UNIVERSIDAD EAFIT

EVALUACIÓN DE RENDIMIENTO DE DIFERENTES MODELO GRANDES DE LENGUAJE PARA EL RECONOCIMIENTO DE EMOCIONES EN TEXTO

Performance evaluation of different large language models for emotion recognition in text

DAVID ALEJANDRO LÓPEZ ATEHORTÚA

Proyecto de grado

Asesor

Edwin Nelson Montoya Munera

UNIVERSIDAD EAFIT

ESCUELA DE CIENCIAS APLICADAS E INGENIERÍA

MAESTRÍA EN CIENCIAS DE LOS DATOS Y LA ANALÍTICA

MEDELLÍN

2024

CONTENIDO

LISTA DE FIGURAS

LISTA DE ILUSTRACIONES

LISTA DE TABLAS

RESUMEN

Cada vez es más común que las personas expresen sus opiniones en textos cortos a través de diferentes medios gracias a la expansión del acceso a internet. Comprender y analizar de una manera eficiente el sentimiento de un individuo a partir de un texto es una tarea que es de utilidad en múltiples escenarios. Por lo anterior, una rama de las ciencias computacionales llamada Procesamiento de Lenguaje Natural (NLP) se ha dedicado al desarrollo de técnicas para entender todo lo relacionado con el lenguaje humano.

Las técnicas tradicionales, se basan en la frecuencia de una palabra o un grupo de palabras consecutivas para clasificar el texto en un sentimiento positivo, negativo o neutral. Estas técnicas tienen limitaciones dado que no logran capturar por completo el contexto de cada palabra en una oración, lo que afecta su precisión y capacidad para detectar un espectro de emociones más detallado.

Recientemente, los Modelos Largos de Lenguaje (LLMs) o Transformers revolucionaron la forma en que se realiza NLP gracias a su capacidad de capturar el contexto alrededor de cada palabra en un texto. Esto permite la detección de sentimientos de una manera más precisa e incluso, la clasificación del texto en una emoción más específica como alegría, optimismo, rabia, tristeza u otros.

Este proyecto, busca evaluar el rendimiento de diferentes LLMs para encontrar el que mejor se desempeñe en la detección de emociones a partir de textos cortos en inglés utilizando conjuntos de datos típicamente utilizados en investigaciones relacionadas con modelos de NLP.

Palabras clave: Grandes modelos de lenguaje, Transformadores, Reconocimiento de emociones, Procesamiento del Lenguaje Natural.

ABSTRACT

It is becoming more common for people to express their opinions in short texts through different media thanks to the expansion of internet access. Understanding and efficiently analyzing an individual's sentiment from a text is a task that is useful in multiple scenarios. For the above, a branch of computer science called Natural Language Processing (NLP) has been dedicated to developing techniques to understand everything related to human language.

Traditional techniques, based on the frequency of a word or a group of consecutive words to classify the text in a positive, negative or neutral sentiment. These techniques have limitations because they fail to capture the full context of each word in a sentence, affecting their accuracy and ability to detect a more detailed spectrum of emotions.

Recently, Long Language Models (LLMs) or Transformers revolutionized the way NLP is performed thanks to their ability to capture the context around each word in a text. This allows for the detection of feelings in a more precise way and even, the classification of the text into a more specific emotion such as joy, optimism, anger, sadness or others.

This project aims to evaluate the performance of different LLMs to find the best performing one in emotion detection from short texts in English using datasets typically used in research related to NLP models.

Keywords: Large Language Models, Transformers, Emotion Recognition, Natural Language Processing.

INTRODUCCIÓN

En los últimos años, el procesamiento de lenguaje natural (NLP) ha avanzado enormemente, y el surgimiento de modelos de lenguaje a gran escala (LLMs) ha revolucionado la manera de abordar el análisis de emociones en textos. Actualmente, muchas personas expresan sus pensamientos y sentimientos a través de textos cortos en redes sociales y otros medios digitales, y poder analizar estas emociones con precisión es esencial en contextos como la gestión de imagen pública, el análisis de satisfacción de clientes, o el estudio de tendencias sociales.

Este trabajo se enfoca en evaluar el rendimiento de distintos modelos de lenguaje de gran escala para detectar emociones en textos cortos, una tarea que presenta desafíos específicos para el NLP. Las técnicas tradicionales no logran capturar el contexto de las palabras, lo que afecta la precisión al momento de identificar emociones. En cambio, los LLMs basados en Transformers han demostrado una capacidad notable para comprender el contexto de cada palabra en una oración, permitiendo clasificaciones más precisas.

Para llevar a cabo esta investigación, se sigue la metodología CRISP-DM, un enfoque estándar en proyectos de ciencia de datos, que permite estructurar el análisis de datos y evaluación de modelos de manera sistemática. La importancia de este estudio radica en su potencial para contribuir tanto a la academia como a la industria, ya que proporciona un análisis comparativo de modelos avanzados de NLP aplicados a la detección de emociones, con miras a mejorar la comprensión y el análisis de los sentimientos expresados en textos breves.

PLANTEAMIENTO DEL PROBLEMA

Las técnicas convencionales de procesamiento de lenguaje natural (NLP) no tienen los niveles de precisión esperados de una buena detección de emociones dadas sus limitaciones para interpretar adecuadamente el contexto en textos cortos. Es necesario utilizar métodos avanzados para detectar de una manera más precisa y eficiente emociones en textos.

JUSTIFICACIÓN

En el contexto de las figuras públicas, como artistas o políticos, suele ser importante analizar la percepción que tiene determinado grupo de interés sobre ellos con el objetivo de gestionar sus acciones y aumentar su popularidad. Para lograr esto, usualmente se toman textos cortos de opiniones en diferentes plataformas como X (antes twitter) para luego ser clasificados con técnicas tradicionales de Procesamiento de Lenguaje Natual (NLP) en un sentimiento positivo, negativo o neutral. Sin embargo, esto impide un entendimiento detallado de las emociones que se están generando.

El surgimiento de los Modelos Grandes de Lenguaje (LLM) o Transformers revolucionó la forma en que se realizan las tareas relacionadas con el entendimiento del lenguaje humano. La mejora en la precisión para la detección de sentimientos, la capacidad de clasificar un texto en espectro de emociones más detallados y otras funcionalidades, los han convertido en el estado del arte. Algunos autores han demostrado que en comparación con técnicas tradicionales de NLP, los Transformers ofrecen una mayor eficiencia y una mejor adaptabilidad a contextos específicos, permitiendo una comprensión más profunda y matizada de las emociones (Prottasha et al., 2022).

Por tanto, transformar la manera tradicional de analizar los sentimientos y emociones a partir de textos cortos al utilizar estas novedosas técnicas, beneficia a las personas públicas en la gestión de su imagen y podría implementarse en diversas áreas de la industria que se beneficiarían al analizar el lenguaje humano de manera más eficiente.

Para esto, se van a comparar diferentes LLM en la tarea de clasificar textos cortos en emociones, utilizando algunos conjuntos de datos que han sido referentes en investigaciones similares, para finalmente identificar cuál de ellos tiene un mejor desempeño respecto a precisión en la clasificación.

OBJETIVOS

GENERAL

Evaluar diferentes modelos de LLM para la clasificación de emociones en textos cortos, basado en diferentes métricas de precisión de los modelos.

ESPECÍFICOS

- Analizar diferentes técnicas tradicionales y avanzadas que se usan para tareas de NLP para la detección de emociones.

- Seleccionar los modelos que sean más relevantes y prometedores para el análisis detallado de emociones en texto.

- Implementar los modelos seleccionados y evaluar su rendimiento en términos de precisión para determinar cuál es el mejor en la identificación de emociones en texto.

- Realizar un prototipo funcional del mejor modelo para la identificación de emociones en texto que permita realizar una evaluación cualitativa del modelo basado en nuevos textos.

MARCO TEÓRICO O MARCO CONCEPTUAL

DEFINICIÓN, TÉCNICAS Y MODELOS DE NLP

El procesamiento del lenguaje natural (NLP), es una disciplina que se encuentra en la intersección de la computación y la lingüística que busca que las máquinas puedan entender lo relacionado con el lenguaje humano. Los orígenes de NLP se remontan a los años 50, con el trabajo pionero de Alan Turing, quien propuso la famosa prueba de Turing como un criterio de inteligencia en máquinas, lo que indirectamente estimuló la investigación en el procesamiento del lenguaje natural (Turing, 1950).

En el NLP se han desarrollado varias técnicas a lo largo del tiempo, cada una adaptada a las capacidades tecnológicas y necesidades de su época:

i. Técnicas basadas en reglas

Las técnicas basadas en reglas es uno de los enfoques más antiguos en el campo del NLP, centrada en la creación manual de un conjunto de reglas lingüísticas predefinidas basadas en la gramática, la sintaxis y la semántica del lenguaje. Esta metodología requiere una comprensión detallada de la estructura del lenguaje por parte de los lingüistas y programadores que desarrollan estas reglas (Lees et al., 1957). En la práctica, un sistema de análisis sintáctico basado en reglas utiliza estas para definir cómo deben combinarse las palabras para formar frases válidas, y se aplica también en la extracción de entidades nombradas (NER), especificando patrones para identificar nombres de personas, organizaciones o ubicaciones en un texto. Un ejemplo histórico de aplicación de este enfoque es el sistema ELIZA, creado por Joseph Weizenbaum en la década de 1960, que simulaba una conversación mediante la reescritura de entradas del usuario siguiendo reglas predefinidas (Weizenbaum, 1966).

Las ventajas de los sistemas basados en reglas incluyen su transparencia y facilidad de auditoría, ya que cada decisión puede ser rastreada a una regla

específica, lo que los hace valiosos en aplicaciones críticas como los sistemas médicos o jurídicos. Sin embargo, enfrentan limitaciones significativas en términos de escalabilidad y flexibilidad, ya que crear y mantener un conjunto completo de reglas que pueda abarcar todas las variaciones y complejidades del lenguaje humano es extremadamente desafiante. Además, estos sistemas son propensos a errores si las entradas no se ajustan exactamente a las expectativas codificadas en las reglas (Lees et al., 1957).

Aunque los métodos basados en reglas han sido superados por enfoques estadísticos y de aprendizaje automático en muchas aplicaciones modernas de NLP, siguen siendo relevantes, especialmente en áreas que requieren un alto grado de explicabilidad.

Sin embargo, para un contexto de detección de emociones no son la mejor opción dada su limitada capacidad para manejar la variedad y sutileza del lenguaje natural en contextos complejos y variados.

ii. Técnicas estadísticas

Las técnicas estadísticas de NLP representan una evolución significativa desde los métodos basados en reglas, ofreciendo una mayor adaptabilidad y capacidad para manejar datos complejos y voluminosos. Utilizan modelos matemáticos para aprender la estructura y el significado del lenguaje basándose en el principio de que las características del lenguaje pueden modelarse eficazmente mediante probabilidades. Entre estas técnicas, el Modelo de Bolsa de Palabras (Bag of Words, BoW), TF-IDF y Latent Dirichlet Allocation (LDA) son algunos de los modelos más influyentes y utilizados para tareas de clasificación y análisis temático.

- ***Modelo de Bolsa de Palabras (BoW)***

El modelo BoW es una representación simplificada del texto que convierte el texto en un conjunto de términos, descartando cualquier información sobre el orden o la estructura gramatical de las palabras, pero manteniendo su frecuencia en el documento. En esencia, BoW transforma texto en un vector de números donde cada número representa la frecuencia de una palabra específica en el documento. A pesar de su simplicidad, BoW ha sido fundamental para numerosas aplicaciones prácticas en NLP, como la clasificación de documentos y el filtrado de spam.

- ***TF-IDF (Frecuencia de Término - Frecuencia Inversa de Documento)***

TF-IDF es una técnica que pondera la importancia de cada palabra en un documento en relación con una colección de documentos o corpus. El valor TF-IDF aumenta proporcionalmente al número de veces que una palabra aparece en el documento, pero se compensa por la frecuencia de la palabra en el corpus, lo que ayuda a ajustar por el hecho de que algunas palabras aparecen más frecuentemente en general. Este método es especialmente útil en sistemas de recuperación de información y en la clasificación de documentos donde la relevancia de los términos es crucial para identificar y clasificar contenidos (Salton et al., 1986).

- ***LDA (Latent Dirichlet Allocation)***

LDA es un modelo generativo probabilístico que asume que los documentos son una mezcla de temas y que cada tema es una mezcla de palabras. LDA es útil para la exploración de grandes volúmenes de texto, ya que puede identificar temas latentes en el material y agrupar documentos similares basados en el contenido semántico compartido, más allá de la simple coincidencia de palabras específicas (Blei et al., 2003).

A pesar de su utilidad, tanto BoW como TF-IDF y LDA tienen limitaciones. BoW y TF-IDF pueden fallar en captar el contexto completo en el que se usan las palabras, ya que no consideran la posición o la gramática de las palabras en el texto. LDA, siendo un modelo probabilístico, puede resultar en sobreajustes o subajustes dependiendo de cómo se configuren los parámetros lo que lo hace desafiante, además es subjetivo respecto a la interpretación de los temas resultantes, dado que no retorna las etiquetas de los temas encontrados.

Por lo tanto, para la identificación de emociones en textos cortos, técnicas como BoW, TF-IDF y LDA proporcionan herramientas valiosas para el preprocesamiento y la exploración temática. Sin embargo, su utilidad puede ser limitada sin integrar técnicas más avanzadas.

iii. Técnicas de aprendizaje automático

Las técnicas de aprendizaje automático en NLP utilizan modelos matemáticos y estadísticos para aprender a partir de datos de texto, ofreciendo un enfoque más flexible y potente que los métodos estadísticos básicos. Los modelos se adaptan especialmente bien para tareas donde los patrones de uso del lenguaje pueden identificarse y usarse para hacer predicciones o clasificaciones.

Modelos de Aprendizaje Supervisado.

El aprendizaje supervisado en NLP implica modelos que aprenden a partir de un conjunto de datos etiquetado, donde cada muestra de texto viene con una etiqueta o respuesta correcta que el modelo intenta predecir.

- **Regresión Logística:**
  Utilizada comúnmente para la clasificación binaria o multinomial, la regresión logística modela la probabilidad de que una entrada pertenezca a una categoría particular. En NLP, este modelo es efectivo para clasificar textos en categorías como positivo o negativo. Su simplicidad y eficacia la hacen

popular, especialmente en conjuntos de datos donde las relaciones entre las características son lineales (Scott et al., 1991).

- **Máquinas de Soporte Vectorial (SVM):**
  Las SVM son particularmente robustas en espacios de alta dimensión, como los que se encuentran en NLP. Operan encontrando el hiperplano que mejor divide las clases de datos en el espacio de características. Para NLP, las SVM se utilizan para distinguir entre diferentes categorías de texto basándose en la presencia de palabras y frases, siendo capaces de manejar la no linealidad mediante el uso de kernels (Cortes et al., 1995).

- **Árboles de Decisión y Modelos de Ensamble:**
  Los árboles de decisión clasifican los datos aprendiendo una serie de reglas inferidas de las características. Son transparentes y fáciles de interpretar, pero individualmente pueden ser propensos a sobreajustar los datos. Por ello, se combinan en métodos de ensamble como Random Forests o Gradient Boosting Machines, que mejoran la robustez y precisión mediante la agregación de múltiples árboles de decisión para formar un modelo más poderoso y generalizable (Breiman, 2001).

Modelos de Aprendizaje No Supervisado

En contextos donde no se dispone de datos etiquetados, el aprendizaje no supervisado puede descubrir estructuras ocultas en los datos de texto. Algunos de estos modelos son k-means o clustering jerárquico, utilizados para agrupar textos en categorías basadas en su similitud. No se contemplan en este proyecto, ya que los datos con los que se van a comparar los modelos tienen etiquetas.

iv. Técnicas con Redes Neuronales:

Las técnicas de redes neuronales y deep learning han transformado el campo del NLP, permitiendo modelos que pueden aprender y modelar complejidades del lenguaje de formas que los métodos tradicionales no pueden. Estos modelos capturan contextos y sutilezas lingüísticas, esenciales para una amplia variedad de aplicaciones de NLP.

- ***Representación de Textos en Modelos de Deep Learning:***
  Antes de procesar texto con modelos de deep learning, las palabras se transforman en vectores numéricos utilizando métodos como:

  a) **Codificación One-Hot:** Cada palabra se representa como un vector único en el cual solo un elemento es "1" y todos los demás son "0". Este método es directo, pero genera vectores de alta dimensionalidad y no captura relaciones semánticas entre palabras.

  b) **Embeddings de Palabras:** Métodos como Word2Vec (Pascanu et al., 2013) y GloVe (Pennington et al., 2014) proporcionan una representación densa y de baja dimensión que refleja relaciones semánticas y sintácticas. Estos embeddings posicionan palabras con significados similares cerca unas de otras en el espacio vectorial, ofreciendo una base rica para modelos más complejos.

- ***Procesamiento de Texto con Redes Neuronales:***
  Diferentes tipos de redes neuronales se utilizan para procesar vectores de texto, cada uno adaptado a tareas específicas de NLP:

  a) Redes Neuronales Convolucionales (CNNs): Inicialmente diseñadas para el procesamiento de imágenes, las CNNs también son efectivas en NLP para detectar patrones locales en los datos de texto. Las CNNs aplican filtros a los embeddings de palabras para extraer características

útiles para tareas como clasificación de textos y análisis de sentimientos (Kim, 2014).

b) **Redes Neuronales Recurrentes (RNNs):** Las RNNs son adecuadas para datos secuenciales como el texto. Mantienen un estado que se actualiza secuencialmente mientras procesan cada palabra, lo que les permite recordar información a lo largo del tiempo. Sin embargo, las RNNs tradicionales enfrentan problemas como el desvanecimiento del gradiente cuando se procesan secuencias largas, lo que significa que su memoria es de corto plazo (Pascanu et al., 2013).

c) **LSTM y GRU:** Las Long Short-Term Memory (LSTM) y las Gated Recurrent Units (GRU) son variantes de las RNNs que incluyen puertas para controlar el flujo de información. Estas estructuras permiten que el modelo retenga información relevante a largo plazo y descarte la que no es relevante, mejorando la capacidad de procesar y generar texto coherente (Hochreiter et al., 1997;Cho et al., 2014).

Estas redes neuronales pueden ser muy potentes, pero requieren de grandes cantidades de datos para su entrenamiento, adicionalmente suelen ser computacionalmente intensivas porque se debe iterar el proceso de aprendizaje muchas veces.

v. Técnicas de Modelos Largos de Lenguaje (LLM):

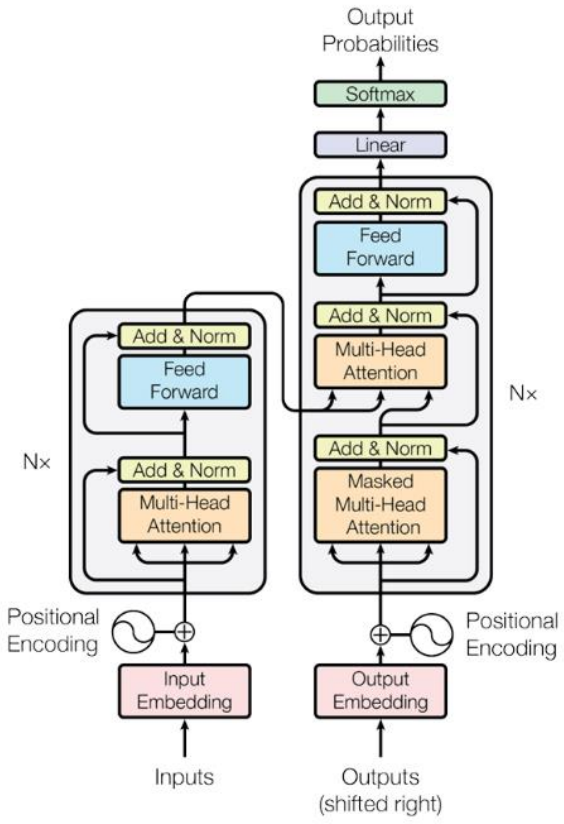
a) Modelos de Transformers.

Los modelos de Transformers, introducidos en el artículo "Attention is All You Need" de Vaswani et al. (2017) representan un antes y un después en el procesamiento del lenguaje natural al introducir una arquitectura basada

completamente en mecanismos de atención, sin depender de las recurrentes redes neuronales que eran comunes hasta ese momento, lo que permite que sean entrenados en pocos pasos y sean fácilmente paralelizados en múltiples GPU's. Estos modelos han revolucionado las tareas de NLP debido a su capacidad para manejar secuencias de datos de manera paralela y efectiva, permitiendo un entrenamiento más rápido y eficiente sobre grandes conjuntos de datos.

La arquitectura original del modelo fue propuesta para tareas de traducción y se representa en la siguiente figura.

Figura 1: Arquitectura Transformers



Fuente: Figura tomada de Attention is all you need (Vaswani et al., 2017).

La arquitectura se divide en dos componentes principales: el encoder y el decoder. Ambos componentes utilizan una combinación de capas de atención multi-head y redes feed-foward, junto con normalizaciones y conexiones residuales para mejorar el flujo de gradiente durante el entrenamiento.

A un nivel general, el modelo Transformer para tareas de traducción automática funciona alimentando el modelo con pares de frases que significan lo mismo en idiomas diferentes. El encoder recibe como entrada oraciones en el idioma de origen (por ejemplo, inglés) y el decoder recibe las correspondientes traducciones en el idioma destino (por ejemplo, español), añadiendo un token inicial SOS a cada entrada del decoder. Durante el entrenamiento, el encoder transforma cada palabra de la entrada para reflejar su significado exacto en el contexto de la frase, mejorando la precisión de la traducción. En la inferencia, el proceso se realiza palabra por palabra, utilizando las salidas parciales del decoder para generar la traducción final.

El encoder ajusta la representación de cada palabra para capturar su significado contextual, mientras que el decoder trabaja para convertir esta representación en la palabra correspondiente del idioma destino. Cada palabra en la salida del decoder se transforma progresivamente hasta generar la traducción completa, terminando con un token EOS que indica el final de la oración. Este proceso iterativo y acumulativo asegura que la traducción sea coherente y contextualmente apropiada.

A un nivel más detallado, el encoder que se encuentra en la parte izquierda, se encarga de recibir un texto y codificar cada palabra en una representación vectorial, que no solo representa la palabra por si sola, sino que captura el contexto de las palabras que están a su alrededor. Tiene los siguientes elementos:

- **Input Embeddings**: Este proceso transforma las palabras de entrada en vectores numéricos. Los modelos de deep learning requieren datos en

forma numérica para procesar, y los embeddings convierten cada palabra en un vector que captura aspectos del significado y uso de esa palabra en diferentes contextos.

- **Positional Encodig**: A los embeddings se les añade información sobre la posición de cada palabra en la secuencia. Como los Transformers no procesan las palabras en orden secuencial, este paso es crucial para que el modelo entienda el orden original de las palabras y cómo cada palabra se relaciona con las demás en la frase o párrafo.

- **Multi-Head Attention**: En esta capa, el modelo procesa los embeddings mejorados con un mecanismo llamado atención multi-cabeza. Cada "head" de atención realiza una operación de atención independientemente, y los resultados se combinan. Este mecanismo permite al modelo enfocarse en diferentes partes del texto de entrada de manera simultánea para capturar diversos aspectos contextuales. Esencialmente, ayuda al modelo a decidir qué palabras son importantes para entender mejor los aspectos contextuales de la secuencia.

- **Add & Norm**: Después de la atención, cada subcapa (atención y luego red feed-forward) en el encoder tiene una conexión residual alrededor de ella seguida de una normalización. Esto significa que la salida de cada subcapa es sumada con su entrada, y este resultado es normalizado. Esto ayuda a evitar problemas en el entrenamiento de redes profundas, permitiendo que los gradientes fluyan bien a través de la red durante el aprendizaje.

- **Feed Forward (Encoder):** Cada capa del encoder también incluye una pequeña red neuronal feed-forward que procesa secuencialmente cada

posición de la entrada por separado. Esto proporciona una capa adicional de transformación y abstracción de los datos.

En la parte derecha se encuentra el decoder, este se encarga de recibir como input el vector generado por el encoder y otra secuencia de texto (la traducción de la secuencia ingresada al encoder) con el objetivo de decodificar el vector nuevamente en texto. Está construido con los siguientes elementos:

- **Output Embedding & Positional Encoding (Decode**r): Similar al encoder, el decoder transforma las palabras de salida (hasta el momento) en vectores y les añade positional encoding. Esto prepara los datos de salida para ser procesados de manera similar a los datos de entrada.

- **Masked Multi-Head Attention (Decoder):** Esta capa es similar a la multi-head attention del encoder, pero se añade una máscara para evitar que las futuras posiciones influyan en la predicción de la posición actual. Esto es esencial para asegurar que la predicción de cada palabra solo pueda utilizar la información de las palabras anteriores, manteniendo el proceso de generación de texto coherente y cronológico.

- **Multi-Head Attention to Encoder Output:** Aquí, el decoder no solo considera su propio output hasta el momento, sino que también observa la salida completa del encoder. Esto le permite al decoder centrarse en las partes relevantes de la entrada mientras genera cada palabra siguiente, lo que es crucial para tareas como la traducción, donde la salida debe corresponder estrechamente a la entrada.

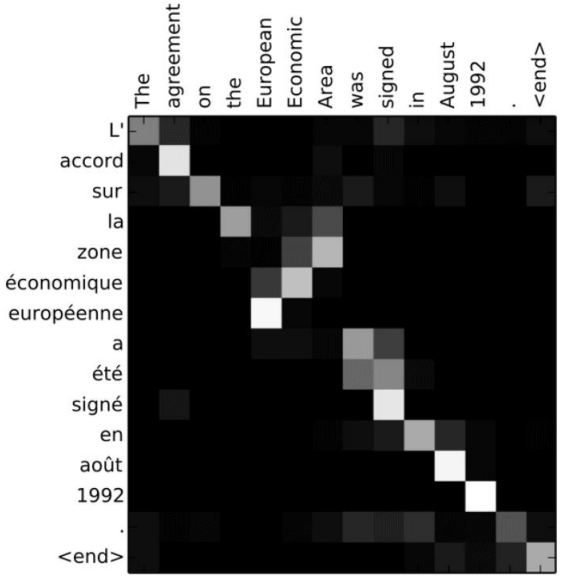
- **Feed Forward (Decoder):** Al igual que en el encoder, después de procesar la atención, cada capa del decoder pasa los datos a través de su propia red feed-forward para una transformación adicional.

- **Linear & Softmax:** Finalmente, la salida del último decoder se transforma a través de una capa lineal y luego se pasa por una función softmax para convertirla en una distribución de probabilidad sobre posibles palabras siguientes. Esto determina la palabra más probable que sigue en la secuencia de salida.

Si se analiza cada uno de los componentes de la arquitectura, los mecanismos de atención fueron la novedad respecto a las técnicas de Redes Neuronales.

La atención le asigna un peso a cada palabra que compone una oración, lo que le permite al modelo prestar especial atención a las palabras más importantes y descartar las que no son significativas para cumplir su objetivo. La siguiente figura muestra un ejemplo tomado del libro de Tunstall et al. (2022), que ilustra los pesos para una traducción de inglés a francés, hay que destacar que el decoder alinea correctamente las palabras y les asigna un peso representado por el color del píxel.

Figura 2: Alineación de palabras en inglés y su traducción generada en francés.



Fuente: Figura tomada de *Natural Language Processing with Transformers* (Tunstall et al., 2022, Cap. 1, p. 5)

Además, los mecanismos de self-attention permiten que los modelos presten atención a todas las entradas de la misma capa simultáneamente, eliminando la necesidad de procesar secuencialmente como en las RNN. Esto hace posible entrenar los modelos de Transformers de manera más rápida y eficiente, ya que las operaciones de atención pueden ser paralelizadas a lo largo de la secuencia completa (Tunstall et al., 2022). Esto es una gran ventaja sobre los modelos recurrentes, que requieren iteraciones múltiples y no pueden ser paralelizados de la misma manera.

Aunque en la arquitectura original se utilizan en conjunto los encoders y los decores, estos se pueden utilizar para tareas de NLP de manera independiente, lo que dio paso a una avalancha de diferentes modelos basados en atención.

b) Modelos Encoders

Los Modelos encoders o también conocidos como auto-encoding models se caracterizan por tener atención bi-direccional, esto quiere decir que el vector que representa una palabra, captura el contexto de las demás palabras que se encuentran a la izquierda y a la derecha. Su entrenamiento usualmente se basa en ocultar una palabra y pedir al modelo que construya la oración completa. Este mecanismo los hace buenos para extraer información importante, por lo que son usualmente utilizados para clasificación de texto, preguntas y respuestas o resumen de textos.

El primero modelo encoder fue BERT (Bidirectional Encoder Representations from Transformers), lanzado por Google en 2018 (Devlin et al., 2018) marcó un hito importante en el campo del procesamiento del lenguaje natural. Este modelo utiliza una arquitectura de Transformer exclusivamente en la sección del encoder e introduce un enfoque bidireccional para el entrenamiento previo supervisado, que fue un cambio significativo respecto a la técnica anterior.

BERT se basa en dos tareas principales durante su entrenamiento previo:

- **Modelo de Lenguaje Enmascarado (MLM):** En esta tarea, alrededor del 15% de las palabras en cada oración se enmascaran de manera aleatoria, y el objetivo del modelo es predecir estas palabras ocultas. Por ejemplo, si la oración original es “Ella se divirtió en la fiesta de cumpleaños”, entonces el modelo crea la secuencia “Ella <mask> divirtió en la <mask> de cumpleaños” y debe predecir las palabras “se” y “fiesta” respectivamente. Esto permite que el modelo aprenda un con texto bidireccional, entendiendo el texto desde ambos lados de la palabra enmascarada.

- **Predicción de Siguiente Oración (NSP):** En esta tarea, se le presenta al modelo pares de oraciones y debe predecir si la segunda oración en el par es la consecutiva de la primera en el texto original. Por ejemplo, debe predecir que “El semáforo está en rojo” y “Puede cruzar la calle” son consecutivas, mientras que “El semáforo está en rojo” y “La tierra gira alrededor del sol” no son consecutivas. Esta tarea ayuda a BERT a entender las relaciones entre oraciones, aunque investigaciones posteriores sugirieron que esta tarea no era tan crucial como se pensaba inicialmente.

Durante el entrenamiento, BERT alterna entre estas dos tareas, refinando sus capacidades de comprensión del texto en un nivel profundo y contextual. El modelo se entrena primero en un vasto corpus de texto de manera no supervisada, lo que significa que aprende solo del texto sin etiquetas humanas. El diseño bidireccional de BERT es lo que realmente establece su distinción, permitiendo que cada palabra se procese en el contexto de todas las demás palabras en una oración, no solo las que la preceden, lo que le da una comprensión más rica y matizada del lenguaje.

La introducción de BERT marcó un hito histórico debido a su impacto significativo en la mejora del rendimiento en muchas tareas de NLP. Estos avances han

demostrado ser superiores en el análisis de texto, especialmente en la detección de emociones, debido a su capacidad para manejar las sutilezas del lenguaje y reconocer un espectro más amplio de emociones humanas (Cortiz, 2021; Prottasha et al., 2022).

Su enfoque ha establecido un nuevo estándar para cómo los modelos de lenguaje son desarrollados, a tal punto de ser la base para múltiples modelos con variantes de BERT, como ALBERT, RoBERTa, DistilBERT y muchos más.

c) Modelos Decoders.

Son muy similares a los encoders, pero difieren principalmente en el mecanismo de atención. En los decores se usa “masked self-attention”, mecanismo que oculta los valores de las palabras a la derecha. Por lo tanto, cada vector representa una palabra y el contexto que tiene esa palabra respecto a las demás que se encuentran a la izquierda. El entrenamiento de estos modelos usualmente se basa en predecir la siguiente palabra en una oración, por ejemplo, dada la entrada “me gusta el fútbol”, se enmascara la secuencia como “me gusta el <mask>” y debería predecir “futbol”. Estos modelos son autorregresivos, quiere decir que la palabra que predicen en una iteración se utiliza como input para predecir la siguiente, por ejemplo, después de predecir “me gusta el fútbol”, sigue iterando hasta completar la oración “me gusta el futbol solo cuando gana mi equipo”.

Dada su forma de entrenarse, son especialmente buenos para generar secuencias de texto. Algunos ejemplos de estos modelos son GPT-2 y GPT Neo.

d) Transfer Learning y Fine-tuning.

A pesar de que estos modelos no son recurrentes y se pueden entrenar en pocos pasos, necesitan de un enorme volumen de información para aprender las sutilezas del lenguaje humano. Por lo tanto, entrenar un modelo de estos desde

cero, puede costar miles o millones de dólares porque se requiere de hardware potente y puede tardar semanas o meses para su entrenamiento.

En este contexto, el *transfer learning* se presenta como una solución eficaz para abordar estos desafíos. El *transfer learning* se hace posible gracias a que grandes compañías invierten en el entrenamiento de LLM y luego hacen públicos los parámetros del modelo pre-entrenado. Esto permite que investigadores y desarrolladores puedan utilizar estos modelos avanzados en sus casos particulares.

En caso de que los parámetros del modelo pre-entrenado no den buenos resultados, es posible adaptarlos a tareas específicas mediante fine-tuning con conjuntos de datos más pequeños y especializado en la aplicación objetivo. Técnicamente, se preservan los conocimientos generales que el modelo ha aprendido de grandes cantidades de datos, mientras que el fine-tuning permite que los pesos de las capas del modelo se reconfiguren para captar patrones y relaciones más específicos de la nueva tarea (Tunstall et al., 2022).

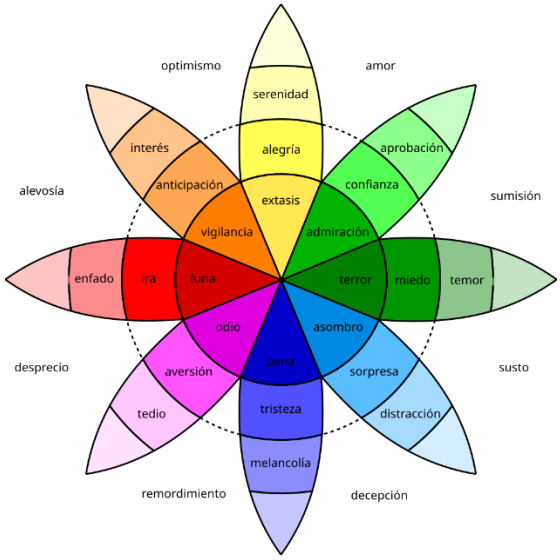
Estos enfoques reducen significativamente la barrera de entrada para trabajar con tecnologías de NLP de última generación, minimizando la necesidad de recursos computacionales extensos y el alto costo que implicaría entrenar estos modelos desde cero (Pan et al., 2010).

DEFINICIÓN DE EMOCIONES EN NLP

Hay un debate en la ciencia afectiva, la neurociencia, la psicología y la filosofía sobre qué constituye una emoción y cuántas existen realmente. Sin embargo, en NLP, a menudo se asume que las emociones se estructuran en un conjunto de emociones básicas. Una de las teorías más reconocidas es la de Robert Plutchik, quien propuso un modelo que incluye un espectro de ocho emociones básicas: alegría, confianza, miedo, sorpresa, tristeza, aversión, ira y anticipación. Este modelo, conocido como la Rueda de las Emociones de Plutchik, sugiere que estas

emociones pueden combinarse de distintas maneras para crear emociones más complejas, proporcionando una base sólida para entender las respuestas emocionales humanas (Plutchik, 1980).

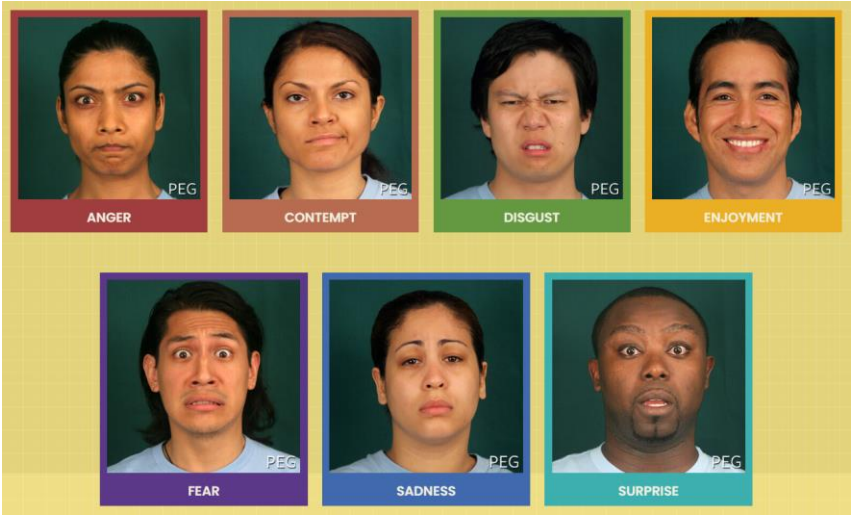
Figura 3: Rueda de las Emociones Plutchik



Fuente: Figura tomada de Wikipedia. (2024, 2 de noviembre). *Robert Plutchik*. Wikipedia. https://es.wikipedia.org/wiki/Robert_Plutchik

Otra propuesta fundamental en el estudio de las emociones es la Teoría básica de las emociones de Paul Ekman (Ekman, 1992), donde se identifica un conjunto de emociones universales que son reconocibles y expresadas de manera similar en todas las culturas. Según esta teoría, hay 7 emociones innatas y biológicamente programadas, lo que implica que todas las personas, independientemente de su cultura, experimentan y expresan estas emociones de maneras similares.

Figura 4: Emociones básicas de Paul Ekman



Fuente: Figura tomada de Paul Ekman Group. (2024, 2 de noviembre). *Universal Emotions*. Paul Ekman. Universal Emotions | What are Emotions? | Paul Ekman Group

Por otra parte, La teoría del espacio semántico, propuesta por Cowen y Keltner (Cowen & Keltner, 2021), adopta un enfoque computacional para explorar una amplia gama de estímulos naturales y utiliza técnicas estadísticas para capturar la variación en los comportamientos emocionales. Esta teoría identifica más de 25 variedades de experiencias emocionales, cada una con diferentes antecedentes y expresiones, y sugiere que las emociones son altamente dimensionales y categóricas, permitiendo una mayor profundidad en el análisis emocional.

ESTADO DE ARTE

Dada de la importancia del NLP, diferentes investigaciones han explorado y comparado varios modelos en función de la precisión en la detección de emociones. En esta sección, se revisan estudios clave que comparan modelos para el reconocimiento de emociones utilizando datasets específicos, usando como referencia "Exploring Transformers in Emotion Recognition: a comparison of BERT, DistillBERT, RoBERTa, XLNet and ELECTRA" (Cortiz, 2021).

Polignano et al., (2019) propusieron un enfoque basado en redes neuronales profundas, combinando BiLSTM, CNN y una capa de autoatención para capturar relaciones distantes entre palabras. Utilizaron embeddings preentrenados como GoogleEmb, GloVe y FastText para mejorar la generalización del modelo. Los resultados mostraron que FastText proporcionó el mejor desempeño en la clasificación de emociones utilizando el dataset ISEAR (Scherer & Wallbott, 1994). En la comparación de modelos, el enfoque BiLSTM+CNN+Self-Attention con embeddings FastText alcanzó un Macro-F1 de 0.63, superando a otros modelos como SVM y Random Forest.

Figura 5: Resultados de la investigación de Polígono et al. (2019) sobre conjunto de datos ISEAR

Model	Embeddings	Macro-F1
SVM	-	0.55
Random Forest	-	0.49
BiLSTM+CNN+ Self-Attention	GoobleEmb	0.62
	GloVE	0.62
	FastText	0.63

Fuente: Figura tomada de *Exploring Transformers in Emotion Recognition: a comparison of BERT, DistillBERT, RoBERTa, XLNet and ELECTRA. (Cortiz, 2021).*

Batbaatar et al. (2019) desarrollaron una arquitectura llamada SENN (Semantic-Emotion Neural Network) que combina BiLSTM para capturar información semántica y CNN para extraer características emocionales. Utilizaron embeddings

enriquecidos emocionalmente (EWE) además de Word2Vec, GloVe y FastText. SENN mostró un mejor rendimiento que los modelos basados en embeddings generales, utilizando el dataset ISEAR. En este estudio, el modelo SENN con GloVe+EWE obtuvo un Macro-F1 de 0.746, destacándose sobre las demás combinaciones de embeddings.

Figura 6: Resultados de la investigación de Barbaatar et al. (2019) sobre conjunto de datos ISEAR

Model	Embeddings	Macro-F1
SENN	Word2Vec + EWE	0.737
	GloVe + EWE	0.746
	FastText + EWE	0.745

Fuente: Figura tomada de *Exploring Transformers in Emotion Recognition: a comparison of BERT, DistillBERT, RoBERTa, XLNet and ELECTRA. (Cortiz, 2021).*

Adoma et al. (2020) realizaron un análisis comparativo de modelos basados en transformers (BERT, RoBERTa, DistilBERT y XLNet) para la tarea de reconocimiento de emociones utilizando el dataset ISEAR. Los resultados indicaron que RoBERTa superó a los demás modelos en términos de precisión y F1-score. RoBERTa obtuvo un Macro-F1 de 0.742, seguido por XLNet con 0.731, BERT con 0.702 y DistilBERT con 0.693. Los autores concluyeron que RoBERTa fue el modelo más efectivo en la detección de emociones en textos, aunque DistilBERT resultó ser el modelo más rápido en términos de tiempo de entrenamiento y evaluación.

Figura 7: Resultados de la investigación de Adoma et al. (2020) sobre conjunto de datos ISEAR

Model	Macro-F1
BERT	0.702
RoBERTa	0.742
DistilBERT	0.693
XLNet	0.731

Fuente: Figura tomada de *Exploring Transformers in Emotion Recognition: a comparison of BERT, DistillBERT, RoBERTa, XLNet and ELECTRA. (Cortiz, 2021).*

Además, Demszky et al. (2020) presentaron el dataset GoEmotions, que incluye 58 mil comentarios de Reddit anotados para 27 categorías emocionales. Utilizaron modelos BiLSTM y BERT para evaluar el rendimiento en la clasificación de emociones. BERT mostró un mejor desempeño con un F1-score promedio de 0.46, superando significativamente al modelo BiLSTM.

Finalmente, Cortiz (2021) llevaron a cabo un análisis comparativo similar, pero utilizando modelos BERT, RoBERTa, DistilBERT y XLNet con el dataset GoEmotions. Los resultados mostraron que RoBERTa alcanzó el mejor F1-score (0.49), seguido por DistilBERT y XLNet (ambos con 0.48), y BERT con un F1-score de 0.46. En términos de tiempo de entrenamiento, DistilBERT fue el modelo más rápido, mientras que XLNet fue el más lento.

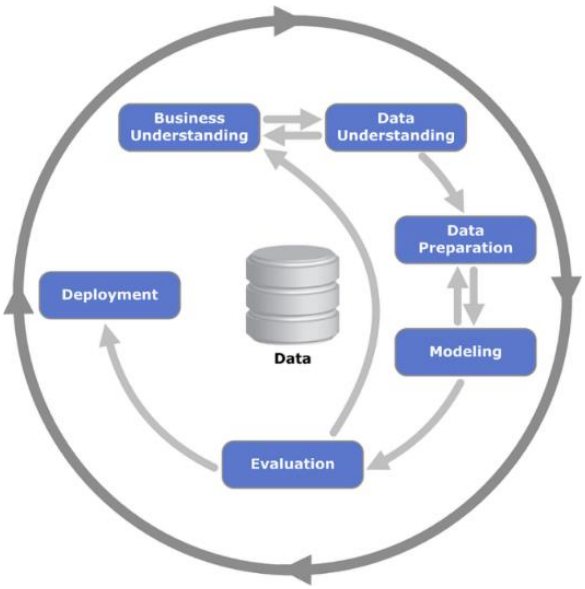
Lo anterior refleja una característica común en los proyectos de NLP que abordan las emociones: la suposición de que las emociones están estructuradas en un conjunto de emociones básicas, generalmente entre 6 y 8 categorías. Aunque en algunos casos se han utilizado la teoría constructivista que clasifica las emociones en más de 25 categorías. También se destaca que los modelos basados en Transformers han demostrado ser superiores en la tarea de clasificación de emociones en texto. Modelos como RoBERTa, BERT y XLNet han mostrado consistentemente un buen desempeño, destacándose RoBERTa por su precisión y DistilBERT por su eficiencia en términos de tiempo de entrenamiento.

DISEÑO METODOLÓGICO

En esta sección de describe la metodología de trabajo, el conjunto de datos y los pasos que se siguen para llevar a cabo el estudio de comparación de diferentes LLM en la tarea de reconocimiento de emociones en textos cortos.

Se usa como guía para este proyecto la metodología CRISP-DM (Cross Industry Process for Data Mining) dado que es típicamente utilizada en proyectos de Ciencia de Datos. Dicho marco de trabajo se representa en la figura 3 y está compuesto por las siguientes fases:

Figura 8: Diagrama metodología CRISP-DM



Fuente: Figura tomada de Wikipedia. (2023, 2 de noviembre). *Cross Industry Standard Process for Data Mining*. Wikipedia. https://es.wikipedia.org/wiki/Cross_Industry_Standard_Process_for_Data_Mining

i. Entendimiento del Negocio:

Tal y como se mencionó en secciones anteriores, se identifica que los modelos convencionales para realizar NLP fueron superados por los LLM dada su capacidad superior de capturar contexto en las palabras, se obtienen modelos con una mayor precisión para identificar emociones en textos. Por lo tanto, se requiere comparar diferentes LLM en la identificación de emociones en texto para encontrar el que mejor se desempeñe en términos de diversas métricas de evaluación.

ii. **Entendimiento de los Datos:**

- **Descripción de los datos:**

Se usa el conjunto de datos ISEAR (International Survey on Emotion Antecedents and Reactions) publicado por Scherer et al. (1994). Consiste en una encuesta realizada en los años 90s en 37 países distribuidos en todo el mundo. Cada una de sus 7.666 observaciones corresponde a una descripción textual de una situación y la emoción correspondiente que experimentó el individuo. Estas emociones se clasifican de una manera básica en: alegría, miedo, ira, tristeza, disgusto, vergüenza o culpa.

- **Implicaciones del idioma del conjunto de datos:**

El hecho de que el conjunto de datos esté en inglés podría tener algunas implicaciones dependiendo del tipo de modelo que se utilice. Con técnicas tradicionales de aprendizaje automático, los modelos solo pueden predecir textos en el mismo idioma con el que fue entrenado, por lo tanto, sería poco preciso para detectar emociones en textos en español u otro idioma diferente al inglés. Por otra parte, si se utilizan LLM, es muy probable que el modelo también tenga buen

desempeño con textos en idiomas diferentes al inglés, dado que su base de entrenamiento son extensos cuerpos de texto en múltiples idiomas.

- **Plan de gestión de datos:**

El conjunto de datos ISEAR ha sido puesto a disposición de la comunidad por lo que no requiere de licenciamiento o permisos de uso específicos. Desde un punto de vista ético, estos datos son anónimos y no se va a hacer ningún esfuerzo para tratar de obtener información adicional sobre los encuestados.

Los datos serán usados únicamente en la etapa de entrenamiento y comparación de modelos, por lo que no requieren de un almacenamiento especial que mantenga su disponibilidad en el tiempo. Sin embargo, se agregarán al repositorio de GitHub del proyecto para asegurar su disponibilidad y facilitar su consulta, uso por otros investigadores y reproducibilidad de los experimentos.

Para la evaluación cualitativa, se tomarán tweets en inglés donde sea evidente la emoción del autor. Solo se usará el cuerpo del tweet, por lo que no se va a adquirir y almacenar ningún tipo de información personal sobre usuarios. Adicionalmente, se tienen en cuenta los términos y condiciones de X (antes twitter), por lo que no se hará web scrapping y, en caso de que el prototipo funcional implementado escale y se utilice en un entorno productivo para clasificar un volumen considerado de tweets, se utilizará la API oficial de la compañía asumiendo sus costos asociados.

iii. **Preparación de los Datos:** En esta etapa se realiza una exploración de los datos con el objetivo de identificar y limpiar errores. Además, se debe asegurar que los datos estén en el formato adecuado para que puedan

ser procesados por cada modelo, para esto, la secuencia de texto debe ser convertida en vectores a través de la tokenización.

iv. **Modelado:** Se realiza Transfer Learning y Fine-Tuning sobre diferentes modelos de uso abierto y de uso por pago sobre el conjunto de datos ISEAR. El fine-tuning se realiza solo para ajustar los parámetros de los modelos que son viables en términos de recursos computacionales y disponibilidad. Algunos de los modelos evaluados son DistilBERT, XLNet, ChatGPT, Claude, Gemini y otros.

v. **Evaluación:** Se evalúa el rendimiento de los modelos utilizando como métrica el F1-score. Este análisis permite identificar cuál de los modelos es más efectivo en términos de precisión y eficiencia para la clasificación de emociones en textos cortos. El F1-score es particularmente útil en tareas de clasificación multiclase, ya que proporciona una "evaluación equilibrada de la precisión y el recall, lo cual es esencial en aplicaciones donde es igual de importante evitar falsos positivos y falsos negativos" (Sokolova & Lapalme, 2009). Adicionalmente se realiza una evaluación cualitativa donde a partir de un prototipo funcional se validan las respuestas del mejor modelo con textos de 5 usuarios de prueba.

vi. **Despliegue:** Se desarrolla un Producto Mínimo Viable (MVP) utilizando el modelo que demuestre el mejor rendimiento para permitir pruebas y evaluaciones cualitativas adicionales, facilitando su utilización para validar su efectividad con datos no observados durante el entrenamiento. Adicionalmente, se hace entrega de este documento detallado del proyecto de investigación, incluyendo todos los hallazgos y conclusiones.

DESARROLLO DEL TRABAJO

En esta sección se describe todo el proceso realizado para evaluar el rendimiento de diferentes Modelos grandes de lenguaje (LLM) en la identificación de emociones en texto siguiendo la metodología CRISP-DM. Comenzando por una etapa de comprensión del negocio, donde se identifican los objetivos y la estrategia del proyecto. A continuación, en las etapas de entendimiento y preparación de datos se realiza un Análisis Exploratorio de Datos (EDA) y se aplican técnicas de limpieza. Luego, en la etapa de modelado, se implementan y comparan varios modelos de aprendizaje automático y LLM, evaluando su capacidad en términos de f1-score de clasificar textos en emociones. Finalmente, se despliega un prototipo funcional con el modelo que tiene un mejor desempeño y se realizan pruebas cualitativas para reconfirmar su efectividad antes de ser llevado a un entorno productivo. A continuación, se detalla cada una de las etapas.

ENTENDIMIENTO DEL NEGOCIO

Después de tener claro que los LLM representan un avance significativo para el análisis del lenguaje humano, se plantea que la implementación de estos modelos en un contexto de identificación de emociones en textos agrega valor en industrias interesadas en comprender las emociones que están generando en su público de interés. Por lo tanto, se define el objetivo principal de evaluar diferentes LLM en la identificación de emociones en texto para encontrar el que mejor se desempeñe en términos de f1-score.

Para esto se establece la estrategia de utilizar un conjunto de datos previamente etiquetado para realizar experimentos de aprendizaje supervisado que permitan realizar evaluaciones cuantitativas y cualitativas de los hallazgos realizados.

ENTENDIMIENTO DEL CONJUNTO DE DATOS.

El conjunto de datos seleccionado para realizar este experimento es ISEAR (International Survey on Emotion Antecedents and Reactions) publicado por Scherer et al. (1994). Consiste en una encuesta realizada en los años 90s en 37 países distribuidos en todo el mundo. Cada una de sus 7.666 observaciones corresponde a una emoción presentada al encuestado y a un texto en inglés que corresponde a la situación que el encuestado respondió en la que ha sentido dicha emoción. Estas emociones se clasifican de acuerdo con la teoría básica de las emociones en: alegría, miedo, ira, tristeza, disgusto, vergüenza o culpa. A continuación, se representa una muestra de los datos:

Tabla 1: Muestra del conjunto de datos ISEAR

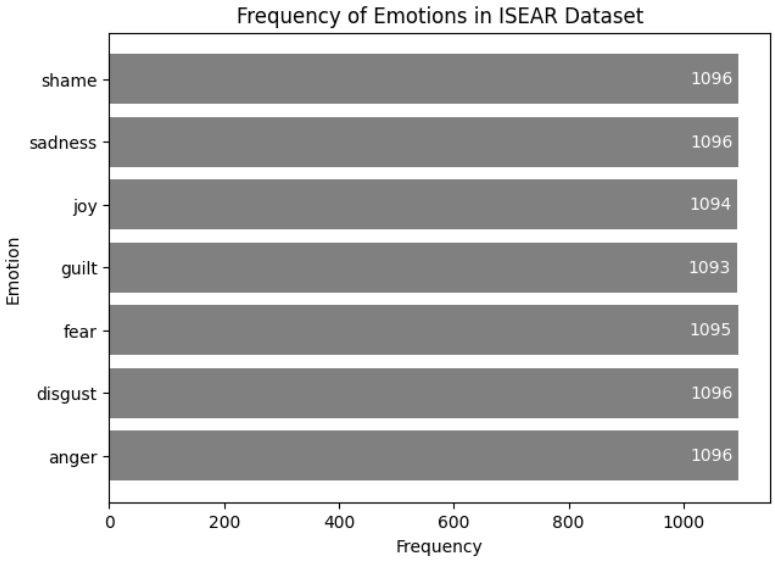
text	**emotion**
When my mother died in August 1978.	sadness
Finding out that I made a stupid mistake in the exam	fear
An unjust calumny concerning me	anger

Fuente: Elaboración propia con datos de ISEAR

Estos datos han sido utilizados en otras investigaciones de NLP por lo que brinda un punto de comparación con otros modelos propuestos para esta tarea por otros autores.

De acuerdo con el Análisis Exploratorio de Datos (EDA) se puede identificar que hay balanceo de clases, dado que cada emoción tiene alrededor de 1095 observaciones, tal y como se muestra en la ilustración 1.

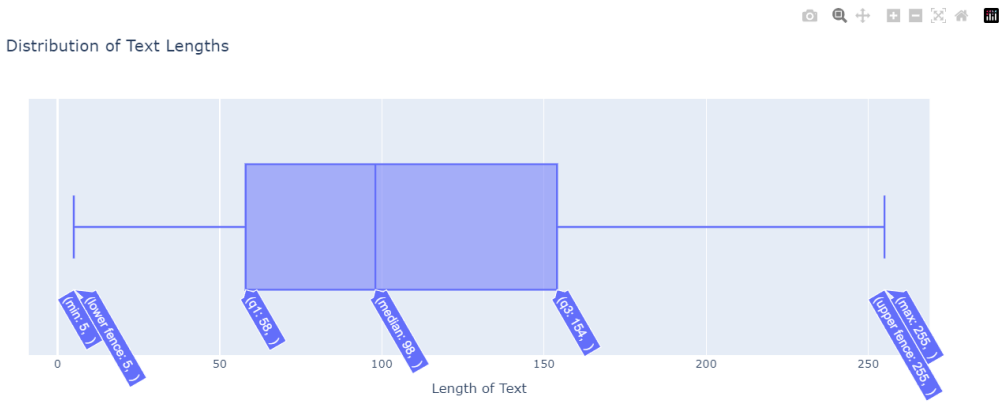
Ilustración 1: Frecuencia de emociones en ISEAR



Fuente: Elaboración propia con datos de ISEAR

Se identifica en la ilustración 2 que la distribución de la longitud de los textos tiene un valor mínimo de 5 caracteres y un valor máximo de 255.

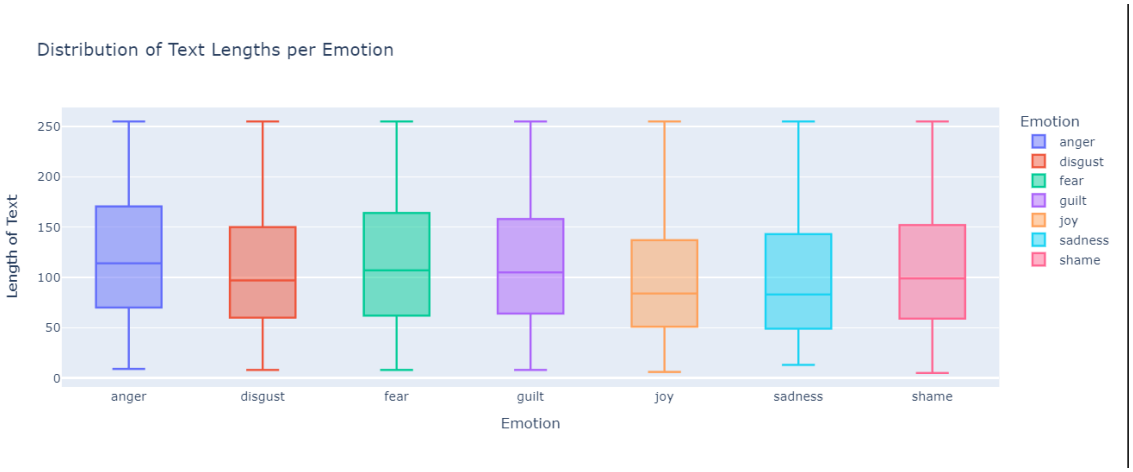
Ilustración 2: Distribución de la longitud de los textos.



Fuente: Elaboración propia con datos de ISEAR

De acuerdo con la ilustración 3, la distribución del número de caracteres por texto es similar para todas las emociones, sin embargo, los textos clasificados como “joy” tienden a tener menos caracteres que los demás.

Ilustración 3: Distribución de la longitud de los textos por emoción.



Fuente: Elaboración propia con datos de ISEAR

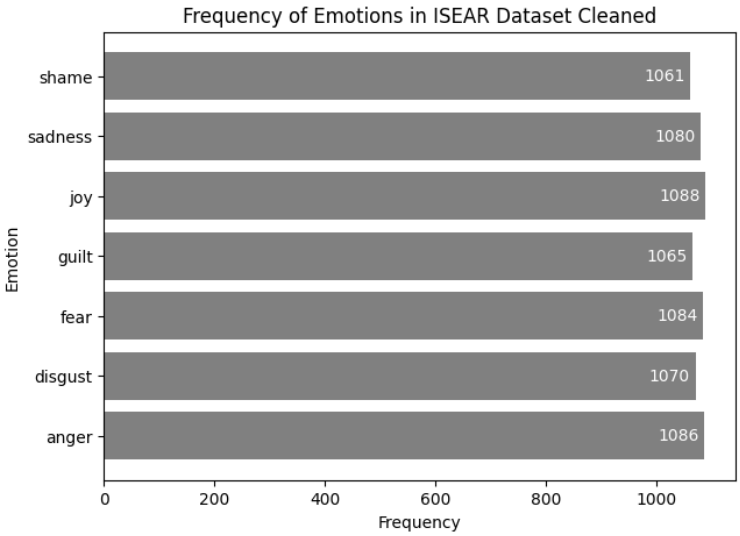
Se evidencia que hay algunas observaciones que tienen muy pocos caracteres como para representar una frase significativa, por lo que se analiza en detalle cada uno de los textos prestando especial atención a los más cortos. Se encuentra que efectivamente algunos textos no tienen sentido, por lo que se procesan en la siguiente etapa del proyecto.

PREPARACIÓN DE DATOS.

Hay algunas observaciones donde el texto no corresponde a una respuesta válida, como, por ejemplo: "NO RESPONSE" o "None". Por lo tanto, se identifican todos los textos inválidos y se eliminan del conjunto de datos. Como resultado de este proceso se eliminan 132 observaciones, pasando de 7666 a 7534, lo que representa una reducción del 1.72% de datos disponibles. Aunque se eliminan más observaciones correspondientes a algunas emociones respecto a otras, se puede

ver en la ilustración 4 que el conjunto de datos sigue balanceado, con observaciones entre 1061 y 1088 para cada emoción.

Ilustración 4: Frecuencia de emociones en ISEAR limpio



Fuente: Elaboración propia con datos de ISEAR

En la siguiente tabla se detalla el número de observaciones eliminadas para cada emoción y su respectiva variación porcentual.

Tabla 2: Detalle de observaciones eliminadas por emoción.

emotion	observaciones antes de limpieza	observaciones después de limpieza	observaciones eliminadas	%variación
anger	1096	1086	10	-0.91%
disgust	1096	1070	26	-2.37%
fear	1095	1084	11	-1.00%
guilt	1093	1065	28	-2.56%
joy	1094	1088	6	-0.55%
sadness	1096	1080	16	-1.46%
shame	1096	1061	35	-3.19%

Fuente: Elaboración propia

Luego de esta limpieza, se realiza ingeniería de caracteres donde se remueven caracteres no útiles como "[]" que se encuentran presentes en algunos textos, se eliminan los "." finales, se elimina “á\n” dado que aparece en cada salto de línea y se transforma todo el texto a minúscula. Estos cambios no afectan significativamente la distribución de la longitud de los textos, pero si impacta de manera positiva la calidad de la información de entrada de los modelos y, por lo tanto, sus resultados en la clasificación.

Finalmente, se toman las 7532 observaciones del conjunto de datos limpio y se divide de manera aleatoria en los subconjuntos de datos de entrenamiento, validación y pruebas. Se realiza una distribución en 80% (6027), 10% (753) y 10% (754) de las observaciones respectivamente. Aunque es un proceso aleatorio, se utiliza una semilla aleatoria para garantizar reproducibilidad y se estratifica por la emoción para mantener el balanceo de clases en cada uno de los subconjuntos.

Estos subconjuntos de datos limpios se persisten en el repositorio del proyecto y se usan en los experimentos de cada uno de los modelos con el fin de que la comparación de los resultados tenga sentido, dado que se elimina la posibilidad de que las diferencias encontradas se puedan explicar porque los subconjuntos de datos utilizados sean diferentes.

MODELADO DE DATOS

En esta etapa se evalúa el rendimiento de algunos modelos de aprendizaje automático y LLM para la clasificación de textos en una de siete posibles emociones (anger, disgust, fear, joy, sadness, shame and guilt). La evaluación se realiza en términos de f1-score utilizando el conjunto de datos ISEAR.

Clasificación de texto en emociones usando modelos de aprendizaje automático.

Los modelos Regresión Logística, Árbol de decisión, Bosques Aleatorios y Support Vector Classifier (SVC) son seleccionados para realizar el experimento de clasificación de modelos de aprendizaje automático, esto brinda un punto de partida para comparar qué tanto mejoran los modelos de LLM. Adicionalmente, se utiliza un modelo dummy de clasificación que siempre asigna la clase más común, para tener una línea base de referencia.

Para que los modelos puedan procesar textos, estos deben vectorizarse, es decir, transformarse a una representación numérica. La técnica que se utilice para la vectorización puede afectar significativamente la capacidad de identificación de patrones en los modelos y, por lo tanto, la efectividad en la clasificación. También afecta si los textos son preprocesados antes de ser vectorizados, por ejemplo, si se decide remover palabras comunes (stop words).

Otros factores que influyen en la capacidad de aprendizaje del modelo sobre los datos es la forma en que se utilicen los subconjuntos de entrenamiento, validación y pruebas y, los hiperparámetros de configuración de los modelos.

En este caso, se usa Grid Search Cross-Validaition para encontrar la combinación óptima de hiperparámetros, es una técnica que combina dos conceptos: Grid Search y Cross-Validaition. Grid Search prueba todas las combinaciones posibles de parámetros entregados por el usuario y evalúa el rendimiento en cada combinación para encontrar la mejor. Cross Validation, se encarga de dividir los datos en subconjuntos (folds), entrenar el modelo en algunos subconjuntos y evaluar en el

restante, repitiendo el proceso hasta que cada subconjunto haya sido utilizado en la evaluación. La ventaja de utilizar esta técnica es que la evaluación en diferentes subconjuntos de datos permite medir la capacidad de generalización, es decir, cómo se comportará en datos no observados.

Dado que esta técnica de optimización se encarga de dividir los datos en entrenamiento y validación de manera automática y aleatoria, se consolidan los subconjuntos de entrenamiento y validación en un solo dataset de entrenamiento con 6.780 observaciones, las 754 observaciones restantes se utilizan para evaluar el rendimiento del modelo entrenado sobre datos nuevos.

Teniendo en cuenta todo lo anterior, para cada uno de los modelos se realiza el proceso de entrenamiento y evaluación utilizando diferentes métodos de vectorización tfidf, word2vec, fasttext, y glove; con preprocesamiento y sin preprocesamiento de texto y diferentes hiperparámetros para encontrar la combinación que optimiza el rendimiento del modelo.

En el caso de preprocesamiento, donde se remueven “stop words”, caracteres especiales y se realiza lematización; siempre se obtenían peores resultados independientemente del método de vectorización. Esto tiene sentido, dado que este preprocesamiento tiende a eliminar sutilezas de los textos, lo que dificulta la capacidad de encontrar factores diferenciadores.

El método de vectorización que demostró mejores resultados en todos los modelos es GloVe. En la siguiente tabla se muestra el detalle de la eficiencia de cada modelo.

Tabla 3: F1-score para modelos de aprendizaje automático con Glove y sin preprocesamiento del texto.

Modelo	Vectorización	Test Macro F1-Score
SVC	glove	0.5103
Logistic Regression	glove	0.4825
Random Forest	glove	0.4363
Decision Tree	glove	0.2817

Dummy Classifier	glove	0.1264

Fuente: Elaboración propia.

El modelo con mejor rendimiento en el experimento de aprendizaje automático fue SVC con los parámetros {‘C': 30, ‘class_weight': None, ‘gamma': 'scale', 'classifier__kernel': 'rbf'} obteniendo un f1-score de 0.5103 sobre el conjunto de datos de pruebas.

Además de estos modelos, se prueba AutoML de Databricks, una herramienta que recibe el conjunto de datos para evaluar de manera automática diferentes modelos y configuraciones de hiperparámetros y retornar el mejor modelo que encuentre con todo el detalle de su proceso de entrenamiento (Microsoft, n.d.). Para problemas de clasificación los modelos que se evalúan son: Árboles de Decisión, Bosques aleatorios, Regresión logística, XGBoost y LightGBM. En este caso, el mejor modelo encontrado fue Regresión Logística con los parámetros {‘C': 1.0892186949935163, ‘l1_ratio’: 5.526615652657683e-08, ‘penalty: 'elasticnet', ’solver’: ‘saga’} obteniendo un f1-score de 0.53 sobre el conjunto de datos de pruebas.

Estos resultados de 0.51 y 0.53 son consistentes con los resultados mencionados en el estado del arte. Otros autores han realizado estudios similares sobre el mismo conjunto de datos y obtuvieron un f1-score de 0.49 con un modelo de Bosques aleatorios y 0.55 con un modelo de Máquinas de Soportes Vectoriales.

En conclusión, los modelos de aprendizaje automático no son los más eficientes para clasificar textos en emociones, dado que un f1-score de 0.53 se encuentra lejos de los mejores resultados obtenidos en los estudios mencionados en el estado del arte con LLMs (0.74).

Clasificación de texto en emociones usando LLMs.

Una de las características de los LLMs es que están entrenados sobre un gran conjunto de datos, lo cual es costoso y puede tomar semanas o meses si se decide hacer desde cero. Esto causa que la manera más ágil y común de aplicar LLMs es

utilizar modelos preentrenados que se ponen a disposición de la comunidad con uso gratuito o de pago por parte de sus creadores.

Existen múltiples formas de aplicar LLMs preentrenados a problemas de clasificación de texto en emociones, entre ellos Prompt Engineering, Retrival Augmented Generation (RAG), Transfer Learning y Fine-Tuning.

La forma y el costo de utilizar estos modelos varía dependiendo de la forma en que los desarrolladores publiquen o den acceso a sus modelos.

En esta etapa del proyecto, se evalúa el rendimiento de múltiples modelos de uso abierto y de pago para la clasificación de textos en emociones utilizando técnicas de Promt Engineering y Fine-Tuning. En la tabla 4 se detallan los modelos a evaluar.

Tabla 4: Detalle de modelos de uso abierto y de pago a evaluar.

Uso	Proveedor	Modelo
Abierto	DistilBERT community	distilbert
Abierto	FacebookAI	roberta
Abierto	FacebookAI	xlm-roberta
Abierto	Meta	llama
Abierto	XLNet Community	xlnet
De pago	Anthropic	claude
De pago	Google	gemini
De pago	OpenAI	gpt

Fuente: Elaboración propia.

Prompt Engineering.

En esta sección se evalúan varias versiones de los modelos llama, Claude, Gpt y Gemini para clasificar textos en emociones a partir de la construcción de un prompt sobre el modelo sin realizarle ninguna modificación a los parámetros preentrenados. Para que tenga sentido comparar la métrica con los demás modelos de este experimento se debe utilizar el mismo conjunto de datos, es decir, las 753 observaciones que pertenecen al subconjunto de pruebas.

Uno de los elementos clave en el prompt engineering es darle un rol, unas instrucciones claras al modelo y ejemplos. Dependiendo de la API para consumir el modelo, en algunos casos se pueden enviar estas instrucciones como mensaje general del sistema, mientras que en otros se debe configurar la pregunta de tal forma que incluya estas instrucciones claras.

A nivel general, el proceso que se realiza es el siguiente:

1. **Investigar y configurar los prerrequisitos:** Cada uno de los modelos de esta sección tienen une prerrequisitos específicos que deben configurarse para poder usar los modelos.
    - **Modelos llama**: Todos los modelos de meta son de uso abierto, sin embargo, para poder usarlos se debe aceptar las condiciones de uso y solicitar su acceso a través de Kaggle o HuggingFace. En este caso se solicitan por Hugging Face, por lo que es necesario autenticarse con una cuenta de Hugging Face, ir a la tarjeta del modelo, llenar el formulario de solicitud y aceptar los términos y condiciones. Finalmente, una vez el acceso sea aprobado, se debe autenticar en los notebooks donde se vayan a utilizar el modelo con la cuenta que tiene los permisos de uso asignados.

    - **Modelos Gpt, Gemini y Claude**: Los modelos de OpenAI, Google y Anthropic son de uso por pago, por lo tanto, antes de poder utilizarlos se debe crear una cuenta, recargar créditos y generar una API Key asociada a la cuenta con créditos disponibles. Finalmente, se deben configurar en los notebooks el cliente de servicio con el API Key correspondiente.

2. **Construir un Prompt**: El prompt usualmente se compone de un mensaje del sistema que da un rol e instrucciones al modelo y, el mensaje del usuario que se espera que sea respondido por el modelo.

2.1. **Mensaje del sistema:** Se construye el mensaje del sistema que se muestra en la ilustración 5 que incluye el rol que debe asumir el modelo, instrucciones, restricciones y un ejemplo de la pregunta del usuario y la respuesta esperada.

Ilustración 5: Mensaje del sistema del prompt para los LLMs

```
system_message = """
You are an advanced assistant specialized in analyzing and detecting emotions in short text.
You will be provided with a text, and your task is to classify it into **exactly one emotion**
from the following list: [shame, sadness, joy, guilt, fear, disgust, anger].

**Important Rules:**
1. You must return only **one** of the emotions from the list without aditional text.
2. Do **not** create or infer any emotions outside the list.
3. If the text does not match any emotion exactly, return the closest emotion from the list.
4. Do not return aditional text, only **exactly one emotion** of the list.

For example:

Text: when the week for exams came I got afraid as to how I would tackle the questions.
Response: fear
"""
```

Fuente: Elaboración propia.

2.2. Mensaje del usuario: Se define una función y se itera sobre cada observación del conjunto de datos de prueba para extraer el texto y convertirlo en el mensaje del usuario. Tal y como se detalla en el siguiente paso.

3. **Preparar texto de entrada de los modelos:** Cada una de las APIs de los modelos a probar, tienen su forma específica de recibir el texto de entrada, por lo que en este paso se realiza dicha preparación. Por ejemplo, en el caso de la API de OpenAI, se deben configurar como se muestra en el siguiente ejemplo:

```
[{"role": "system", "content": system_message}
{"role": "user", "content": user_message}]
```

Donde, system_message es el mencionado en el punto 2.1. y el user_messge es el texto para clasificar del conjunto de datos de pruebas.

El ejemplo mencionado anteriormente, corresponde al caso de OpenAI. Se puede ver el detalle de preparación de cada uno de los modelos en el repositorio de GitHub que se comparte en anexos.

4. **Realizar petición de clasificación a los modelos:** Una vez construido el prompt y preparado el texto en el formato adecuado que esperan los modelos, se realiza la petición de clasificación del texto en una emoción. De manera implícita, estas peticiones transforman los textos en las representaciones numéricas aceptadas por los modelos. En los casos de Gpt, Gemini y Claude, esta petición se envía a la plataforma del proveedor, se procesa en sus máquinas y se recibe la respuesta. Por otra parte, en el caso de llama, los textos son procesados con los hiperparámetros del modelo preentrenado en la máquina local para obtener la emoción de respuesta.

   Estas peticiones tienen parámetros que deben ser configurados y que pueden afectar el resultado de la predicción, entre ellos, la temperatura y el número de tokens de salida. Para todos los casos se asignó la temperatura mínima, dado que no se busca que el modelo no sea creativo, adicionalmente, se indica un número máximo de 2 tokens de salida porque son suficientes para la longitud de las emociones que se esperan como resultado.

   Finalmente, para el caso de Gemini, se deben configurar los filtros de seguridad que trae por defecto, dado que algunos textos se bloquean por tener alta probabilidad de contener texto ofensivo u otra categoría no admitida. Dado que no se quería omitir ninguna observación, se modifican los filtros al mínimo para que ninguna observación fuese bloqueada.

Todos estos pasos se realizan sobre múltiples versiones de los modelos, en algunos casos el experimento es exitoso, mientras que, en otros casos, las respuestas no seguían las restricciones indicadas, por lo que se considera que fueron fallidos. En la siguiente tabla se detallan todos los modelos experimentados y se indica si el resultado fue exitoso o fallido.

Tabla 5: Lista de LLMs evaluados en el experimento de prompt engineering.

Uso	Proveedor	Modelo	Resultado
De pago	Anthropic	claude-3-5-sonnet-20240620	Exitoso
De pago	Anthropic	claude-3-haiku-20240307	Exitoso
De pago	Anthropic	claude-3-opus-20240229	Exitoso
De pago	Anthropic	claude-3-sonnet-20240229	Exitoso
De pago	Google	gemini-1.5-flash	Exitoso
De pago	Google	gemini-1.5-pro-latest	Exitoso
De pago	OpenAI	gpt-4o-mini	Exitoso
Abierto	Meta	llama-3.1-8b-instruct	Exitoso
Abierto	Meta	llama-3.2-1B	Fallido
Abierto	Meta	llama-3.2-1B-Instruct	Fallido

Fuente: Elaboración propia.

Fine-Tuning.

En esta sección se realiza el proceso de Fine-Tuning sobre modelos de uso abierto como Distilbert, Roberta, XLM-Roberta, llama y XLNet; y sobre modelos de uso de pago como GPT, Claude y Gemini. Este proceso consiste en ajustar todos los parámetros del modelo de acuerdo con los datos de entrenamiento y validación para finalmente realizar la evaluación sobre el conjunto de datos de pruebas.

A nivel general, el proceso que se realiza se describe por los siguientes pasos:

1. **Investigar y configurar los prerrequisitos:** Al igual que en la sección anterior, se verifican y configuran los prerrequisitos, en caso de existir, para realizar fine-tuning se los modelos seleccionados.

    - **Modelos de uso abierto:** Los modelos Distilbert, RoBERTa, XLM-RoBERTa y XLNet son de uso abierto y pueden ser usados a través de la librería de Hugging Face llamada transformers sin configuración de prerrequisitos.

- **Modelos llama:** Las mismas configuraciones mencionadas en la sección de prompt engineering.

- **Modelos GPT y Gemini:** Las mismas configuraciones mencionadas en la sección de prompt engineering.

- **Modelos Claude:** A la fecha de este desarrollo los modelos de Anthropic se encuentran disponibles para fine-tuning exclusivamente a través del servicio de AWS llamado Amazon Bedrock. Para poder acceder a ellos, se debe realizar una solicitud de manera formal al equipo de AWS y esperar su respuesta. Es posible que esta solicitud de acceso sea negada. En caso de que la respuesta sea favorable, se debe disponer de créditos en la cuenta de AWS dado que el uso del modelo fine-tuneado tiene costo.

2. **Preparar texto de entrada para los modelos:** En este paso se le da formato a los datos para que sean compatibles con las entradas esperadas de cada uno de los modelos. Esta preparación varía dependiendo del modelo, pero pueden catalogarse en dos etapas:

    2.1. Formato esperado por las API: Consiste en la estructura de datos que es compatible con las API de los modelos y a nivel general se puede resumir en las siguientes dos categorías.

    - **Modelos de Hugging Face de uso abierto:** Todos los modelos que se consumen a través de la librería de Transformers esperan que los datos de entrada de encuentren en formato “Datasets”, por lo tanto, se debe importar dicha librería y convertir los dataframes de pandas a datasets de hugging face.

- **Modelos de uso de pago:** Generalmente, los modelos de uso de pago como GPT, Gemini y Claude, esperan que los datos de encuentren en formato json y que simulen una conversación entre el modelo y el usuario. Por lo tanto, se realiza dicha transformación al dataframe de pandas y se almacenan los datos en formato json para poderse enviar a la API en el paso de entrenamiento.

2.2. Representación numérica de los textos: En los modelos de uso abierto, antes de enviar los textos al proceso de fine-tuning se deben vectorizar utilizando los embeddings o las representaciones vectoriales correspondientes a cada uno de los modelos. Los modelos solo admiten su respectivo vectorizador, porque están configurados para representar los textos en matrices de dimensiones que espera el modelo en su capa de entrada. Si las dimensiones no coinciden, el proceso falla.

Para los modelos de uso de pago, este proceso se realiza de manera interna en la plataforma del proveedor, por lo que no se aplica vectorización antes de enviar los datos al trabajo de ajuste fino.

3. **Fine-Tuning de modelos:** Una vez que se tienen configurados todos los prerrequisitos y se preparan los datos en el formato adecuado, se procede a realizar el proceso de fine-tuning de los modelos. Para esto, los modelos utilizan los conjuntos de datos de entrenamiento y de validación. Este es un proceso computacionalmente más intensivo que prompt enginering**,** lo que implica tiempos de espera muy prolongados e incluso, en algunos casos, intentos fallidos por falta de capacidad computacional.

   Para los modelos de uso abierto, el proceso de fine-tuning se realiza en cómputo local con capacidad reducida, por esto, el ajuste del modelo llama

fue fallido. Sin embargo, los modelos Distilbert, RoBERTa, XLM-RoBERTa y XLNet se lograron ajustar de manera exitosa.

Por otra parte, para los modelos de uso de pago GPT y Gemini, el proceso de fine-tuning se realiza en cómputo del proveedor, por lo que se crea un job de entrenamiento y se envían los datos en el formato esperado para finalmente obtener un id del modelo fine-tuneado que queda hospedado en OpenAI o Google para su uso.

Finalmente, para el modelo de Claude, la respuesta a la solicitud de acceso al modelo fue negativa, por lo que se descarta del experimento.

El proceso de fine-tuning tiene ciertos hiperparámetros que pueden determinar la efectividad en el aprendizaje, los más relevante son:

- **Epochs**: Representan el número de iteraciones completas sobre el conjunto de datos de entrenamiento durante el ajuste del modelo. En los casos en los que podía ser configurado, se asignaron valores entre 5 y 20, dependiendo del tiempo que tomaba realizar una sola iteración de ajuste del modelo.
- **Learning Rate:** Define el tamaño del paso que da el modelo al ajustar los pesos durante el entrenamiento en cada iteración. Dado que el conjunto de datos no es demasiado grande, se utiliza un learning_rate bajo.
- **Early Stop:** Boolean que detiene el entrenamiento si el modelo deja de mejorar en un conjunto de validación. Se indica como verdadero dado que se quiere evitar seguir iterando cuando el modelo ha dejado de aprender.
- **Patience**: Número de épocas adicionales que se espera sin mejoras antes de activar el Early Stop. Se asigna el valor de 2 a este parámetro.

Este proceso se realiza para múltiples versiones de las familias de modelos. En este caso, no todos los modelos seleccionados se lograron fine-tunear por falta de

acceso o falta de recursos computacionales. En la tabla 6 se detallan todas las versiones de modelos que fueron fine-tuneadas y se indica si su proceso de ajuste se pudo finalizar con éxito o fue fallido.

Tabla 6: Lista de LLMs evaluados en el experimento de fine-tuning.

Uso	Proveedor	Modelo	Resultado
Open	DistilBERT community	distilbert-base-uncased-2	Exitoso
Close	Google	gemini-1.5-flash-5e_v1	Exitoso
Close	Google	gemini-1.5-flash-5e_v2	Exitoso
Close	OpenAI	gpt-3.5-turbo-1106	Exitoso
Close	OpenAI	gpt-4o-mini-2024-07-18	Exitoso
Close	Clause	claude-3-haiku	Fallido
Open	FacebookAI	roberta-base	Exitoso
Open	FacebookAI	roberta-large	Exitoso
Open	FacebookAI	xlm-roberta-base	Exitoso
Open	FacebookAI	xlm-roberta-large	Exitoso
Open	XLNet Community	xlnet-base-cased	Exitoso
Open	Meta	llama-3.2-1b	Fallido

Fuente: Elaboración propia.

EVALUACIÓN DE LOS MODELOS

La evaluación de los modelos se divide en una evaluación cuantitativa y una evaluación cualitativa. La evaluación cuantitativa se utiliza para definir cuál es el modelo que tiene mejor rendimiento para la clasificación de textos en emociones utilizando el conjunto de datos ISEAR, mientras que la evaluación cualitativa, busca evaluar el mejor modelo con textos ingresados por una muestra de 5 usuarios y obtener una retroalimentación que indique si la clasificación tiene sentido o no. En esta sección se describe cómo se realiza cada uno de los tipos de evaluaciones, sin embargo, los resultados de estas se detallan en la siguiente sección del proyecto llamada “Resultados”.

Evaluación cuantitativa.

Para evaluar la eficiencia de los modelos para clasificar textos que no ha observado en su proceso de entrenamiento, se utilizan los 752 textos del conjunto de datos de prueba.

En el caso de prompt engineering, se utiliza el prompt construido para obtener una clasificación para todas las observaciones de prueba. Por otra parte, para el caso de fine-tuning, se utiliza la nueva versión ajustada de los modelos para clasificar los textos en una emoción.

Posteriormente, se utilizan las clasificaciones de emociones del modelo y las emociones reales para calcular el f1-score y analizar la matriz de confusión con el objetivo de identificar cuál de los modelos tuvo el mejor rendimiento.

De acuerdo con esta evaluación cuantitativa, se encuentra que el mejor modelo para esta tarea es la versión más reciente al momento de este experimento de gpt-4o-mini. Por lo tanto, se procede a crear un prototipo funcional con el cuál se realiza la evaluación cualitativa.

Evaluación cualitativa.

Para esta evaluación, se les pide a 5 personas que realicen pruebas con 5 textos propios para que entreguen una retroalimentación de cada uno de los textos donde indiquen si la clasificación tiene sentido o no tiene sentido para ellos.

DESPLIEGUE DEL MEJOR MODELO

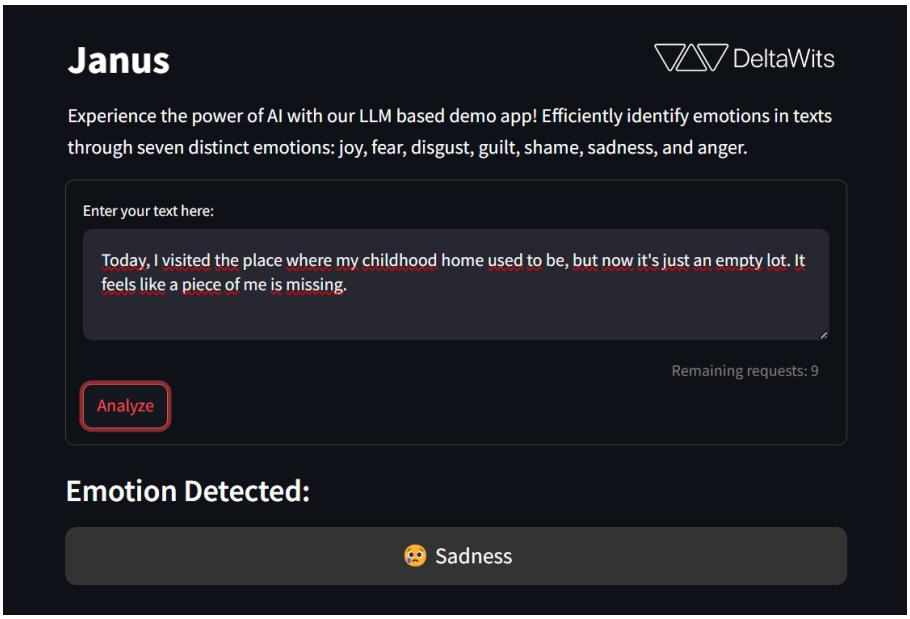
En esta etapa del proyecto se busca implementar el modelo que ha demostrado mejor rendimiento en la clasificación de textos en emociones. Para esto, se plantean dos pasos.

1. **Despliegue de prototipo funcional:** En este paso se desarrolla y despliega un prototipo funcional llamado Janus utilizando el lenguaje de Python con la librería de Streamlit. El objetivo de este prototipo funcional es validar rápidamente la hipótesis de que el modelo realiza clasificaciones que tienen

sentido en textos que no han sido observados en el proceso de entrenamiento o de evaluación cuantitativa.

En la siguiente ilustración se muestra un ejemplo de la interacción con el prototipo funcional, donde se ingresa un texto que evidentemente está relacionado con una emoción y se retorna la emoción que le modelo identifica.

Ilustración 6: Ejemplo de interacción con prototipo funcional.



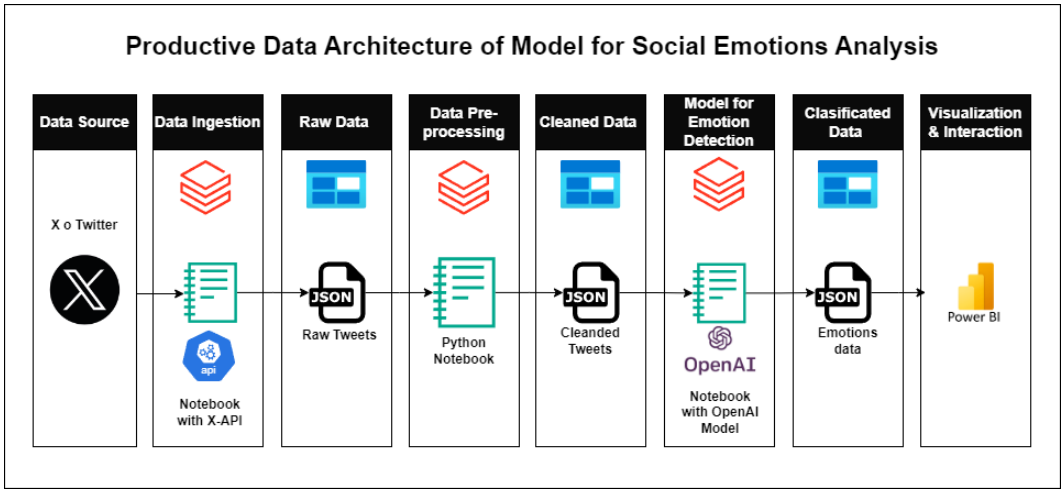
Fuente: Elaboración propia.

2. **Despliegue productivo del modelo:** Una vez se valide la hipótesis de que el modelo es efectivo para detectar emociones en textos, se planea utilizar en un proyecto de análisis de emociones sociales a partir de tweets. Es importante mencionar que la solución que se describe a continuación no se encuentra desarrollada, pero se considera que es necesario detallarla dado que hace parte de la etapa final de la metodología seleccionada en el proyecto.

   En la siguiente ilustración de muestra la arquitectura propuesta para esta solución. A nivel general, consiste en extraer tweets alrededor de un tema

(por ejemplo #DonaldTrump) utilizando la API oficial de X, almacenar los datos crudos en un datalake, posteriormente un notebook se ejecuta de manera para transformar estos textos y llevarlos a una capa de almacenamiento de datos limpios, desde allí el modelo de clasificación los ingesta y retorna las emociones detectadas, finalmente, dichas emociones se utilizan para crear un reporte que permita interacción y análisis. Se propone blob storage de azure como datalake y databricks como plataforma de ejecución de los notebooks.

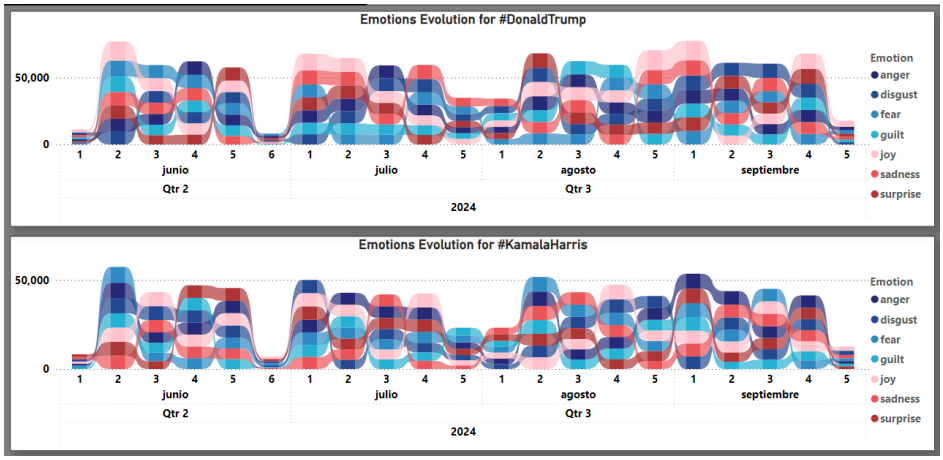
Ilustración 7: Arquitectura de solución de análisis de emociones sociales



Fuente: Elaboración propia.

Una demostración del uso de este análisis se representa en la ilustración 8, donde se muestra con datos simulados, cómo el análisis de emociones alrededor de dos temas de Twitter podría ser útil para entender lo que la sociedad percibe alrededor de temas tan importantes como las elecciones de un país.

Ilustración 8: Demo de análisis de emociones alrededor de las elecciones de USA



Fuente: Elaboración propia con datos simulados.

RESULTADOS

En esta sección se muestran los resultados obtenidos en las evaluaciones cuantitativas y cualitativas. En la evaluación cuantitativa se evidencian los resultados obtenidos en los LLM experimentados en los procesos de prompt engineering y fine-tuning. Para la evaluación cualitativa se detallan las evaluaciones realizadas por 5 usuarios sobre el prototipo funcional del mejor modelo.

RESULTADOS DE EVALUACIÓN CUANTITATIVA DE LOS LLM.

Para evaluar cuantitativamente los modelos, se utiliza la emoción identificada por los modelos y la emoción real de las observaciones del conjunto de datos de prueba para calcular el f1-score para todos los modelos experimentados con los métodos de prompt engineernig y fine-tuning.

RESULTADOS DE DETECCIÓN DE EMOCIONES CON PROMPT ENGINEERING.

Se evalúan en total 8 LLMs con prompt engineering en la clasificación de textos en emociones. En la tabla 7 se puede observar el nombre del modelo, el macro f1-score sobre el conjunto de datos de prueba, el proveedor y la columna source que indica si el modelo es de uso abierto (open) o de uso de pago (close).

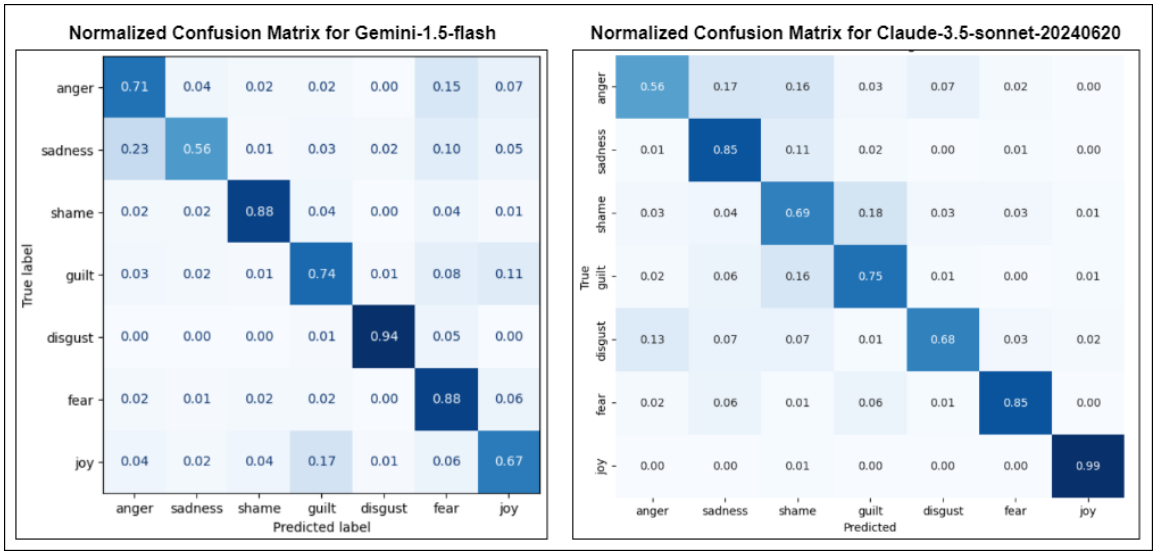
Tabla 7: F1-score para diferentes LLMs con Prompt Enginering.

Proveedor	Model	Source	F1-Score
Google	gemini-1.5-flash	Close	**0.76729**
Anthropic	claude-3-5-sonnet-20240620	Close	**0.76710**
Google	gemini-1.5-pro-latest	Close	0.76459
OpenAI	gpt-4o-mini	Close	0.75422
Anthropic	claude-3-opus-20240229	Close	0.74494
Anthropic	claude-3-sonnet-20240229	Close	0.73916
Meta	llama-3.1-8b-instruct	Open	0.67199
Anthropic	claude-3-haiku-20240307	Close	0.65701

Fuente: Elaboración propia.

Se puede evidenciar que, en casi todos los casos, los modelos de uso por pago tienen mejores resultados. El mejor modelo fue Gemini-1.5-flash de Google con un f1-score de 0.76729. Sin embargo, da resultados muy similares a los modelos Claude-3-5-sonnet-20240620. Para definir cuál de estos dos es el mejor modelo, se analiza la ilustración 9, que representa la matriz de confusión normalizada para los modelos Gemini-1.5-flash y Claude-3.5-sonnet, en el eje y se encuentra la emoción real y en el eje x se encuentra la emoción asignada por el modelo.

Ilustración 9: Matriz de confusión normalizada de Gemini-1.5-flash y Claude-3.5-Sonnet sobre el conjunto de datos de pruebas con Prompt Engineering.



Fuente: Elaboración propia.

Para el modelo Gemini-1.5-flash se puede identificar que el modelo es especialmente bueno para identificar las emociones disgusto, miedo y culpa. Tiende a confundir la alegría con culpa, la tristeza con rabia y la rabia con miedo. La tristeza es la emoción con más errores en la clasificación, con una precisión de 0.56.

Por otra parte, en la ilustración del modelo Claude-3.5-sonnet se evidencia que el modelo es especialmente bueno para identificar las emociones alegría, miedo y tristeza. Tiende a confundir la rabia con la rabia con la tristeza y la pena, pena con culpa y disgusto con rabia. La rabia es la emoción con más errores en la clasificación, con una precisión 0.56.

En general, ambos modelos son buenos identificando algunas emociones, pero son cuestionables para identificar otras. Por lo tanto, la elección del mejor modelo depende de cuáles emociones se priorizan para tener mayor certeza en la clasificación. Por ejemplo, si se consideran más importante las emociones alegría y tristeza, el mejor modelo sería el de Claude; pero si se considera más importante el disgusto y la rabia, el mejor modelo sería el de Gemini.

RESULTADOS DE DETECCIÓN DE EMOCIONES CON FINE-TUNING.

Se evalúan en total 10 LLMs de uso abierto y de uso por pago para clasificar los mismos datos de prueba en una emoción. En la tabla 8 se muestran los nombres de los modelos, el macro f1-score sobre el conjunto de datos de prueba, el proveedor y la columna source que indica si el modelo es de uso abierto (open) o de uso de pago (close).

Tabla 8: F1-score para diferentes LLMs con Fine-tuning.

Proveedor	**Model**	**Source**	**F1-Score**
OpenAI	gpt-4o-mini-2024-07-18	Close	**0.81505**
OpenAI	gpt-3.5-turbo-1106	Close	0.81063
Google	gemini-1.5-flash-5e	Close	0.79583
FacebookAI	roberta-large	Open	0.77358
FacebookAI	xlm-roberta-large	Open	0.74438
FacebookAI	roberta-base	Open	0.74005
XLNet Community	xlnet-base-cased	Open	0.72208

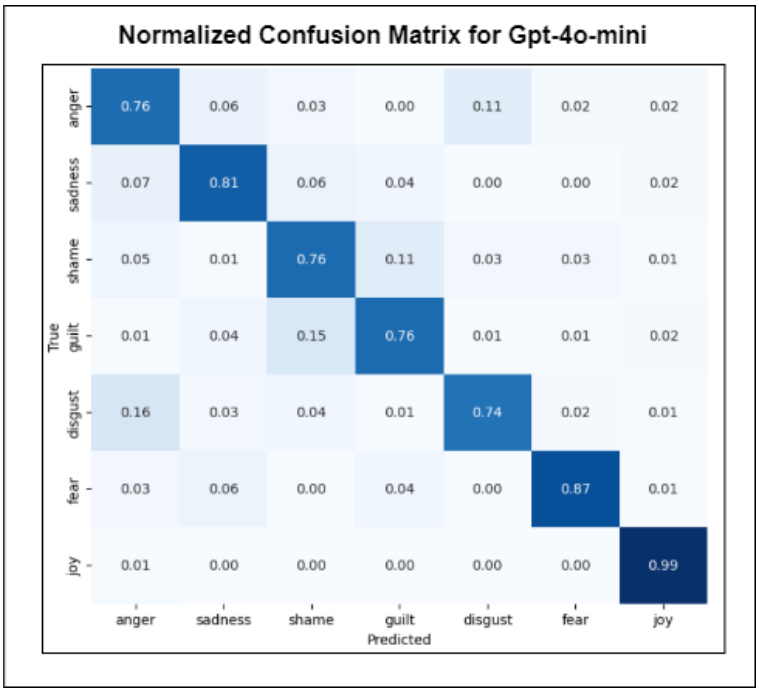
FacebookAI	xlm-roberta-base	Open	0.70088
DistilBERT community	distilbert-base-uncased-2	Open	0.69600

Fuente: Elaboración propia.

Se destaca que el mejor modelo en este caso es gpt-4o-mini-2024-07-18 de OpenAI con un f1-score de 0.81505. La diferencia entre estos resultados y el f1-score del mejor modelo y de la misma versión del modelo con Prompt engineering es de 0.04776 y 0.06083 puntos respectivamente.

En la ilustración 10 se representa la matriz de confusión normalizada para el modelo Gpt-4o-mini donde se puede identificar que le modelo es especialmente bueno para alegría y miedo. Aunque tiende a confundir rabia con disgusto y la culpa con pena, su rendimiento general es mucho mejor que los casos de prompt engineering, dado que su peor proporción de aciertos es de 0.74 para la emoción de disgusto.

Ilustración 10: Matriz de confusión normalizada de Gpt-4o-mini sobre el conjunto de datos de pruebas con Fine-tuning.



Fuente: Elaboración propia.

Por lo anterior, se puede concluir que el mejor modelo brinda una buena probabilidad de acierto en la identificación de todas las emociones y que, casi nunca se va a equivocar para identificar alegría.

RESULTADOS DE DETECCIÓN DE EMOCIONES CON LLMS CONSOLIDADO.

Si bien el mejor desempeño fue alcanzado con un modelo ajustado mediante fine-tuning, los modelos basados en prompt engineering también mostraron un rendimiento competitivo. Se puede identificar en la tabla 9 que, en algunos casos, los modelos con prompt engineering logran superar los resultados de los modelos ajustados, posicionándose como una alternativa viable cuando el fine-tuning no es factible debido a limitaciones computacionales o económicas.

Tabla 9: F1-score para diferentes LLMs con Fine-tuning y Prompt engineering.

Proveedor	**Model**	**Source**	**Method**	**F1-Score**
OpenAI	gpt-4o-mini-2024-07-18	Close	Fine-Tuning	**0.81505**
OpenAI	gpt-3.5-turbo-1106	Close	Fine-Tuning	0.81063
Google	gemini-1.5-flash-5e_v1	Close	Fine-Tuning	0.79583
Google	gemini-1.5-flash-5e_v2	Close	Fine-Tuning	0.77728
FacebookAI	roberta-large	Open	Fine-Tuning	0.77358
Google	gemini-1.5-flash	Close	Prompt Engineering	**0.76729**
Anthropic	claude-3-5-sonnet-20240620	Close	Prompt Engineering	0.76710
Google	gemini-1.5-pro-latest	Close	Prompt Engineering	0.76459
OpenAI	gpt-4o-mini	Close	Prompt Engineering	0.75422
Anthropic	claude-3-opus-20240229	Close	Prompt Engineering	0.74494
FacebookAI	xlm-roberta-large	Open	Fine-Tuning	0.74438
FacebookAI	roberta-base	Open	Fine-Tuning	0.74005
Anthropic	claude-3-sonnet-20240229	Close	Prompt Engineering	0.73916
XLNet Community	xlnet-base-cased	Open	Fine-Tuning	0.72208
FacebookAI	xlm-roberta-base	Open	Fine-Tuning	0.70088

DistilBERT community	distilbert-base-uncased-2	Open	Fine-Tuning	0.69600
Meta	llama-3.1-8b-instruct	Open	Prompt Engineering	0.67199
Anthropic	claude-3-haiku-20240307	Close	Prompt Engineering	0.65701

Fuente: Elaboración propia.

RESULTADOS DE EVALUACIÓN CUALITATIVA DE LOS LLM.

El mejor modelo encontrado con la evaluación cuantitativa fue la versión fine-tuneada de gpt-4o-mini. Con esta versión se despliega un prototipo funcional y se pide a 5 usuarios probar el modelo con 5 textos en español o en inglés con el objetivo de identificar si las clasificaciones del modelo tienen sentido o no.

En la tabla 10 se muestra el detalle de los textos ingresados por los usuarios, la clasificación del modelo y una columna que indica si la respuesta tuvo sentido para el usuario.

Tabla 10: Detalle evaluación de usuarios del mejor modelo.

Usuario	Texto	Clasificación	¿Tiene Sentido?
1	Mi madre ha muerto hoy	Sadness	SI
1	Destapé una bolsa de leche y tenía olor a podrido	Disgust	SI
1	Me gané la loteria y nadie más lo sabe	Joy	SI
1	Un asesino entro a mi casa y yo estoy escondido	Fear	SI
1	Hoy robé un dulce, y no puedo dormir	Guilt	SI
2	My dog was hit by a car yesterday.	Sadness	SI
2	Today I received the great news of a new job	Joy	SI
2	I discovered my employee stealing money from the store	Anger	SI
2	I am in a cabin alone, far from the city, and I hear strange noises that won't let me sleep	Fear	SI

2	Today they are going to reveal the gender of my baby. I am in the waiting room	Joy	SI
3	My teacher changed my notes, I got a good grade and he changed it to a bad one	Anger	SI
3	I did put a lot of effort to a homework and I got an amazing grade	Joy	SI
3	I gave a present to my cousin and she didn't like it	Shame	SI
3	I was in a road trip with my family and suddenly my brother threw up in the car	Disgust	SI
3	I borrowed my mother's car and I crashed	Guilt	SI
4	I had an appointment with my doctor and i arrived on hour late because of the traffic jam	Guilt	SI
4	I was shopping with my mom and we didn't have cash and we had to pay with coins	Shame	SI
4	I was starving and a friend invited me a pizza	joy	SI
4	I am home alone and I am hearing some voices in the house	fear	SI
4	I was walking in a sunny day and i bought an ice cream and i fell and i drop my ice cream	sadness	SI
5	I finally got my hands on that deam job! I swear, I couldn't stop smiling all day	Joy	SI
5	I was walking home late at night and when I heard a noise behind me, I felt my heart jump to my trhoat	Fear	SI
5	I opened the fridge, and man, the smell was so nasty I almost threw up right there	Disgust	SI
5	I borrowed my buddy's bike and scratched it; now I fell like I owe him the world to make it up	Guilt	SI
5	When I saw the old house beign torn down, I felt like a piece of my childhood was gone forever	Sadness	SI

Fuente: Elaboración propia.

Se puede identificar en el detalle entregado por los usuarios de prueba para todos los textos evaluados, que la respuesta tuvo sentido para los usuarios, por lo que se confirma la eficiencia del modelo con datos no observados. Además, se destaca que textos en español también fueron clasificados de manera exitosa.

CONCLUSIONES

En este trabajo se ha realizado la evaluación de rendimiento de diversos modelos en la clasificación de emociones a partir de textos, abordando desde técnicas tradicionales de aprendizaje automático hasta técnicas avanzadas de LLMs como prompt engineering y fine-tuning.

Se puede concluir que los LLM representan un avance significativo para tareas de NLP, dado que todos los modelos evaluados entregaron mejores resultados que cualquier modelo de aprendizaje automático. Adicionalmente, se comprobó que los LLMs ajustados mediante fine-tuning ofrecen los mejores resultados en términos de F1-score, siendo GPT-4o-mini el mejor modelo. Sin embargo, los modelos basados en prompt engineering demostraron ser competitivos, superando en ciertos casos a los modelos ajustados y posicionándose como una solución viable para escenarios con restricciones computacionales o económicas.

Por último, la implementación de un prototipo funcional validó la eficacia de estos modelos en escenarios reales, demostrando su potencial para aplicaciones prácticas en diversas industrias, como el análisis de emociones en redes sociales, contribuyendo significativamente al entendimiento y aprovechamiento del lenguaje humano.

REFERENCIAS

Adoma, A. F., Henry, N. M., & Chen, W. (2020). Comparative Analyses of Bert, Roberta, Distilbert, and Xlnet for Text-Based Emotion Recognition. *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing, ICCWAMTIP 2020*. https://doi.org/10.1109/ICCWAMTIP51612.2020.9317379

Alan, M. (1950). Turing. Computing machinery and intelligence. *Mind*, *59*(236).

Batbaatar, E., Li, M., & Ryu, K. H. (2019). Semantic-Emotion Neural Network for Emotion Recognition from Text. *IEEE Access*, *7*. https://doi.org/10.1109/ACCESS.2019.2934529

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, *3*(4–5). https://doi.org/10.7551/mitpress/1120.003.0082

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1). https://doi.org/10.1023/A:1010933404324

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*. https://doi.org/10.3115/v1/d14-1179

Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, *20*(3). https://doi.org/10.1023/A:1022627411411

Cortiz, D. (2021). Exploring Transformers in Emotion Recognition: a comparison of BERT, DistillBERT, RoBERTa, XLNet and ELECTRA. *ArXiv*.

Cowen, A. S., & Keltner, D. (2021). Semantic Space Theory: A Computational Approach to Emotion. In *Trends in Cognitive Sciences* (Vol. 25, Issue 2). https://doi.org/10.1016/j.tics.2020.11.004

Demszky, D., Movshovitz-Attias, D., Ko, J., Cowen, A., Nemade, G., & Ravi, S. (2020). GoEmotions: A dataset of fine-grained emotions. *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. https://doi.org/10.18653/v1/2020.acl-main.372

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). 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*, *1*.

Ekman, P. (1992). An Argument for Basic Emotions. *Cognition and Emotion*, *6*(3–4). https://doi.org/10.1080/02699939208411068

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8). https://doi.org/10.1162/neco.1997.9.8.1735

Kim, Y. (2014). Convolutional neural networks for sentence classification. *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*. https://doi.org/10.3115/v1/d14-1181

Lees, R. B., & Chomsky, N. (1957). Syntactic Structures. *Language*, *33*(3). https://doi.org/10.2307/411160

Microsoft. (n.d.). *¿Qué es Mosaic AutoML?* Https://Learn.Microsoft.Com/Es-Es/Azure/Databricks/Machine-Learning/Automl/.

Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. In *IEEE Transactions on Knowledge and Data Engineering* (Vol. 22, Issue 10). https://doi.org/10.1109/TKDE.2009.191

Pascanu, R., Mikolov, T., & Bengio, Y. (2013). On the difficulty of training recurrent neural networks. *30th International Conference on Machine Learning, ICML 2013*, *PART 3*.

Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*. https://doi.org/10.3115/v1/d14-1162

Plutchik, R. (1980). Emotion: A Psychoevolutionary Synthesis. *The American Journal of Psychology*, *93*(4). https://doi.org/10.2307/1422394

Polignano, M., De Gemmis, M., Basile, P., & Semeraro, G. (2019). A comparison of Word-Embeddings in Emotion Detection from Text using BiLSTM, CNN and Self-Attention. *ACM UMAP 2019 Adjunct - Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization*. https://doi.org/10.1145/3314183.3324983

Prottasha, N. J., Sami, A. A., Kowsher, M., Murad, S. A., Bairagi, A. K., Masud, M., & Baz, M. (2022). Transfer Learning for Sentiment Analysis Using BERT Based Supervised Fine-Tuning. *Sensors*, *22*(11). https://doi.org/10.3390/s22114157

Salton, G., & J.McGill, M. (1986). *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc.

Scherer, K. R., & Wallbott, H. G. (1994). Evidence for Universality and Cultural Variation of Differential Emotion Response Patterning. *Journal of Personality and Social Psychology*, *66*(2). https://doi.org/10.1037/0022-3514.66.2.310

Scott, A. J., Hosmer, D. W., & Lemeshow, S. (1991). Applied Logistic Regression. *Biometrics*, *47*(4). https://doi.org/10.2307/2532419

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, *45*(4). https://doi.org/10.1016/j.ipm.2009.03.002

Tunstall, L., Von Werra, L., & Wolf, T. (2022). Natural Language Processing with Transformers (Revised Edition). In *O'Reilly Media* (Vol. 19, Issue 1).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *2017-December*.

Weizenbaum, J. (1966). ELIZA-A computer program for the study of natural language communication between man and machine. *Communications of the ACM*, *9*(1). https://doi.org/10.1145/365153.365168

ANEXOS

En esta sección se encuentran los anexos relacionados con el detalle del desarrollo del proyecto y el prototipo funcional

1. **Repositorio de Github de Experimentos.**

   En el siguiente repositorio se encuentran todos los experimentos realizados, incluyendo las etapas de exploración y transformación de datos, modelado y evaluación de diferentes modelos: https://github.com/dlopeza98/TrabajoGradosEAFIT

2. **Repositorio de Github del prototipo funcional.**

   En el siguiente repositorio de encuentra el código fuente del prototipo funcional: https://github.com/dlopeza98/jannus

3. **Acceso al prototipo funcional.**

   El prototipo funcional se mantendrá habilitado por un tiempo. Para acceder a él póngase en contacto con el autor en el correo dalopeza@eafit.edu.co.