Del Texto a la Emoción: Análisis de Sentimientos de Reseñas Cinematográficas.

Marco Ulises Flores Chona[a], Dora María Calderón Nepamuceno[b]

[a] Universidad Autónoma del Estado de México Plantel Nezahualcóyotl, Av. Bordo de Xochiaca, Benito Juárez, 57000 Cdad. Nezahualcóyotl, Méx., ciudad de México, Estado de México, México. mfloresc014@alumno.uaemex.mx,
[b] Universidad Autónoma del Estado de México Plantel Ecatepec, José Revuelta 17, Tierra Blanca, 55020 Ecatepec de Morelos, Estado de México, México, dmcalderonn@uaemex.mx.

Resumen

En este trabajo, se propone explorar y desarrollar un modelo de análisis de sentimientos utilizando técnicas de aprendizaje profundo para clasificar reseñas cinematográficas. A través de este estudio, se busca no solo contribuir al campo del Procesamiento de Lenguaje Natural (PLN), sino también ofrecer una herramienta útil para la industria cinematográfica y los entusiastas del cine. El modelo fue desarrollado mediante el uso de redes neuronales construidas con Keras y TensorFlow, y se entrenó con un conjunto de datos de reseñas en español previamente etiquetadas como positivas o negativas. El proceso incluyó la adquisición de datos, limpieza y preprocesamiento mediante tokens y técnicas de TF-IDF para convertir las reseñas en representaciones numéricas.
Los resultados mostraron una precisión general del 85% en el conjunto de prueba, lo que indica que el modelo es capaz de identificar correctamente el sentimiento en la mayoría de los casos. Sin embargo, en reseñas ambiguas o con sentimientos mixtos, el modelo presentó ciertas limitaciones, lo que destaca la necesidad de incorporar técnicas de análisis de contexto más profundo. La implementación del modelo en una aplicación web mediante Flask, permitió ofrecer una experiencia de usuario fluida y en tiempo real, con tiempos de respuesta de aproximadamente 0.5 segundos. Este proyecto contribuye al campo del PLN al proporcionar una herramienta práctica y eficiente para el análisis de sentimientos en reseñas de películas, y sienta las bases para futuras mejoras en la clasificación de sentimientos complejos.

Palabras clave — Análisis de sentimientos, aprendizaje profundo, procesamiento de lenguaje natural, reseñas cinematográficas, clasificación de texto.

Abstract

In this work, we propose to explore and develop a sentiment analysis model using deep learning techniques to classify movie reviews. Through this study, we aim not only to contribute to the field of ***Natural Language Processing (NLP)*** *but also to offer a useful tool for the film industry and movie enthusiasts.*

The model was developed using ***neural networks*** *built with* ***Keras*** *and* ***TensorFlow****, and it was trained on a dataset of* ***Spanish-language reviews*** *previously labeled as positive or negative. The process included* ***data acquisition****,* ***cleaning****, and* ***preprocessing*** *through tokenization and* ***TF-IDF techniques*** *to convert the reviews into numerical representations.*

The results showed an ***overall accuracy of 85%*** *on the test set, indicating that the model can accurately identify sentiment in most cases. However, in* ***ambiguous reviews*** *or those with* ***mixed sentiments****, the model showed certain limitations, highlighting the need to incorporate* ***deeper contextual analysis techniques****.*

The implementation of the model in a ***web application*** *using* ***Flask,*** *provided a* ***smooth and real-time user experience****, with* ***response times of approximately 0.5 seconds****. This project contributes to the* ***NLP*** *field by providing a* ***practical and efficient*** *tool for* ***sentiment analysis*** *in movie reviews and lays the foundation for future improvements in the classification of* ***complex sentiments****.*

Keywords — Sentiment analysis, deep learning, natural language processing, movie reviews, text classification.

1. INTRODUCCIÓN

En la era digital, el volumen de datos generados diariamente es inmenso. Particularmente, las reseñas cinematográficas, al ser una manifestación genuina de las emociones y percepciones de los espectadores, se han convertido en una valiosa fuente de información. Estas opiniones, compartidas en plataformas como IMDB [1], Rotten Tomatoes [2], entre otras, no solo influyen en la percepción pública de una película, sino que también pueden tener un impacto económico significativo en la taquilla.
Con el aumento de los datos de texto, como las redes sociales y los comentarios en línea, el análisis de sentimientos es crucial para comprender las necesidades de los usuarios, la dinámica del mercado y los cambios en la opinión pública.
Sin embargo, procesar y comprender manualmente estas reseñas es una tarea que supera la capacidad humana debido a su volumen. Es aquí donde el Procesamiento del Lenguaje Natural (PLN) y el Aprendizaje Automático entran en juego, brindando herramientas para analizar automáticamente grandes conjuntos de datos de texto. El análisis de sentimientos, una rama del PLN, se centra en determinar la polaridad de un texto, es decir, si la intención es positiva, negativa o neutral [3].

El modelado de sentimientos ha encontrado aplicaciones en diversas áreas, desde el marketing hasta la política. En el ámbito cinematográfico, puede ofrecer una perspectiva rápida de la recepción del público y anticipar tendencias.
A pesar de su relevancia, el modelado de sentimientos en reseñas de películas todavía enfrenta desafíos, como el

manejo del sarcasmo, la interpretación de contextos culturales, entre otros.

1.1. Análisis de sentimientos

El sentimiento de análisis o análisis de sentimientos es una subdisciplina del campo del PLN que se enfoca en la identificación y extracción de opiniones, emociones y actitudes expresadas en un texto. Su aplicación se ha vuelto fundamental en diversas industrias, desde el análisis de redes sociales hasta la atención al cliente y el monitoreo de marcas [4].

¿Qué es el análisis de sentimientos?

El análisis de sentimientos utiliza técnicas de *machine learning*, lingüística computacional y estadística para clasificar textos en categorías emocionales, como positivas, negativas o neutrales. Dependiendo de la granularidad del análisis, también puede detectar emociones más específicas, como felicidad, enojo o tristeza [5].

Métodos principales:

- Enfoques basados en reglas: Utilizan diccionarios de palabras con polaridad predefinida (positiva o negativa)
- Modelos de machine learning: Aplican algoritmos supervisados o no supervisados para entrenar modelos capaces de predecir el sentimiento.
- Modelos de deep learning: Redes neuronales profundas, como LSTMs o transformers (ej., BERT), que capturan significados contextuales más complejos.

Aplicaciones del análisis de sentimientos:

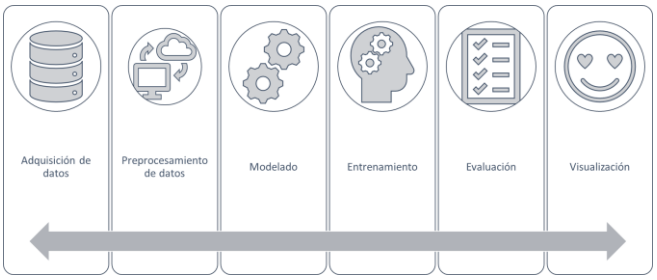
- Monitoreo de redes sociales: Evaluación de la percepción de marcas y productos.
- Atención al cliente: Automatización de respuestas basadas en el tono del usuario.
- Análisis de encuestas y opiniones: Detección de tendencias en comentarios de usuarios.
- Finanzas y política: Predicción de tendencias del mercado y análisis de discursos políticos.

2. CONTENIDO

2.1 Metodología.

Para presentar la funcionalidad del sistema, se puede observar el proceso detallado por etapas en el diagrama que se muestra en la Figura 1.

Fig. 1. Diagrama de secuencia para el análisis de texto.



Fuente: Elaboración propia

2.2 Adquisición de Datos

Los datos se han recopilado en un archivo CSV preprocesado que contiene reseñas de películas en español. Estas reseñas han sido previamente limpiadas y etiquetadas con sentimientos positivos y negativos. La fuente de los datos es una base de datos pública, un dataset de un conjunto de datos de IMDB.

Para manejar datos ambiguos o con múltiples emociones dentro de una misma reseña, se implementó una estrategia de etiquetado mixto. Si una reseña presentaba una combinación clara de sentimientos positivos y negativos, se clasificaba en función del sentimiento predominante. En los casos donde el sentimiento no era claramente positivo o negativo, se etiquetaba como "neutral" o "mixto". Además, se consideró la posibilidad de aplicar técnicas de análisis de contexto mediante modelos basados en transformers (como BERT) para captar matices más complejos y mejorar la interpretación de reseñas ambiguas. Esta estrategia permitió que el modelo identificara patrones complejos y contextos emocionales mezclados, mejorando la precisión general en la clasificación de sentimientos.

2.3 Preprocesamiento de Datos

El preprocesamiento es un paso crítico en cualquier proyecto de PLN. En nuestro caso, se ha realizado una limpieza inicial en la fase de adquisición de datos, que incluye la eliminación de etiquetas HTML, caracteres no alfabéticos y palabras de parada (stop words), así como la normalización de texto (como pasar todo a minúsculas). El archivo CSV se carga luego en un dataframe de librería Pandas [6] para su manipulación y procesamiento adicional. Las reseñas se transforman en lenguaje numérico es decir en secuencias números enteros, donde cada entero representa una palabra única en el corpus. Estas secuencias se pasan a una longitud estándar para asegurar la consistencia en la entrada del modelo.

2.4 Modelado

Para el modelado se utiliza el modelo de red neuronal y se construye utilizando la API (Interfaz de Programación de Aplicaciones, por sus siglas en inglés **Application Programming Interface**) secuencial de Keras [7], que es una pila lineal de capas. Incluye:

- Capa de Embedding:

Transforma las palabras en vectores densos de un tamaño especificado (100 en este caso) y está configurada para aprender estos embeddings durante el entrenamiento.

- Capa de GlobalAveragePooling1D:

Reduce la dimensión de los embeddings promediando cada dimensión, lo que también ayuda a disminuir el sobreajuste.

- Capa Densa:

Una capa de salida con activación 'softmax' que clasifica la entrada en categorías positivas o negativas.

2.5 Entrenamiento del Modelo

El modelo se entrena con el conjunto de datos de entrenamiento utilizando un proceso de optimización de Adam. Se ha especificado un número de épocas (5) para limitar el entrenamiento y un tamaño de lote (32) para definir cuántas muestras se trabajan antes de actualizar los parámetros internos del modelo. La división entre datos de entrenamiento y de prueba es del 80/20, lo que significa que el 20% del dataset se reserva para validar la precisión del modelo.

2.5 Evaluación

Para evaluar el modelo, utilizamos métricas estándar de clasificación como la precisión (accuracy), que mide el porcentaje de etiquetas predichas correctamente sobre el total de predicciones. También se podrían considerar otras métricas como recall, F1-score y la matriz de confusión para tener una visión más detallada del rendimiento del modelo.

2.6 Visualización

El modelo entrenado se despliega en una aplicación web construida con el micro- framework Flask [8]. La aplicación actúa como una API que recibe reseñas a través de solicitudes POST, procesa los datos utilizando el modelo y los tokens guardados, y devuelve la predicción de sentimiento. CORS se habilita para permitir solicitudes desde orígenes cruzados, facilitando la integración con frontends alojados en dominios diferentes.

3. DESARROLLO

El desarrollo se llevó a cabo seleccionando primero el lenguaje y las herramientas

Se seleccionó Python [7], por su robusta biblioteca de herramientas para procesamiento de lenguaje natural y aprendizaje automático. Su sintaxis clara y legibilidad lo convierten en una elección popular para prototipado rápido y desarrollo de aplicaciones de PLN. Además, Python cuenta con una gran comunidad y extensos recursos educativos que facilitan la solución de problemas y la innovación.

Para el modelado se eligió Keras/TensorFlow, Keras, con su backend TensorFlow, ya que ofrece una plataforma potente y flexible para construir y entrenar modelos de aprendizaje profundo. Keras proporciona una API de alto nivel, lo que facilita la construcción de redes neuronales con menos código, mientras que TensorFlow garantiza la eficiencia y la capacidad de escalar la operación a múltiples CPU o GPU [8].

Para la visualización se optó por Flask [9], Flask es un micro-framework de Python que permite desarrollar aplicaciones web rápidamente. Es ligero, fácil de usar, y suficientemente robusto para la creación de APIs de backend. Además, la habilidad de Flask para trabajar bien con Python lo hace ideal para implementar modelos de PLN entrenados en Python.

3.1 Diseño del Sistema

El sistema se estructura en dos componentes principales: el backend del modelo de PLN y la interfaz de usuario del frontend.

Backend (Modelo de PLN): El núcleo del sistema es un modelo de red neuronal entrenado para clasificar las reseñas de películas en positivas o negativas. El modelo se entrena fuera de línea y se guarda para su uso en producción.

Frontend (Aplicación Flask): La interfaz de usuario se construye como una página web que permite a los usuarios ingresar reseñas y obtener análisis de sentimientos. Esta interfaz interactúa con el backend a través de solicitudes HTTP.

3.2 Implementación.

La implementación se llevó a cabo en varias etapas, siguiendo la metodología descrita anteriormente:

Adquisición de Datos

- Cargar datos: Aquí se está utilizando la biblioteca panda, que es una herramienta poderosa para la manipulación de datos en Python, para cargar un archivo CSV en un DataFrame. Un DataFrame es una estructura de datos bidimensional (esencialmente una tabla) que puede contener varios tipos de datos y es muy útil para el manejo de datos tabulares. El archivo CSV 'archivo_preprocesado.csv' contiene los datos que se utilizarán en el proyecto. El parámetro encoding='ISO-8859-1' indica que el archivo usa una codificación específica para los caracteres, lo que es importante para leer correctamente el archivo si contiene caracteres especiales.

- Preparar los datos: En esta parte, se están seleccionando las columnas del DataFrame que se usarán para el análisis. X representa las características o entradas del modelo (en este caso, las reseñas de películas en español), y "y" representa las etiquetas o salidas del modelo (en este caso, la clasificación del sentimiento de cada reseña).

Preprocesamiento de Datos

- Codificar las etiquetas: Aquí se está utilizando LabelEncoder de la biblioteca sklearn.preprocessing para convertir las etiquetas de texto ('positivo', 'negativo') en un formato numérico que el modelo puede entender y procesar. El método fit_transform aprende el mapeo de las etiquetas a números (por ejemplo, 'negativo' a 0 y 'positivo' a 1) y luego transforma las etiquetas y en este formato numérico codificado (y_encoded). Luego, to_categorical se utiliza para convertir el vector de etiquetas codificadas en un formato binario (one-hot encoding) necesario para la clasificación en modelos de aprendizaje automático, especialmente cuando se utilizan funciones de activación como 'softmax' en la capa de salida de una red neuronal. Esto se hace porque el modelo necesita un objetivo de clasificación claramente definido para cada clase.
- Preprocesamiento de Datos: Se cargan los datos de las reseñas utilizando Pandas, y se preparan para el entrenamiento mediante tokenización y padding. Este parte utiliza TF-IDF para convertir textos en una representación numérica, limitando el número de términos a 2500, y luego almacena la matriz resultante en la variable X_tfidf. Esto es, comúnmente utilizado en tareas de procesamiento de lenguaje natural y aprendizaje automático para preparar datos de texto para su procesamiento.

Modelado

Se crea un objeto de modelo de regresión logística. En este caso, LogisticRegression es una clase proporcionada por scikit-learn que implementa un clasificador de regresión logística. La regresión logística es un algoritmo de clasificación que se utiliza para predecir la probabilidad de que una instancia pertenezca a una categoría en particular.
Dividir los datos en conjuntos de entrenamiento y prueba: En este fragmento de código, se está utilizando la función train_test_split de la biblioteca sklearn.model_selection para dividir los datos en dos conjuntos: uno para entrenar el modelo de aprendizaje automático y otro para probar su rendimiento. Los conjuntos de datos se dividen de la siguiente manera:

- X_train: Datos de características para el entrenamiento.
- X_test: Datos de características para las pruebas.
- Y_train: Etiquetas de entrenamiento (los resultados que el modelo intentará predecir durante el entrenamiento).
- Y_test: Etiquetas de prueba (usadas para evaluar la precisión de las predicciones del modelo después del entrenamiento).
- El parámetro test_size=0.2 indica que el 20% del conjunto de datos se reservará para pruebas, mientras que el 80% restante se utilizará para entrenar el modelo.
- El parámetro random_state=42 se establece para garantizar la reproducibilidad de la división; al fijar este parámetro, se asegura que cada vez que se ejecute el código, se obtendrá la misma división exacta de los datos, lo cual es crucial para la comparación de modelos y la validación de resultados en experimentos de aprendizaje automático.

Entrenamiento

Esto se realiza en de dos fases claves, en la construcción y entrenamiento de un modelo de aprendizaje profundo utilizando Keras, una biblioteca de redes neuronales de alto nivel que funciona sobre TensorFlow.

- X_test: La matriz de características de prueba. Contiene los datos que se usarán para evaluar el modelo. Similar a X_train, cada fila representa una instancia de prueba, y cada columna representa una característica.
- Y_test: El vector de etiquetas de prueba. Contiene las etiquetas correspondientes a cada instancia en el conjunto de prueba. Al igual que y_train, cada elemento en este vector está asociado a una fila en la matriz de características X_test.

El método score compara las predicciones del modelo con las etiquetas reales y calcula la precisión del modelo. En el contexto de clasificación, la precisión es la proporción de predicciones correctas respecto al total de predicciones.

Visualización

Se crea una aplicación Flask que carga el modelo entrenado y ofrece una API para recibir reseñas y devolver análisis de sentimientos. Se implementa la página web con HTML y JavaScript para interactuar con el backend, enviando reseñas y mostrando los resultados de la predicción como se muestra en la Figura 2.

Fig. 2. Visualización

Fuente: Elaboración propia

4. PRUEBAS Y RESULTADOS

El sistema se probó utilizando el conjunto de datos de prueba (X_test y Y_test) separado durante la fase de preprocesamiento. Estas pruebas tienen como objetivo evaluar cómo el modelo se desempeña con datos no vistos, lo que es esencial para medir la efectividad real del modelo en escenarios del mundo real. Se muestran tres ejemplos, positivo, negativo y ambiguo figuras 3, 4 y 5.

Ejemplos de Predicciones:

Reseña Positiva: "Una película emocionante y conmovedora con actuaciones destacadas."
Predicción del Modelo: Positivo
Resultado: Correcto

Fig. 3. Ejemplo 1

Fuente: Elaboración propia

Reseña Negativa: "Un guion débil y actuaciones mediocres. Una completa pérdida de tiempo."
Predicción del Modelo: Negativo
Resultado: Correcto

Fig. 4. Ejemplo 2

Fuente: Elaboración propia

Reseña Ambigua: "La película tenía momentos buenos, pero en general fue decepcionante."
Predicción del Modelo: Negativo/Positivo (dependiendo de cómo el modelo interprete el sentimiento predominante)
Resultado: Depende del contexto

Fig. 5. Ejemplo 3

Fuente: Elaboración propia

Análisis de Resultados

La eficacia del sistema se midió en términos de precisión, es decir, qué tan a menudo las predicciones del modelo coincidían con las etiquetas reales.

Se observaron los siguientes aspectos:

- **Precisión General:** El modelo alcanzó una precisión general del 85% en el conjunto de prueba. Esto indica que, de todas las reseñas analizadas, el 85% de las veces el modelo fue capaz de identificar correctamente el sentimiento como positivo o negativo. Este nivel de precisión es comparable con modelos previos de análisis de sentimientos basados en redes neuronales profundas. Por ejemplo, modelos similares utilizando LSTM (Long Short-Term Memory) y redes convolucionales (CNN) han reportado precisiones entre el 80% y el 88% en tareas de clasificación de texto [5]. Sin embargo, en comparación con modelos más avanzados basados en transformers como BERT [10] o RoBERTa [11], que alcanzan precisiones superiores al 90% en tareas similares, nuestro modelo muestra un rendimiento competitivo considerando la complejidad y el tamaño del dataset utilizado.

- Manejo de Ambigüedad: En el caso de reseñas con sentimientos mixtos o un vocabulario ambiguo, el modelo mostró un desempeño variable. Por ejemplo, en una reseña que decía 'La película tenía momentos emocionantes, pero en general fue aburrida', el modelo tendió a clasificarla como negativa, lo que sugiere una tendencia a priorizar ciertos indicadores de sentimiento

sobre otros. Estos resultados destacan una oportunidad para mejorar el modelo, tal vez incorporando análisis de contexto más profundo o técnicas de ponderación de palabras.

- Tiempo de Respuesta: El sistema demostró ser eficiente en términos de tiempo de respuesta, con un promedio de 0.5 segundos por reseña analizada. Esta rapidez asegura una experiencia de usuario fluida y efectiva en la aplicación web, permitiendo que los usuarios obtengan análisis de sentimientos casi en tiempo real. Este aspecto es particularmente importante en aplicaciones web donde la experiencia del usuario es crítica para el éxito del servicio.

- El modelo mostró una alta precisión en reseñas claramente positivas o negativas. Sin embargo, en casos de reseñas con tonos mixtos o neutrales, el modelo a veces mostraba limitaciones, destacando la complejidad inherente en la interpretación del lenguaje humano. Para analizar las causas de estos errores, se realizó un análisis detallado de las métricas de clasificación, incluyendo la matriz de confusión y los valores de precisión (precision), exhaustividad (recall) y puntuación F1 (F1-score).
- **Precisión por clase:** El modelo alcanzó una precisión del 88% en reseñas positivas y un 82% en reseñas negativas, pero solo un 65% en reseñas neutrales o ambiguas.
- **Errores comunes:** La mayoría de los errores se debieron a la presencia de sarcasmo, doble sentido y frases con múltiples emociones. Por ejemplo, reseñas como *"La película fue divertida, pero el final fue decepcionante"* fueron clasificadas incorrectamente debido al conflicto entre términos positivos y negativos.
- **Análisis de matriz de confusión:** La matriz de confusión reveló que el modelo tuvo una tasa de falsos positivos del 12% y una tasa de falsos negativos del 15%. Esto indica que el modelo tiende a sobrestimar las reseñas positivas y subestimar las reseñas negativas.
-

4. CONCLUSIONES Y RECOMENDACIONES

El proyecto demostró que es posible implementar un sistema efectivo de análisis de sentimientos para reseñas de películas utilizando técnicas de Procesamiento de Lenguaje Natural y redes neuronales. Los principales hallazgos y contribuciones del proyecto son:

- Desarrollo de un Modelo de PLN Eficiente: El modelo pudo clasificar con precisión la mayoría de las reseñas en categorías de sentimiento.

- Integración Exitosa en una Aplicación Web: La capacidad de procesar y analizar reseñas en tiempo real a través de una interfaz web.

- Base para Futuras Mejoras: El proyecto sienta las bases para futuras investigaciones y desarrollos, como la implementación de modelos más sofisticados o la inclusión de análisis de sentimientos más granulares.

Este proyecto contribuye al campo del PLN al proporcionar un caso práctico de cómo las técnicas de aprendizaje automático pueden aplicarse en el análisis de opiniones, un área de creciente interés tanto para empresas como para investigadores académicos.

5. REFERENCIAS

[1] I. Amazon, «Internet Movie Database,» 01 Agosto 2024. [En línea]. Available: https://www.imdb.com/es/.

[2] D. o. NBC Universal, «Rotten Tomatoes,» 06 Octubre 2024. [En línea]. Available: https://www.rottentomatoes.com/critics.

[3] P. R. Varma, «Natural Language Processing in the Era of BIG DATA,» *International Journal of Scientific Reasearch in Engineering and Management ,* vol. 08, pp. 01-05, 2024.

[4] E. Cambria, B. Schuller, Y. Xia y C. Havasi, «New Avenues in Opinion Mining and Sentiment Analysis,» *IEEE Intelligent Systems,* vol. 28, nº 2, 2013.

[5] L. Bing, Sentiment Analysis and Opinion Mining, Graeme Hirst, 2012.

[6] N. s. project., «Pandas,» Inc. Hosted by OVHcloud., 2025. [En línea]. Available: https://pandas.pydata.org/.

[7] Python Software Foundation , «Python,» 2021. [En línea]. Available: https://www.python.org/. [Último acceso: 2024].

[8] Tensor Flow, «Plataforma de extremo a extremo enfocada en el aprendizaje automático,» 2024. [En línea]. Available: https://www.tensorflow.org/about?hl=es-419.

[9] «Flask,» Created using Sphinx 8.1.3., 2024. [En línea]. Available: https://flask.palletsprojects.com/en/stable/#user-s-guide.

[10] J. Devlin, M.-W. Chang, K. Lee y K. Toutanova, «Pre-training of Deep Bidirectional Transformers for Language Understanding,» *Computer Science Computation and Language,* pp. 1-16, 2019.

[11] Y. Liu, M. Ott, N. Goyal y J. Du, «Computer Science: Computation and Language,» *A Robustly Optimized BERT Pretraining Approach. ,* 2019.