endaligned this compiled orfs and

describe the angular sensitivity of the four pulsars for the pure and mode of the sgwb respectively. the black solid curve the blue dashed curve the red dotted curve the dark red space dotted curve and the green long dashed curve represent respectivelytitlefigwidth a a function of the angular separation between the two pulsars. we extend the generalized orfs to cases with circular polarizations characterized by the parameter. the orfs for the quadrupole are depicted. using well and we get then we define hence the dimensionless quantity in is given by where the spherical harmonics are orthogonal and normalized using we obtain without loss of generality we normalize the monopole moment so becomes the time of arrival of radio pulses from the pulsar is affected by gws

La tabla muestra los resultados de las métricas obtenidas del modelo K-means, los cuales fueron obtenidos utilizando el mismo conjunto de datos para garantizar la consistencia en la comparación.

Métrica	Tipo	Media	Desviación Estándar
ROUGE-1	r	0.3653	0.0900
	p	0.2829	0.0861
	f	0.3098	0.0713
ROUGE-2	r	0.1075	0.0613
	p	0.0847	0.0577
	f	0.0913	0.0543
ROUGE-L	r	0.2561	0.0755
	p	0.1980	0.0670
	f	0.2171	0.0599
BERTScore	r	0.8343	0.0192
	p	0.8208	0.0249
	f	0.8273	0.0195

Cuadro 5.5: Resultados de las métricas de evaluación de K-Means

5.2. Evaluación y análisis

5.2.1. ROUGE-1

Modelo	Tipo	Media	Desviación Estándar
Experimento 1	r	0.1733	0.0438
	p	0.3232	0.0726
	f	0.2213	0.0422
Experimento 2	r	0.1819	0.0463
	p	0.3743	0.0795
	f	0.2410	0.0488
Experimento 3	r	0.1577	0.0462
	p	0.4168	0.1006
	f	0.2251	0.0571
Luhn	r	0.4472	0.1068
	p	0.2844	0.0987
	f	0.3371	0.0833
K-Means	r	0.3653	0.0900
	p	0.2829	0.0861
	f	0.3098	0.0713

Cuadro 5.6: Resultados de la métrica ROUGE-1 por modelo

Del análisis de los resultados en la Tabla 5.6, se destacan los modelos con el mayor y menor desempeño en cada métrica:

1. **Recuperación (r):**
   - **Mayor recuperación:** Luhn (r=0.4472) muestra la mejor capacidad para identificar información relevante, destacándose claramente frente a los demás.
   - **Menor recuperación:** Experimento 3 (r=0.1577) tiene la recuperación más baja, reflejando una menor capacidad para abarcar información relevante.
2. **Precisión (p):**
   - **Mayor precisión:** Experimento 2 (p=0.3743p) logra el valor más alto, indicando un modelo más selectivo en la inclusión de información relevante.
   - **Menor precisión:** Luhn (p=0.2844) y K-Means (p=0.2829) presentan valores similares, inferiores a los experimentos.
3. **Medida F1 (f):**

- **Mejor equilibrio (F1):** Luhn (f=0.3371) obtiene la mayor medida F1, equilibrando efectivamente la recuperación y precisión.
- **Menor F1:** Experimento 1 (f=0.2213) presenta el menor balance entre estas métricas.

5.2.1.1. Observación general

El modelo **Luhn** se distingue por su alto desempeño en recuperación y medida F1, mostrando un equilibrio robusto y capturando gran cantidad de información relevante. Por su parte, **Experimento 2** destaca únicamente en precisión, lo que sugiere un enfoque más selectivo, aunque menos equilibrado en términos globales. Los experimentos presentan un rendimiento menor en recuperación y F1 en comparación con Luhn y K-Means.

5.2.2. ROUGE-2

Modelo	Tipo	Media	Desviación Estándar
Experimento 1	r	0.0268	0.0165
	p	0.0304	0.0169
	f	0.0277	0.0151
Experimento 2	r	0.0311	0.0197
	p	0.0360	0.0251
	f	0.0326	0.0208
Experimento 3	r	0.0233	0.0146
	p	0.0377	0.0250
	f	0.0281	0.0172
Luhn	r	0.1535	0.0838
	p	0.0920	0.0760
	f	0.1097	0.0692
K-Means	r	0.1075	0.0613
	p	0.0847	0.0577
	f	0.0913	0.0543

Cuadro 5.7: Resultados de la métrica ROUGE-2 por modelo

A partir de los resultados obtenidos en la Tabla 5.7, se destacan las siguientes observaciones para la métrica ROUGE-2:

1. **Recuperación (r):**
   - **Mayor recuperación:** Luhn (r=0.1535) es el modelo que capta la mayor cantidad de información relevante en términos de bigramas.

- **Menor recuperación:** Experimento 3 (r=0.0233) tiene el valor más bajo, con un desempeño considerablemente inferior a los demás.

2. **Precisión (p):**

- **Mayor precisión:** Experimento 3 (p=0.0377) se posiciona como el modelo más selectivo en la inclusión de bigramas relevantes.

- **Menor precisión:** Luhn (p=0.0920) y K-Means (p=0.0847) tienen valores cercanos, inferiores a los experimentos.

3. **Medida F1 (f):**

- **Mejor equilibrio (F1):** Luhn (f=0.1097) logra el mejor balance entre recuperación y precisión.

- **Menor F1:** Experimento 1 (f=0.0277) tiene el valor más bajo, reflejando una capacidad limitada para equilibrar ambas métricas.

5.2.2.1. Observación general

El modelo **Luhn** nuevamente destaca por su recuperación y medida F1, posicionándose como el modelo más equilibrado y efectivo en términos globales. **Experimento 3**, aunque lidera en precisión, muestra una recuperación considerablemente menor, lo que afecta su medida F1. Los demás experimentos presentan un rendimiento más limitado en comparación con Luhn y K-Means, especialmente en recuperación y F1, mostrando su enfoque más selectivo pero menos efectivo para la métrica ROUGE-2.

5.2.3. ROUGE-L

Modelo	Tipo	Media	Desviación Estándar
Experimento 1	r	0.1318	0.0389
	p	0.2419	0.0528
	f	0.1673	0.0371
Experimento 2	r	0.1366	0.0347
	p	0.2810	0.0575
	f	0.1810	0.0359
Experimento 3	r	0.1192	0.0347
	p	0.3135	0.0771
	f	0.1699	0.0435
Luhn	r	0.3150	0.0957
	p	0.2002	0.0833
	f	0.2372	0.0746
K-Means	r	0.2561	0.0755
	p	0.1980	0.0670
	f	0.2171	0.0599

Cuadro 5.8: Resultados de la métrica ROUGE-L por modelo

Con base en los resultados de la Tabla 5.8, se destacan las siguientes observaciones relacionadas con la métrica ROUGE-L:

1. **Recuperación (r):**
   - **Mayor recuperación:** Luhn (r=0.3150) demuestra la mayor capacidad para identificar información relevante basada en la longitud de las coincidencias comunes.
   - **Menor recuperación:** Experimento 3 (r=0.1192) obtiene el valor más bajo, destacando una menor cobertura de información relevante.
2. **Precisión (p):**
   - **Mayor precisión:** Experimento 3 (p=0.3135) lidera en esta métrica, indicando que este modelo es más selectivo al incluir información relevante.
   - **Menor precisión:** K-Means (p=0.1980) y Luhn (p=0.2002) tienen valores similares y son menos selectivos en comparación con los experimentos.
3. **Medida F1 (f):**

- **Mejor equilibrio (F1):** Luhn (f=0.2372) obtiene el mayor balance entre recuperación y precisión, consolidándose como el modelo más equilibrado.
- **Menor F1:** Experimento 1 (f=0.1673) presenta un equilibrio más limitado, reflejando menor efectividad en ambas métricas.

5.2.3.1. Observación general

El modelo **Luhn** se distingue por su alto desempeño en recuperación y medida F1, consolidándose como el más equilibrado y eficiente en ROUGE-L. Aunque **Experimento 3** destaca en precisión, su baja recuperación afecta su rendimiento global. En términos generales, los experimentos muestran un enfoque más selectivo pero menos equilibrado en comparación con Luhn y K-Means, siendo estos últimos más robustos para tareas de resumen basadas en coincidencias de longitud.

5.2.4. BERTScore

Modelo	Tipo	Media	Desviación Estándar
Experimento 1	r	0.7969	0.0095
	p	0.8190	0.0089
	f	0.8077	0.0077
Experimento 2	r	0.7922	0.0092
	p	0.8099	0.0117
	f	0.8009	0.0089
Experimento 3	r	0.7908	0.0104
	p	0.8104	0.0162
	f	0.8004	0.0114
Luhn	r	0.8417	0.0219
	p	0.8156	0.0276
	f	0.8283	0.0222
K-Means	r	0.8343	0.0192
	p	0.8208	0.0249
	f	0.8273	0.0195

Cuadro 5.9: Resultados de la métrica BERTScore por modelo

Del análisis de la Tabla 5.9, se obtienen las siguientes observaciones sobre el desempeño de los modelos basados en la métrica **BERTScore**:

1. **Recuperación (r):**

- **Mayor recuperación:** Luhn (r=0.8417) tiene el mejor desempeño, capturando la mayor cantidad de información semánticamente relevante.
- **Menor recuperación:** Experimento 3 (r=0.7908) obtiene el valor más bajo, aunque con diferencias mínimas frente a los demás experimentos.

2. **Precisión (p):**

- **Mayor precisión:** K-Means (p=0.8208) destaca como el modelo más selectivo en la inclusión de información semánticamente relevante, seguido muy de cerca por Luhn (p=0.8156).
- **Menor precisión:** Experimento 1 (p=0.8190) y los demás experimentos muestran valores similares, con diferencias prácticamente insignificantes.

3. **Medida F1 (f):**

- **Mayor equilibrio (F1):** Luhn (f=0.8283) logra el mejor balance entre recuperación y precisión, seguido de cerca por K-Means (f=0.8273).
- **Menor F1:** Experimento 3 (f=0.8004) tiene el valor más bajo, aunque con diferencias menores respecto a los otros experimentos.

5.2.4.1. Observación general

El modelo **Luhn** lidera en recuperación y medida F1, mostrando un equilibrio sólido y una capacidad destacada para identificar información relevante a nivel semántico. **K-Means**, aunque ligeramente inferior, también demuestra un rendimiento robusto y balanceado. Por otro lado, los **experimentos** presentan resultados consistentes pero inferiores, con pequeñas variaciones en recuperación y F1, destacando en precisión con valores competitivos respecto a los mejores modelos. En general, las diferencias entre los modelos son pequeñas, lo que sugiere un alto nivel de competencia en esta métrica.

Capítulo 6

Conclusiones y Proyecciones Futuras

6.1. Conclusiones

En este proyecto, se exploraron y desarrollaron diversos modelos aplicando tanto técnicas de aprendizaje automático como métodos estadísticos, con un enfoque principal en la generación automática de resúmenes para textos académicos. El objetivo principal fue evaluar la capacidad de cada modelo para generar resúmenes, considerando enfoques tanto extractivos como abstractivos. A lo largo del desarrollo del proyecto, se identificaron varios retos y observaciones que fueron clave en la toma de decisiones, permitiendo así cumplir con el objetivo principal de ser capaces de sintetizar la información en estos tipos de textos en particular.

Tras evaluar los diferentes modelos aplicados, se observó que el modelo Luhn, basado en heurísticas, mostró el mejor desempeño según las métricas utilizadas. En particular, los resultados obtenidos con ROUGE y BERTScore fueron superiores, lo que indica que este modelo fue el más efectivo para generar resúmenes que mantuvieran la coherencia y relevancia de los textos académicos. Este modelo demostró ser el más adecuado para la tarea, destacándose frente a otros enfoques que no lograron el mismo nivel de precisión. No obstante, otro modelo que mostró un desempeño notable fue K-Means, que, aunque se encuentra dentro de los enfoques de aprendizaje no supervisado, obtuvo métricas muy positivas. Su capacidad para agrupar de manera efectiva las características clave del texto lo posiciona como la opción más relevante dentro de los enfoques explorados en aprendizaje automático.

En cuanto a las métricas utilizadas, se notó que ROUGE, aunque ampliamente utilizada, tiende a ser más estricta debido a su enfoque en la coincidencia exacta de términos. Esto puede llevar a que se subestime la calidad de los resúmenes generados por modelos abstractivos, como los Seq2Seq, que pueden producir resúmenes con una estructura diferente pero igualmente coherentes. Por otro lado, métricas como BERTScore, que evalúan la similitud semántica entre los resúmenes generados y los originales, permiten una comparación más flexible y cercana a los enfoques extractivos. En este caso, BERTScore mostró una mayor capacidad para igualar los enfoques abstractivos con los extractivos, ya que toma en cuenta la relación semántica entre las palabras y no solo su coincidencia exacta. Esto permitió que los modelos Seq2Seq, a pesar de sus limitaciones computacionales, obtuvieran mejores puntuaciones en comparación con las que obtenían con ROUGE.

Por ultimo, las principales dificultades en el proyecto estuvieron relacionadas con las limitaciones de los modelos Seq2Seq, especialmente debido a los altos requerimientos computacionales para procesar textos largos y complejos. Estas restricciones afectaron significativamente el rendimiento de los modelos, impidiendo que alcanzaran su máximo potencial. Como enseñanza, es crucial tener en cuenta la infraestructura disponible al trabajar con este tipo de modelos y considerar la optimización

de parámetros o el uso de modelos preentrenados para reducir los recursos necesarios. Además, técnicas de preprocesamiento, como la reducción de la longitud de los textos, podrían mejorar la eficiencia sin sacrificar la calidad de los resúmenes generados.

6.2. Proyecciones Futuras

Este proyecto en particular, presenta diversas proyecciones futuras que son realmente interesantes de explorar, a continuación mencionamos algunas de las posibilidades:

Optimización del Modelo Actual

Una de las principales áreas de mejora es la optimización del desempeño de los modelos utilizados. Esto podría lograrse entrenando el sistema con conjuntos de datos más amplios y diversificados que abarquen diversas disciplinas académicas. Asimismo, la incorporación de modelos de última generación, como GPT-4 o T5, permitiría mejorar tanto la coherencia como la precisión de los resúmenes generados.

Ampliación de Funcionalidades

En el futuro, el proyecto podría extenderse para incluir funcionalidades como:

- Soporte para múltiples idiomas, permitiendo ya no solo generar el resumen, sino obtenerlo en el lenguaje que se requiera.
- Integración de resúmenes multimodales que combinen texto con elementos visuales como gráficos, tablas o diagramas.
- Herramientas para ajustar el nivel de detalle de los resúmenes según las necesidades del usuario.

Aplicaciones en Contextos Específicos

Este sistema podría adaptarse para abordar necesidades específicas en campos como:

- Medicina, donde los resúmenes podrían ayudar a procesar grandes volúmenes de literatura clínica.
- Derecho, facilitando el análisis de documentos legales extensos.
- Educación, asistiendo a estudiantes y profesores en la síntesis de materiales académicos.

Bibliografía

[1] J. Devlin, M.-W. Chang, K. Lee, y K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,"*arXiv preprint arXiv:1810.04805*, 2018.

[2] S. Subramanian, R. Li, J. Pilault, y C. Pal, .ºn Extractive and Abstractive Neural Document Summarization with Transformer Language Models,"*arXiv preprint arXiv:1909.03186*, 2019.

[3] Wang Guan, Ivan Smetannikov, and Man Tianxing. 2021. Survey on Automatic Text Summarization and Transformer Models Applicability. In Proceedings of the 2020 1st International Conference on Control, Robotics and Intelligent System (CCRIS '20). Association for Computing Machinery, New York, NY, USA, 176–184. https://doi.org/10.1145/3437802.3437832

[4] T. R. N and R. Gupta, "A Survey on Machine Learning Approaches and Its Techniques:," 2020 IEEE International Students' Conference on Electrical,Electronics and Computer Science (SCEECS), Bhopal, India, 2020, pp. 1-6, doi: 10.1109/SCEECS48394.2020.190.

[5] M. Zhang and C. Luan, "Extractive-Abstractive Text Automatic Summary Based on the BERT Graph Model," 2024 International Conference on Electrical Drives, Power Electronics & Engineering (EDPEE), Athens, Greece, 2024, pp. 619-623, doi: 10.1109/EDPEE61724.2024.00121.

[6] N. F. Baarir, S. Bourekkache and A. Aloui, "The use of Deep Learning techniques in E-Learning systems and MOOCs," 2022 International Symposium on iNnovative Informatics of Biskra (ISNIB), Biskra, Algeria, 2022, pp. 1-6, doi: 10.1109/ISNIB57382.2022.10076002.

[7] M. Chandra, V. Gupta and S. K. Paul, "A Statistical Approach for Automatic Text Summarization by Extraction," 2011 International Conference on Communication Systems and Network Technologies, Katra, India, 2011, pp. 268-271, doi: 10.1109/CSNT.2011.65.

[8] K. Srividya, S. K. Bommuluri, V. V. Vinod Kumar Asapu, T. R. Illa, V. R. Basa and R. V. Surendra Chatradi, "A Hybrid Approach for Automatic Text Summarization and Translation based On Luhn, Pegasus, and Textrank Algorithms," 2022 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON), Bangalore, India, 2022, pp. 1-8, doi: 10.1109/SMARTGENCON56628.2022.10083732.

[9] T. R. N and R. Gupta, "A Survey on Machine Learning Approaches and Its Techniques:" 2020 IEEE International Students' Conference on Electrical,Electronics and Computer Science (SCEECS), Bhopal, India, 2020, pp. 1-6, doi: 10.1109/SCEECS48394.2020.190.

[10] M. M. Haider, M. A. Hossin, H. R. Mahi and H. Arif, "Automatic Text Summarization Using Gensim Word2Vec and K-Means Clustering Algorithm," 2020 IEEE Region 10 Symposium (TENSYMP), Dhaka, Bangladesh, 2020, pp. 283-286, doi: 10.1109/TENSYMP50017.2020.9230670.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 6000–6010.

[12] C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, Barcelona, Spain, Jul. 2004.

[13] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "BERTScore: Evaluating Text Generation with BERT,ïn Proceedings of the International Conference on Learning Representations (ICLR), 2020. Available: https://openreview.net/forum?id=SkeHuCVFDr.

[14] I. M. K. Karo, A. Perdana and S. Dewi, "Automatic Text Review Summarization of Digital Library System Application using TextRank Algorithm and TF-IDF," 2024 4th International Conference of Science and Information Technology in Smart Administration (ICSINTESA), Balikpapan, Indonesia, 2024, pp. 570-575, doi: 10.1109/ICSINTESA62455.2024.10747952.

[15] S. M. Tasnimul Hasan, M. A. Rahman, M. M. Hasan, M. R. Hasan and M. M. Hoque, "Advancing Abstractive Bangla Text Summarization: A Deep Learning Approach Using Seq2seq Encoder- Decoder Model and T5 Transformer" 2023 5th International Conference on Sustainable Technologies for Industry 5.0 (STI), Dhaka, Bangladesh, 2023, pp. 1-6, doi: 10.1109/STI59863.2023.10464712.

[16] https://www.tensorflow.org/datasets/catalog/scientific_papers?hl=es-419