COMPARACIÓN DE METODOLOGÍAS PEFT APLICADAS A MODELOS DE LENGUAJE GRANDES ENFOCADOS EN GENERAR RESUMENES DE TEXTOS

BRYAN STIVEN FLOREZ PAZOS
Código: 8985092

CARLOS ANDRES ARÉVALO RODRÍGUEZ
Código: 8986106

SEBASTIÁN BARRERA SÁENZ
Código: 8986386

Proyecto Aplicado para optar al título de

Magister en Ciencia de Datos

Director(a)

Omar Andrés Castaño

FACULTAD DE INGENIERÍA Y CIENCIAS

MAESTRÍA EN CIENCIA DE DATOS

SANTIAGO DE CALI, DICIEMBRE 5 DE 2024

TABLA DE CONTENIDO

1 INTRODUCCION 6
2 DEFINICIÓN DEL PROBLEMA 7
2.1 PLANTEAMIENTO DEL PROBLEMA 7
2.2 FORMULACIÓN DEL PROBLEMA 8
3 OBJETIVOS DEL PROYECTO 10
3.1 OBJETIVO GENERAL 10
3.2 OBJETIVOS ESPECÍFICOS 10
4 MARCO DE REFERENCIA 11
4.1 MARCO TEÓRICO 11
4.1.1 GENERACIÓN AUTOMÁTICA DE RESÚMENES 11
4.1.2 ANALÍTICA DE TEXTOS 11
4.1.3 PROCESAMIENTO DE TEXTOS 11
4.1.4 METRICAS DE EVALUACIÓN DE TEXTOS 12
4.1.5 MODELO T5 13
4.1.6 MODELO GPT-2 13
4.1.7 DEEP LEARNING 13
4.1.8 PROCESAMIENTO DE LENGUAJE NATURAL 15
4.1.9 MODELOS DE LENGUAJE GRANDE (LLM) 15
4.1.10 MODELOS DE LENGUAJE PRE-ENTRENADOS 15
4.1.11 FINE-TUNING 16
4.1.12 PARAMETER EFFICIENT FINE-TUNING (PEFT) 16
4.2 ANTECEDENTES Y TRABAJOS RELACIONADOS 16
5 DESARROLLO E IMPLEMENTACION 22
5.1 RECOLECCIÓN DE DATOS 23
5.1.1 SELECCIÓN DE MODELOS LLM MÁS CONVENIENTES PARA TEXT SUMMARIZING 23
5.1.2 ANÁLISIS EXPLORATORIO DE LOS DATASET SELECCIONADOS 25
5.1.3 PREPARACIÓN DE LOS DATOS 31
5.2 IMPLEMENTACIÓN DE METODOLOGÍAS PEFT 36
5.3 EVALUACIÓN DE RESULTADOS 41
5.4 RENDIMIENTO COMPUTACIONAL 45
6 CONCLUSIONES Y TRABAJOS FUTUROS 48
6.1 CONCLUSIONES 48
6.2 TRABAJOS FUTUROS 49
7 REFERENCIAS BIBLIOGRÁFICAS 51

LISTA DE TABLAS

Tabla 1. Información modelos seleccionados (Elaboración propia).....25
Tabla 2 Ejemplo de información contenida en el dataset Dialogsum: [30]. 26
Tabla 3. Ejemplo de información contenida en el dataset Samsum. [31]. 28
Tabla 4. Ejemplo de información contenida en el dataset cnn_dailymail [32]. 30
Tabla 5. Hiperparametros entrenamiento modelo T5. 37
Tabla 6. Hiperparametros entrenamiento modelo GPT-2. 37
Tabla 7.Resultados preliminares obtenidos (Elaboración propia). 41
Tabla 8. Resultados obtenidos. (Elaboración propia).....43
Tabla 9. Ocupación en disco de los modelos (Elaboración propia). 45
Tabla 10. Consumo de recursos durante el entrenamiento (Elaboración propia). 46

LISTA DE FIGURAS

Figura 1. Métodos PEFT mencionados con sus tres clases principales y subgrupos descritos [20].....17
Figura 2. Pipeline utilizado PEFT-MedAware [20]. 19
Figura 3. Metodología de desarrollo (Elaboración propia). 22
Figura 4. Arquitectura modelo T5 [28]. 24
Figura 5. Arquitectura GPT-2 [29].....24
Figura 6. Histograma con la cantidad de tokens en los textos de entrada (text) – Dataset dialogsum (Elaboración propia) 27
Figura 7. Histograma con la cantidad de tokens de entrada en la columna de resumen (summary) – Dataset dialogsum (Elaboración propia) 27
Figura 8. Histograma con la cantidad de tokens en los textos de entrada (text) - Dataset SamSum. Fuente: Elaboración Propia. 28
Figura 9. Histograma con la cantidad de tokens de entrada en la columna de resumen (summary) - Dataset SamSum (Elaboración propia).....29
Figura 10. Histograma con la cantidad de tokens en los textos de entrada (text) - Dataset Cnn_dailymail (Elaboración propia) 30
Figura 11. Histograma con la cantidad de tokens de entrada en la columna de resumen (summary) - Dataset Cnn_dailymai (Elaboración propia).....31
Figura 12. Histograma con la cantidad de tokens para la columna combinada (merge_tokens) – Dataset dialogsum (Elaboración propia).....33
Figura 13. Histograma con la cantidad de tokens para la columna combinada (merge_tokens) - Dataset SamSum (Elaboración propia) 34
Figura 14. Histograma con la cantidad de tokens para la columna combinada (merge_tokens) – Dataset Cnn_dailymail (Elaboración propia).....35
Figura 15. Diagrama de flujo - Proceso de entrenamiento y aplicación de Fine Tuning (Elaboración propia).....39
Figura 16. Implementación de metodologías PEFT (Elaboración propia). 40

1 INTRODUCCION

En la era digital actual, caracterizada por la abrumadora cantidad de información disponible, los modelos de lenguaje grandes destacan como poderosas herramientas para el procesamiento y comprensión de textos extensos. Entre las diversas tareas que estos modelos abordan, la generación de resúmenes de texto se destaca como una función esencial para simplificar la información y hacerla más accesible. Sin embargo, la eficacia de esta tarea depende en gran medida de la metodología subyacente, y en el contexto de modelos de lenguaje avanzados, la elección de la técnica adecuada se convierte en un desafío significativo.

Nos enfrentamos a la incertidumbre de determinar qué modelos de LLM, junto con los métodos de Fine-Tuning dentro de las metodologías PEFT, son más adecuados para generar resúmenes eficientemente. Los tiempos de generación, el consumo energético y otros parámetros varían según el método y el modelo seleccionado. Por tanto, fue crucial elegir una metodología que optimizara el proceso, como las PEFT, que reducen los parámetros del modelo para agilizar el aprendizaje y la generación de resúmenes.

La aplicación de metodologías PEFT en la generación de resúmenes mediante LLM planteó el desafío de identificar el enfoque más adaptable y eficaz. La diversidad de modelos y la dinámica del lenguaje agregaron complejidad, resaltando la necesidad de una evaluación exhaustiva de estas metodologías. Nuestro proyecto analizó los diferentes métodos de Fine-Tuning y metodologías aplicadas a la generación de resúmenes, con el objetivo de comprender mejor su idoneidad y contribuir a la selección informada de técnicas en el procesamiento de lenguaje natural.

En esta investigación se adoptó un enfoque integral que combinó la exploración teórica con la experimentación práctica. Para esto la estrategia ejecutada consistió en realizar una revisión exhaustiva de la aplicación de metodologías de Fine-Tuning en modelos de procesamiento de lenguaje natural (NLP) en la cual se destacaron sus principios fundamentales, fortalezas y limitaciones. Este análisis teórico sirvió como fundamento para comprender la diversidad de enfoques y sus aplicaciones en la generación de resúmenes de texto. También se realizaron experimentos prácticos mediante la implementación de dichas metodologías en modelos de lenguaje grandes. Estos experimentos se diseñaron cuidadosamente para evaluar el rendimiento de cada enfoque en términos de precisión, coherencia y adaptabilidad a la complejidad lingüística.

Este enfoque permitió abordar no solo la complejidad teórica de las metodologías PEFT y métodos de Fine-Tuning, sino también su viabilidad y eficacia en escenarios prácticos donde el objetivo es realizar resúmenes de texto. La combinación de análisis teóricos y experimentación empírica constituye la base sólida sobre la cual se construyen conclusiones informadas, contribuyendo así a resolver la incertidumbre sobre elección de métodos y metodologías de Fine-Tuning son los más apropiados para la generación de resúmenes en modelos de lenguaje grandes.

Finalmente, se proporciona una comparación detallada del rendimiento de diferentes metodologías PEFT en términos de precisión, coherencia y eficacia en la generación de resúmenes de texto. En modelos LLM base como: GPT-2 (Normal, Medium y Large) y T5 (Normal y Large). Evaluando métricas: Bleu, n-gram precision y Rouge. Lo cual nos permitió identificar que en modelos de tamaño mediano y grande (GPT-2 Large y GPT-2 Medium) y T5 Large presentan una mejora rendimiento al aplicarse las metodologías PEFT, en contraste con los modelos pequeños (GPT-2 y T5) se obtuvo una ligera disminución del rendimiento cuando se aplican las metodologías PEFT sobre estos.

2 DEFINICIÓN DEL PROBLEMA

2.1 PLANTEAMIENTO DEL PROBLEMA

Cada día, el flujo de información aumenta en diversos ámbitos como: empresas, internet, aplicaciones, redes, entre otras. Este crecimiento plantea un desafío considerable que radica en procesar grandes volúmenes de información, donde gran parte de ésta se encuentra en textos extensos, lo cual dificulta la lectura bien sea por su magnitud o el tiempo disponible para ejecutar el ejercicio de lectura. En este contexto, los modelos de procesamiento de lenguaje natural (NLP) y Deep Learning se han vuelto esenciales, pues no solo permiten extraer las ideas más importantes de textos extensos, sino que también son capaces de entender el contexto y la intención detrás de las palabras utilizando técnicas de NLP, más concretamente Large Language Models diseñados específicamente para comprender y generar lenguaje humano. Asimismo, dichos modelos utilizan grandes cantidades de datos para aprender patrones lingüísticos y generar texto coherente y relevante, combinados con la capacidad de los modelos de Deep learning para aprender representaciones complejas y patrones en los datos, se logra generar resúmenes precisos y contextuales. Un ejemplo de esto son los modelos pre-entrenados, como BERT o GPT, que han aprendido representaciones lingüísticas complejas a partir de grandes volúmenes de datos, lo que les permite capturar matices semánticos y contextuales en el texto.

Ahora bien, es esencial poder leer y comprender la gran cantidad de datos e información que encontramos día a día en textos de todo tipo como: periódicos, informes, libros, revistas, entre otros. Es aquí donde los métodos para simplificar o resumir esa información aparecen como ayuda a dicho propósito. En este punto la generación de resúmenes es una solución al problema relacionado con la sobrecarga de información en la era digital. El poder gestionar eficientemente grandes volúmenes de texto permite condensar información extensa en versiones concisas, proporcionando a los usuarios acceso rápido a la esencia del contenido sin la necesidad de revisar textos completos. Además, en entornos profesionales y académicos, donde la velocidad en la obtención de información es crucial, la generación automática de resúmenes se presenta como una herramienta esencial para la toma de decisiones informadas y la optimización del tiempo. Además, otra problemática encontrada es la personalización de la información. Al adaptarse mediante técnicas como el fine-tuning a dominios específicos, la generación automática de resúmenes garantiza la relevancia contextual y la captura precisa de detalles importantes, superando la limitación de enfoques estándar que podrían pasar por alto matices cruciales.

Para lograr un resumen apropiado y eficaz se necesita un modelo adecuado, así como un texto que cumpla con ciertos criterios para ejecutar un resumen adecuado, pero la creación de un modelo de lenguaje desde cero presenta desafíos significativos, principalmente por la complejidad inherente del procesamiento del lenguaje natural y la cantidad de datos requeridos para su correcto entrenamiento. Las dificultades incluyen la demanda computacional masiva, ya que el entrenamiento de modelos de lenguaje requiere recursos significativos, como unidades de procesamiento gráfico (GPU) potentes y gran capacidad de memoria. Además, la elaboración de arquitecturas efectivas y la afinación de hiper-parámetros para garantizar el rendimiento óptimo se convierten en un proceso extenso e intensivo de ensayo y error.

En este contexto el Fine-Tuning es un tipo especifico de transferencia de aprendizaje (transfer learning) que busca aprovechar un modelo pre-entrenado como punto de partida para que aprenda una nueva tarea con un conjunto de datos nuevo. Esto se consigue ajustando el peso de los

parámetros de entrada del modelo con el nuevo dataset. En el caso de modelos de lenguaje natural este proceso ofrece ventajas significativas al superar barreras asociadas con la creación de modelos desde cero. Al adaptar un LLM existente a tareas específicas, se capitaliza el conocimiento previo y las complejas representaciones lingüísticas aprendidas durante el preentrenamiento en grandes conjuntos de datos. Esto resulta en una eficiencia sustancial en términos de recursos computacionales y tiempo de entrenamiento, ya que el modelo pre-entrenado posee una comprensión profunda del lenguaje natural. Además, el Fine-Tuning permite la adaptación a conjuntos de datos más pequeños y específicos de la tarea, lo cual es crucial cuando los datos disponibles son limitados.

Normalmente para realizar Fine Tuning se deben ajustar todos los parámetros de un modelo para cada tarea posterior, pero debido a los requerimientos que este solicita es extremadamente costoso en términos de tiempo y dinero, dado que esto requiere una infraestructura robusta para realizarse como capacidad de almacenamiento y procesamiento (GPU), sin mencionar la enorme cantidad de parámetros en los modelos actuales que se deben tener en cuenta. Para solucionar esta problemática las metodologías PEFT (Parameter Efficent Fine-Tuning) son un conjunto de métodos que buscan optimizar el proceso de fine-tuning de un modelo, identificando y actualizando los parámetros más relevantes del modelo pre-entrenado para que se alineen mejor con la tarea específica y poder así mejorar significativamente la calidad y precisión de los resultados generados.[1]

Este proyecto implementó un modelo de procesamiento de lenguaje natural (NLP) enfocado en realizar resúmenes de textos y a su vez utilizar diferentes técnicas de Fine-Tuning implementando las metodologías PEFT, las cuales permiten adaptar de manera eficiente un modelo de lenguaje previamente entrenado (PLM) a varias aplicaciones posteriores sin ajustar todos los parámetros del modelo. Esto se traduce en una reducción significativa de los costos computacionales y de almacenamiento, ya que el Fine-Tuning a gran escala es demasiado costoso. Para lograr esto, uno de los métodos usados en PEFT es la adaptación de rango bajo (LoRA), la cual consiste en entrenar una cantidad menor de parámetros rápidos o utilizar un método de reparametrización que ayuda a sacar mejor provecho al modelo que será posteriormente usado, en este caso, para generar resúmenes consistentes y de una manera más favorable tanto para los desarrolladores como para su implementación en un ámbito real [2]. Además del método anteriormente mencionado, existen otras técnicas que contribuyen a resolver las problemáticas discutidas, facilitando el entrenamiento de los modelos seleccionados, y haciendo más efectiva, rápida y económica la generación automática de resúmenes.

Teniendo en cuenta esto, se plantea la pregunta principal por la cual se lleva a cabo el proyecto y la cual es respondida con el desarrollo de este.

2.2 FORMULACIÓN DEL PROBLEMA

¿Qué método de Fine-Tuning en el marco de las metodologías PEFT optimiza el rendimiento y la eficiencia computacional en un modelo de lenguaje preentrenado para generar resúmenes adaptables a distintos contextos (académico, profesional e informativo)?

Teniendo en cuenta esta pregunta clave y también la necesidad de profundizar en los detalles del proyecto, se formulan las siguientes cuestiones que guiaron el proceso y ayudaron a responder los objetivos planteados, tales como:

1. ¿Qué criterios se deben considerar para seleccionar artículos, informes o conversaciones cortas adecuados para entrenar un modelo LLM en generación de resúmenes?

2. ¿Qué diferencias se observan en el rendimiento del modelo al aplicar distintas técnicas de Fine-Tuning enfocadas en la optimización de recursos?

3. ¿Cómo varían los resultados de los modelos al utilizar diferentes enfoques de PEFT en términos de precisión, consistencia y calidad de los resúmenes generados?

4. ¿Cuál es el impacto de estas metodologías en el proceso de adaptación y Fine-Tuning en modelos LLM destinados a la generación de resúmenes?

3 OBJETIVOS DEL PROYECTO

3.1 OBJETIVO GENERAL

Determinar qué modelo/s de procesamiento de lenguaje natural de tipo LLM resume textos de forma eficaz, al ser ajustados haciendo uso de Fine-Tuning y diferentes metodologías PEFT (Parameter Efficient Fine-Tuning), buscando mejorar la velocidad de inferencia, reducción de costos computacionales y lograr resultados óptimos al aplicarse a textos de conversaciones cortas, así como a artículos publicados en páginas de noticias y/o revistas.

3.2 OBJETIVOS ESPECÍFICOS

- Recolectar datos que sean adecuados para el entrenamiento de modelos de procesamiento de textos enfocados en LLM, dentro de los cuales se propone artículos, informes o posts cortos de diferentes medios de información y/o conversaciones cortas.
- Implementar metodologías PEFT sobre los modelos LLM seleccionados buscando determinar cómo estas mejoran la capacidad del modelo para generar resúmenes de texto.
- Evaluar los resultados obtenidos en la ejecución de los modelos comparando métricas de precisión, coherencia y relevancia, para identificar el modelo que demuestra mayor eficacia en la generación de resúmenes de textos.
- Comparar la eficiencia en el uso de recursos computacionales al emplear la metodología PEFT (LoRA) en comparación con el fine-tuning convencional, contrastando ventajas y desventajas.

4 MARCO DE REFERENCIA

4.1 MARCO TEÓRICO

En el avance de la inteligencia artificial, el procesamiento del lenguaje natural ha surgido como un campo de estudio fundamental, con aplicaciones que abarcan desde la traducción automática hasta la generación de resúmenes de texto. En este contexto, el fine-tuning en apoyo con metodologías eficientes como por ejemplo PEFT, han demostrado ser una estrategia eficaz para adaptar modelos pre-entrenados a tareas específicas. Este marco teórico se sumerge en la convergencia de estas dos áreas, explorando la aplicación del fine-tuning en la generación de resúmenes textuales. Se examinará la base conceptual del procesamiento del lenguaje natural, los modelos pre-entrenados, y se profundizará en la técnica del fine-tuning con metodologías eficientes como instrumento clave para mejorar la capacidad de los modelos en la tarea específica de resumir información textual. Al comprender y evaluar estas intersecciones, se busca aportar una perspectiva integral y crítica a la aplicación de técnicas de fine-tuning en el contexto del resumen automático de textos.

4.1.1 GENERACIÓN AUTOMÁTICA DE RESÚMENES

La generación automática de resúmenes de texto ha experimentado notables mejoras gracias a la implementación de técnicas avanzadas de aprendizaje automático. En este contexto, el ajuste de modelos de lenguaje preentrenados ha surgido como una estrategia prometedora para adaptar estos modelos a tareas específicas como la generación de resúmenes [2].

Ahora bien, dada la creciente demanda de modelos preentrenados como BERT o GPT-3, realizar *fine-tuning* sobre estos modelos se ha convertido en el estándar para optimizar su rendimiento en la tarea deseada [3]. En el caso de la generación de resúmenes, este enfoque permite personalizar el modelo para capturar la esencia y relevancia del contenido en un dominio particular. Esto es esencial, ya que los modelos preentrenados a menudo carecen de información específica sobre el contexto y la terminología de un campo particular. Además, el *fine-tuning* mejora significativamente la capacidad del modelo para generar resúmenes concisos y coherentes, manteniendo la calidad lingüística y semántica del texto original [4].

4.1.2 ANALÍTICA DE TEXTOS

La analítica de textos desempeña un papel fundamental en la generación automática de resúmenes al proporcionar una profunda comprensión del contenido textual, dada la capacidad de extraer información relevante, identificar patrones y entender el contexto esencial para la generación de resúmenes efectivos. La analítica de textos no permite solo procesar la información de manera estructura, sino también capturar la semántica y la intención subyacente en el texto original. Al integrar técnicas analíticas avanzadas, como el análisis de sentimientos, identificación de entidades y extracción de características clave, los sistemas de generación de resúmenes pueden adaptarse de manera más precisa a la naturaleza específica de los datos, produciendo resúmenes que no solo son concisos, sino también informativos y contextualmente ricos [5].

4.1.3 PROCESAMIENTO DE TEXTOS

Constituye el primer paso para garantizar la calidad y relevancia de los resúmenes resultantes. En esta fase, se lleva a cabo un minucioso preprocesamiento del texto original, que abarca desde la eliminación de caracteres especiales y etiquetas, así como procesos de tokenización y lematización

para simplificar la representación del contenido. La limpieza y estructuración del texto son fundamentales para garantizar que el modelo de generación pueda capturar de manera efectiva las características esenciales y la semántica subyacente[6]. Además, la cuidadosa selección de técnicas de representación del texto, como incrustaciones de palabras o modelos pre-entrenados, contribuye a la construcción de una base sólida para el análisis y la síntesis que se realizará en etapas posteriores del proceso.

- **Tokenización:** Es el paso de romper una cadena de caracteres o texto en unidades atómicas usadas en el modelo. Hay varias estrategias de tokenización que se pueden adoptar y la división óptima de palabras en las subunidades generalmente se aprenden del corpus [5]. En otras palabras, la tokenización permite convertir el texto en una estructura más manejable y estructurada, facilitando el análisis y la comprensión por parte de los modelos de procesamiento de lenguaje natural ya que, al segmentar el texto en unidades más pequeñas, se logra una representación más granular del contenido y preserva la semántica del texto al mantener la integridad de las palabras individuales o sub-palabras, permitiendo que el modelo capture con precisión la información esencial presente en el texto original.
- **Lematización:** La lematización es el proceso de reducir palabras a su forma base o de diccionario, conocida como lema. Implica identificar la parte del discurso y el contexto de la palabra para determinar el lema correcto[2]. A diferencia de la extracción de raíces, la lematización considera la morfología y la gramática del idioma, generando formas que son palabras válidas y que tienen significado. Este proceso contribuye significativamente a la coherencia semántica del texto al unificar variantes de una palabra y reducir la dimensionalidad del vocabulario.

4.1.4 METRICAS DE EVALUACIÓN DE TEXTOS

- **Blue Score (Bilingual Evaluation Understudy)**: El BLEU score evalúa de manera automática comparando n-gramas entre el texto generado y un texto de referencia. Se mide la precisión a nivel de n-gramas (palabras o secuencias de palabras) coincidentes. El sistema incluye una penalización por brevedad para evitar que los textos generados sean demasiado cortos, asegurando que la longitud sea cercana a la de la referencia. BLEU es especialmente efectivo cuando se usa para comparar múltiples textos de referencia [7].

- **ROUGE (Recall-Oriented Understudy for Gisting Evaluation):** Es una métrica de evaluación utilizada principalmente para tareas de resumen automático de texto. ROUGE mide la superposición entre un resumen generado automáticamente y un resumen de referencia. Existen diferentes variantes de ROUGE, como ROUGE-N basado en la coincidencia de n-gramas, ROUGE-L basado en la coincidencia de las subsecuencias más largas y ROUGE-S que mide pares de palabras con saltos intermedios. ROUGE es popular porque correlaciona bien con la evaluación humana en tareas de resumen [8]. Como su nombre lo indica, ROUGE está más orientado al recall, lo que significa que la métrica favorece aquellos resúmenes en los que más palabras del texto de referencia aparecen en la salida del modelo, sin penalizar la presencia de palabras adicionales que no estén en el resumen de referencia. ROUGE es ampliamente utilizado debido a su alta correlación con la evaluación humana en tareas de resumen, lo que lo convierte en una herramienta confiable para medir la calidad de resúmenes generados automáticamente.

4.1.5 MODELO T5

El modelo T5 (Text-To-Text Transfer Transformer) es un modelo de lenguaje desarrollado por Google Research, basado en la arquitectura Transformer y diseñado específicamente para tareas de generación de texto, traducción automática, resumen de textos, entre otras. T5 introduce un enfoque unificado en el que todas las tareas de procesamiento del lenguaje natural (NLP) se reformulan como un problema de conversión de texto a texto (*text-to-text*), facilitando la transferencia de conocimientos entre diferentes tareas.

Variantes del modelo T5

T5 cuenta con varias versiones que difieren en tamaño y capacidad computacional, adaptándose a diferentes requerimientos de entrenamiento e inferencia. Entre las principales variantes se encuentran:

- **T5-Small**: 60 millones de parámetros. Diseñado para tareas que requieren menor capacidad computacional sin comprometer excesivamente el rendimiento.
- **T5-Base**: 220 millones de parámetros. Balance entre eficiencia y capacidad predictiva.
- **T5-Large**: 770 millones de parámetros. Ofrece mejor rendimiento en tareas complejas, aunque con mayor demanda de recursos computacionales.

4.1.6 MODELO GPT-2

El modelo GPT-2 (Generative Pre-trained Transformer 2) es un modelo de lenguaje basado en la arquitectura Transformer, desarrollado por OpenAI, y diseñado para tareas de generación de texto. A diferencia de modelos basados en enfoques supervisados, GPT-2 emplea un aprendizaje auto-regresivo (*causal language modeling*), donde el modelo predice la siguiente palabra en una secuencia dada, utilizando únicamente información de los tokens previos.

Variantes de GPT-2

GPT-2 se presenta en distintas versiones que varían en la cantidad de parámetros, lo que impacta su capacidad de generación y los recursos computacionales requeridos:

- **GPT-2 Small**: 124 millones de parámetros. Versión más ligera, útil para tareas menos exigentes en términos de recursos.
- **GPT-2 Medium**: 355 millones de parámetros. Proporciona un mejor equilibrio entre rendimiento y eficiencia.
- **GPT-2 Large**: 774 millones de parámetros. Capaz de generar texto con mayor coherencia y contexto.
- **GPT-2 XL**: 1.500 millones de parámetros. La versión más grande de GPT-2, con el mejor desempeño en generación de texto, pero con un alto costo computacional.

4.1.7 DEEP LEARNING

El Deep Learning o el aprendizaje profundo es una rama avanzada de la inteligencia artificial que se enfoca en la construcción y entrenamiento de modelos computacionales denominados redes neuronales. A diferencia de los enfoques tradicionales de aprendizaje automático, el Deep Learning se caracteriza por su capacidad para aprender automáticamente patrones y representaciones complejas a través de múltiples capas de procesamiento. Estas redes neuronales, inspiradas en la

organización del cerebro humano, son capaces de descubrir características jerárquicas y abstractas en conjuntos de datos, permitiendo la extracción de representaciones significativas de información. El Deep Learning ha demostrado ser especialmente efectivo en tareas complejas de procesamiento del lenguaje natural, visión por computadora y otras áreas, destacándose por su capacidad para abordar problemas de gran escala y su adaptabilidad a una amplia variedad de dominios. En este contexto, el aprendizaje profundo se presenta como un componente fundamental en la comprensión y aplicación de técnicas avanzadas para el resumen automático de textos mediante el ajuste fino de modelos pre entrenados [10].

- **Diferencia entre Deep Learning y Machine Learning**

Su principal diferencia radica en cómo aprenden los modelos y, así mismo, en los algoritmos que se utilizan para estos. Los modelos tradicionales de aprendizaje automático, también conocidos como aprendizaje superficial, generalmente utilizan algoritmos que no involucran múltiples capas de procesamiento. Estos modelos suelen ser más simples en términos de estructura y su capacidad para captar relaciones complejas en los datos puede ser limitada. También cabe aclarar que estos trabajan de mejor manera con datos estructurados y etiquetados para realizar mejores predicciones, lo que conlleva una definición específica para sus datos utilizados en el modelo y para su organización[11]. Esto no significa que no utilice datos no estructurados, solo que normalmente pasa por algún procesamiento previo para organizarlos en un formato estructurado.

El Deep Learning utiliza arquitecturas de redes neuronales que consisten en múltiples capas. Estas capas permiten la extracción automática de características y representaciones de datos en niveles cada vez más abstractos, lo que facilita la captura de patrones complejos y la resolución de tareas más sofisticadas. Estos algoritmos pueden capturar y procesar datos no estructurados, como texto e imágenes, y automatizan la extracción de características, eliminando parte de la dependencia de humanos [12].

- **Deep Learning y redes neuronales**

Las redes neuronales fueron inspiradas en la estructura del cerebro humano, se caracterizan por tener múltiples capas de nodos interconectados, lo que permite la extracción automática de representaciones complejas de datos. A diferencia de enfoques tradicionales de aprendizaje automático, el aprendizaje profundo es capaz de aprender de manera autónoma características y patrones jerárquicos, lo que lo hace especialmente efectivo en tareas como reconocimiento de imágenes, procesamiento de lenguaje natural y resumen de textos. La capacidad de las redes neuronales profundas para adaptarse y generalizar a partir de datos masivos ha posicionado al deep learning como una herramienta fundamental en la resolución de problemas complejos en diversas disciplinas [13].

En estas redes se encuentran múltiples capas de nodos interconectados, cada una de las cuales realizan operaciones matemáticas para procesar y transformar la información de entrada. Este enfoque jerárquico de procesamiento de datos permite a las redes neuronales profundas aprender patrones y representaciones cada vez más complejas a medida que avanzan en las capas.

4.1.8 PROCESAMIENTO DE LENGUAJE NATURAL

El procesamiento del lenguaje natural es un área de investigación en informática e inteligencia artificial (IA) que se ocupa del procesamiento de lenguajes naturales como el inglés o el mandarín. Este procesamiento generalmente implica traducir el lenguaje natural en datos (números) que una computadora puede usar para aprender sobre el mundo. Y esta comprensión del mundo a veces se utiliza para generar texto en lenguaje natural que refleja esa comprensión [14].

En el contexto de resúmenes de texto, el Procesamiento de Lenguaje Natural (PLN) desempeña un papel crucial al permitir que las máquinas comprendan, interpreten y generen resúmenes coherentes y significativos a partir de textos extensos. El PLN facilita la extracción de información esencial al identificar patrones semánticos, relaciones contextuales y matices lingüísticos presentes en el lenguaje humano. Al aplicar técnicas avanzadas de PLN, como el análisis semántico y la desambiguación del significado, los modelos pueden discernir las ideas clave y la estructura del texto original. Además, el NLP posibilita la adaptación de modelos de lenguaje pre-entrenados mediante técnicas como el fine-tuning, permitiendo una generación de resúmenes más precisa y contextualizada. En este proceso, el PLN actúa como la piedra angular que capacita a las máquinas para comprender y sintetizar información de manera eficiente, proporcionando un enfoque esencial para la automatización efectiva de la generación de resúmenes de texto.

4.1.9 MODELOS DE LENGUAJE GRANDE (LLM)

Un modelo de lenguaje grande (LLM) es un tipo de programa de inteligencia artificial (AI) que puede reconocer y generar texto, entre otras tareas. Los LLM están entrenados con enormes conjuntos de datos, de ahí el nombre "grande", y se basan en el aprendizaje automático: específicamente, un tipo de red neuronal llamada modelo transformador. En términos más simples, un LLM es un programa informático que ha recibido suficientes ejemplos para poder reconocer e interpretar el lenguaje humano u otros tipos de datos complejos. Muchos LLM están capacitados con datos recopilados de Internet: miles o millones de gigabytes de texto. Pero la calidad de las muestras afecta qué tan bien los LLM aprenderán el lenguaje natural, por lo que los programadores de un LLM pueden usar un conjunto de datos más seleccionado [15].

En el contexto del Procesamiento del Lenguaje Natural (NLP), los modelos de lenguaje grandes (LLM) desempeñan un papel crucial al transformar la forma en que las máquinas comprenden y generan texto. Estos modelos, como BERT, GPT, entre otros, son entrenados con vastos conjuntos de datos, capturando patrones complejos y matices semánticos del lenguaje humano [3]. Su intervención se manifiesta en la mejora significativa de tareas NLP, como la generación de resúmenes, traducción automática y análisis de sentimientos. Estos LLM, al aprender representaciones lingüísticas profundas, permiten una comprensión más precisa del contexto y la intención detrás de las palabras, facilitando así la creación de sistemas más inteligentes y contextuales en aplicaciones de procesamiento de lenguaje natural [16].

4.1.10 MODELOS DE LENGUAJE PRE-ENTRENADOS

Un modelo de IA previamente entrenado es un modelo de Deep learning que se entrena en grandes conjuntos de datos para realizar una tarea específica, y se puede usar tal como está o personalizarse para satisfacer los requisitos de una aplicación en múltiples industrias. Los modelos previamente entrenados orientados al lenguaje se utilizan para la traducción, los chatbots y otras aplicaciones de procesamiento de idiomas naturales. Los grandes modelos de idiomas, a menudo basados en la arquitectura del modelo transformer, son una extensión de los modelos previamente entrenados. [17]

Los LLM previamente descritos son ejemplos de modelos pre-entrenados orientados al lenguaje o NLP. Y como se menciona dentro de los ejemplos tenemos BERT, CHAT GPT, y NVIDIA NeMo Megatron.

4.1.11 FINE-TUNING

En el ámbito del aprendizaje automático, el fine-tuning, se presenta como una estrategia esencial para adaptar modelos pre-entrenados a tareas específicas, destacando su importancia en el contexto del Deep Learning. Cuando se utilizan modelos previamente entrenados en conjuntos de datos masivos, el fine-tuning permite refinar estos modelos para que se desempeñen de manera óptima en tareas más especializadas. Este proceso implica la modificación cuidadosa de los pesos y parámetros del modelo para alinearlos con las características y patrones relevantes del nuevo conjunto de datos objetivo. El fine-tuning no solo acelera el entrenamiento de modelos, sino que también potencia su capacidad de generalización y adaptación a contextos específicos. El fine-tuning en este orden de ideas se podría ver como una clave técnica para mejorar la eficiencia y la precisión de modelos de aprendizaje profundo en tareas específicas, en este caso, sería la de resumir textos.

- **Transfer Learning y Fine-Tuning**

El Transfer Learning y Fine-tuning emergen como estrategias fundamentales en el ámbito del aprendizaje profundo. El Transfer Learning elige un enfoque clave que busca capitalizar el conocimiento adquirido en una tarea general para mejorar el rendimiento en tareas más específicas y relacionadas. Este paradigma implica entrenar modelos en conjuntos de datos extensos y diversos para capturar patrones generales, trasladando luego este aprendizaje a nuevas tareas. Dentro de esta práctica, el Fine-tuning se presenta como una técnica esencial. Este proceso se materializa mediante la adaptación de modelos pre-entrenados a través de ajustes precisos en sus pesos y parámetros para alinearlos con las peculiaridades de la tarea objetivo. Así, el Transfer Learning, respaldado por el Fine-tuning, no solo acelera el proceso de entrenamiento, sino que también mejora la capacidad de los modelos para generalizar y desempeñarse de manera óptima en contextos más especializados [18].

4.1.12 PARAMETER EFFICIENT FINE-TUNING (PEFT)

El Fine-Tuning sobre modelos pre-entrenados es una tarea que computacionalmente es costosa al trabajar sobre todos los parámetros de un modelo. El ajuste fino eficiente en parámetros (PEFT) como su nombre lo indica, ofrece una alternativa diferente de hacer fine tuning, donde se enfoca en entrenar un pequeño conjunto de parámetros de manera eficiente en vez de ajustar todos los parámetros, esto quiere decir que permite que un modelo realice una nueva tarea con actualizaciones mínimas. Para resumir con un método PEFT un modelo previamente entrenado se ajusta actualizando solo una pequeña cantidad de parámetros agregados o seleccionados. Lo que hace que usar métodos PEFT el costo computacional reduzca de manera muy notable [1], [19].

4.2 ANTECEDENTES Y TRABAJOS RELACIONADOS

En los proyectos mencionados a continuación, exploraremos el panorama de la generación automática de resúmenes, un campo intrínseco al procesamiento del lenguaje natural y al Deep Learning. Este análisis retrospectivo se centró en investigaciones previas que han sentado las bases para comprender la complejidad y la diversidad de los enfoques utilizados en la síntesis automática de resúmenes. Con un énfasis particular en la técnica de Fine-Tuning, examinaremos cómo la adaptación precisa de modelos pre-entrenados ha surgido como una estrategia prometedora para

mejorar la eficacia y la coherencia en la generación de resúmenes automáticos. Este recorrido por los antecedentes nos proporcionará un marco contextual esencial para comprender el estado actual del campo y para delinear las contribuciones específicas que esta investigación aportará a la evolución de la generación de resúmenes en el contexto del aprendizaje profundo.

Dentro de los antecedentes encontramos los siguientes proyectos:

Scaling Down to Scale Up: A Guide to Parameter-Efficient Fine-Tuning

Este proyecto realiza una comparación exhaustiva de más de 40 métodos PEFT basados en más de 40 artículos publicados entre febrero de 2019 y febrero de 2023, con un enfoque en el bajo costo computacional. Los métodos PEFT se clasifican en tres grandes grupos: suma, selección y reparación (Figura 1). El grupo basado en suma incluye dos subgrupos: métodos similares a adaptadores y avisos suaves. Esta amplia comparación ofrece un panorama general de los métodos existentes, destacando aquellos que optimizan recursos computacionales, lo que facilita la elección de técnicas más eficientes para tareas específicas, como la generación de resúmenes de texto en LLM.Haga clic o pulse aquí para escribir texto.

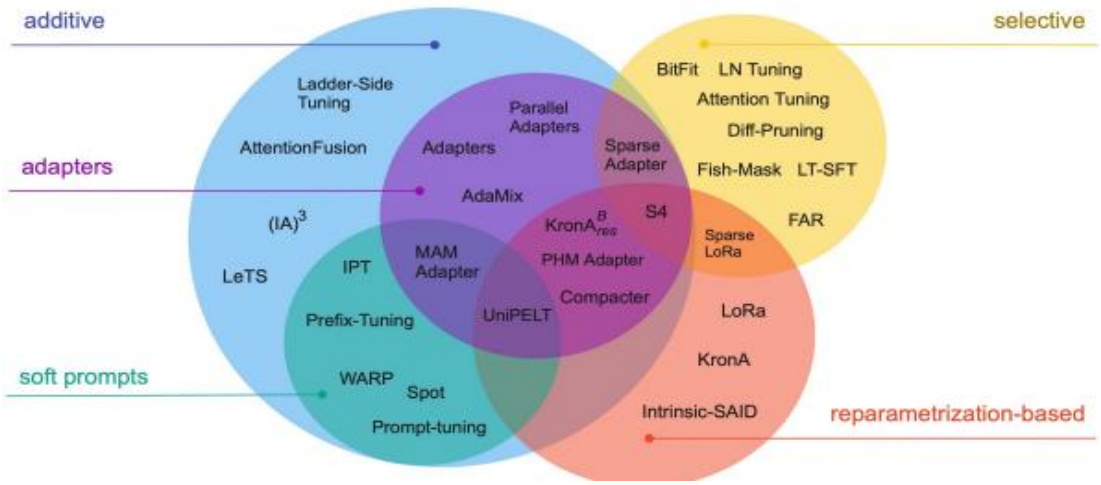


Figura 1. Métodos PEFT mencionados con sus tres clases principales y subgrupos descritos [20].

Entre los aspectos positivos del proyecto, destaca que permite a la comunidad investigadora utilizar y mejorar grandes modelos de lenguaje sin depender de recursos tecnológicos costosos. Además, técnicas como *Adapters* y *LoRa* han demostrado ser prácticas y eficientes, especialmente *LoRa* que, como técnica de reparación de bajo rango, reducen significativamente los parámetros entrenables y el costo computacional.

Sin embargo, persisten los desafíos. Potencializar el rendimiento de la afinación eficiente de parámetros sigue siendo complejo en comparación con la afinación tradicional. Además, aunque los métodos son eficientes, no siempre considere el procesamiento de información a través de las múltiples capas de los modelos, lo que puede limitar su precisión y efectividad.

Towards A Unified View of Parameter-Efficient Transfer Learning

Un trabajo precursor al proyecto “Scaling Down to Scale Up: A Guide to Parameter-Efficient Fine-Tuning” aborda de manera integral el conocimiento sobre los métodos PEFT, estableciendo una base teórica que se conecta estrechamente con la investigación posterior. Este estudio no solo compara la eficiencia de diferentes métodos, incluyendo el Full Fine-Tuning (donde se ajustan todos los parámetros del modelo), sino que también explora cómo el transfer learning hace posible realizar fine-tuning sobre modelos de lenguaje preentrenados (PLM). Este análisis retrospectivo no solo profundiza en el origen del fine-tuning, sino que también destaca cómo los métodos PEFT aprovechan este enfoque para optimizar recursos y mejorar la eficiencia [21].

Además, el trabajo introduce un marco (*framework*) que organiza y unifica múltiples métodos de ajuste de parámetros, proporcionando una herramienta para el desarrollo de técnicas más eficientes que buscan igualar el rendimiento del *Full Fine-Tuning*. Este marco, al servir como guía, representa una contribución clave para investigaciones futuras, fomentando la creación de metodologías innovadoras. Aunque algunos de los métodos descritos requieren tiempos de entrenamiento más prolongados, logran reducir considerablemente los costos asociados al uso de memoria y servidores durante el despliegue de los modelos.

Sin embargo, persisten desafíos que limitan su implementación. Por ejemplo, aún no se ha explorado en profundidad cómo estos métodos podrían influir en cuestiones éticas, como la generación de textos sesgados o la filtración de información privada. Asimismo, el tiempo adicional necesario para entrenar los modelos puede representar una barrera significativa, especialmente en escenarios con recursos limitados. Finalmente, la necesidad de más épocas de entrenamiento para lograr una convergencia adecuada podría ser menos ventajosa en comparación con la afinación completa.

LLM-Adapters: An Adapter Family for Parameter-Efficient Fine-Tuning of Large Language Models

Siguiendo la clasificación presentada en “Scaling Down to Scale Up: A Guide to Parameter-Efficient Fine-Tuning”, se identificaron proyectos enfocados en aplicar métodos PEFT específicamente en grandes modelos de lenguaje (LLM). Entre ellos destaca “LLM-Adapters: An Adapter Family for Parameter-Efficient Fine-Tuning of Large Language Models”, un estudio centrado en el subgrupo de métodos Aditivos conocido como *Adapters*. Estos métodos, aunque se basan principalmente en un enfoque aditivo, pueden incorporar elementos de las metodologías de reparametrización o selección, ampliando así su versatilidad (Figura 1).

En este proyecto, se desarrolló un *framework* diseñado para trabajar con LLM de acceso abierto como LLaMA, BLOOM y GPT-J. A través de experimentación empírica y comparación de resultados, los autores propusieron estrategias eficientes para aplicar los *Adapters* en distintas tareas. Específicamente, el estudio se enfocó en el razonamiento aritmético y el razonamiento de sentido común, utilizando LLM adaptados al objetivo de cada tarea.

Este trabajo pone de relieve la capacidad de los *Adapters* para ejecutar *fine-tuning* eficiente sobre modelos preentrenados, demostrando cómo estas técnicas permiten adaptar LLM a tareas específicas con resultados sobresalientes. Los hallazgos indican que el uso de métodos PEFT basados en *Adapters* en modelos más pequeños con menos parámetros puede igualar, e incluso superar, el rendimiento de modelos de mayor tamaño. Esto refuerza la efectividad y eficiencia de este enfoque, haciendo que sea una opción viable para escenarios con limitaciones de recursos computacionales[22].

Sin embargo, el proyecto también enfrentó limitaciones significativas. No se evaluaron modelos de mayor escala, como LLaMA-33B y LLaMA-65B, debido a restricciones computacionales. Esto deja abierta la pregunta de si los resultados observados pueden replicarse en modelos más grandes, lo que representa un área importante para futuras investigaciones.

PEFT-MedAware: Large Language Model for Medical Awareness

Las metodologías PEFT demuestran su versatilidad no solo en la generación de resúmenes de texto, sino también en aplicaciones prácticas como el desarrollo de chatbots especializados. Un ejemplo de esto es el proyecto "PEFT-MedAware", que plantea un modelo diseñado para responder preguntas médicas con precisión en dominios específicos, aprovechando recursos computacionales limitados. Este enfoque es especialmente útil en entornos con restricciones tecnológicas, donde el *Full Fine-Tuning* resulta inviable debido a su elevado costo computacional. En este proyecto, se desarrolló un modelo basado en PEFT-MedAware (Figura 2), optimizando el modelo Falcon-1b, un modelo de lenguaje grande, mediante el uso de datos especializados provenientes de MedQuAD. Este conjunto de datos incluye 16,407 pares de control de calidad médica. Lo innovador del enfoque radica en que solo se utiliza el 0,44% de los parámetros entrenables, lo cual permite mejorar la eficiencia computacional sin sacrificar la precisión del modelo. Los resultados demuestran que PEFT puede superar a los modelos estándar en tareas específicas del ámbito médico, logrando un balance óptimo entre especialización y uso eficiente de recursos [23].

El modelo Falcon-1b, que cuenta con más de 700 millones de parámetros, fue ajustado utilizando únicamente 3 millones de datos gracias a las metodologías PEFT implementadas. Este nivel de eficiencia resalta las ventajas de PEFT en entornos donde las limitaciones de infraestructura son un desafío.

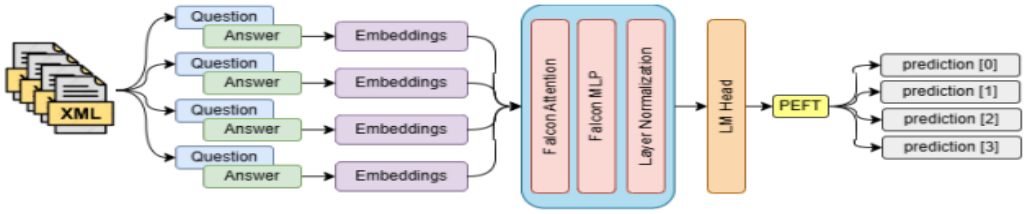


Figura 2. Pipeline utilizado PEFT-MedAware [20].

Además, este enfoque se conecta con estudios como *Parameter-Efficient Fine-Tuning of Large-Scale Pre-Trained Language Models*, que abordan el desafío de adaptar y almacenar modelos de lenguaje preentrenados (PLMs) a gran escala. Ambos trabajos refuerzan la idea de que las metodologías PEFT no solo reducen el costo computacional, sino que también amplían el alcance de los modelos a aplicaciones prácticas que requieren soluciones especializadas y sostenibles.

Parameter-efficient fine-tuning of large-scale pre-trained language models

En este trabajo, los autores abordan el desafío asociado con el uso de modelos de lenguaje preentrenados (PLMs), destacando que, aunque modelos más grandes tienden a ofrecer un mejor rendimiento, la adaptación y almacenamiento de todos los parámetros de modelos a gran escala resultan costosos e impracticables. Como respuesta, La investigación se centra en la adaptación eficiente de PLMs, enfocándose en optimizar una pequeña porción de los parámetros mientras se mantienen el resto fijos, lo cual reduce significativamente los costos de cómputo y almacenamiento.

Introducen el término "delta-tuning" para describir este enfoque, donde "delta" se refiere a la porción de parámetros que cambian durante el entrenamiento [24].

Los autores formalizan el problema y proponen un criterio de categorización unificado para los métodos de delta-tuning, explorando sus correlaciones y diferencias, y discuten los principios teóricos subyacentes a la eficacia del delta-tuning desde las perspectivas de la optimización y el control óptimo. A partir de evidencia empírica, presentan dos marcos teóricos para discutir el delta-tuning, arrojando luz sobre las referencias teóricas de un diseño novedoso para estos métodos, con la esperanza de inspirar una comprensión más profunda de la adaptación de modelos para PLMs.

Empíricamente, realizan extensos experimentos en más de 100 tareas de procesamiento del lenguaje natural, evaluando propiedades combinatorias, la influencia de la escala y la transferibilidad del delta-tuning. En términos de rendimiento, observan que el delta-tuning puede estar ligeramente por detrás o ser comparable al fine-tuning en diversas tareas, con una brecha que se reduce a medida que el modelo escala. En cuanto a la eficiencia, destacan el potencial del delta-tuning para reducir significativamente el espacio de almacenamiento y el uso de memoria, así como para acelerar la retro propagación. En resumen, concluyen que el delta-tuning muestra un considerable potencial para estimular PLMs a gran escala y abogan por un mayor estudio teórico y práctico de este paradigma.

LoRA: Low-Rank Adaptation of Large Language Models

En el procesamiento del lenguaje natural, un enfoque fundamental consiste en preentrenar modelos a gran escala utilizando datos de dominio general y luego adaptarlos a tareas o dominios específicos. Sin embargo, a medida que los modelos de lenguaje se vuelven más grandes, como el **GPT-3 175B**, el ajuste fino completo se vuelve menos práctico debido a los altos costos computacionales y de almacenamiento asociados con el reentrenamiento de todos los parámetros del modelo.

Para abordar este desafío, se propone **LoRA** (**Low-Rank Adaptation**), una técnica que optimiza el ajuste fino al congelar los pesos del modelo preentrenado y añadir matrices de bajo rango entrenables en cada capa de la arquitectura Transformer. Esta metodología reduce significativamente el número de parámetros que deben ser entrenados, disminuyendo en 10,000 veces la cantidad de parámetros ajustables y reduciendo el requisito de memoria de la GPU en 3 veces en comparación con el ajuste fino completo de GPT-3 175B utilizando Adam [25].

LoRA ha demostrado un rendimiento comparable o incluso superior al ajuste fino tradicional en modelos como **RoBERTa**, **DeBERTa**, **GPT-2** y **GPT-3**, a pesar de manejar menos parámetros entrenables. Además, ofrece ventajas adicionales como un mayor rendimiento de entrenamiento y la eliminación de la latencia de inferencia adicional que a menudo se asocia con técnicas de adaptación como los adaptadores [25].

El estudio incluye una investigación empírica sobre la deficiencia de rango en la adaptación de modelos de lenguaje, proporcionando evidencia de la eficacia de LoRA en la mejora del ajuste fino de grandes modelos. Además, se ha publicado un paquete que facilita la integración de LoRA con modelos en PyTorch, junto con implementaciones y puntos de control para RoBERTa, DeBERTa y GPT-2 [26].

Luego observaremos un proyecto similar al nuestro en el que se desarrolló también con los modelos GPT-2 y T5.

Prefix-Tuning: Optimizing Continuous Prompts for Generation

El ajuste fino es el método estándar para adaptar grandes modelos de lenguaje preentrenados a tareas específicas, pero implica modificar todos los parámetros del modelo, lo que requiere almacenar una copia completa para cada tarea individual. Este enfoque puede ser costoso en términos de almacenamiento y recursos computacionales.

Para superar estas limitaciones, el artículo propone **prefix-tuning**, una alternativa más eficiente al ajuste fino que mantiene congelados los parámetros del modelo de lenguaje preentrenado y en su lugar optimiza un pequeño vector continuo específico para la tarea, denominado "prefijo". Esta técnica se basa en el concepto de *prompting*, permitiendo que los tokens generados presten atención a este prefijo como si fuera un conjunto de "tokens virtuales", ajustando así el comportamiento del modelo para la tarea particular [27].

El método de ajuste de prefijo se aplicó a GPT-2 para la generación de texto a partir de tablas y a BART para la tarea de resumen. Los resultados muestran que, al ajustar solo el 0.1% de los parámetros, prefix-tuning logra un rendimiento comparable al del ajuste fino completo en escenarios con datos abundantes. Además, supera al ajuste fino en configuraciones con pocos datos y muestra una mejor capacidad de extrapolación a ejemplos con temas no vistos durante el entrenamiento [27].

5 DESARROLLO E IMPLEMENTACION

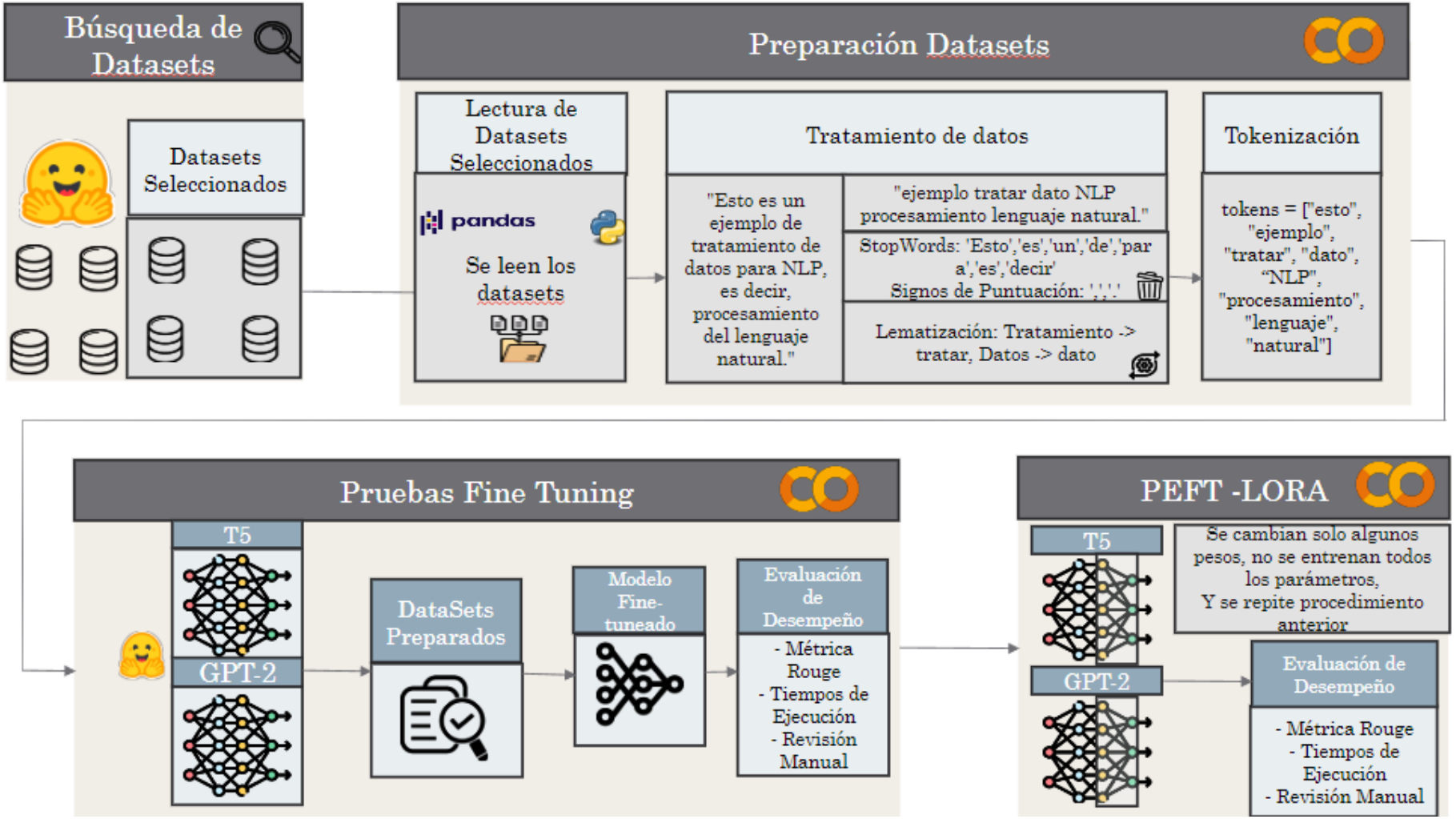


Figura 3. Metodología de desarrollo (Elaboración propia).

Para dar mayor claridad sobre el desarrollo e implementación del modelo, se presenta un diagrama de flujo general en la Figura 3, que ilustra cada etapa del proceso. Este capítulo detalla la implementación y los resultados del proyecto, centrado en evaluar metodologías de ajuste eficiente de parámetros (PEFT) en modelos de lenguaje grande (LLM) para generar resúmenes de texto.

El proceso se inicia con la recolección de datos relevantes, seguido por la aplicación de técnicas PEFT para ajustar los modelos específicamente a la tarea de resumen. Posteriormente, se comparan los modelos mediante métricas de precisión, coherencia y relevancia para identificar aquellos que demuestran mayor eficacia. Finalmente, se analiza la eficiencia computacional de cada método, destacando las metodologías PEFT que optimizan el uso de recursos sin comprometer la calidad de los resúmenes generados.

5.1 RECOLECCIÓN DE DATOS

La ejecución de esta actividad implicó, en primer lugar, seleccionar los modelos de procesamiento de lenguaje natural y los conjuntos de datos (datasets) usados durante el entrenamiento de los modelos. Posteriormente, se llevó a cabo el preprocesamiento y preparación de los datos, que consistió en limpiar, organizar y transformar los datasets en un formato adecuado para alimentar los modelos seleccionados durante la etapa de entrenamiento. Esta fase de preparación de datos es crucial, ya que determinó la calidad de los resultados que se obtuvieron al finalizar el entrenamiento de los modelos.

5.1.1 SELECCIÓN DE MODELOS LLM MÁS CONVENIENTES PARA TEXT SUMMARIZING

Paralelo a la búsqueda de los conjuntos de datos se eligieron modelos apropiados no solo para la generación de resúmenes de texto, si no también que se adaptaran a nuestras capacidades computacionales y a las necesidades para llevar a cabo el Fine-Tuning en los modelos. Por lo cual seleccionamos dos modelos, teniendo en cuenta el tiempo de desarrollo del proyecto y lo requerido en su implementación. Por lo tanto, los modelos seleccionados son el T5 y el GPT-2.

T5 y GPT-2 son LLM, T5 fue desarrollado por Google Research, mientras que GPT-2 fue desarrollado por OPENAI. Ambos son modelos de lenguaje basados en la arquitectura Transformers, y provienen de organizaciones diferentes y tienen enfoques y características ligeramente diferentes. Mientras que T5 tiene un enfoque *"text-to-text"* lo que significa que recibe como input un texto y genera de salida un output de texto. En contraparte GPT-2 es "*texto a continuación*", es decir que de entrada un texto, genera una continuación basada en el texto de la entrada. En cuanto a ventajas y desventajas "text to text" es más versátil en las tareas y es un poco más pesado computacionalmente, mientras que GPT-2 suele tener mejor rendimiento en tareas conversacionales, y como asistente de escritura.

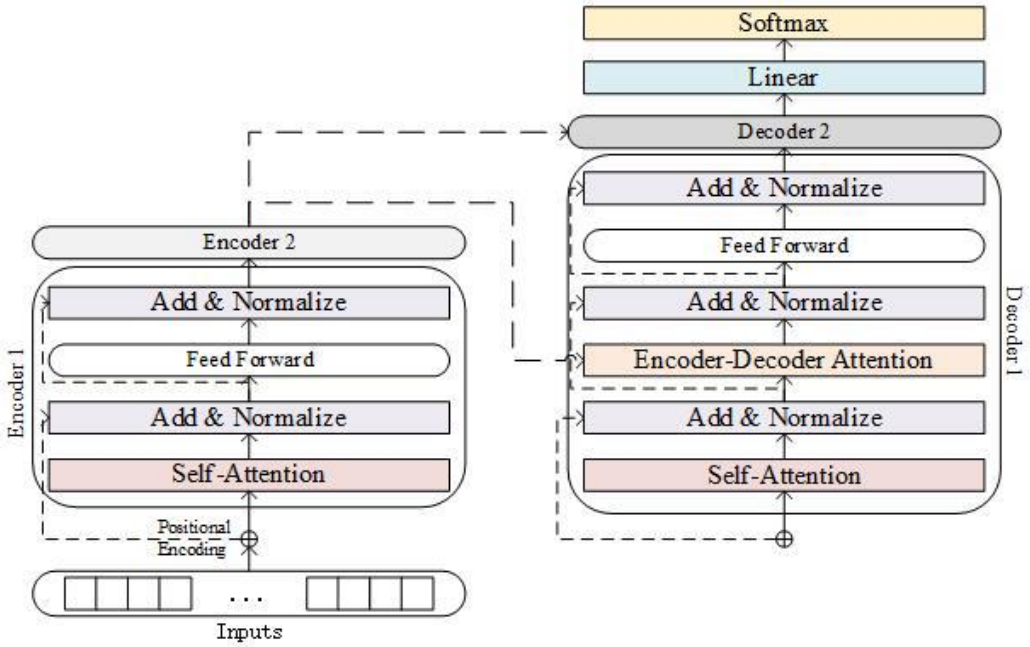


Figura 4. Arquitectura modelo T5 [28].

El modelo T5 utiliza una arquitectura seq2seq (Figura 4), donde todas las tareas de NLP se formulan como tareas de "texto a texto". Esto significa que tanto el input como el output son texto, y el modelo se entrena para mapear un texto de entrada a un texto de salida. Y como podemos ver usa solo un codificador-decodificador de T5 captura toda la información relevante del input utilizando el codificador y luego genera una respuesta coherente utilizando el decodificador. Además, este cuenta con una ventana de contexto de 512 tokens por procesamiento.

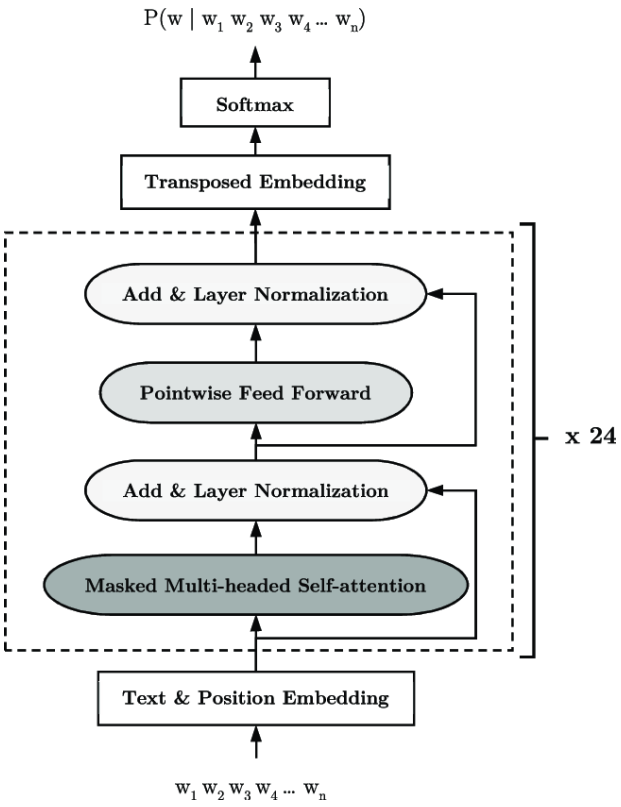


Figura 5. Arquitectura GPT-2 [29].

GPT-2 utiliza una arquitectura de autoregresión (Figura 5), donde el modelo genera una secuencia de salida token por token, condicionada por los tokens previamente generados. No hay una distinción explícita entre un codificador y un decodificador; el modelo simplemente continúa generando texto basado en el contexto proporcionado por los tokens de entrada con ventana de contexto es de 1024 tokens.

Los modelos seleccionados considerando las investigaciones y lecturas realizadas sobre la generación de resúmenes, donde se evidenció que estos modelos se utilizan en proyectos similares a este, por tener más referencias y más seguridad se eligieron los ya mencionados modelos.

5.1.2 ANÁLISIS EXPLORATORIO DE LOS DATASET SELECCIONADOS

Selección de datasets y de los datos:

El preprocesamiento ejecutado en los dataset se realizó para determinar la cantidad de tokens, lo cual nos permitió establecer vectores numéricos sobre las palabras requeridas para el entrenamiento de los modelos. Este proceso de tokenización es crucial, ya que la cantidad de tokens no puede superar la ventana de contexto de los modelos (Tabla 1), por lo cual, a partir de este criterio se pudieron descartar algunos datasets que no satisfacían esta condición y de esta forma se garantizó que los datasets elegidos se pudieran procesar adecuadamente en los modelos seleccionados.

Tabla 1. Información modelos seleccionados (Elaboración propia)

Modelo	Ventana de contexto (Tokens)
GPT-2	1024
T5	512

Durante la búsqueda de información, se consultaron sitios web como HuggingFace y Kaggle, donde se encontraron diversos conjuntos de datos de acceso público, que incluían conversaciones cortas, noticias, informes, entre otros. Teniendo esto en cuenta, se eligieron diferentes datasets para llevar a cabo el desarrollo del proyecto, los cuales se centran en noticias y conversaciones cortas, ya que estas presentan un contexto claro y son fáciles de comprender para cualquier lector, lo cual facilita el proceso de validación al evaluar la coherencia de los resúmenes generados por los modelos.

Los 3 dataset se encuentran en el repositorio de HuggingFace los cuales se titulan neil-code/dialogsum-test [30], Samsung/samsum [31] y abisee/cnn_dailymail [32], junto con la documentación relacionada, que sirvió como insumo para comprender las variables y la información que contiene [33].

A continuación, hacemos un análisis exploratorio inicial de los datasets seleccionados:

Dataset dialogsum

Fuente: Huggingface

Campos:

- **dialogue**: texto del diálogo.
- **summary**: resumen escrito por una persona del diálogo.
- **topic**: tema o una línea que describe el diálogo, escrito por una persona.
- **id**: identificador único del archivo de un ejemplo.

Tamaño dataset:

- **train**: 12460
- **val**: 500
- **test**: 1500

Idioma: Ingles

Para tener una idea del tipo de contenido que contienen los datasets, presentamos un ejemplo de los datos que tenemos (Tabla 2).

Tabla 2 Ejemplo de información contenida en el dataset Dialogsum: [30].

dialogue	summary	topic	id
#Person1#: Hi, Mr. Smith. I'm Doctor Hawkins. Why are you here today?\n#Person2#: I found it would be a good idea to get a check-up.\n#Person1#: Yes, well, you haven't had one for 5 years. You should have one every year.\n#Person2#: I know. I figure as long as there is nothing wrong, why go see the doctor?\n#Person1#: Well, the best way to avoid serious illnesses is to find out about them early. So try to come at least once a year for your own good.\n#Person2#: Ok.\n#Person1#: Let me see here. Your eyes and ears look	Mr. Smith's getting a check-up, and Doctor Hawkins advises him to have one every year. Hawkins'll give some information about their classes and medications to help Mr. Smith quit smoking.	'get a check-up'	'train_0'

Ahora lo que realmente nos interesa para es saber si los datos pueden ser utilizados por los modelos seleccionados para probar, sabemos que tenemos una ventana de contexto de 512 para T5 y 1024 para GPT-2, a continuación, en la Figura 6. y la Figura 7. se ve la distribución de tokens para el dialogsum .

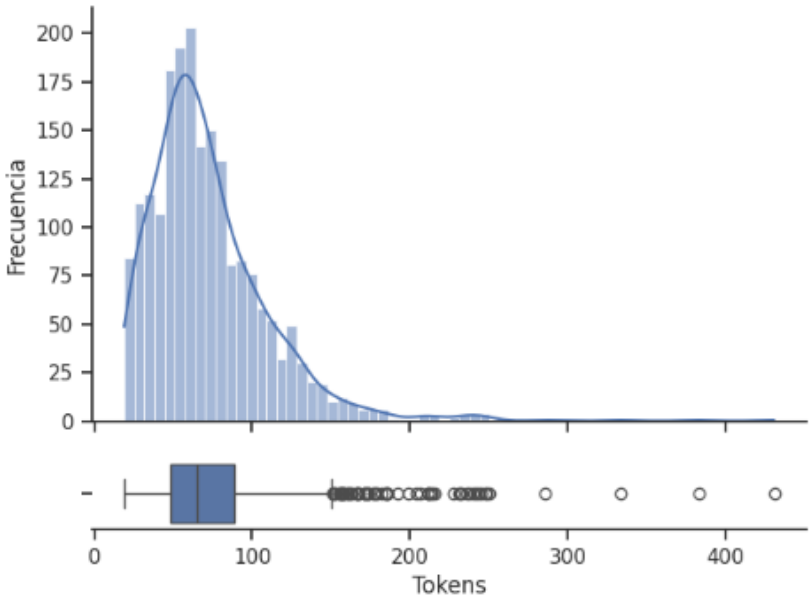


Figura 6. Histograma con la cantidad de tokens en los textos de entrada (text) – Dataset dialogsum (Elaboración propia)

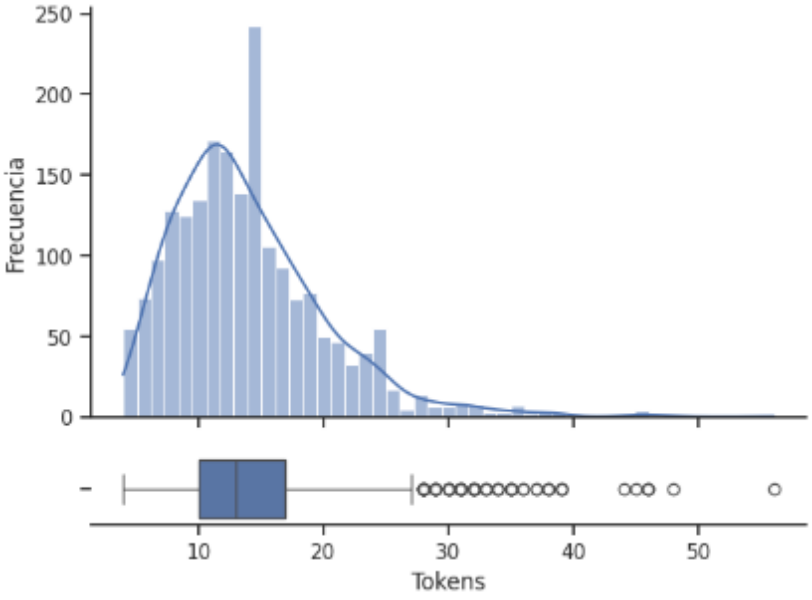


Figura 7. Histograma con la cantidad de tokens de entrada en la columna de resumen (summary) – Dataset dialogsum (Elaboración propia)

Tenemos que no hay ningún tipo de problema en usar este dataset, ya que ambas distribuciones no superan los 500 tokens. Y puede ser utilizado tanto por T5 como GPT-2

Dataset Samsum

Fuente: Huggingface

Campos:

- **dialogue**: texto del diálogo.
- **summary**: resumen escrito por una persona del diálogo.
- **id**: identificador único de un ejemplo.

Tamaño dataset:

- **train**: 14732

- **val**: 818
- **test**: 819

Idioma: Ingles

Ejemplo:

Tabla 3. Ejemplo de información contenida en el dataset Samsum. [31].

id	dialogue	summary
13818513	"Amanda: I baked cookies. Do you want some?\r\nJerry: Sure!\r\nAmanda: I'll bring you tomorrow :-)"	“Amanda baked cookies and will bring Jerry some tomorrow.”

Con el dialogo anterior (Tabla 3) podemos observar que no hay mucho texto en el diálogo, y esperando que los demas diálogos sean igual de cortos, no debería haber problemas por la cantidad de tokens que generen para los modelos que nos interesan.

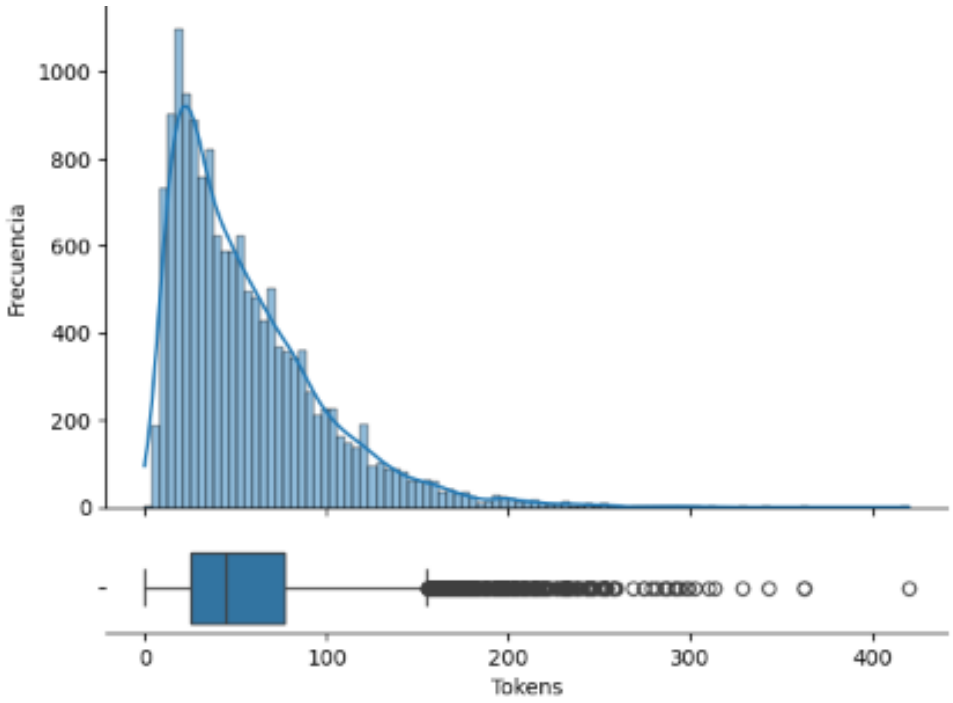


Figura 8. Histograma con la cantidad de tokens en los textos de entrada (text) - Dataset SamSum. Fuente: Elaboración Propia.

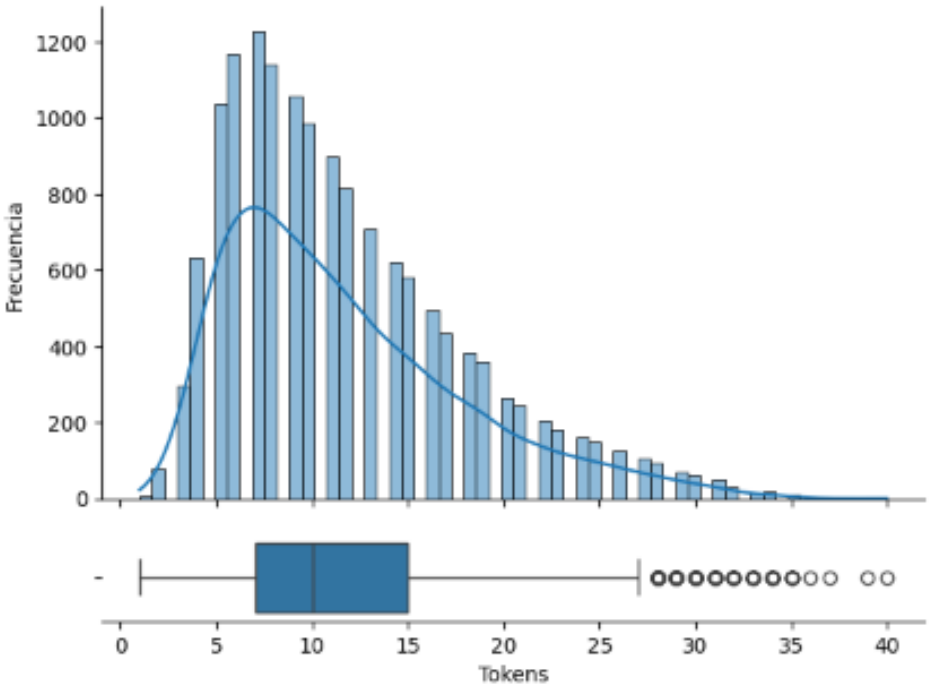


Figura 9. Histograma con la cantidad de tokens de entrada en la columna de resumen (summary) - Dataset SamSum (Elaboración propia)

En la Figura 8 y Figura 9, se observa que en efecto no hay problema con la cantidad de tokens y que el dataset es adecuado para el entrenamiento de estos modelos. Observamos que no se sobrepasan los 500 tokens

Dataset cnn_dailymail

Fuente: Hugging face

Campos:

- **id**: un string que contiene el hash SHA1 en formato hexadecimal de la URL de donde se recuperó la historia.
- **article**: una cadena que contiene el cuerpo del artículo de noticias.
- **highlights**: una cadena que contiene los puntos destacados del artículo tal como fueron escritos por el autor del artículo.

Tamaño dataset:

- **Train**: 287.113
- **Validation**:13.368
- **Test**: 1.,490

Idioma: Ingles

Ejemplo:

Tabla 4. Ejemplo de información contenida en el dataset cnn_dailymail [32].

id	article	highlights
42c027e4ff9730fbb3de84c1af0d2c506e41c3e4	'LONDON, England (Reuters) -- Harry Potter star Daniel Radcliffe gains access to a reported £20 million ($41.1 million) fortune as he turns 18 on Monday, but he insists the money won\'t cast a spell on him. Daniel Radcliffe as Harry Potter in "Harry Potter and the Order of the Phoenix" To the disappointment of gossip columnists around the world, the young actor says he has no plans to fritter his cash away on latest outing as the boy wizard in "Harry Potter and the Order of the Phoenix" is breaking record.....	"Harry Potter star Daniel Radcliffe gets £20M fortune as he turns 18 Monday .\nYoung actor says he has no plans to fritter his cash away .\nRadcliffe's earnings from first five Potter films have been held in trust fund ."

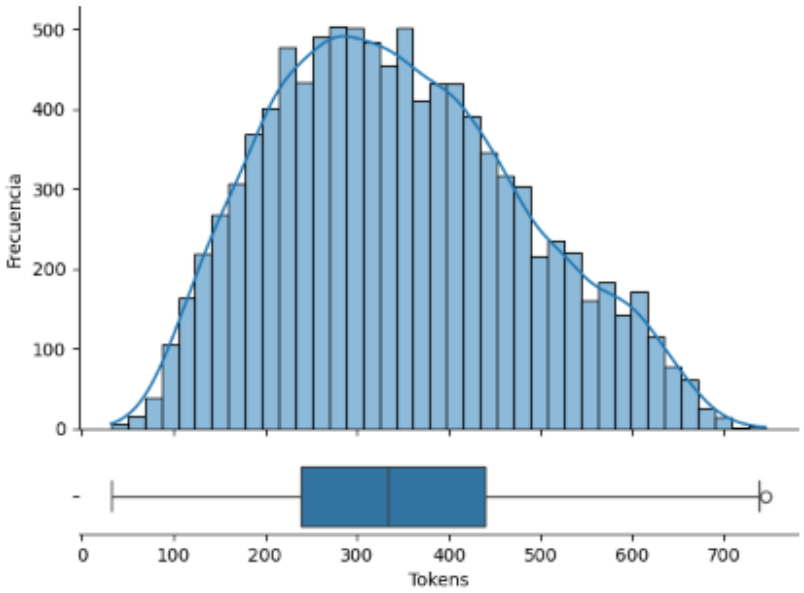


Figura 10. Histograma con la cantidad de tokens en los textos de entrada (text) - Dataset Cnn_dailymail (Elaboración propia)

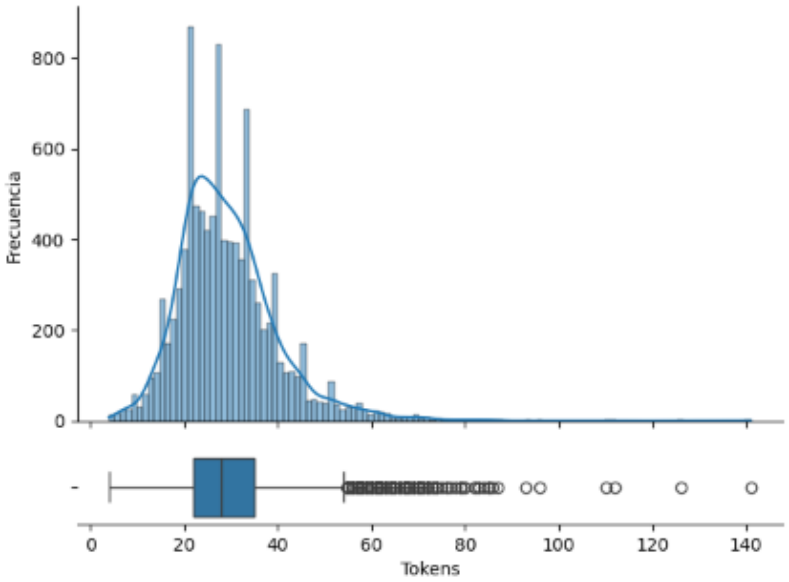


Figura 11. Histograma con la cantidad de tokens de entrada en la columna de resumen (summary) - Dataset Cnn_dailymai (Elaboración propia)

Acorde la información presentada en la Figura 10 y Figura 11, en este caso se realizó un recorte del dataset para solo aquellos que no sobrepasen la ventana de contexto para el T5, pero gracias al boxplot se observa que realmente la mayoría de los datos no sobrepasan el umbral de los 512 tokens por lo que no debería haber problema al momento de generar el modelo, ya que el percentil 75 parece estar debajo de los 450 tokens.

En este caso si será necesario hacer un recorte en los datos de entrenamiento para obtener mejores resultados durante el procesamiento de los datos

5.1.3 PREPARACIÓN DE LOS DATOS

Ahora, explorados los datos, se prepara en los datos, para ello debemos considerar que, en el contexto de trabajar con lenguaje natural, debemos llevar estas palabras de letras a números, para que el modelo pueda entender los datos de entrada.

A continuación, se describen los pasos realizados en el preprocesamiento fueron los siguientes:

1. **Lematización**: Proceso lingüístico que consiste en reducir las palabras a su forma base o lema. Esto implica transformar palabras flexionadas o conjugadas a su forma más simple y canónica., ejemplo:
   - Reírse → reír
   - Comiendo → comer
   - Corriendo → correr
   - Hablando → hablar

2. **Eliminar Stopwords:** Consiste en eliminar las palabras como conectores o palabras que se repiten mucho y que por sí solas no aportan al contexto, por ejemplo: El, ella, con, y, nos, etc.

3. **Transformación de mayúsculas a minúsculas:** Esto se hace para estandarizar el texto y así manejarlo todo de la misma forma, este proceso es importante porque ayuda a evitar discrepancias en la identificación de palabras o en la comparación de cadenas de texto, ya que

el texto normalizado tendrá una representación única independientemente de la capitalización original.

4. **Eliminación de Jergas:** Se eliminan las palabras que se usan en forma de jerga o como expresión en momentos específicos del texto.

5. **Ruido:** Refiere a palabras erradas o palabras que se utilizan unas pocas veces y no son de gran importancia en el análisis y el contexto del texto.

6. **Tokenización:** Este paso se realizó después del preprocesamiento del texto, dado que al realizar el preprocesamiento se eliminan muchas palabras lo que reduce el número de tokens, de esta forma se hace más fácil para el procesamiento del modelo.

Posteriormente, se realiza la tokenización, uno de los pasos más importantes en el proceso. La tokenización juega un papel fundamental en la generación de resúmenes de texto al proporcionar la base para identificar las unidades de información clave y determinar su relevancia en el contexto del texto completo. Para llevar a cabo esta tokenización correctamente, se unieron las dos columnas del dataset en las que se llevaba a cabo la conversación. Después, se preprocesó este texto. Utilizando la librería Spacy, se realizó la tokenización y se contó el número de tokens resultantes por cada fila, agregando esta información en una nueva columna denominada 'merge_tokens'. Esto permitió conocer la cantidad de tokens disponibles y verificar si coincidían con la ventana de contexto de los modelos elegidos. Además, se añadió una columna que mostraba los tokens unidos y otras dos columnas que contenían los tokens de las columnas 'text' y 'Summary'[1]. Todo esto se realizó para tener un control preciso sobre la cantidad de tokens en cada sentencia y asegurarse de no sobrepasar los límites permitidos por el modelo.

A partir del procesamiento descrito anteriormente, se presentan a continuación los resultados obtenidos en cada uno de los dataset.

Dataset dialogsum

Teniendo en cuenta la cantidad de tokens obtenidos, se generó un histograma y boxplot para las variables “text”, “Summary” y “merge_tokens”[2]. De este análisis se destacó que la media de tokens para la variable “merge_tokens” es 86.5, con un mínimo de 24 y máximo de 469. Por lo tanto, se puede concluir que este dataset es apropiado para utilizarse como insumo en el entrenamiento de los modelos propuestos, ya que la cantidad de tokens por registro no supera la ventana de contexto de cada modelo.

A continuación, se muestra la distribución de los tokens pertenecientes a cada uno de los textos de entrada y resumen combinados (Figura 12) obtenidos para el dataset neil-code/dialogsum-test.

[1] Durante el procesamiento de los dataset, se renombraron las variables de entrada como “text” y “Summary”, las cuales indican el texto original y su respectivo resumen.
[2] Combinación de la columna text y summary.

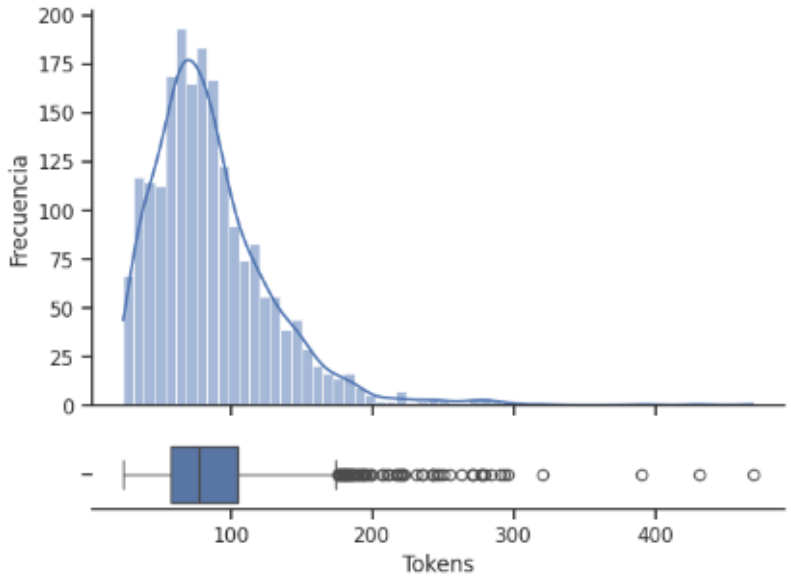


Figura 12. Histograma con la cantidad de tokens para la columna combinada (merge_tokens) – Dataset dialogsum (Elaboración propia)

Es importante aclarar que se realizó exactamente el mismo preprocesamiento y tokenizacion con los otros dos datasets, esto para manejar la misma cantidad de datos y de tokens para facilidad al momento de entrenar el modelo. Es por esta razón que encontraremos las mismas columnas agregadas en el dataset anterior en los próximos dos datasets. La diferencia entre estos es un poco más notoria mediante los histogramas y los diagramas de cajas, ya que es aquí donde podemos ver sus medias, valores atípicos y demás datos relevantes de cada dataset.

Dataset Samsum

Ahora bien, considerando el proceso aplicado al anterior dataset, se elaboró un histograma que muestra la cantidad de tokens en cada registro del dataset Samsum. De este análisis, se observó que la media de tokens es de 68, con un mínimo de 3 y un máximo de 434. En consecuencia, se puede determinar que el dataset Samsum es adecuado para su uso en el entrenamiento de los modelos propuestos, ya que la cantidad de tokens por registro se encuentra dentro de los límites de la ventana de contexto de cada modelo.

A continuación, se muestra la distribución de los tokens y diagrama de cajas pertenecientes a cada uno de los textos de entrada, resumen y combinación (Figura 13) para el dataset de Samsum.

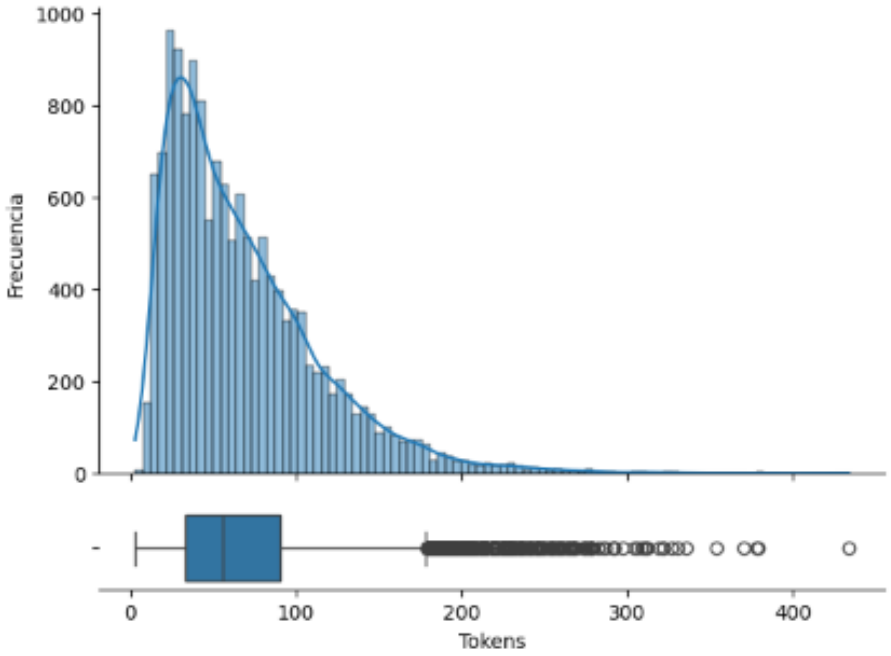


Figura 13. Histograma con la cantidad de tokens para la columna combinada (merge_tokens) - Dataset SamSum (Elaboración propia)

Al igual que con el dataset anterior, tampoco se alcanza con este el máximo valor de la ventana de tokens permitido por los modelos que se van a utilizar. Lo que indica que es posible trabajar con este Dataset y que si así lo quisiéramos podríamos elegir oraciones más largas o un dataset con conversaciones un poco más extensas y se realizaría el procedimiento de manera óptima igualmente. Por motivos de capacidad de cómputo, estos datos son apropiados para ejecutarse en las máquinas de poseemos.

Dataset cnn_dailymail

Ahora bien, se realizó el mismo proceso al dataset faltante abisee/cnn_dailymail para terminar con el proceso de exploración y preparación de los dataset y validación de sus datos, alcances y limitaciones. En este dataset al igual que en los anteriores dos, se puso limitación a 1000 filas para tener seguridad de que será soportado en el proceso de entrenamiento de los modelos en las maquinas en las que estamos desarrollando el proyecto.

A continuación, se muestra la distribución de los tokens pertenecientes a cada uno de los textos de entrada y resumen combinados (Figura 14) obtenidos para el dataset faltante abisee/cnn_dailymail.

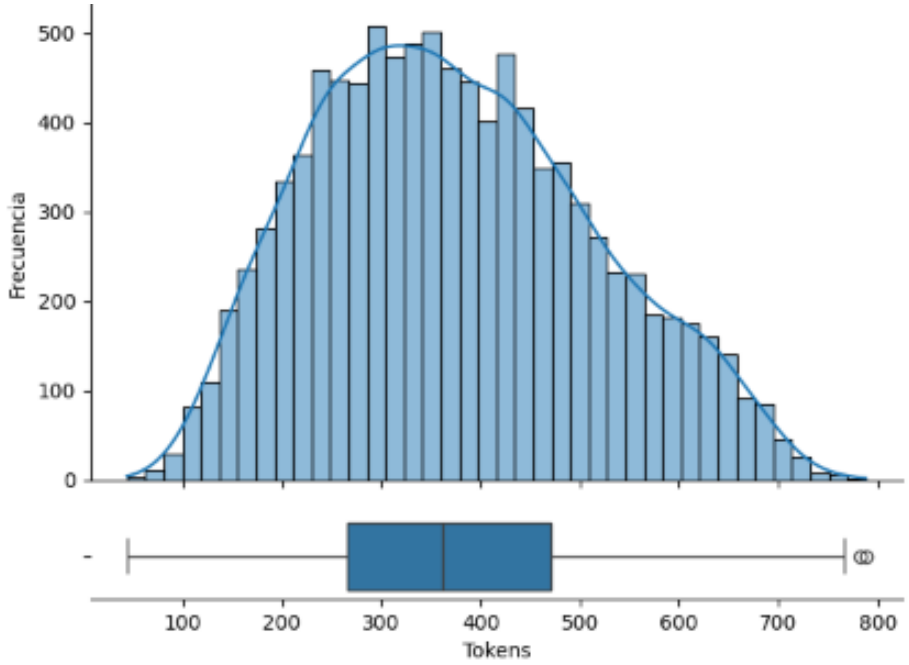


Figura 14. Histograma con la cantidad de tokens para la columna combinada (merge_tokens) – Dataset Cnn_dailymail (Elaboración propia)

Es importante aclarar que en este dataset la cantidad de tokens supera la ventana de contexto que limita el modelo T5 de 512, por lo tanto, se realizó un filtro sobre el dataset para registros que no superaran los 500 tokens, del resultado obtenido sobre este dataset, es posible concluir que se puede tomar un subconjunto del dataset cercano a 15.000 registros.

En este punto del proyecto es importante identificar que metodologías PEFT se podrían adaptar a las necesidades del proyecto, ya que buscamos optimizar el uso de parámetros durante el proceso de ajuste fino de modelos de aprendizaje profundo, relevante en contextos donde los recursos computacionales son limitados o donde se requiere una implementación ágil y escalable. Este enfoque implica una evaluación detallada de diversas técnicas que facilitan el ajuste eficiente, incluyendo métodos como el ajuste de capas parciales, técnicas de regularización adaptativa y la utilización de subespacios parametrizados.

A través de una revisión exhaustiva de la literatura y un análisis comparativo de diversas metodologías PEFT, buscamos identificar aquellas que proporcionen un marco robusto para la optimización de nuestro modelo. Este proceso nos permitirá no solo mejorar el rendimiento del modelo, sino también reducir el tiempo y los recursos necesarios para su entrenamiento y ajuste.

5.2 IMPLEMENTACIÓN DE METODOLOGÍAS PEFT

En la realización de este proceso, es crucial destacar la importancia del fine-tuning en esta etapa. No es posible implementar las metodologías PEFT (Parameter-Efficient Fine-Tuning) sin antes haber llevado a cabo el proceso adecuado de fine-tuning. No obstante, a continuación, se detalla el proceso implementado para llevar a cabo el entrenamiento del modelo y Fine-Tuning, seguido de la evaluación del modelo y obtención de métricas:

Entrenamiento del modelo y Fine Tuning

1. **Definición de los modelos a utilizar**
   En este paso, se seleccionan el modelo pre-entrenado que se afinara para la tarea específica de generación de resúmenes. La elección del modelo es crucial porque determina la arquitectura subyacente, la capacidad del modelo y las características con las que comenzará el entrenamiento personalizado.
2. **Cargue y preparación de dataset**
   En este paso, se carga el conjunto de datos necesario para el fine-tuning de los modelos GPT-2 y T5 en la tarea de generación de resúmenes. Es importante destacar que se utiliza el ecosistema de Hugging Face para acceder y manejar los datos, aprovechando sus librerías y métodos especializados.
3. **Definir el tokenizador**
   Se define el tokenizador que se utilizará para convertir el texto en una secuencia de tokens, siendo estas las unidades mínimas que los modelos de lenguaje como GPT-2 y T5 procesan. Este paso es crucial para preparar los datos de entrada de manera que sean comprensibles para el modelo.
4. **Definición de la plantilla y cargue del prompt**
   En este paso, se define una plantilla de prompt que será utilizada para generar las entradas de texto que el modelo procesará. Este punto es esencial para guiar al modelo en la tarea específica de resumen, proporcionándole el contexto necesario para producir una salida adecuada.
5. **Creación y ajuste de percentiles y cuantiles para la longitud de los tokens**
   Este paso se enfoca en analizar la longitud de los tokens en los diálogos y resúmenes del conjunto de datos para establecer umbrales que permitan filtrar ejemplos demasiado largos. El objetivo es optimizar el rendimiento del modelo durante el entrenamiento, asegurando que los ejemplos sean manejables y representativos.
6. **Cargue y preparación del tokenizador**
   Seguidamente, se carga y prepara el tokenizador para su uso con el modelo de lenguaje. Esto incluye la configuración de los tokens especiales, como el token de padding, que es esencial para alinear las secuencias de entrada y garantizar que el modelo procese los datos correctamente.
7. **Definición del data loader**
   En este punto, se define un data collator personalizado, que es una función encargada de procesar y preparar los lotes de datos (batches) que se usarán durante el entrenamiento del modelo. Este data collator se encarga de tokenizar las entradas y salidas, aplicar padding, y manejar las máscaras de atención, todo esto para que el modelo pueda entrenarse de manera eficiente.

8. Entrenamiento del modelo

Este paso se centra en la configuración y ejecución del proceso de entrenamiento para los modelos GPT-2 y T5 utilizando el conjunto de datos preparado. Se define cómo se llevará a cabo el entrenamiento, incluyendo los parámetros de optimización, estrategias de guardado, y configuración del Trainer de Hugging Face.

En este punto se configuran parámetros como como el número de épocas, el tamaño del lote, la tasa de aprendizaje, y las estrategias de guardado y evaluación.

Tabla 5. Hiperparametros entrenamiento modelo T5.

T5 hiperparametros	
num_train_epochs	8
optim	"adamw_torch"
per_device_train_batch_size	30
per_device_eval_batch_size	30
gradient_accumulation_steps	5
max_steps	-1
weight_decay	0.01
learning_rate	5,00E-05
remove_unused_columns	FALSE
max_grad_norm	1
gradient_checkpointing	TRUE
do_train	TRUE
do_eval	TRUE
lr_scheduler_type	"constant"
warmup_ratio	5
logging_strategy	"epoch"
logging_steps	100
evaluation_strategy	"epoch"
eval_steps	100
group_by_length	FALSE
save_strategy	"epoch"
save_steps	100
save_total_limit	100
load_best_model_at_end	TRUE

Tabla 6. Hiperparametros entrenamiento modelo GPT-2.

GPT2 hiperparametros	
num_train_epochs	8
optim	"adamw_torch"
per_device_train_batch_size	20
per_device_eval_batch_size	20
gradient_accumulation_steps	5

GPT2 hiperparametros	
max_steps	-1
weight_decay	0.01
learning_rate	5,00E-05
remove_unused_columns	FALSE
max_grad_norm	1
gradient_checkpointing	TRUE
do_train	TRUE
do_eval	TRUE
lr_scheduler_type	"constant"
warmup_ratio	5
logging_strategy	"epoch"
logging_steps	100
evaluation_strategy	"epoch"
eval_steps	100
group_by_length	FALSE
save_strategy	"epoch"
save_steps	100
save_total_limit	100
load_best_model_at_end	TRUE

Evaluación del modelo y estimación de métricas

Después del entrenamiento del modelo, el siguiente paso fue determinar las métricas que permitieran medir su efectividad en la generación de resúmenes. Los procedimientos del paso 1 al 4 en esta etapa son equivalentes a los pasos 2 a 5 realizados durante el entrenamiento, con la diferencia en el conjunto de datos utilizado y la cantidad de registros. Durante el entrenamiento, se empleó un volumen significativamente mayor de datos.

5. **Cargue del modelo con Fine Tuning y tokenizador**
   En esta fase, se cargó el modelo previamente ajustado mediante *fine-tuning*, junto con su tokenizador correspondiente. Ambos fueron utilizados para la generación de texto y la evaluación sobre el conjunto de prueba.
6. **Generación de métricas y guardado en archivos**
   Para evaluar el desempeño del modelo ajustado, se generaron predicciones a partir del conjunto de prueba y se calcularon métricas relevantes, incluyendo *BLEU score*, precisión *n-gram* (1-gram, 2-gram, 3-gram, 4-gram), *ROUGE-1*, *ROUGE-2*, *ROUGE-L* y *ROUGE-Lsum*. Tanto las predicciones como las métricas obtenidas fueron almacenadas en archivos para su posterior análisis.

La metodología previamente descrita se presenta de forma gráfica en la Figura 15.

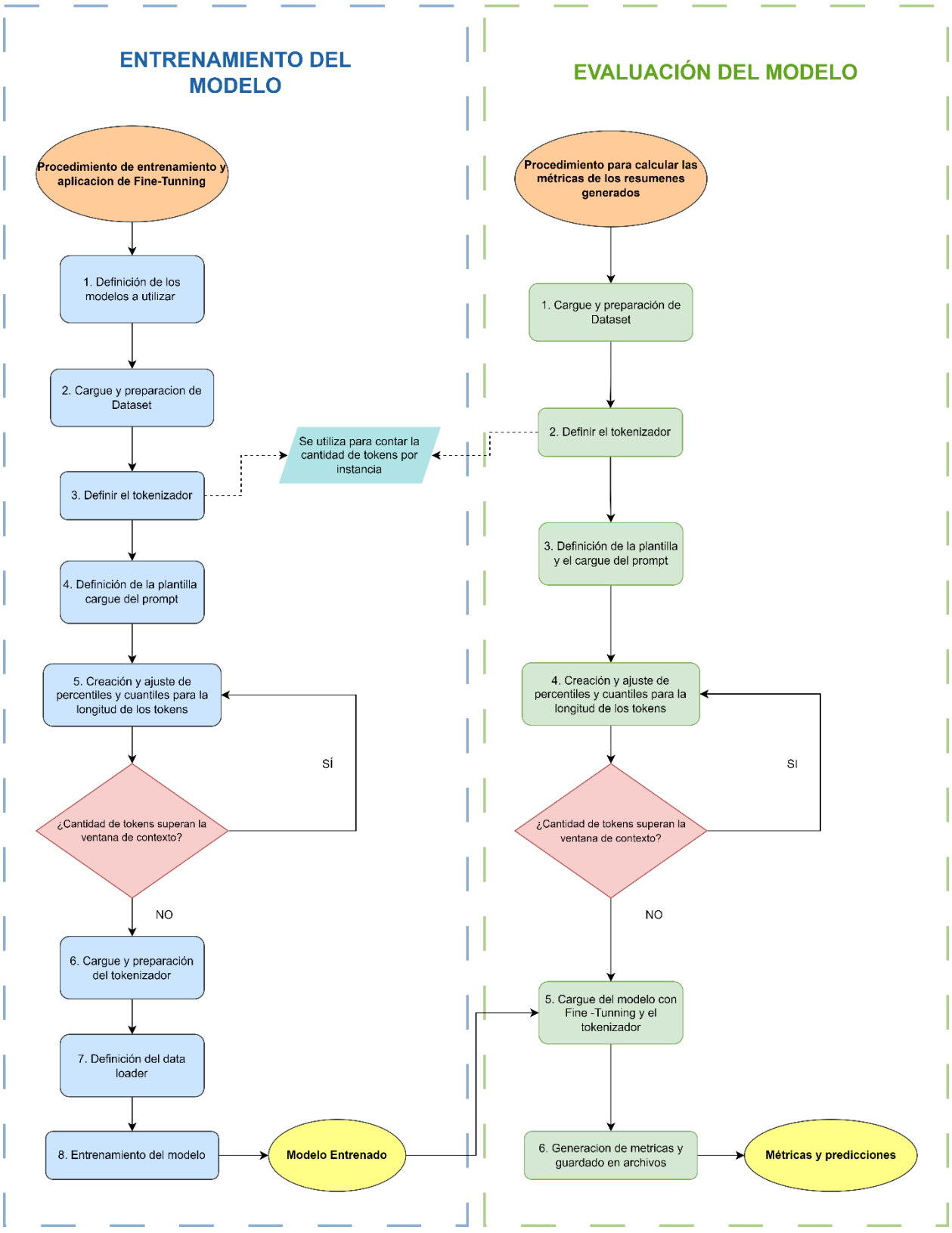


Figura 15. Diagrama de flujo - Proceso de entrenamiento y aplicación de Fine Tuning (Elaboración propia).

El proceso para implementar las metodologías PEFT es similar al del Fine-tuning tradicional (Figura 16), pero aprovechando la versión PEFT del modelo. Esta aproximación simplifica considerablemente

la experimentación y la obtención de resultados, facilitando la transición de técnicas de Fine-tuning convencionales a enfoques más eficientes.

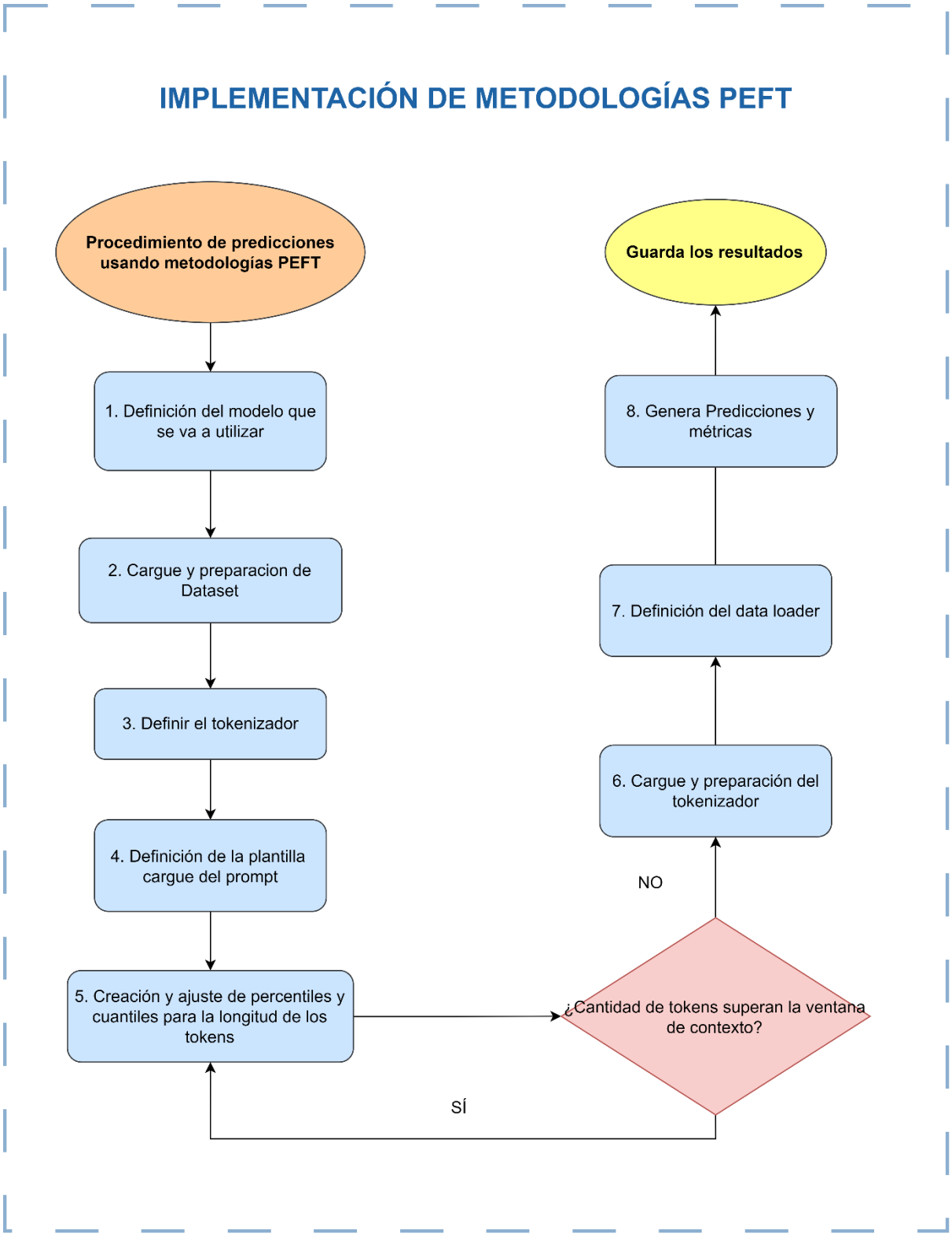


Figura 16. Implementación de metodologías PEFT (Elaboración propia).

5.3 EVALUACIÓN DE RESULTADOS

Al realizar el proceso de evaluación descrito en el paso anterior, se puede comparar los resúmenes originales presentes en el dataset con el generado por el modelo. Esta información permite una evaluación detallada de la calidad de las predicciones en comparación con los resúmenes referenciales del dataset. Adicionalmente observamos las primeras métricas obtenidas, con las cuales podemos validar la coherencia, la relevancia y la concisión de los resúmenes, lo que facilita la identificación de áreas de mejora en el rendimiento del modelo.

Es importante destacar que, aunque estas métricas proporcionan una valiosa orientación sobre la calidad de los resúmenes generados, no son completamente precisas y, en muchos casos, no reflejan con exactitud la calidad real de los resúmenes. Esto se debe a la ausencia de un estándar absoluto que determine cuán bien está elaborado un resumen. No obstante, estas métricas y predicciones nos permiten obtener una comprensión clara del rendimiento de los modelos empleados, así como de su configuración.

En la Tabla 7 se presenta un ejemplo de las predicciones obtenidas para el dataset DialogSum, donde se puede observar que el modelo es capaz de captar el contexto general de las conversaciones identificando actores principales y resaltando los hechos más relevantes de forma concreta. Esta misma capacidad se observó al evaluar los resultados obtenidos al con otros dataset (SamSum y CNN Daily Mail).

Tabla 7.Resultados preliminares obtenidos (Elaboración propia).

input	output	prediction
Dylan: what else did he tell you? Abby: That they left kind of illegally Abby: it was a big mess and extreme poverty everywhere Abby: then suddenly the border was open and they just left Abby: people were boarding available ships, whatever, just to get out of there Abby: he showed me some pictures, like <file_photo> Dylan: insane Abby: yes, and his father was among the people Dylan: scary but interesting Abby: very!	Miro speaks Albanian with his parents. His family left Albania illegally in 1990s.	Abby met Miro yesterday at the party. Miro's father came to the US in the early 1990s. Miro's father left Albania for the seacoast. Miro's father told Abby about the border opening and people boarding available ships.
Summarize the following conversation. ### Conversation: Abdellilah: Where are you? Sam: work	Sam won't finish work till 5. Sam is bringing him over about 9 am. Sam will see Abdellilah in the morning.	Sam is not bringing him over tonight. Sam will be at work around 9.

Abdellilah: What time you finish? Sam: Not til 5 Abdellilah: Are your bringing him over tonight: Sam: No in the morning: Abdellilah: ok, what time? Sam: About 9. Is that ok? Abdellilah: ok - see you then		
Summarize the following conversation. ### Conversation: Abigail: Why didn't you attend the part last night? :/ Ethan: I am currently in Los Angeles dat's why it was impossible for me Abigail: When did you leave for LA? :o You didnt even tell me Ethan: Why did you wanted know about it ? Abigail: Well! you are my friend and I dont even know that you are outta town. Dont seems good Ethan: Don't worry. I would be back after 3 or 2 days Abigail: Let me know when you reach back Ethan: Sure. Will meet soon	Ethan didn't come to the party last night because he is in Los Angeles. Abigail didn't know about it. Ethan will be back in a couple of days, the he will reach out to Abigail.	Ethan didn't attend the part last night because he was in Los Angeles. Ethan will be back after 3 or 2 days.

Resultados

Ahora bien, se presentan los resultados obtenidos para las métricas que miden el desempeño del modelo (Tabla 8) sobre una serie de modelos preentrenados enfocados en la tarea de generación automática de resúmenes de texto. Los modelos evaluados incluyen variantes de GPT-2 (GPT-2, GPT-2 Medium y GPT-2 Large) y de T5 (T5 y T5 Large), todos ampliamente utilizados en el campo de procesamiento de lenguaje natural (NLP).

El propósito de este análisis es identificar el impacto que las técnicas de PEFT tienen en la capacidad de los modelos para generar resúmenes efectivos y precisos, evaluados mediante métricas estándar como BLEU y ROUGE. A través de esta comparación, se busca determinar qué combinación de modelo y metodología PEFT ofrece el mejor balance entre rendimiento y eficiencia computacional, considerando que los modelos más grandes presentan mayores desafíos en términos de recursos.

Tabla 8. Resultados obtenidos. (Elaboración propia)

DATASET	MODELO	bleu_score	1-gram precision	2-gram precision	3-gram precision	4-gram precision	rouge1	rouge2	rougeL	rougeLsum
DialogSum	GPT2	0,256	0,558	0,352	0,204	0,116	0,460	0,204	0,377	0,377
	GPT2-PEFT	0,243	0,601	0,334	0,204	0,120	0,450	0,210	0,365	0,365
	GPT2 MEDIUM	0,301	0,595	0,372	0,245	0,151	0,494	0,245	0,418	0,418
	GPT2 MEDIUM - PEFT	0,260	0,566	0,333	0,209	0,117	0,464	0,212	0,381	0,382
	GPT2 LARGE	0,295	0,607	0,370	0,240	0,141	0,479	0,227	0,400	0,400
	GPT2 LARGE - PEFT	0,253	0,539	0,326	0,205	0,114	0,461	0,211	0,378	0,378
	T5	0,323	0,623	0,399	0,267	0,165	0,519	0,268	0,442	0,442
	T5 - PEFT	0,300	0,593	0,380	0,260	0,145	0,501	0,240	0,442	0,442
SamSum	GPT2	0,136	0,480	0,200	0,153	0,101	0,436	0,190	0,354	0,355
	GPT2-PEFT	0,112	0,431	0,192	0,142	0,100	0,403	0,157	0,348	0,348
	GPT2 MEDIUM	0,149	0,507	0,231	0,131	0,100	0,453	0,213	0,378	0,378
	GPT2 MEDIUM - PEFT	0,160	0,510	0,220	0,131	0,100	0,481	0,240	0,365	0,365
	GPT2 LARGE	0,181	0,490	0,220	0,131	0,100	0,430	0,189	0,363	0,363
	GPT2 LARGE - PEFT	0,190	0,506	0,216	0,161	0,100	0,441	0,194	0,365	0,365
	T5	0,185	0,514	0,250	0,150	0,093	0,487	0,246	0,412	0,412
	T5 - PEFT	0,200	0,490	0,230	0,160	0,100	0,421	0,220	0,400	0,400
cnn_daily_mail	T5	0,172	0,422	0,203	0,123	0,083	0,412	0,204	0,312	0,363
	T5 - PEFT	0,168	0,413	0,196	0,119	0,082	0,408	0,200	0,308	0,359
	T5 LARGE	0,182	0,440	0,215	0,131	0,090	0,426	0,214	0,320	0,374
	T5 LARGE - PEFT	0,186	0,440	0,219	0,134	0,092	0,430	0,219	0,323	0,378

A continuación, se presentan los principales hallazgos encontrados para cada modelo y dataset:

- Los resultados obtenidos revelan una gran variabilidad en el desempeño de los modelos en función del dataset. DialogSum donde los modelos tienen los mejores resultados en general en contraste con las versiones base. Esta variabilidad subraya la importancia de evaluar los modelos en una amplia gama de datasets para garantizar su robustez y generalización. Además, sugiere que las características específicas de cada dataset, como la complejidad lingüística y la diversidad de temas, pueden influir significativamente en la elección de la arquitectura del modelo y la configuración de los hiperparámetros.

- **BLEU SCORE:** El análisis del BLEU Score indica que la arquitectura del modelo juega un papel crucial en el desempeño. Valores superiores a 0,3 indican buenos resultados, el dataset DialogSum, con los modelos T5 y GPT2 Large en su versión base obtienen los mejores resultados en 1y 2 gram precision, lo que sugiere que estas arquitecturas son más adecuadas para capturar las características de estos conjuntos de datos. Sin embargo, en los datasets SamSum y CNN Daily Mail, el patrón es menos claro, esto sugiere que la complejidad de este dataset requiere un ajuste fino más específico.

- **ROUGE:** Los resultados obtenidos con la métrica ROUGE indican que, en general la versión base de los modelos supera ligeramente a las versiones PEFT en los tres conjuntos de datos con valores de las métricas por encima de 0,3 incluso llegan a 0,5 lo que indican buens resultados respecto a la inferencia. No obstante, las diferencias son mínimas y no siempre consistentes a lo largo de todos los subconjuntos de ROUGE (ROUGE-1, ROUGE-2, ROUGE-L, ROUGE-Lsum). Esto sugiere que, si bien PEFT puede mejorar el desempeño en tareas específicas, en este caso no ha logrado una ventaja significativa en términos de coincidencia con las referencias.

- Aunque las variantes PEFT de los modelos GPT-2 no alcanzan los niveles de rendimiento de los modelos completos, logran mantener un rendimiento competitivo. Por ejemplo, el modelo GPT-2 PEFT mejora la precisión de 1-gram en comparación con el modelo base GPT-2, sugiriendo que la optimización de parámetros ayuda a mejorar aspectos importantes sin requerir recursos computacionales elevados.

- Los modelos con PEFT ofrecen un compromiso interesante para entornos con restricciones computacionales. Aunque los modelos completos (en particular T5) son preferibles en términos de calidad de resumen, las versiones PEFT mantienen resultados razonablemente cercanos, lo que los convierte en una opción viable cuando se necesita reducir la carga computacional.

- El dataset CNN_DAILY MAIL tuvo mayor dificultad por su longitud en las oraciones y la cantidad de datos recolectados, ya que este es un dataset con entrevistas de varios meses, lo que generaba que los tokens resultantes sobrepasaran la ventana de contexto y se descartaran.

Nota: El código utilizado para el entrenamiento de los diferentes modelos se encuentra disponible en Github [34].

5.4 RENDIMIENTO COMPUTACIONAL

Al finalizar el entrenamiento de cada modelo, se obtuvieron archivos que contenían los tensores correspondientes a los pesos y sesgos aprendidos. Estos archivos, cuyo tamaño dependía directamente de la complejidad del modelo, sirvieron como base para realizar las predicciones. Asimismo, se registró el tiempo de ejecución, consumo de RAM, VRAM y espacio en disco durante el entrenamiento de cada modelo, estos datos son el insumo para poder comparar el consumo de recursos. A continuación, se detallan los resultados experimentales:

Tabla 9. Ocupación en disco de los modelos (Elaboración propia).

Modelo General	Modelo	Peso Tensores (MB)
GPT2	GPT2	474,79
	GPT2 -PEFT	5,57
	GPT2 MEDIUM	1.353,48
	GPT2 MEDIUM PEFT	15,89
T5	T5 SMALL	293
	T5 SMALL - PEFT	3,95
	T5	944
	T5 - PEFT	10,1
	T5 LARGE	2.910
	T5 LARGE - PEFT	27,0

Se observa cómo el uso de metodologías como **PEFT** (Parameter-Efficient Fine-Tuning) en modelos grandes de lenguaje (**LLMs**) impacta significativamente en la cantidad de parámetros (tensores) ajustados y almacenados durante el entrenamiento (Tabla 9). Este impacto se traduce en una reducción sustancial del tamaño de los pesos almacenados, como demuestran los datos analizados.

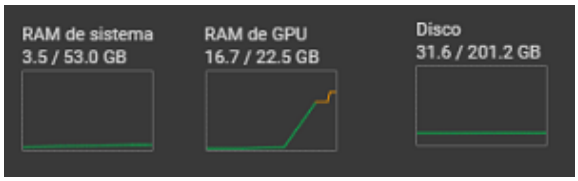
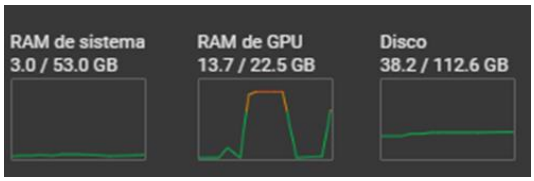
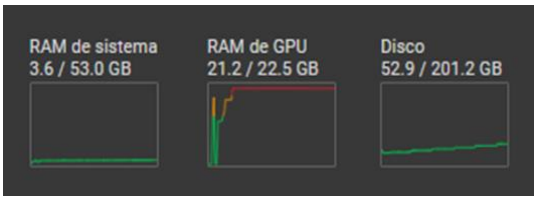
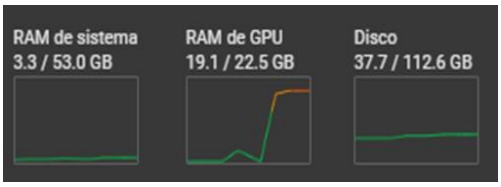
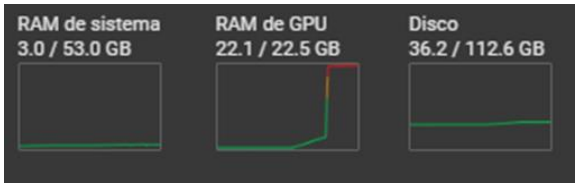
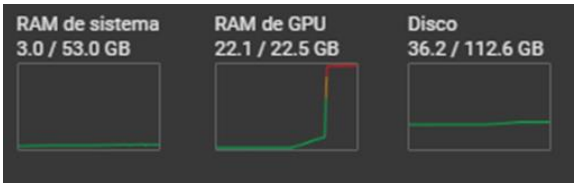
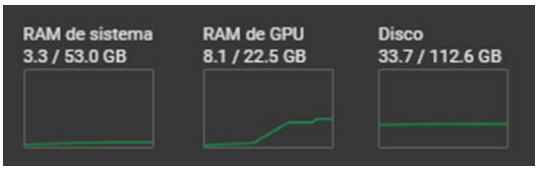
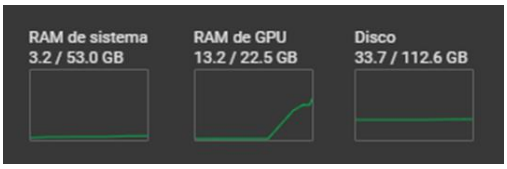
Por ejemplo, en el caso del modelo **T5 Large**, se logró reducir el tamaño de los pesos de **2.910 MB a 27 MB**, lo que representa una disminución del **99%**. En promedio, el peso de los tensores almacenados experimentó una reducción del **98.88%**.

Este enfoque no solo optimiza el uso de recursos computacionales, sino que también facilita escenarios más ágiles para la experimentación, el despliegue y la reutilización de modelos pre-entrenados, especialmente en entornos con hardware limitado.

Además, se evalúa el desempeño computacional de los modelos utilizados durante el desarrollo de esta investigación, considerando el uso de recursos como la memoria RAM del sistema, la memoria de la GPU y el almacenamiento en disco, así como el tiempo total de entrenamiento. Estas métricas son fundamentales para comparar la eficiencia de los modelos y metodologías empleadas, especialmente al trabajar con modelos de lenguaje grandes y técnicas de ajuste fino (Fine-Tuning). A continuación, se presentan los resultados obtenidos para los modelos GPT-2, GPT-2 Medium y T5 bajo diferentes configuraciones de hardware y carga de trabajo.

A continuación, observaremos el consumo de varias ejecuciones de modelos, al igual que los tiempos que les tomo el entrenamiento, con esto podemos evidenciar las diferencias entre el consumo de los modelos al igual que la diferencia de estos implementados con las metodologías PEFT, en términos de memoria RAM de GPU y memoria, RAM del sistema, y tamaño de disco utilizado.

Tabla 10. Consumo de recursos durante el entrenamiento (Elaboración propia).

GPT 2	GPT2 - PEFT
 Duración: 1 h 42 m 33 s	 Duración: 19 m 30 s
GPT 2 MEDIUM	**GPT2 MEDIUM - PEFT**
 Duración: 4 h 27 m 45 s	 Duración: 58 min 50 s
GPT 2 LARGE	**GPT2 LARGE - PEFT**
 Duración: 4 h 50 min 34 s	 Duración: 4 h 35 min 02 s
T5	**T5 - PEFT**
 Duración: 5 h 43 m 46 s	 Duración: 5 h 20 m 21 s

De las imágenes en la Tabla 10. se observa que el consumo de memoria RAM de sistema (CPU) es constante en todos los casos, oscilando entre 3.0 y 3.6 GB tanto en el Fine Tuning convencional como en PEFT. Este comportamiento es esperable, ya que el entrenamiento se configura para hacer uso principalmente de los recursos de la GPU.

En cuanto a la memoria RAM de la GPU, se presentan comportamientos diferenciados según el modelo. En el caso de GPT-2 Medium, el uso disminuyó de 21.2 GB en el Fine-Tuning convencional a 19.1 GB con PEFT, lo que indica una optimización en el uso de este recurso. Sin embargo, para GPT-2 Large, el consumo se mantuvo constante en 22.1 GB, lo que evidencia que PEFT no necesariamente reduce el uso de memoria GPU en todos los casos. Por otro lado, el modelo T5 muestra un comportamiento opuesto, donde el consumo de GPU aumentó de 8.1 GB a 13.2 GB al aplicar PEFT, sugiriendo que esta técnica puede requerir más memoria dependiendo de las características del modelo.

En términos de duración de entrenamiento, PEFT demuestra una reducción significativa en los tiempos en la mayoría de los casos. Por ejemplo:

- **GPT-2**: De **1 hora, 42 minutos y 33 segundos** a **19 minutos y 30 segundos** (reducción del **81.0%**).
- **GPT-2 Medium**: De **4 horas, 27 minutos y 45 segundos** a **58 minutos y 50 segundos** (reducción del **78.0%**).
- **T5**: De **5 horas, 43 minutos y 46 segundos** a **5 horas, 20 minutos y 21 segundos** (reducción del **6.8%**).

Aunque la reducción de tiempo en el modelo T5 es más modesta, en los demás casos resulta sustancial. Este ahorro de tiempo tiene implicaciones importantes, dado el elevado costo energético asociado al uso prolongado de GPU. En escenarios donde sea necesario realizar múltiples ajustes con diferentes configuraciones, las ventajas de PEFT en términos de eficiencia temporal pueden traducirse en un ahorro económico significativo y en una menor huella de carbono asociada al entrenamiento de modelos. Además, estos resultados resaltan la necesidad de evaluar caso por caso el impacto de PEFT, ya que sus beneficios pueden variar ampliamente según el modelo y los recursos computacionales disponibles.

6 CONCLUSIONES Y TRABAJOS FUTUROS

6.1 CONCLUSIONES

Este estudio evaluó la implementación de metodologías de ajuste eficiente de parámetros (PEFT) en modelos de lenguaje grande (LLM) para la tarea de generación automática de resúmenes de texto. Su objetivo principal fue analizar cómo estas metodologías permiten adaptar los modelos a tareas específicas de manera eficiente, reduciendo la carga computacional sin afectar significativamente el desempeño. Este enfoque es especialmente relevante en escenarios con recursos limitados, donde la capacidad de aprovechar modelos avanzados suele estar restringida.

A lo largo del análisis, se evaluaron modelos de la familia GPT-2 (en sus versiones base, Medium y Large) y T5, tanto en su forma estándar como con PEFT, empleando métricas como BLEU y ROUGE sobre los datasets DialogSum, SamSum y CNN Daily Mail. Los resultados muestran diferencias significativas según el modelo y el dataset, lo que resalta la importancia de adaptar los modelos al contexto de aplicación.

En el dataset DialogSum, T5 sin PEFT obtuvo el mejor rendimiento global (BLEU = 0.323, ROUGE-1 = 0.519), seguido por GPT-2 Medium sin PEFT (BLEU = 0.301, ROUGE-1 = 0.494). Sin embargo, la aplicación de PEFT en estos modelos mostró una ligera reducción en las métricas, indicando que, aunque mejora la eficiencia computacional, puede introducir una penalización en la calidad del resumen generado.

En SamSum, se observó un desempeño más equilibrado, con GPT-2 Large con PEFT alcanzando el mejor BLEU (0.19) y T5 sin PEFT obteniendo el mejor ROUGE-1 (0.487). Esto sugiere que modelos más grandes pueden beneficiarse del ajuste con PEFT en este tipo de datos, aunque con una diferencia marginal respecto a las versiones sin ajuste.

Por otro lado, en CNN Daily Mail, T5 Large con PEFT obtuvo el mejor BLEU (0.186) y ROUGE-1 (0.43), superando a sus versiones sin ajuste. Esto podría deberse a la mayor capacidad del modelo para manejar textos extensos y su arquitectura enfocada en tareas de resumen. No obstante, la limitación de la ventana de contexto en modelos autoregresivos como GPT-2 afectó su desempeño en este dataset, evidenciando la necesidad de estrategias específicas para manejar textos largos.

Desde una perspectiva de eficiencia computacional, las metodologías PEFT permitieron una reducción significativa en los tiempos de entrenamiento (hasta un 81% en modelos como GPT-2), lo que representa un ahorro en consumo energético y costos de hardware. Sin embargo, el impacto en el uso de memoria GPU no fue uniforme: mientras que GPT-2 Medium con PEFT mostró una reducción en el consumo, T5 con PEFT evidenció un incremento, lo que sugiere que la efectividad de estas metodologías depende del tipo de modelo y su arquitectura.

En términos de balance entre calidad y eficiencia, T5 demostró mayor adaptabilidad a datos complejos, consolidándose como una opción sólida para tareas de resumen. GPT-2 Medium emergió como una alternativa viable en entornos con restricciones de hardware, ofreciendo un rendimiento aceptable con menores costos computacionales.

Este estudio confirma que PEFT representa una solución eficiente para adaptar modelos LLM a tareas específicas, especialmente en escenarios con limitaciones de recursos. Aunque estas metodologías no buscan mejorar directamente la precisión del modelo, su capacidad para reducir costos

computacionales y facilitar su implementación en entornos con recursos limitados las convierte en una herramienta clave para la democratización del acceso a modelos avanzados.

En el futuro, se recomienda profundizar en estrategias que mitiguen la pérdida de rendimiento observada con PEFT en ciertos modelos, así como explorar su impacto en el uso de memoria y optimización del balance entre calidad y eficiencia. Esto permitirá que los modelos de lenguaje grande sean adoptados más ampliamente en contextos productivos y académicos, fortaleciendo su aplicación en la síntesis y análisis de grandes volúmenes de información.

En el futuro, se espera que la investigación continúe explorando maneras de optimizar el uso de PEFT, particularmente en la reducción del consumo de memoria y el balance entre calidad y eficiencia. Esto permitirá que los modelos de lenguaje grande se adopten de manera más generalizada en contextos productivos y académicos, fortaleciendo su papel en la síntesis y análisis de grandes volúmenes de información.

6.2 TRABAJOS FUTUROS

A partir de los resultados obtenidos en esta investigación, se identifican dos áreas clave de desarrollo que podrían contribuir a mejorar la eficacia y eficiencia del modelo de generación de resúmenes. Estos enfoques propuestos se centran en ajustes técnicos específicos y mejoras en la arquitectura del modelo, en lugar de explorar un listado amplio de opciones en el área. La intención es profundizar en métodos que directamente impacten la calidad y precisión de los resúmenes generados, aprovechando los hallazgos de este trabajo.

En primer lugar, resulta relevante investigar la aplicación de otras metodologías PEFT (Parameter-Efficient Fine-Tuning) además de LoRA, que en este estudio ha demostrado ser eficaz para ajustar modelos de lenguaje grandes (LLM). Metodologías como Adapters, Prefix Tuning o P-Tuning ofrecen un potencial considerable, ya que se enfocan en minimizar la necesidad de ajustes en todos los parámetros del modelo, lo que a su vez reduce la carga computacional sin sacrificar precisión. La implementación de estos métodos podría mejorar la calidad de los resúmenes generados, especialmente en casos de uso donde se requiere un mayor grado de generalización o la adaptación a diferentes estilos y contextos de texto. Este enfoque sería una extensión directa del trabajo realizado, buscando mejorar los resultados de resumen con una mayor eficiencia en el ajuste fino.

Otra línea de investigación valiosa sería la experimentación con modelos de lenguaje más recientes y avanzados, como GPT-3 o BART. La inclusión de estos modelos en futuros estudios permitiría analizar si arquitecturas más sofisticadas y de mayor capacidad pueden aportar mejoras en términos de precisión y relevancia en la generación de resúmenes. Además, estos modelos podrían ser más adecuados para tareas que requieren una comprensión más profunda del contexto y un nivel elevado de abstracción, algo que podría mejorar la adaptabilidad de los resúmenes en áreas específicas o dominios complejos. La exploración de estas arquitecturas podría aportar una comparación valiosa con los resultados obtenidos en esta investigación y ofrecer una visión más amplia de las posibilidades de ajuste fino en modelos de lenguaje avanzados.

Ambas propuestas están orientadas a mejorar las técnicas y resultados obtenidos en esta investigación, optimizando el ajuste de los modelos de lenguaje tanto en calidad como en uso eficiente de recursos. Estos enfoques futuros, al centrarse en metodologías PEFT adicionales y en la evaluación de modelos más avanzados, no solo contribuirán a generar resúmenes de mayor precisión y adaptabilidad, sino que también facilitarán una mayor aplicabilidad en sectores que requieren

procesamiento de grandes volúmenes de texto. La evolución constante de los modelos y técnicas de ajuste fino continúa siendo un área prometedora para la innovación en procesamiento de lenguaje natural.

7 REFERENCIAS BIBLIOGRÁFICAS

[1] Hugging Face, "PEFT." Accessed: Nov. 18, 2023. [Online]. Available: https://huggingface.co/docs/peft/index

[2] "Speech and Language Processing." Accessed: Apr. 20, 2024. [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/

[3] T. Wolf *et al.*, "HuggingFace's Transformers: State-of-the-art Natural Language Processing," Oct. 2019.

[4] L. Dong *et al.*, "Unified Language Model Pre-training for Natural Language Understanding and Generation," *Adv Neural Inf Process Syst*, vol. 32, May 2019.

[5] L. Tunstall, L. Von Werra, and T. Wolf, *Natural Language Processing with Transformers Building Language Applications with Hugging Face*. O'Reilly, 2022.

[6] D. Manning Christopher, Raghavan Prabhakar, and Shutze Hinrich, "Introduction to Information Retrieval," *Cambridge University Press*, May 2008.

[7] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: a Method for Automatic Evaluation of Machine Translation," *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia*, pp. 311–318, Jul. 2002.

[8] C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," University of Southern California, Marina del Rey.

[9] S. Raschka, *Python machine learning : aprendizaje automático y aprendizaje profundo con Python, scikit-learn y TensorFlow*, 2ª ed. Barcelona: Marcombo, 2019.

[10] IBM, "¿Qué es Deep Learning?" Accessed: Nov. 18, 2023. [Online]. Available: https://www.ibm.com/es-es/topics/deep-learning

[11] G. James, D. Witten, T. Hastie, and R. Tibshirani, "Springer Texts in Statistics An Introduction to Statistical Learning".

[12] "Capítulo 8 Métodos basados en Deep Learning | Notas sobre pronóstico del flujo de tráfico en la ciudad de Madrid." Accessed: Nov. 18, 2023. [Online]. Available: https://bookdown.org/amanas/traficomadrid/m%C3%A9todos-basados-en-deep-learning.html

[13] I. Goodfellow, Y. Bengio, and A. Courville, "Deep Learning".

[14] H. Hapke, C. Howard, and H. Lane, *Natural Language Processing in Action: Understanding, analyzing, and generating text with Python*. Simon and Schuster, 2019.

[15] Cloudflare, "What is a large language model (LLM)?" Accessed: Nov. 18, 2023. [Online]. Available: https://www.cloudflare.com/learning/ai/what-is-large-language-model/

[16] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, vol. 1, pp. 4171–4186, Oct. 2018.

[17] NVIDIA, "¿Qué Es un Modelo de IA Previamente Entrenado?" Accessed: Nov. 18, 2023. [Online]. Available: https://la.blogs.nvidia.com/2023/01/11/que-es-un-modelo-de-ia-previamente-entrenado/

[18] Keras, "Transfer learning & fine-tuning." Accessed: Nov. 18, 2023. [Online]. Available: https://keras.io/guides/transfer_learning/

[19] P. Peng and J. Wang, "How to fine-tune deep neural networks in few-shot learning?," Dec. 2020, Accessed: Nov. 18, 2023. [Online]. Available: https://arxiv.org/abs/2012.00204v1

[20] V. Lialin, V. Deshpande, U. Lowell, A. Rumshisky, and A. Ai, "Scaling Down to Scale Up: A Guide to Parameter-Efficient Fine-Tuning," Mar. 2023.

[21] J. He, C. Zhou, X. Ma, T. Berg-Kirkpatrick, and G. Neubig, "TOWARDS A UNIFIED VIEW OF PARAMETER-EFFICIENT TRANSFER LEARNING," ICLR, 2022.

[22] Z. Hu *et al.*, "LLM-Adapters: An Adapter Family for Parameter-Efficient Fine-Tuning of Large Language Models," Apr. 2023.

[23] K. Pandya, "PEFT-MedAware: Large Language Model for Medical Awareness," Nov. 2023.

[24] N. Ding *et al.*, "Parameter-efficient fine-tuning of large-scale pre-trained language models," *Nat Mach Intell*, vol. 5, no. 3, pp. 220–235, Mar. 2023, doi: 10.1038/s42256-023-00626-4.

[25] E. Hu *et al.*, "LoRA: Low-Rank Adaptation of Large Language Models," *ICLR 2022 - 10th International Conference on Learning Representations*, Jun. 2021.

[26] "GitHub - Ryota-Kawamura/Generative-AI-with-LLMs: In Generative AI with Large Language Models (LLMs), you'll learn the fundamentals of how generative AI works, and how to deploy it in real-world applications." Accessed: Jul. 27, 2024. [Online]. Available: https://github.com/Ryota-Kawamura/Generative-AI-with-LLMs

[27] X. L. Li and P. Liang, "Prefix-Tuning: Optimizing Continuous Prompts for Generation," *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, pp. 4582–4597, Jan. 2021, doi: 10.18653/v1/2021.acl-long.353.

[28] M. Wang, P. Xie, Y. Du, and X. Hu, "T5-Based Model for Abstractive Summarization: A Semi-Supervised Learning Approach with Consistency Loss Functions," *Applied Sciences (Switzerland)*, vol. 13, no. 12, Jun. 2023, doi: 10.3390/app13127111.

[29] B. Tarján, T. Fegyó, and P. Mihajlik, "Morphology aware data augmentation with neural language models for online hybrid ASR," *Acta Linguistica Academica*, vol. 69, no. 4, pp. 581–598, Dec. 2022, doi: 10.1556/2062.2022.00582.

[30] "neil-code/dialogsum-test · Datasets at Hugging Face." Accessed: May 25, 2024. [Online]. Available: https://huggingface.co/datasets/neil-code/dialogsum-test

[31] "Samsung/samsum · Datasets at Hugging Face." Accessed: May 25, 2024. [Online]. Available: https://huggingface.co/datasets/Samsung/samsum

[32] "abisee/cnn_dailymail · Datasets at Hugging Face." Accessed: May 25, 2024. [Online]. Available: https://huggingface.co/datasets/abisee/cnn_dailymail

[33] “Fine-Tune T5 for Conversational Model.” Accessed: Apr. 20, 2024. [Online]. Available: https://www.kaggle.com/code/kreeshrajani/fine-tune-t5-for-conversational-model/input?select=Conversation.csv

[34] “COMPARACION DE METODOLOGIAS PEFT APLICADAS A LLM ENFOCADOS EN GENERAR RESUMENES DE TEXTOS.” Accessed: Dec. 03, 2024. [Online]. Available: https://github.com/Caarevalor/COMPARACION_DE_METODOLOGIAS_PEFT_APLICADAS_A_LLM_ENFOCADOS_EN_GENERAR_RESUMENES_DE_TEXTOS