UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

TITULO DEL TRABAJO DE TITULACIÓN

Aplicación de Técnicas de Word Embedding para Detectar Emociones en Texto en Idioma Español Usando Algoritmos de Machine Learning y Deep Learning.

AUTOR

González Mora , Mayerli Anahi

PROYECTO DE UNIDAD DE INTEGRACIÓN CURRICULAR

Previo a la obtención del grado académico en
INGENIERO EN TECNOLOGÍAS DE LA INFORMACIÓN

TUTOR

Ing. Haz López Lídice Victoria, MSI.

Santa Elena, Ecuador

Año 2025

UPSE

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

TRIBUNAL DE SUSTENTACIÓN

Ing. José Sánchez Aquino, Mgt.
DIRECTOR DE LA CARRERA

Ing. Lídice Haz López, Msi.
TUTOR

Ing. Walter Orozco Iguasnia, Msc.
DOCENTE ESPECIALISTA

Ing. Marjorie Coronel Suárez, Mgt.
DOCENTE GUÍA UIC

UNIVERSIDAD ESTATAL PENÍNSULA

DE SANTA ELENA

FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

CERTIFICACIÓN

Certifico que luego de haber dirigido científica y técnicamente el desarrollo y estructura final del trabajo, este cumple y se ajusta a los estándares académicos, razón por el cual apruebo en todas sus partes el presente trabajo de titulación que fue realizado en su totalidad por GONZÁLEZ MORA MAYERLI ANAHI, como requerimiento para la obtención del título de Ingeniero en Tecnologías de la Información.

La Libertad, a los 19 días del mes de junio del año 2025

TUTOR

Firmado electrónicamente por:
LIDICE VICTORIA HAZ LOPEZ
Validar únicamente con FirmaEC

Ing. Haz López Lídice Victoria

UNIVERSIDAD ESTATAL PENÍNSULA
DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

Yo, **González Mora Mayerli Anahi**

DECLARO QUE:

El trabajo de Titulación, Aplicación de Técnicas de Word Embedding para Detectar Emociones en Texto en Idioma Español Usando Algoritmos de Machine Learning y Deep Learning previo a la obtención del título en Ingeniero en Tecnologías de la Información, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

La Libertad, a los 19 días del mes de junio del año 2025

EL AUTOR

Mayerli González

Mayerli Anahi González Mora

UNIVERSIDAD ESTATAL PENÍNSULA
DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

CERTIFICACIÓN DE ANTIPLAGIO

Certifico que después de revisar el documento final del trabajo de titulación denominado Aplicación de Técnicas de Word Embedding para Detectar Emociones en Texto en Idioma Español Usando Algoritmos de Machine Learning y Deep Learning, presentado por el estudiante, GONZÁLEZ MORA MAYERLI ANAHI fue enviado al Sistema Antiplagio, presentando un porcentaje de similitud correspondiente al 5%, por lo que se aprueba el trabajo para que continúe con el proceso de titulación.


INFORME DE ANÁLISIS
magister

UIC (González Mora Mayerli Anahi)

5% Textos sospechosos

< 1% Similitudes (ignorado)
< 1% similitudes entre comillas
0% entre las fuentes mencionadas
5% Idiomas no reconocidos
8% Textos potencialmente generados por la IA (ignorado)

Nombre del documento: UIC (González Mora Mayerli Anahi).docx
ID del documento: cb7c4d6d13d7922b4bd728741334de725e0d4f34
Tamaño del documento original: 2,67 MB

Depositante: LIDICE VICTORIA HAZ LÓPEZ
Fecha de depósito: 16/6/2025
Tipo de carga: interface
fecha de fin de análisis: 16/6/2025

Número de palabras: 21.711
Número de caracteres: 154.134

Ubicación de las similitudes en el documento:

TUTOR

Firmado electrónicamente por:
LIDICE VICTORIA HAZ LOPEZ
Validar únicamente con FirmaEC

ING. HAZ LÓPEZ LÍDICE, MSI

UNIVERSIDAD ESTATAL PENÍNSULA

DE SANTA ELENA

FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

AUTORIZACIÓN

Yo, **GONZÁLEZ MORA MAYERLI ANAHI**

Autorizo a la Universidad Estatal Península de Santa Elena, para que haga de este trabajo de titulación o parte de él, un documento disponible para su lectura consulta y procesos de investigación, según las normas de la Institución.

Cedo los derechos en línea patrimoniales del presente trabajo de titulación con fines de difusión pública, además apruebo la reproducción dentro de las regulaciones de la Universidad, siempre y cuando esta reproducción no suponga una ganancia económica y se realice respetando mis derechos de autor.

Santa Elena, a los 19 días del mes de junio del año 2025

EL AUTOR

Mayerli Anahi González Mora

AGRADECIMIENTO

Agradezco, en primer lugar, a Dios, por darme la fuerza, la salud y la sabiduría necesarias para culminar esta etapa tan importante de mi vida.

A mis padres, por ser mi mayor inspiración, por su amor incondicional, por apoyarme en todo momento y por enseñarme el valor del esfuerzo y la perseverancia. Gracias por ser el pilar de mi formación personal y profesional.

A mi Asesora de Tesis, la Ing. Lídice Haz, por su acompañamiento, paciencia y por guiarme con profesionalismo durante este proyecto. Gracias por su orientación y apoyo.

A mis amigos que formaron parte de esta etapa universitaria, especialmente a Melany Reyes y Angie Carvajal.

Mayerli Anahi, González Mora

DEDICATORIA

Le dedico este proyecto realizado a mi familia, quienes me han apoyado a lo largo de este camino con su paciencia, comprensión y por creer en mí. Gracias por ser buenos padres. También a Milo, mi gato, por su silenciosa pero constante compañía.

Mayerli Anahi, González Mora

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

La presente investigación aborda la detección automática de emociones en textos en español mediante técnicas de Word Embedding (Word2Vec y FastText) combinadas con algoritmos de Machine Learning (SVM, Regresión Logística, Naïve Bayes Gaussiano) y Deep Learning (BiLSTM, GRU, LSTM). El objetivo principal fue aplicar y comparar estos enfoques para identificar múltiples emociones en comentarios extraídos de un conjunto de datos de Kaggle, reetiquetados con base en criterios psicolingüísticos. La metodología fue cuantitativa, descriptiva y experimental, utilizando Google Colab como entorno de desarrollo. Los resultados demostraron que los modelos de Deep Learning con vectores FastText ofrecieron mejor desempeño, destacando BiLSTM por su capacidad para capturar relaciones contextuales en secuencias de texto. Se concluye que la combinación de representaciones semánticas y modelos secuenciales mejora significativamente la detección de emociones, aportando al desarrollo de herramientas automatizadas para el análisis emocional en idioma español.

Palabras claves: Word Embedding, Machine Learning, Deep Learning.

ABSTRACT

The present research addresses the automatic detection of emotions in Spanish texts using Word Embedding techniques (Word2Vec and FastText) combined with Machine Learning (SVM, Logistic Regression, Naïve Bayes Gaussian) and Deep Learning (BiLSTM, GRU, LSTM) algorithms. The main objective was to apply and compare these approaches to identify multiple emotions in comments extracted from a Kaggle dataset, relabeled based on psycholinguistic criteria. The methodology was quantitative, descriptive and experimental, using Google Colab as the development environment. The results showed that Deep Learning models with FastText vectors offered better performance, with BiLSTM standing out for its ability to capture contextual relationships in text sequences. It is concluded that the combination of semantic representations and sequential models significantly improves emotion detection, contributing to the development of automated tools for emotional analysis in Spanish.

Keywords: Word Embedding, Machine Learning, Deep Learning.

INTRODUCCIÓN

En la actualidad, los usuarios generan grandes volúmenes de contenido textual a través de las redes sociales y otras plataformas digitales, donde las personas comparten constantemente opiniones y experiencias. Esta información se ha convertido en una fuente valiosa de análisis para diversas áreas como la psicología, la atención al cliente, el marketing y la política. Sin embargo, analizar este contenido de forma manual es una tarea compleja y poco eficiente, por lo que se ha vuelto importante el uso de herramientas computacionales capaces de procesar el lenguaje humano para identificar las emociones de manera automática.

El reconocimiento automático de emociones en texto presenta múltiples desafíos, especialmente cuando un mismo mensaje puede expresar diversas emociones a la vez, lo que lo convierte en una tarea de clasificación multietiqueta. En el caso del idioma español, dicha dificultad aumenta, debido a su complejidad lingüística, la ambigüedad de términos y las variaciones regionales, además de contar con menos recursos para el análisis en comparación con otros idiomas como el inglés.

Esta investigación tiene como finalidad aplicar técnicas de Word Embedding, específicamente FastText y Word2Vec, que permiten convertir palabras en vectores numéricos preservando su significado semántico. Estas representaciones vectoriales se utilizan para entrenar y evaluar modelos de clasificación. Para ello, se utilizan algoritmos de Machine Learning como SVM, regresión logística y Naïve Bayes Gaussiano, y para Deep Learning, el BiLSTM, GRU y LSTM. El objetivo es comparar el rendimiento de estos modelos y determinar cuál ofrece mejores resultados en la detección de emociones en textos en español.

El estudio adopta un enfoque cuantitativo, descriptivo y experimental, y se desarrolla en la plataforma Google Colab. La metodología se divide en cinco fases: recolección de datos, preprocesamiento, vectorización de palabras, entrenamiento y evaluación de los resultados. Para la evaluación del desempeño de los modelos se emplean métricas como la precisión, recuperación (recall), f1-score y el área bajo la curva (AUC), además, la matriz de confusión de un modelo con la finalidad de visualizar su rendimiento real.

El documento consta de tres capítulos: en el capítulo 1 se aborda la fundamentación teórica y metodología, los objetivos y el alcance; en el capítulo 2, se explican los conceptos y el contexto legal; por último, en el capítulo 3 se describe el desarrollo práctico y la evaluación de los resultados.

CAPÍTULO 1. FUNDAMENTACIÓN

1.1. Antecedentes

El desarrollo de la web 2.0 ha provocado un crecimiento exponencial en la cantidad de información disponible en internet. Además, el rápido aumento del contenido generado por los usuarios en plataformas como redes sociales, blogs o foros ha permitido que las personas compartan sus opiniones personales sobre diversos aspectos de la vida cotidiana[1] Este auge ha transformado la forma en que interactuamos, dando lugar a una abundancia de datos que requieren herramientas especializadas para ser procesados y comprendidos.

Actualmente, una gran cantidad de información se encuentra en formato no estructurado, lo que exige el uso de técnicas de procesamiento automático para su tratamiento efectivo[1] En este contexto, el análisis de sentimientos ha cobrado relevancia como una disciplina de procesamiento de del Lenguaje Natural (PLN), enfocada en extraer y clasificar emociones o polaridades dentro del texto.

Los primeros trabajos en el campo de análisis de sentimiento fueron desarrollados a partir de 2000 que se basaron en enfoques que abarcaban dos paradigmas principales: el simbólico, que se centra en el uso de recursos léxicos y reglas para manipular símbolos, y el estadístico, que hace uso de técnicas de aprendizaje automático[2]. Si bien ambos enfoques fueron fundamentales para el desarrollo del análisis de emociones, su alcance era limitado, ya que dependían en gran medida de la calidad de los recursos o de la necesidad de grandes cantidades de datos etiquetados, lo que los hacía poco escalables, abriendo el camino para nuevas soluciones.

El trabajo de Goldberg analiza la evolución de los métodos basados en redes neuronales para el procesamiento del lenguaje natural (PLN), destacando el impacto de Word Embedding en la representación de semántica de palabras. Estos modelos convierten palabras en vectores dentro de un espacio matemático, permitiendo identificar relaciones semánticas y sintácticas. Para tareas de predicción y análisis de secuencias textuales, se han utilizado algoritmos de Machine Learning, como SVM y Naïve Bayes, así como modelos avanzados de Deep Learning, como redes convolucionales (CNN) y recurrentes (RNN), las cuales han demostrado ser

efectivos en múltiples aplicaciones, incluyendo el análisis de sentimientos, detección de emociones y generación de texto[3].

El análisis de sentimientos en textos abarca diversas tereas, como la clasificación de polaridad (positivo o negativo), la identificación de intensidades emocionales (muy negativo, negativo, neutral, positivo, muy positivo) y la identificación de emoción básicas (enojo, alegría, tristeza, etc.). Sin embargo, no todas han recibido el mismo nivel de desarrollo. En este contexto, el Taller de Análisis de Sentimientos en español de 2020 destacó la falta de avance en la detección de emociones. Aunque la clasificación de polaridad está bien establecida, la detección de emociones sigue siendo un área poco explorada debido a su complejidad. Como resultado, se enfatizó la necesidad de impulsar el desarrollo de este campo para el idioma español[4]. Por lo tanto, la presente investigación busca contribuir en este especto, mediante el uso de técnicas de Word Embedding y algoritmos avanzados de Machine Learning y Deep Learning.

En la actualidad, vivimos en un mundo digital inundado de información, especialmente en plataformas digitales, donde contienen una gran cantidad de opiniones y emociones expresadas por los usuarios. Sin embargo, analizar manualmente esta enorme cantidad de datos resulta una tarea compleja y poco práctica debido a la magnitud y la naturaleza no estructurada de los datos. Además, la mayoría de los sistemas de análisis emocional están diseñados para el idioma inglés, lo que limita su aplicabilidad y precisión al trabajar con textos en español[5].

A continuación, se presentan los siguientes trabajos que se tomaron como referencia para el desarrollo de esta investigación:

El trabajo realizado por Julia Scotto resalta la importancia del análisis de sentimientos en el contexto de las redes sociales, debido a la gran cantidad de datos generados en las plataformas como Twitter y Facebook. En su investigación demostró la efectividad de las técnicas de Word Embedding y evidenció que su integración con modelos avanzados de aprendizaje profundo mejoró significativamente la precisión en la clasificación de emociones. Los resultados obtenidos mostraron un desempeño superior en la detección de emociones en texto

en español, lo que resulta esencial para el desarrollo de aplicaciones orientadas a la inteligencia empresarial y la toma de decisiones informadas[6]

La investigación de Jesús Herrero desarrolló un clasificador de sentimientos para tweets en español, utilizando técnicas de aprendizaje profundo. El proceso incluyó la recopilación y procesamiento de datos, seguido de la implementación de modelos como neuronales convolucionales (CNN) y redes neuronales recurrentes (RNN), para clasificar los sentimientos en tres categorías: positivo, negativo y neutro, el modelo alcanzó una precisión superior al 75%[7]

El estudio de Santiago Palmero abordó el análisis de sentimientos en español, utilizando tanto métodos tradicionales de Machine Learning como técnicas avanzadas de Deep Learning. A pesar de las limitaciones como la escasez de recursos y herramientas específicas, los resultados mostraron que un preprocesamiento adecuado mejora la precisión de los modelos. Además, las redes neuronales profundas, en particular las GRU y LSTM, superaron a los métodos tradicionales, como regresión logística y las máquinas de vectores soporte (SVM)[8]. La investigación resaltó la importancia de las capas de embedding en redes profundas, ya que permiten representar las palabras de manera más precisa en función de su significado.

En conclusión, se ha observado un avance significativo en el análisis de emociones en texto mediante el uso de Word Embedding y algoritmos de Machine Learning y Deep Learning. A pesar de los progresos en la clasificación de polaridad y análisis de sentimientos, la identificación precisa de emociones específicas sigue representado un desafío, especialmente en idioma español. Además, los estudios previos han demostrado que la integración de técnicas de Word Embedding con modelos de Machine Learning y Deep Learning ha mejorado notablemente la precisión en la detección de emociones en texto.

1.2. Descripción del Proyecto

En el ámbito del Procesamiento del Lenguaje Natural (NLP), la identificación de emociones en texto, especialmente en aquellos provenientes de plataformas digitales, representa un reto complejo debido a la riqueza y diversidad del lenguaje

humano. Este proyecto tiene como objetivo aplicar y evaluar técnicas de Word Embedding para la clasificación de emociones en textos en español mediante algoritmos de Machine Learning y Deep Learning. Para representar las palabras en un espacio numérico que capture relaciones semánticas, se emplean los modelos de Word2Vec y FastText. Estas representaciones permiten identificar similitudes contextuales en el texto y mejorar el desempeño de los modelos de clasificación.

El proyecto se desarrolla utilizando un conjunto de datos en español obtenido de la plataforma Kaggle, para fortalecer el análisis y la validez psicológica de la clasificación, se realizó una reetiquetación de los comentarios del dataset original. Esta reetiquetación se fundamentó en las características emocionales descritas en estudios psicolingüísticos, como el trabajo de Quevedo-Aguado e Iraegui-Torralbo. Cada comentario fue etiquetado con hasta tres emociones, según su contenido emocional.

El dataset contiene comentarios etiquetados con una, dos o hasta tres emociones por texto. Las once emociones consideradas para este estudio son las siguientes: orgullo, egocentrismo, soberbia, sensualidad, disgusto/desagrado, antipatía, rechazo, enfado/enojo, odio, aceptación, independencia. Debido al desbalance en la distribución de clases, estas emociones fueron agrupadas en siete clases de emociones, basadas en criterios psicológicos y semánticos.

La metodología empleada en este estudio es descriptiva y experimental. Se lleva a cabo un preprocesamiento que incluye funciones como la conversión del texto a minúscula, la eliminación de caracteres especiales, números, emojis, stopwords y la tokeniacion. Posteriormente, se aplican las técnicas Word Embedding para convertir los comentarios en representaciones vectoriales. Estos vectores sirven como entrada para modelos de clasificación basados en algoritmos de Machine Learning (SVM, Regresión Logística y Naïve Bayes Gaussiano (GaussianNB)), y de Deep Learning (BiLSTM, GRU y LSTM), con el propósito de evaluar su rendimiento en la tarea de detección de emociones. Además de un enfoque cuantitativo, dado que se basa en el análisis numérico de las métricas de evaluación de los modelos.

El desarrollo del proyecto se lleva a cabo en Google Colab, utilizando librerías de Python, para medir la efectividad de los modelos, se emplean métricas como precisión, recall, F1-score y AUC (Área Bajo la Curva), que permitirán comparar los mejores enfoques y determinar cual ofrece mejores resultados, además se utilizara la matriz de confusión de un modelo para visualizar su rendimiento real.

Para facilitar la aplicación práctica, se desarrolló una interfaz sencilla utilizando la librería de Streamlit, que emplea el modelo en combinación con la técnica de Word Embedding que obtuvo mejor rendimiento. Esta interfaz permite ingresar un comentario y obtener una predicción de hasta tres emociones, lo que ayuda a visualizar y validar de forma interactiva los resultados del modelo.

No obstante, se reconoce que el modelo puede presentar errores en la clasificación de emociones, especialmente en este contexto de múltiples emociones o aquellas menos representadas en el conjunto de datos. Esta limitación puede deberse a la ambigüedad del lenguaje natural, y la complejidad del idioma. Por ello, aunque el sistema es funcional y ofrece predicciones razonables, no garantiza una precisión absoluta en todos los escenarios, lo que sugiere la necesidad de futuras mejoras basadas en un mayor volumen de datos.

El alcance de este proyecto se limita a textos escritos en español y se enfoca en un conjunto predefinido de emociones específicas. Finalmente, los resultados obtenidos se analizarán mediante visualizaciones gráficas y tablas, lo que facilitará la comparación del desempeño de los modelos. La evaluación de estas métricas permitirá identificar la mejor combinación de técnica de Word Embedding y algoritmo de clasificación para la detección de emociones en textos en español.

El proyecto incluye cinco fases a realizar[9]:

Fase de Recolección de Datos.

En esta fase se obtiene y prepara un conjunto de datos adecuado para el análisis de emociones en textos.

- El dataset utilizado proviene de Kaggle, una plataforma que proporciona dataset públicos, y contiene comentarios extraídos de diversas fuentes digitales.

- Para fortalecer el sustento teórico de la clasificación emocional, el dataset fue reetiquetado con base a sus características, basándose en la propuesta psicolingüística de Quevedo-Aguado e Iraegui-Torralbo, lo cual permite organizar las emociones según criterios psicológicos más robustos.
- Los comentarios del dataset contiene etiquetas con una o más emociones, permitiendo un máximo de tres emociones por texto.
- Las emociones consideradas fueron: orgullo, egocentrismo, soberbia, sensualidad, disgusto/desagrado, antipatía, rechazo, enfado/enojo, odio, aceptación, independencia.
- Debido al desbalance en la distribución de clases, las emociones fueron agrupadas en 7 clases desde un enfoque psicológico y semántico.

Fase de Preprocesamiento.

En la fase de preprocesamiento se limpia, transforman y preparan los datos para ser utilizados en los modelos de aprendizaje automático y aprendizaje profundo.

- Limpieza de datos: Se elimina caracteres especiales, signos de puntuación, números, espacios vacíos, emojis y stopwords, ya que no aportan valor semántico al análisis.
- Tokenización: Se segmenta el texto en unidades significativas (tokens) para facilitar la interpretación de las estructuras del lenguaje.

Fase de vectorización de palabras:

En esta fase, las palabras del texto se transforman en representaciones vectoriales mediante las técnicas de Word2Vec y FastText, lo que permite a los modelos comprender el significado semántico de las palabras.

- Word2Vec y FastText: Estas técnicas transforman las palabras en vectores numéricos, de manera que aquellas con significados similares tienen representaciones cercanas entre sí.

- Promedio de vectores: Se calcula el promedio de los vectores de las palabras en cada comentario para representar cada texto como un único vector de características.
- Secuencia de palabras: Se utiliza una representación secuencial, lo que permite preservar el orden de las palabras en el texto.

Fase de Entrenamiento.

Durante esta fase, se entrenaron diversos modelos de aprendizaje automático y profundo para detectar emociones en textos.

- Modelos de Machine Learning: Se implementaron Máquinas de Soporte Vectorial (SVM), Regresión Logística y el modelo Naïve Bayes Gaussiano), para la clasificación de emociones.
- Modelos de Deep Learning: Se emplean los modelos BiLSTM, GRU y LSTM, capaces de capturar la dependencia contextual y la estructura secuencial del texto.
- División del conjunto de datos: El dataset se dividió en un 80% para entrenamiento y un 20% para validación, con el fin de evaluar el desempeño de los modelos entrenados.

Fase de Evaluación.

En esta última fase se mide la efectividad de los modelos entrenados en la clasificación de emociones.

- Métricas de evaluación: Se utilizaron métricas como precisión, recall, F1-score y AUC (área bajo la curva) para analizar el rendimiento de los modelos.
- Comparación de modelos: Se realiza un análisis comparativo mediante tablas entre los modelos de Machine Learning y Deep Learning para determinar cual ofrece mejores resultados.
- Visualización de resultados: Se utilizaron gráficos y tablas para facilitar la interpretación y comparación de los resultados obtenidos.

Especificaciones de la Computadora Personal

- Procesador: Intel(R) Core(TM) i3-3240 CPU @ 3.40GHz
- Memoria RAM: 4,00 GB
- Disco duro: 222 GB

El presente Proyecto contribuirá en la línea de investigación Tecnología y Sistemas de la Información (TSI), sublínea Inteligencia Computacional.

1.3. Objetivos del Proyecto

1.3.1. Objetivo General

Aplicar técnicas de Word Embedding y modelos de clasificación basados en Deep Learning y Machine Learning para la detección de emociones en textos en español, con el propósito de mejorar la comprensión y análisis de la información emocional.

1.3.2. Objetivos Específicos

- Realizar un estudio bibliográfico para describir el funcionamiento de las técnicas de Word Embedding en NLP.
- Aplicar las técnicas de Word Embedding para extraer las principales características semánticas de un texto en el idioma español.
- Realizar pruebas experimentales para evaluar la eficiencia de las técnicas de Word Embedding de los modelos de Deep Learning y Machine Learning.
- Evaluar el rendimiento de los modelos de clasificación mediante la comparación de métricas de desempeño.

1.4. Justificación del Proyecto

En la actualidad, el análisis de emociones ha adquirido una relevancia significativa en diversas áreas del conocimiento. En el ámbito político, resulta útil para comprender las opiniones del público respecto a políticas y candidatos. En el contexto educativo, constituye una herramienta clave para interpretar las emociones de los estudiantes y, en consecuencia, mejorar la calidad de enseñanza-aprendizaje. En el sector de la salud, se emplea para analizar las emociones de los pacientes, lo que contribuye así a una atención medica más empática y personalizada. En el

entorno empresarial, facilita la gestión de la reputación corporativa y el monitoreo del comportamiento del mercado.

El análisis automatizado de las opiniones del público representa una oportunidad valiosa para empresas como para entidades gubernamentales, al permitir conocer la percepción de la audiencia e identificar tendencias tanto positivas como negativas, es decir, las emociones subyacentes en los textos. Sin embargo, el elevado volumen de información generado por los usuarios, generalmente no estructurada, dificulta su procesamiento manual. Además, los métodos tradicionales como encuestas, talleres y entrevistas, aunque útiles, presentan diversas limitaciones como: altos costos, largos tiempos de ejecución, sesgos en los resultados y baja participación[10]

En este contexto, la aplicación de técnicas de Word Embedding como Word2Vec y FastText adquiere especial relevancia para la detección automática de emociones en textos escritos en idioma español. Estas técnicas permiten representar las palabras en espacios vectoriales que capturan tanto su tanto su significado como las relaciones semánticas entre ellas, facilitando así el análisis emocional desde un enfoque computacional.

El presente proyecto contribuir a la limitada cantidad de estudios centrados en el análisis de emociones en textos en español mediante la aplicación de técnicas avanzadas. Al incorporar algoritmos de Machine Learning como SVM, Naïve Bayes Gaussiano (GaussianNB) y Regresión Logística, y de Deep Learning como BiLSTM, LSTM y GRU, el estudio permitirá realizar una comparación experimental entre diversos enfoques de clasificación multietiqueta. Este análisis tiene como finalidad contribuir al desarrollo de herramientas más precisas y robustas en el campo del Procesamiento del Lenguaje Natural (PLN).

En conclusión, este proyecto radica en la importancia de contar son soluciones más efectivas para identificar emociones en textos en español. Este enfoque resulta relevante no solo desde el punto de vista académico, sino también en su aplicación práctica, ya que puede tener un impacto significativo en áreas como la comunicación digital, el marketing, la educación y el análisis de comportamiento en diversas plataformas digitales. Con esta investigación, se busca contribuir en el

campo del procesamiento de lenguaje natural, promoviendo el desarrollo de tecnologías que ayuden a interpretar de manera más precisas las emociones humanas en entornos digitales.

El presente proyecto de investigación se alinea al Plan de creación de oportunidades en el Eje social.

- **Objetivo 7.-** "Potenciar las capacidades de la ciudadanía y promover una educación innovadora, inclusiva y de calidad en todos los niveles"[11]
- **Política 7**.1**.- "**Promover la modernización y eficiencia del modelo educativo por medio de la innovación y el uso de herramientas tecnológicas"[11].

1.5. Alcance del Proyecto

El alcance de esta propuesta de investigación abarca la aplicación de técnicas de Word Embedding para detectar emociones en texto en idioma español, mediante el uso de algoritmos de Machine Learning y Deep Learning. En particular, se emplearán los modelos de Word2Vec y FastText.

El desarrollo del proyecto abarca cinco fases las cuales son las siguientes:

Fase 1: Recolección de datos

Durante la primera fase se llevará a cabo la recopilación de los datos necesarios para el entrenamiento y evaluación de los modelos de clasificación de emociones. Se emplea un conjunto de datos proveniente de la plataforma Kaggle, el cual contiene comentarios en español extraídos de diversas fuentes digitales, reetiquetado con una o hasta tres emociones por comentario.

Las emociones que se consideran en esta investigación son: orgullo, egocentrismo, soberbia, sensualidad, disgusto, antipatía, rechazo, enfado/enojo, odio, aceptación e independencia. Donde se realiza una agrupación semántica de las emociones, quedando 7 clases. El dataset utilizado está limitado a comentarios de textos en español, lo que restringe la aplicación del modelo a este idioma en específico.

Fase 2: Preprocesamiento

Durante esta etapa, se llevan a cabo una serie de transformaciones con el fin de preparar los datos para su uso en los modelos de aprendizaje automático. El procesamiento incluye la limpieza del texto donde primero se hace la conversión a minúscula, después, se eliminan los caracteres especiales, signos de puntuación, números, emojis y palabras vacías (stopwords), debido a que estos elementos no aportan información relevante al análisis emocional. Posteriormente, se lleva a cabo la tokenización, dividiendo los comentarios en pequeñas unidades llamadas tokens, para facilitar su interpretación.

Fase 3: Vectorización de palabras

En esta fase, las palabras contenidas en los comentarios se transforman en representaciones vectoriales mediante las técnicas de Word Embedding, específicamente Word2Vec y FastText. Estas técnicas permiten que las palabras con significados semánticamente similares se ubiquen cerca unas de otras dentro de un espacio vectorial, lo que facilita el análisis semántico del texto.

Para los modelos de Machine Learning, cada comentario se representa mediante el promedio de los vectores de sus palabras, generando así un único vector que resume sus características semánticas. En cambio, en los modelos de Deep Learning, se conserva la secuencia completa de vectores por palabra, lo que permite al modelo procesar la estructura interna del texto y capturar relaciones contextuales y secuenciales entre los términos.

Fase 4: Entrenamiento

Durante la fase se entrenarán diversos modelos de clasificación con el objetivo de identificar las emociones en los textos. Se utilizan algoritmos de Machine Learning como Máquinas de Soporte Vectorial (SVM), Regresión Logística y Naïve Bayes Gaussiano (GaussianNB); y modelos de Deep Learning como BiLSTM, GRU y LSTM, los cuales son capaces de capturar dependencias contextuales y secuenciales entre las palabras

El dataset se dividió en subconjuntos de 80% entrenamiento y 20% validación, lo que permitirá evaluar el desempeño de los modelos a lo largo de todo el proceso.

Además, dado que el conjunto de datos presenta un desbalance significativo entre clases emociones, se optó por aplicar la estrategia de agrupación semántica, unificando emociones con significados similares.

Fase 5: Evaluación

El desempeño de los modelos se evalúa utilizando métricas como precisión, recall, F1-score y el área bajo la curva (AUC). El propósito de este análisis es determinar qué tan bien los modelos entrenados son capaces de clasificar las emociones en los textos. Además, mediante tablas se realiza una comparación entre los enfoques de Machine Learning y Deep Learning con base a los resultados obtenidos, para identificar cuál de los modelos ofrece un mejor rendimiento.

1.5.1 Población y Muestra

En este proyecto, la población está constituida por textos escritos en español, la muestra utilizada corresponde a un conjunto de datos público extraído de la plataforma Kaggle, que contiene comentarios en español etiquetados. Estos comentarios fueron reetiquetados manualmente conforme a criterios psicolingüísticos, permitiendo la asignación de hasta tres emociones por texto.

1.6. Metodología del Proyecto

1.6.1. Metodología de Investigación

La metodología de investigación aplicada en este proyecto adopta un enfoque cuantitativo, descriptivo y experimental[12]. A continuación, se describen las características de cada tipo de investigación y su aplicación en el desarrollo del trabajo.

Enfoque Cuantitativo

Este enfoque se centra en la medición de las variables numéricas relacionadas con el rendimiento de los modelos en las tareas de clasificación[12]. La evaluación de las técnicas implementadas se realiza mediante el uso de métricas cuantitativas, lo que permite comparar los resultados obtenidos y seleccionar el más educado para el objetivo planteado. Durante todo del proceso experimental, se lleva a cabo un análisis estadístico utilizando métricas como precisión, recall y F1-score, curva

AUC (área bajo la curva), lo que proporciona una base sólida para validar la eficiencia los modelos entrenados.

Enfoque Descriptivo

El enfoque descriptivo[12] se emplea principalmente en las fases iniciales del estudio, con el fin de analizar las técnicas de Word Embedding, específicamente Word2Vec y FastText, así como los modelos de clasificación Machine Learning y Deep Learning. A través de este enfoque, se describe el funcionamiento de dichas técnicas, su aplicación para generar representaciones vectoriales del texto y su relevancia en la clasificación de emociones. Asimismo, se describen las características del conjunto de datos, la distribución de las emociones y las técnicas de procesamiento aplicadas, de manera que facilita la interpretación de los resultados que se obtienen durante la fase experimental.

Enfoque experimental

El enfoque experimental[12], orienta el desarrollo práctico del proyecto, dado que se centra en la aplicación y evaluación de modelos de aprendizaje automático y profundo para detectar emociones en texto en español. A lo largo de las distintas fases del estudio, se llevan a cabo pruebas controladas con el objetivo de medir el rendimiento de los modelos, evaluando su desempeño en función a los resultados obtenidos del conjunto de datos. Este enfoque permite realizar ajustes iterativos durante el desarrollo, adaptando las técnicas y modelos según el comportamiento observado.

1.6.2. Beneficiarios del Proyecto

El presente proyecto de investigación tiene impacto en varias áreas, entre los beneficiarios podemos destacar:

La implementación de sistemas capaces de detectar las emociones de las plataformas digitales tales como, las redes sociales, sitios de opiniones y foros, beneficiara las los usuarios. Estas herramientas ayudarán a las empresas a entender mejor las necesidades de los usuarios, de modo que resultaría en una mejora en la experiencia digital que ofrecen.

Las empresas y organizaciones que deseen mejorar la relación con sus clientes a través del análisis de sentimientos, podrán detectar y comprender emociones expresadas en texto, lo que permitirá adaptar estrategias de comunicación, mejorar la atención al cliente, ajustar productos o servicios para satisfacer mejor las expectativas de sus consumidores.

El proyecto contribuirá a la línea de investigación en Procesamiento del Lenguaje Natural (NLP), proporcionando un estudio comparativo sobre la eficacia de Word Embedding para la detección de emociones en español. Los resultados y la metodología empleada servirán como base para futuros estudios en áreas como el análisis de sentimientos, minería de opiniones.

En el ámbito de la ciberpsicología, tiene un impacto relevante al proporcionar a los profesionales encargados del bienestar emocional una herramienta para analizar el estado emocional de los usuarios en plataformas digitales. La comprensión del lenguaje emocional en entornos virtuales facilitará la intervención psicológica en línea, mejorando la capacidad de respuesta de los psicólogos antes necesidades de los usuarios. Esto será clave para la implementación de intervenciones psicológicas más efectivas en espacios digitales, permitiendo una detección y asistencia más precisa y oportuna.

1.6.3. Variables

Variable independiente: Técnicas de Word Embedding (Word2Vec y FastText).

Variable dependiente: Desempeño de los modelos en la Detección de emociones en textos.

1.6.4. Recolección y Procesamiento de la Información

1.6.4.1 Técnicas de Recolección de Información

La recolección de la información se basó en la obtención de datos digitales a partir de un dataset etiquetado disponible de la plataforma Kaggle , donde los comentarios fueron reetiquetados siguiendo criterios psicolingüísticos, permitiendo asignar hasta tres emociones por texto. Esta reetiquetación se fundamentó en la propuesta de Quevedo-Aguado e Iraegui-Torralbo, lo que permitió organizar las emociones de manera más coherente con la naturaleza del lenguaje emocional. Para el

procesamiento y análisis de datos, se utilizaron herramientas como Google Colab para la ejecución del código, pandas para la manipulación de la información y bibliotecas como NLTK y Spacy para el procesamiento del texto.

Dataset	
Cantidad de comentarios	2531

Tabla 1: Cantidad de comentarios del Dataset: Elaboración propia

Distribución de los Datos	
7 Clases (emociones)	Cantidad
Rechazo_antipatia_disgusto	868
aceptacion	854
orgullo	850
Soberbia_egocentrismo	758
independencia	745
Enfado_enojo_odio	323
sensualidad	226
Total	4624

Tabla 2: Tabla de Distribución de los datos de las 7 clases: Elaboración propia

1.6.4.2 Técnicas para el Procesamiento de la Información

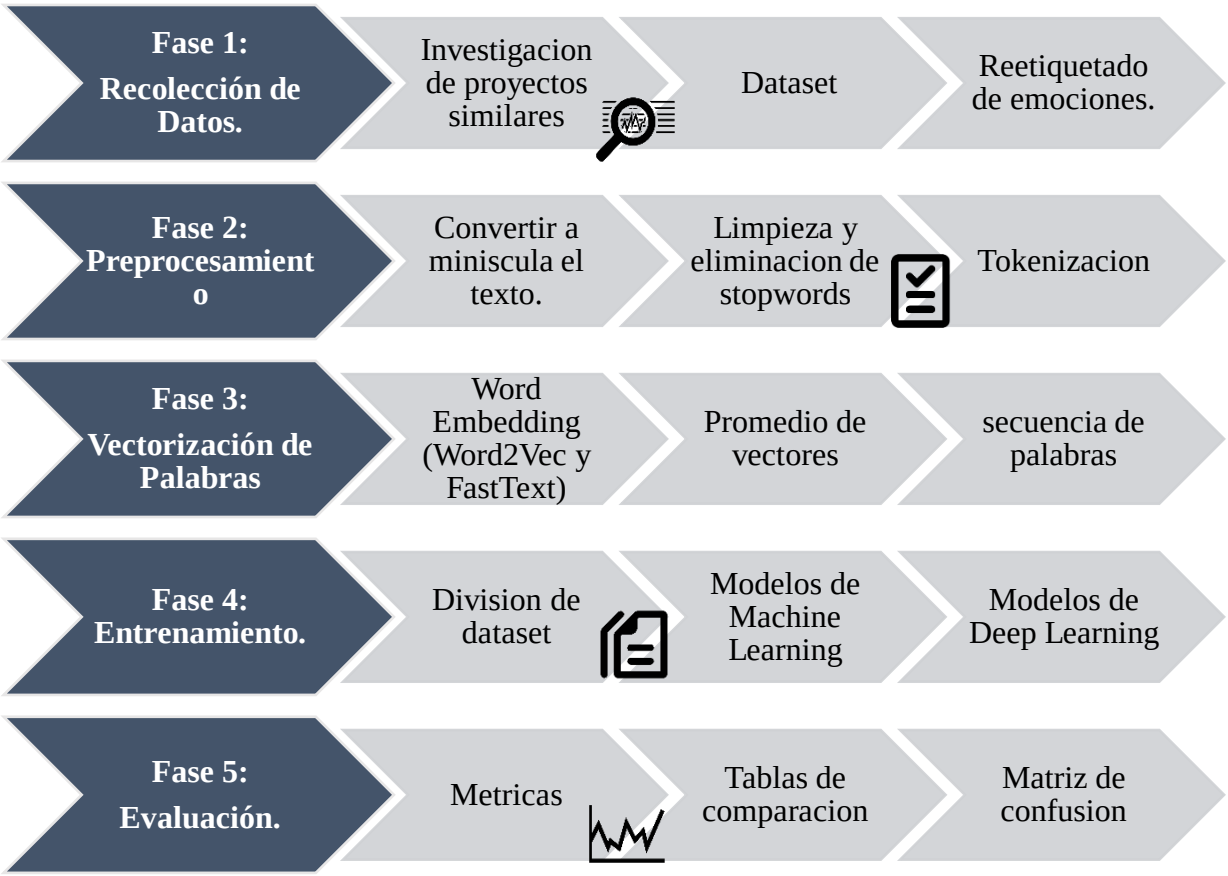
El conjunto de datos debe someterse a un procedimiento adecuado para transformarlo en un formato que permita su análisis mediante las técnicas Machine Learning y Deep Learning. Este procesamiento de la información se realiza en varias etapas las cuales son las siguientes:

Limpieza de datos: Consiste en la eliminación de aquello que no aporta valor semántico y que podría interferir con los modelos de aprendizaje. Entre ellos se encuentran los caracteres especiales, tildes, signos de puntuación, emojis y stopwords.

Tokenización: Se trata de dividir el texto en unidades más pequeñas, conocidas como tokens.

Vectorización de Palabras mediante Word Embedding: Para transformar los términos en representaciones numéricas, se utiliza Word2Vec y FastText, técnicas permiten capturar relaciones semánticas entre palabras en un espacio multidimensional. Para los Modelos de Machine Learning, cada comentario fue representado mediante el promedio de los vectores de sus palabras, obteniendo un vector por texto y para Deep Learning, se utilizó la secuencia de vectores por palabra.

1.7. Metodología de desarrollo



Para el desarrollo de este proyecto, se sigue una metodología estructurada en cinco fases ,basada en el procedimiento propuesto en el trabajo de Desirée García Soriano[9],con el fin de garantizar un análisis riguroso y sistemático en la detección de emociones en textos en español. La metodología adoptada es de carácter cuantitativo, descriptivo y experimental, permitiendo evaluar el desempeño de distintas técnicas de Word Embedding, así como modelos de aprendizaje automático y profundo para la clasificación de emociones. Cada fase abarca desde la recolección y preprocesamiento de datos hasta la vectorización, entrenamiento y evaluación de los modelos, asegurando la implementación completa y estructurada del proceso.

CAPÍTULO 2. PROPUESTA

2.1. Marco Contextual

En los últimos años, la inteligencia artificial ha impulsado el desarrollo de nuevas tecnologías basadas en el aprendizaje automático y las redes neuronales. Estas técnicas han mejorado la capacidad de los sistemas para procesar y generar información, permitiendo su aplicación en diversas áreas como la robótica, el procesamiento de voz, la visión artificial y el procesamiento del lenguaje natural. Gracias a estos avances, los sistemas computacionales han adquirido la capacidad de aprender y mejorar su rendimiento en múltiples tareas. Dentro del PLN, las aplicaciones han avanzado significativamente, destacándose en la generación de textos y la traducción automática. En particular, los chatbots han evolucionado hasta responder de manera similar a los seres humanos[13].

En la sociedad actual, la tecnología y los servicios digitales han revolucionado la manera en que las personas interactúan con su entorno, provocando transformaciones sociales, económicas y culturales. La globalización y el comercio electrónico han facilitado las compras en línea desde cualquier parte del mundo, mientras que las aplicaciones móviles han mejorado el acceso a la información y la navegación urbana. Además, las redes sociales y las aplicaciones de mensajería han transformado la comunicación, especialmente entre los jóvenes[14]. En este contexto, la detección de emociones mediante inteligencia artificial y procesamiento del lenguaje natural se ha vuelto esencial, permitiendo analizar el estado emocional de los usuarios en plataformas digitales, mejorando la interacción con sistemas automáticos y ofreciendo experiencias más personalizadas en sectores como marketing y la atención al cliente.

La inteligencia artificial juega un papel clave en la transformación de diversas industrias, optimizando procesos y mejorando la toma de decisiones. En talento humano, facilita el análisis de las emociones y comportamientos de los empleados, lo que contribuye a mejorar la productividad y satisfacción laboral. En la atención al cliente, permite personalizar la interacción con los usuarios, brindando respuestas automatizadas y adaptadas a sus necesidades. Por otro lado, en marketing ayuda a identificar patrones de comportamiento y segmentar audiencias, aumentando la

efectividad de las campañas publicitarias. En el campo de la inteligencia de negocios, el análisis de sentimientos y la predicción de tendencias permiten a las empresas a tomar decisiones estratégicas basadas en datos. Además, la integración de la psicología en la IA permite una mejor compresión de las emociones humanas, optimizando la interacción con los usuarios y mejorando la experiencia en diversas plataformas digitales[15]. En resumen, la IA no solo aumenta la eficiencia operativa, sino que también fortalece la relación entre empresas, sus clientes y empleados.

El avance de las tecnologías digitales y la expansión del internet ha transformado profundamente la interacción humana, alterando la manera en que las personas se relacionan en su vida cotidiana y profesional. Estos avances han optimizado la comunicación, ampliado el acceso a la información y fomentado el trabajo remoto generando múltiples beneficios. Sin embargo, el uso excesivo ha dado lugar a trastornos psicológicos y sociales, lo que ha impulsado el desarrollo de la ciberpsicología, una disciplina que analiza la influencia del mundo digital en el comportamiento humano y la salud mental[16]. En este contexto, la creciente dependencia de las redes sociales y la constante exposición a entornos virtuales han modificado la forma en que las personas interactúan, dando lugar tanto a oportunidades como a desafíos, entre ellos la adicción a internet y la dependencia de las plataformas digitales.

Esta investigación se desarrolla en el contexto de la Universidad Estatal Península de Santa Elena (UPSE), institución en la que el avance tecnológico y la transformación digital son aspectos importantes para la mejora de la calidad educativa. Considerando el uso de las plataformas digitales dentro de la comunidad universitaria, sería una buena práctica explorar herramientas que faciliten la comprensión de las interacciones entre estudiantes y docentes. En este sentido, el análisis de emociones mediante técnicas de procesamiento del lenguaje natural ofrece una alternativa prometedora para contribuir al mejoramiento de los procesos educativos.

2.1.2 Base legal

2.1.2.1 Normativas nacionales

2.1.2.2 Ley orgánica de protección de datos personales

Articulo 7 - Tratamiento legítimo de datos personales

Establece que el tratamiento de información se considera legítimo y lícito siempre que se cumpla con determinadas condiciones. Una de ellas incluye el uso de datos que provengan de bases de datos que son de acceso público, lo que implica su disponibilidad general para su uso, donde no se requiere autorización del titular[17].

Artículo 9 - Interés legítimo

La normativa indica que el tratamiento de datos personales basado en el interés legítimo solo será válido si se limita estrictamente a aquellos datos que sean esenciales para cumplir con la finalidad establecida[17]. Este enfoque limita el uso de los datos a lo estrictamente indispensable, garantizando así un tratamiento responsable, especialmente en actividades de investigación.

Articulo 18 - Excepciones a los derechos de rectificación, actualización, eliminación, oposición, anulación y portabilidad.

Según la normativa del artículo, cuando los datos personales se utilizan para fines de científicos, históricos o estadísticos, no aplican ciertos derechos del titular, como la eliminación, portabilidad, anulación o rectificación. Esta disposición permite que la información pueda ser usada en proyectos de investigación, siempre que mantenga el respeto a los principios de protección de datos personales[17].

Artículo 26 - Tratamiento de datos sensibles

Establece una prohibición general para el tratamiento de datos personales sensibles; sin embargo, existen excepciones en las que su uso está permitido, como en investigaciones científicas, históricas o estadísticas, siempre que el uso sea adecuado y necesario para cumplir con el objetivo planteado. Además, se debe respetar los derechos fundamentales del titular y establecer medidas adecuadas que garanticen la seguridad y protección de sus datos[17].

2.1.2.3 Normativas internacionales

2.1.2.4 Reglamento General de Protección de Datos (GDPR)

Artículo 6 - Legalidad del procesamiento

- Establece que el tratamiento de datos personales es legal cuando se realiza para cumplir con una tarea en beneficio a la sociedad o interés público[18].
- Permite el tratamiento de datos cuando este responde a un interés legítimo por parte del responsable del tratamiento o de un tercero. Sin embargo, esta base legal no puede prevalecer si afecta a los derechos fundamentales de los titulares, especialmente si se trata de menores de edad[18].

2.1.2.5 Regulaciones específicas sobre el uso de inteligencia artificial

El uso de técnicas de Machine Learning y Deep Learning en la detección de emociones debe alinearse con normativas internacionales que establecen principios éticos y de transparencia en la inteligencia artificial (IA). A continuación, se presentan las principales regulaciones aplicables:

Principios de la OCDE sobre IA

La organización para la cooperación y el Desarrollo Económicos(OCDE) estableció cinco principios fundamentales para el desarrollo y el uso de sistemas de IA: transparencia, equidad, seguridad, rendición de cuentas y robustez[19]. Estos principios garantizan que los modelos de IA sean utilizados de manera ética y responsable.

Regulación de la Unión Europea sobre la IA

La Ley de la inteligencia artificial de la UE establece diferentes niveles de riesgo para los modelos de IA, aunque la clasificación de emociones no se considera un sistema de alto riesgo, se recomienda aplicar medidas de transparencia para evitar malinterpretaciones de los resultados[20].

2.1.2.6 Implicaciones Éticas de la Detección de Emociones

El análisis de emociones mediante inteligencia artificial (IA) presenta desafíos éticos, particularmente en relación con la precisión de los modelos y la posible discriminación algorítmica. Según la UNESCO (2021), los sistemas de IA deben diseñarse para minimizar sesgos y garantizar la equidad en su funcionamiento [21]. En este contexto, la detección de emociones debe adherirse a los principios de

transparencia, equidad y explicabilidad, con el fin de asegurar que los modelos de Machine Learning y Deep Learning utilizados en este estudio se desarrollen de manera ética, responsable y alineada con normativas internacionales**.**

2.2. Marco Conceptual

2.2.1 Procesamiento del Lenguaje Natural (PLN)

El Procesamiento del Lenguaje Natural (PLN) es un campo enmarcado dentro del área de la inteligencia artificial, la computación y la lingüística. Su propósito principal es optimizar la interacción entre los seres humanos y las computadoras a través del uso de lenguajes naturales, permitiendo que los sistemas informáticos comprendan, interpreten y generen texto de manera similar al lenguaje humano[22].

2.2.1.1 Psicolingüística

La psicolingüística es una disciplina experimental que estudia los procesos mentales involucrados en la adquisición, producción y comprensión del lenguaje. Analiza como la mente humana, con base en el funcionamiento del cerebro, utiliza conocimientos previos y mecanismos cognitivos para interpretar y generar lenguaje[23]. Este enfoque en el análisis de textos emocionales, permite comprender como las emociones se expresan y se codifican lingüísticamente.

2.2.1.2 Lingüística semántica

La Lingüística consta de diferentes niveles de estudios entre uno de ellos es el nivel semántico, su principal enfoque se basa en tener un sentido y un significado con las palabras y oraciones[24]. Dado el contexto se introduce la macroestructura semántica que se usa para identificar la predominación de un texto, dicho de otra manera, esto juega un papel calve al momento de proporcionar una versión global del contenido.

2.2.2 Análisis de sentimientos

En el Análisis de Sentimientos (Sentiment Analysis) dentro del área de estudio del procesamiento del lenguaje Natural (PLN), es la encargada de identificar la información ya sea esta de manera automática o mediante técnicas

computacionales, lo realiza haciendo un análisis de cómo se expresa una opinión en un documento. Permitiendo identificar su enfoque positivo o negativo de un tema en específico[25]. En general el análisis es una herramienta fundamental que ayuda a identificar la emoción de un argumento puntual.

2.2.2.1 Sentimientos

"Los sentimientos funcionan como una herramienta emocional que permite la interrelación, toma decisiones, estado de ánimo e, incluso, estado de salud y gesticulación de cada individuo, pueden determinar la disposición o estado de ánimo de un individuo (alegre, triste, desanimado, apasionado, etcétera)[26].

2.2.2.2 Emociones

Las emociones se refieren a la reacción ante un estímulo o alguna situación en específico, Se consideran un fenómeno tanto psicológico como fisiológico llegándose a manifestar a través de la conducta, las percepciones corporales y en la conciencia de las personas. Estas reacciones son adaptativas, ya que generan un impulso que propicia a la acción, se clasifican en emociones positivas, negativas y neutras[27].

2.2.3 Google Colab

Colab o comúnmente conocido como "Colaboratory", es una plataforma que nos brinda un servicio de manera que permite ejecutar y programar en Python en el navegador teniendo ventajas como: Los cuadernos de trabajo que se almacenan en Google Drive, utilización de bibliotecas, no necesita una configuración complicada y acceso gratuito a GPUs[28].

2.2.4 Python

Python es un lenguaje de programación que destaca por su simpleza y versatilidad. Su diseño intuitivo y su capacidad para tipado dinámico facilitan la creación rápida de aplicaciones de manera eficiente en múltiples plataformas. Además, gracias a sus estructuras de datos de alto nivel y su sistema orientado a objetos brindan a los programadores las herramientas necesarias para una integración fluida de sistemas. En pocas palabras estas características hacen de Python una herramienta ideal para proyectos de scripting y desarrollo rápido[29].

2.2.5 Conjunto de datos

Un conjunto de datos está compuesto por información reunida que ha sido observada, colectada o creada con el objetivo de generar resultados originales. La importancia de estos conjuntos es que una vez procesados y transformados en formatos como matrices, donde se pueden validar descubrimientos y establecer relaciones entre propiedades con fundamentos sólidos, basándose en pruebas objetivas, y no solo en suposiciones[30].

2.2.6 Modelo computacional

El modelo computacional representa una herramienta en la ciencia computacional, que se basa en un enfoque de modelos matemáticos para analizar y estudiar cómo se comportan los sistemas complejos. Para realizar esta simulación, este tipo de modelo requiere de un alto consumo de recursos computacionales, ya que su objetivo es simular el comportamiento de dichos sistemas mediante el uso de simulaciones por computadoras[31]. Es decir, son fundamentales para comprender el comportamiento de sistemas que no se pueden estudiar de forma directa.

2.2.7 Ciberpsicología

La Ciberpsicología, una de las ramas más recientes de la psicología, representa como una de las áreas más jóvenes, surgió de la necesidad de comprender como la tecnología influye en las personas. Esta disciplina se enfoca en estudiar los efectos de los avances tecnológicos a medida que las personas hacen un uso mayor o menor de dicha tecnología. Su objetivo es entender los cambios psicológicos que se producen en ellos a lo largo del tiempo[32].

2.2.8 Tokenización

La tokenización es una técnica clave en el ámbito del procesamiento del lenguaje natural, que consiste en división del texto en palabras o frases significativas, denominado “tokens”. Este proceso se realiza mediante la identificación de caracteres delimitadores como puntos, comas o espacios. Su uso es común en diversas aplicaciones, desde la traducción automática hasta el análisis de emociones, ya que facilita la manipulación y análisis de textos al generar conjuntos de palabras que pueden ser estudiados individualmente[33]

2.2.9 Stopwords

Los stopwords o palabras vacías, se refieren a las palabras que, debido a su alta frecuencia de uso, no aporta un valor significativo al análisis semántico de un texto. Estas incluyen artículos, preposiciones, pronombres que son necesarias para la estructura gramatical de las frases, pero que no contribuyen a la identificación de conceptos clave en el procesamiento del lenguaje natural (PLN)[34].

2.2.10 Inteligencia artificial

"Disciplina científica que se ocupa de crear programas informáticos que ejecutan operaciones comparables a las que realiza la mente humana, como el aprendizaje o el razonamiento lógico"[35]. Por lo que se refiere a la inteligencia artificial(IA) se puede considerar un área de la informática que se dedica al desarrollo de tecnologías y sistemas con la finalidad de que cuenten con la capacidad de semejar o imitar a la inteligencia humana. Para llegar a cumplir este objetivo, es a través de diseños de modelos matemáticos y algoritmos. El resultado de su capacidad de aprendizaje es esencial en optimizar procesos y resolver problemas[36].

2.2.10.1 Aprendizaje supervisado

En cuanto al enfoque supervisado se centra en procesar datos etiquetados o clasificados, de tal forma que permita aprender a identificar patrones y hacer predicciones sobre futuras observaciones. Además, facilita un control más preciso sobre las predicciones y categorizaciones[37]. Dicho brevemente "Se enseña al algoritmo cómo realizar su trabajo, con un conjunto de datos clasificados bajo una cierta apreciación o idea para encontrar patrones que puedan aplicarse en un análisis y producir una salida que ya se conoce."[38]

2.2.10.2 Algoritmo

Los algoritmos son conjuntos de instrucciones lógicas y estructuradas, secuencia de operaciones matemáticas o pasos programados que se crean mediante códigos informáticos escritos por seres humanos, que sirven para procesar diferentes tipos de datos. Mediante una serie de pasos que son finitos y ordenados, generan resultados como predicciones, sugerencias o inferencias, ayudando a resolver un problema específico, buscando una solución de manera estructurada y eficiente[39].

2.2.11 Word Embedding

Word Embedding o incrustación de palabras es un método basado en la semántica de la distribución que transforma las palabras en representaciones vectoriales de números reales. Este enfoque permite la agrupación de palabras con significados y estructuras similares, lo que facilita su análisis en tareas de procesamiento del lenguaje natural. Estas técnicas se aplican en modelos de lenguaje y aprendizaje automático, contribuyendo a la capacidad de las máquinas para comprender y procesar texto de manera más eficiente[40].

Características principales de la incrustación de palabras:

- Una incrustación de palabras se representa como un vector único, es decir, una secuencia de números que caracteriza de forma específica a cada palabra.
- Los vectores que representan a las palabras operan en un espacio multidimensional, cuya longitud varía entre 50 y 500 dimensiones , dependiendo del modelo utilizado.
- Los vectores representan el significado de las palabas al capturar su contexto y relación con otros términos en el lenguaje, lo que permite al modelo interpretar de manera efectiva relaciones semánticas[41]

Las palabras con contextos o significados similares se agrupan mediante valores vectoriales cercanos, lo que facilita el análisis y agrupación por los algoritmos[41].

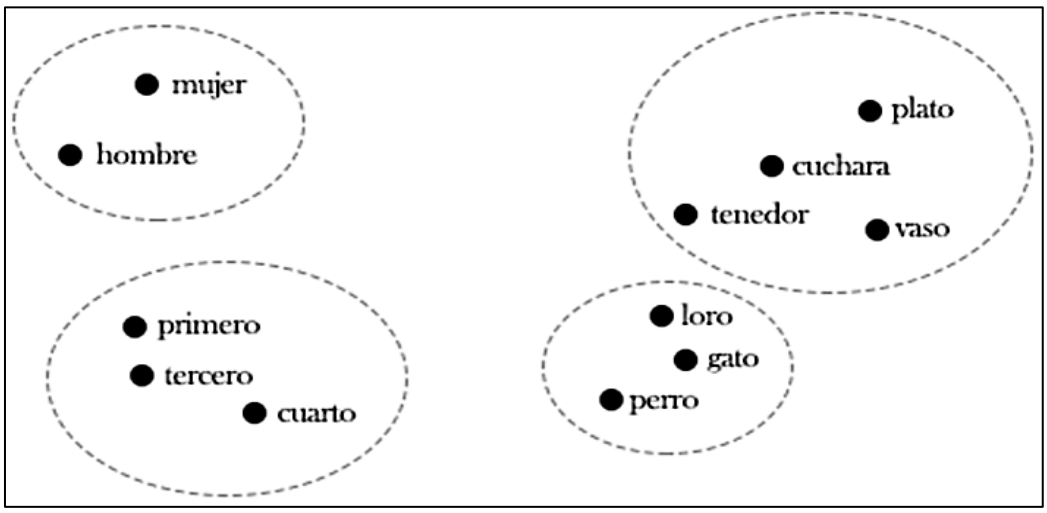


Figura 1: Palabras organizadas en campos vectoriales [5]

2.2.11.1 Word2Vec

Es un modelo de redes neuronales que, a través de su estructura de dos capas, se encarga de aprender la incrustación de palabras. Este algoritmo, basado en aprendizaje no supervisado, durante su fase de entrenamiento, detecta palabras que aparecen en contextos similares y los agrupa en el espacio vectorial. Este enfoque resulta eficaz para capturar relaciones semánticas entre palabras[42].

Word2Vec, emplea una arquitectura neuronal de dos capas para generar representaciones vectoriales de palabras. Existen dos enfoques principales para su entrenamiento: El modelo Skip-gram, que busca predecir el contexto a partir de una palabra objetivo, y el modelo CBOW (Continuous Bag of Words), cuyo objetivo es predecir la palabra objetivo a partir de su contexto[42]. En ambos métodos, los Word Embedding se obtienen desde la primera capa de la red neuronal.

2.2.11.2 FastText

FastText es una técnica de incrustaciones de palabras desarrollada por el laboratorio AI Research (FAIR) de Facebook, introducen el uso de subpalabras mediante la generación de n-gramas de caracteres, en lugar de tratar las palabras como entidades completas. Este método permite capturar el significado entre términos que comparten similitudes morfológicas. Además, su capacidad para manejar palabras fuera del vocabulario lo convierte en una herramienta valiosa para lenguajes con morfología rica o para tareas donde las palabras fuera del vocabulario son comunes[43].

FastText ha transformado el procesamiento del lenguaje natural al incorporar información a nivel de carácter para generar representaciones de palabras más sólidas. Emplea redes neuronales entrenadas bajo los modelos Continuous Bag of Words (CBOW) o Skip-gram, los cuales predicen el contexto a partir de una palabra objetivo o viceversa. Estos modelos ajustan los parámetros de la red neuronal para minimizar la perdida, de manera que FastText aprenda representaciones significativas a partir de grandes volúmenes de texto[43].

2.2.11.3 Skip-gram

El modelo Skip-gram es una técnica de Word Embedding que aprende el significado de las palabras analizando cuáles suelen aparecer cerca de una palabra central. A diferencia del modelo CBOW, que predice una palabra a partir de su contexto, Skip-gram invierte el uso de las palabras objetivo y centrales[42]. Esto permite generar representaciones vectoriales ricas en contenido semántico, al capturar como una palabra se relaciona con su entorno cercano.

2.2.12 Machine Learning

El aprendizaje automático, también conocido como machine Learning, es una rama fundamental de la inteligencia artificial. Esta disciplina permite que las máquinas y sistemas informáticos desarrollen la capacidad de aprender y mejorar su rendimiento sin estar programados específicamente para cada tarea. Este proceso se basa en la implementación de algoritmos que identifican patrones dentro de conjuntos de datos, lo que facilita la toma de decisiones, la generación de predicciones y el suministro de recomendaciones ajustadas a situaciones particulares[44].

2.2.12.1 SVM (Máquinas de Soporte Vectorial)

La máquina de soporte vectorial (SVM) forma parte de las estrategias avanzadas de clasificación supervisada en machine Learning. Su principio básico consiste en encontrar hiperplanos que separen datos en espacios de n dimensiones, siendo dichos hiperplanos representaciones geométricas, como líneas o planos, dependiendo de la cantidad de características involucradas. El algoritmo identifica y utiliza estos hiperplanos para separar los datos de forma clara en distintas clases, optimizando la distancia entre los puntos más cercanos de cada conjunto. Gracias a su capacidad para resolver tanto tareas de clasificación como de regresión, SVM ha ganado un lugar destacado en el análisis y procesamiento de datos en aplicaciones modernas, siendo una herramienta valiosa para el análisis de datos complejos[45].

Es un potente clasificador discriminatorio que trabaja mediante la definición de un hiperplano separador que maximiza la distancia entre las clases de los datos. Este modelo de aprendizaje supervisado utiliza conjuntos de datos etiquetados, para identificar el hiperplano optimo que no solo caracteriza correctamente los ejemplos del conjunto de entrenamiento, sino que también generaliza adecuadamente para datos nuevos. En espacios bidimensionales, dicho hiperplano se visualiza como una línea recta que divide el espacio en dos regiones, cada una correspondiente a una categoría distinta, facilitando así una correcta clasificación[45].

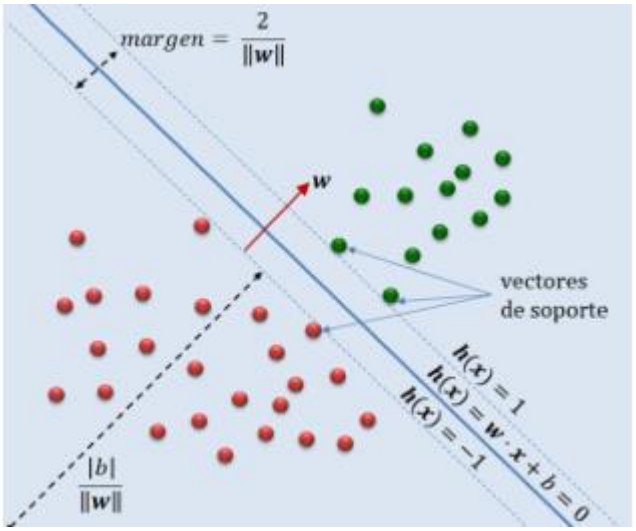


Figura 2: Máquina de soporte vectorial (SVM) [44]

2.2.12.2 Regresión Logística

La regresión logística es un algoritmo de clasificación en Machine Learning que permite predecir la probabilidad de que un dato pertenezca a una clase especifica a partir de variables independientes. El modelo calcula una combinación lineal de las características de entrada, incluyendo en muchos casos un término de sesgo, y aplica la función logística para transformar el resultado[46]. El rango de salida de la regresión logística siempre se encuentra entre 0 y 1, lo que la hace particularmente adecuada para tareas de clasificación binaria. Los Valores cercanos a 1 indican una mayor probabilidad de que la muestra pertenezca a la clase 1,

mientras que los valores cercanos a 0 sugieren que la instancia pertenece a la clase opuesta[46].

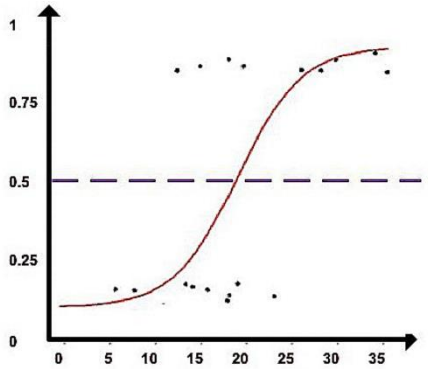


Figura 3: Modelo de Regresión Logística [45]

2.2.12.3 Naïve Bayes Gaussiano (GaussianNB)

El algoritmo Gaussian Naïve Bayes (GaussianNB) es una variante del clasificador Naïve Bayes, que destaca por su aplicación a variables continuas que siguen una distribución normal. El modelo estima la probabilidad de pertinencia a una clase calculando la media y la desviación estándar de cada atributo dentro de cada categoría, lo que permite realizar predicciones bajo el supuesto de independencia incondicional entre las variables[47].

2.2.13 Deep Learning

Los algoritmos de aprendizaje profundo (Deep Learning) es un tipo de machine learning que entrena a una computadora para que realice tareas como las hacemos los seres humanos, como el reconocimiento del habla, la identificación de imágenes o hacer predicciones. A diferencia de otros enfoques que requieren que los datos sean procesados mediante ecuaciones establecidas, deep leaning configura parámetros básicos y permite que la maquina aprenda por si sola a través de la detección de patrones, utilizando redes neuronales profundas que procesan la información en varias capas[48].

2.2.13.1 Redes Neuronales Artificiales (ANN)

Son modelos computacionales inspirados en la estructura y el funcionamiento de las redes neuronales biológicas del cerebro humano. Los elementos que las conforman imitan el comportamiento de las neuronas biológicas en cuanto al

procesamiento de información y la comunicación entre sí. Esta red de elementos interconectados está organizada de forma similar a la disposición neuronal del cerebro, lo que permite que las ANN realizar tareas como el reconocimiento de patrones, la predicción y la clasificación de datos, emulando ciertos procesos cognitivos humanos[49].

2.2.13.2 Redes Neuronales Recurrentes (RNN)

Una red neuronal recurrente(RNN) es un modelo avanzado de redes neuronales artificiales diseñado específicamente para procesar secuencia de datos, permitiendo analizar información en la que el orden de los elementos es crucial. Su característica principal radica en que la salida de cada paso se reutiliza como entrada para el siguiente, lo que permite capturar relaciones temporales y contextuales de los datos[50].

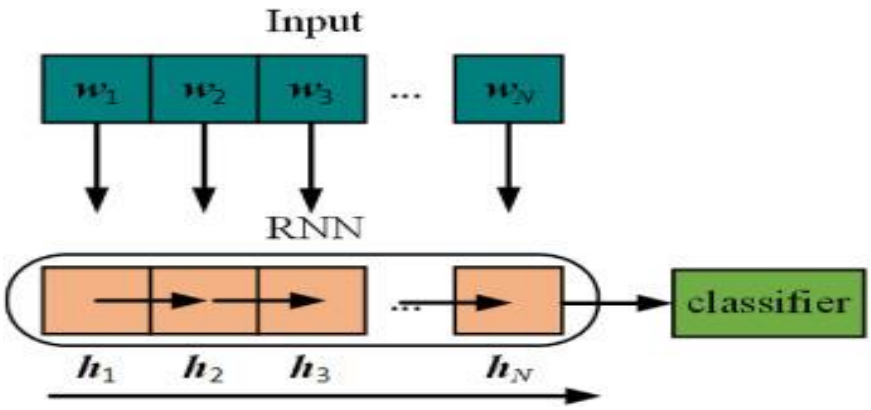


Figura 4: Predicción de emociones basado en RNN [50]

En la figura se representa el esquema de una RNN aplicada a la predicción de emociones, donde cada unidad procesa secuencialmente información del texto y transmite su estado oculto al siguiente paso temporal. El estado final h_N acumula la información contextual de toda la secuencia, y es a partir de este vector que el clasificador realiza la predicción emocional[51].

2.2.13.3 BiLSTM (Memoria a Largo Plazo Bidireccional)

La BiLSTM (Bi-Long Short-Term Memory) surge al agregar capas adicionales a la arquitectura LSTM, transformándola en un modelo bidireccional. A diferencia de un LSTM tradicional, que procesa los datos de entrada solo en una dirección (de izquierda a derecha o de derecha a izquierda), la BiLSTM utiliza dos redes LSTM

que operan de forma simultánea, procesando los datos en ambas direcciones. Esto significa que, durante el entrenamiento, el modelo captura información tanto desde el inicio como desde el final de la secuencia[52].

En el diagrama se observa como el BI-LSTM permite un flujo de información en ambas direcciones: hacia adelante y hacia atrás a través de sus capas. Esto lo convierte en una opción eficiente para resolver tareas secuenciales como clasificación de texto, reconocimiento de voz y diversos modelos de pronósticos. Gracias a su naturaleza bidireccional, el BI-LSTM puede captar relaciones contextuales más amplias en los datos[7].

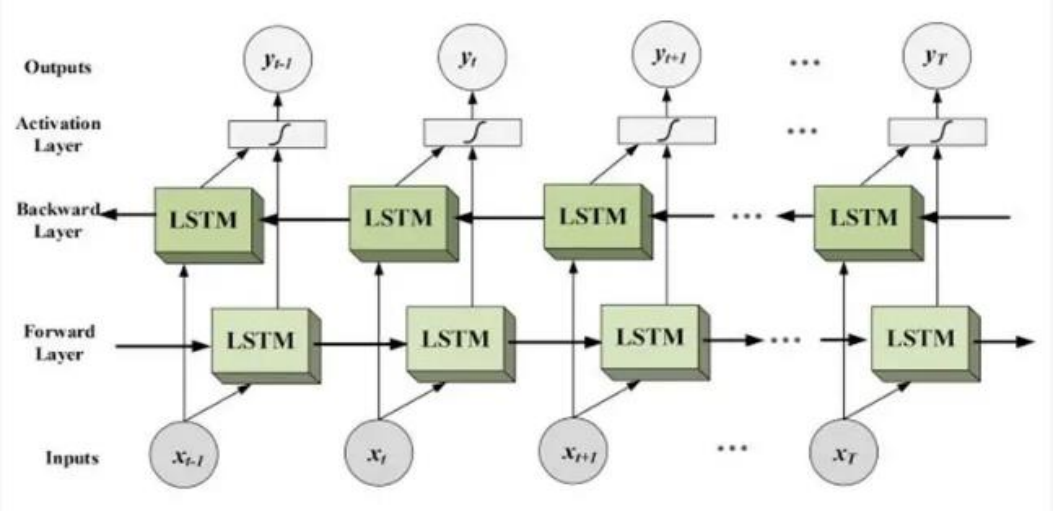


Figura 5: BiLSTM (Bi-Long Short-Term Memory) [7]

2.2.13.4 LSTM(memoria a corto-largo plazo)

Las redes LSTM incorporan mecanismos adicionales en comparación con las redes recurrentes (RNN) tradicionales, con el propósito de mejorar el aprendizaje de dependencia a largo plazo. Su arquitectura se basa en la integración de una celda de memoria y tres tipos de puertas: entrada, salida y olvido. Estas puertas regulan de manera eficiente la información que se retiene y la que se descarta en cada iteración, evitando la perdida de información relevante en secuencias largas[53].

Flujo de datos en la unidad de tiempo t de una unidad de LSTM. La puerta de olvido y la celda de memoria evitan problemas de desvanecimiento o explosión de gradiente[53].

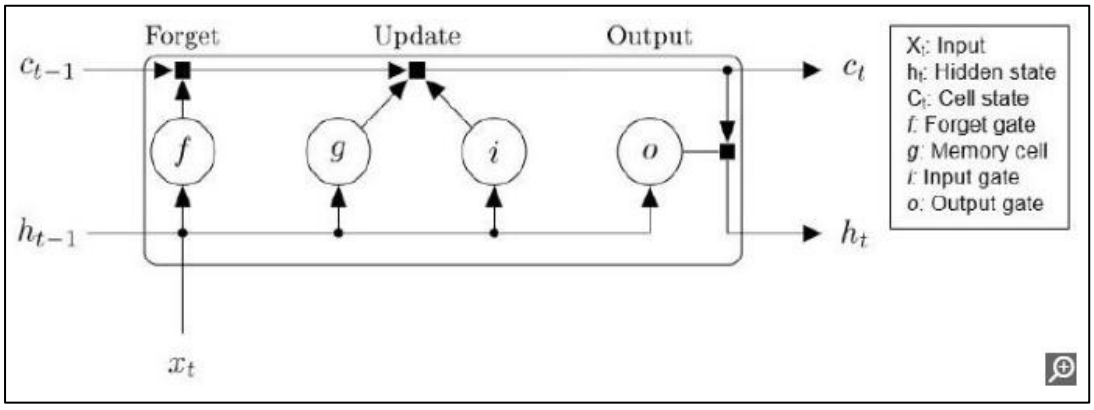


Figura 6: Arquitectura LSTM [51]

2.2.13.5 Red neuronal recurrente (GRU)

Las unidades GRU (Gated Recurrent Unit) representan una variable de las RNN diseñadas para abordar de manera eficiente el problema de las dependencias a largo plazo, una de las principales limitaciones de las RNN tradicionales. Las GRU cuentan con una estructura más simple, ya que combinan las puertas de entrada y olvido en una sola puerta de actualización, además de fusionar la puerta de salida con esta puerta de actualización. Esto simplifica el proceso de entrenamiento y aumenta la eficiencia computacional[50].

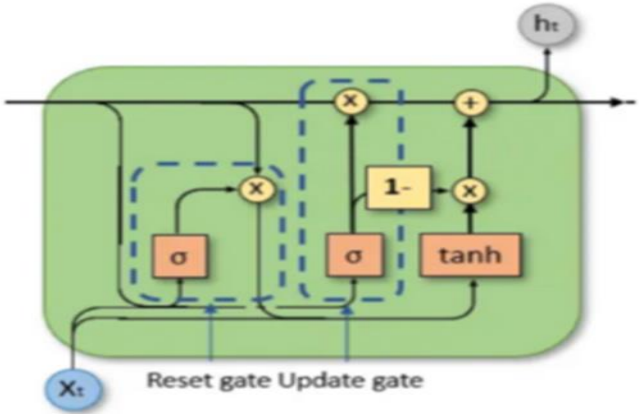


Figura 7: Gated Recurrent Unit (GRU) [49]

2.2.18 Métricas de Evaluación

2.2.18.1 Precisión

Es un indicador del rendimiento de un modelo de clasificación que mide la proporción de instancias correctamente identificadas como positivas en relación con todas las instancias que el modelo ha clasificado como positivas. Se calcula como la relación entre los verdaderos positivos (TP) y la suma de verdaderos positivos y falsos positivos (FP). Un alto nivel de precisión sugiere que el modelo tiene un buen desempeño al asignar correctamente las clases positivas, evitando clasificar erróneamente instancias negativas[54].

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FP}}$$

2.2.18.2 Recall (Recuperación)

El recall mide la capacidad del modelo para identificar correctamente los ejemplos relevantes dentro de un conjunto de datos. Se calcula como la razón entre el número de verdaderos positivos (TP) y la suma de verdaderos positivos y falsos negativos (FN). Un valor alto de recall indica que el modelo logra recuperar la mayoría de los casos relevantes, minimizando la cantidad de instancias omitidas[54].

$$\boldsymbol{Recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FN}}$$

2.2.18.3 F1-score

La puntuación F1 es un indicador clave para evaluar el rendimiento de modelos de clasificación, especialmente en datos desbalanceados. Esta métrica combina precisión y exhaustividad (recall), asegurando un equilibrio entre ambas. La precisión mide que tan confiable es el modelo al predecir una clase positiva, mientras que la exhaustividad evalúa que tan completo es el modelo al identificar todos los casos positivos. Alcanza su valor máximo de 1 cuando ambas son del

100%, mientras que su peor valor es 0 cuando una de las métricas es completamente deficiente[54].

$$F_1Score = \frac{2\ x\ presicion\ x\ recall}{recall + precision}$$

2.2.18.4 AUC (Área Bajo la Curva)

Se emplea en tareas de clasificación binaria y mide la efectividad de un clasificador al comparar la tasa de verdaderos positivos frente a la tasa de falsos positivos. Un valor de AUC igual a 1 indica un clasificador perfecto, mientras que un valor de 0.5 sugiere que el modelo no tiene mayor capacidad de discriminación que un clasificador aleatorio, ya que no puede diferenciar entre las clases de manera significativa[9].

2.2.18.5 Matriz de Confusión

La matriz de confusión es una herramienta visual utilizado para evaluar el rendimiento de los algoritmos de aprendizaje supervisado. Esta matriz permite observar de forma detallada las predicciones correctas e incorrectas realizadas por el modelo durante las fases de entrenamiento y validación. Cada celda de la matriz representa un tipo de resultado de la predicción del modelo contiene valores que representan las métricas de verdaderos positivos, falsos positivos, verdaderos negativos y falsos positivos.[55]

Verdaderos Positivos (TP): Casos que el modelo identifico correctamente como positivos.

Falsos Positivos (FP): Casos que fueron clasificados incorrectamente como positivos, cuando en realidad eran negativas.

Verdaderos Negativos (TN): Casos negativos que el modelo identifico correctamente.

Falsos Negativos (FN): Casos que el modelo clasificado erróneamente como negativas, siendo en realidad positivas.

Prediction		
Label	**TP** **(Verdaderos positivos)**	**FP** **(Falsos positivos)**
	FN **(Falsos Negativos)**	**TN** **(Verdaderos Negativos)**

Figura 8: Matriz de confusión: Elaboración propia

2.2.19 librerías

2.2.19.1 Pandas

Pandas es una biblioteca de software de código abierto diseñada específicamente para la manipulación y el análisis de datos en el lenguaje Python. Su arquitectura permite trabajar de manera eficiente con grandes volúmenes de información, facilitando tareas como la limpieza, transformación y exploración de los datos[6].

2.2.19.2 Spacy

Es una librería de código abierto de Python diseñada para el procesamiento del lenguaje natural en aplicaciones de producción. Es especialmente útil en tareas como la extracción de información y la comprensión del texto, donde se requiere analizar estructuras lingüísticas complejas. Incluye funcionalidades como análisis sintáctico, etiquetado gramatical y reconocimiento de entidad nombrada, para diferentes idiomas[33].

2.2.19.3 NLTK (kit de herramientas de lenguaje natural)

Es un conjunto de librerías y programas en Python diseñado para llevar a cabo diversas tareas relacionadas con el Procesamiento del Lenguaje Natural. Proporciona recursos léxicos y corpus, además de herramientas funcionales para realizar operaciones como tokenización, clasificación, stemming, etiquetado y análisis sintáctico[40].

2.2.19.4 Numpy

Es una librería especializada en cálculo numérico y el análisis de datos, diseñada para trabajar con grandes volúmenes de información. Proporciona soporte para crear vectores y matrices grandes multidimensionales, junto con una gran colección de funciones matemáticas[6].

2.2.19.5 Scikit-learn

Scikit-learn es una biblioteca de Python que proporciona una amplia variedad de algoritmos para tareas de aprendizaje automático, como clasificación, regresión, agrupamiento (clustering) y reducción de dimensionalidad. Es compatible con otras bibliotecas del ecosistema Python, como NumPy, SciPy y matplotlib. Además, se caracteriza por su simplicidad de uso, eficiencia y una documentación extensa que facilita su implementación en proyectos académicos y profesionales[40].

2.2.19.6 Re

El módulo Re proporciona herramientas para trabajar con expresiones regulares, facilitando tareas como la búsqueda, sustitución o segmentación de textos según estructuras definidas. Estas expresiones son útiles para la detección de patrones y manipulación de texto, ya que permite buscar, emparejar y manipular cadenas basadas en estos patrones[40].

2.2.19.7 Gensim

Biblioteca gratuita de Python diseñada para el procesamiento del lenguaje natural y la minería de texto. Permite trabajar con representaciones vectoriales de palabras mediante modelos como Word2Vec o FastText, así como para desarrollar modelos temáticos que extraen patrones semánticos a partir de grandes volúmenes de datos[9].

2.2.19.8 Matplotlib

Matplotlib es una de las bibliotecas más utilizadas en Python para la visualización de datos. Permite generar gráficos estadísticos, animadas e interactivos de alta

calidad. Gracias a su compatibilidad con distintos entornos gráficos la hace adecuada para una amplia gama de aplicaciones en análisis de datos[56].

2.2.19.9 Seaborn

La biblioteca Seaborn, desarrollada sobre matplotlib, proporciona herramientas avanzadas para generar visualización estadísticas atractivas y comprensibles. Facilita la identificación de patrones, tendencias y distribuciones en conjuntos de datos complejos[57].

2.2.19.10 Tensorflow

Es una plataforma de código abierto diseñada para el desarrollo y la implementación de modelos de modelos de aprendizaje automático. Esta biblioteca permite tanto la ejecución de modelos de aprendizaje profundo (deep learning) como algoritmos tradiciones de machine learning, facilitando el manejo eficiente de tareas computacionales complejas[52].

2.2.19.11 Streamlit

Streamlit es un framework de código abierto desarrollado en Python, diseñado para facilitar a científicos de datos e ingenieros en inteligencia artificial y aprendizaje automático la creación de aplicaciones webs interactivas, permite construir interfaces visuales, para la visualización y presentación de modelos y datos[58].

2.3. Marco Teórico

2.3.1 Importancia de la detección de emociones en textos y su aplicación en diferentes áreas

Dentro del procesamiento del lenguaje natural(NLP), el análisis de emociones en texto desempeña un rol esencial, busca identificar, extraer y comprender las emociones que se expresan a través de las palabras. Esta tarea es indispensable debido a que el lenguaje humano combina palabras, estructuras gramaticales, además también expresa emociones que enriquecen la comunicación. [59]

El análisis de emociones en texto es una herramienta poderosa para empresas en contextos como el servicio al cliente, el marketing e investigación del mercado dando como resultado el comprender los sentimientos de los consumidores a través de sus comentarios. Al identificar patrones emocionales, las organizaciones pueden adaptar sus estrategias y mejorar la conexión con la audiencia. Además, el análisis de emociones en redes sociales y cometarios de clientes es útil para prevenir crisis, de modo que las empresas puedan actuar rápidamente ante posibles problemas, permitiendo tomar medidas preventivas para proteger su reputación.[59]

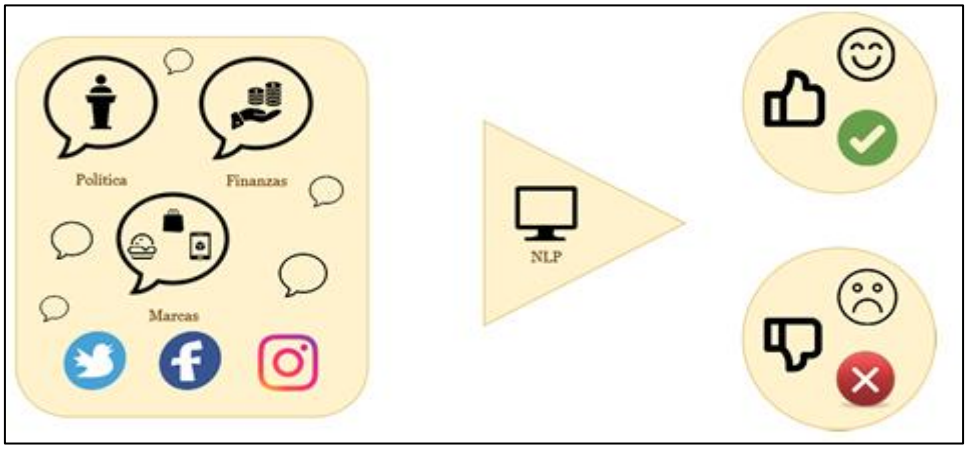


Figura 9: Análisis de Sentimientos en el Procesamiento del Lenguaje Natural. [9]

2.3.2 Desafíos y limitaciones del procesamiento del lenguaje natural

El Procesamiento del Lenguaje Natural (PLN) ha logrado avances significativos en los últimos años, sin embargo, aún enfrenta desafíos que dificultan su implementación en tareas complejas.[60]

Datos no estructurados: La gran cantidad de información textual disponible provienen de fuentes no estructuradas, lo que dificulta su análisis y clasificación eficiente.[60]

Ambigüedad lingüística: Muchas palabras poseen múltiples significados de modo que representa un reto para la interpretación precisa del texto, ya que los modelos deben considerar el contexto en el que se encuentran[60].

Contexto cultural: El lenguaje esta influenciado por factores culturales que pueden modificar el significado de una oración, dificultando su traducción y comprensión automática[60].

Dificultad en la generación de respuestas originales: Los algoritmos tienden a generar respuestas repetitivas debido a su naturaleza de su entrenamiento basado en patrones preexistentes[60].

Comprensión de la ironía y el sarcasmo: La identificación de expresiones irónicas y sarcásticas sigue siendo un reto, ya que requiere un entendimiento del tono y del contexto, aspectos difíciles de modelar computacionalmente[60].

2.3.3 Futuro del Procesamiento del Lenguaje Natural

El futuro del procesamiento del lenguaje natural es prometedor y lleno de posibilidades. A medida que los modelos de lenguaje se vuelven más sofisticados y se combinan con otras tecnologías, se espera que el PLN tenga un impacto transformador en diversas industrias Algunas áreas de desarrollo futuro incluyen[61]:

Avances en Modelos de Lenguaje: Los modelos de lenguaje están en constante mejora, y se espera que en futuro pueden procesar y generar texto de manera aún más precisa. A medida que los modelos avanzan, será posible crear sistemas que ofrezcan interacciones más naturales y cercanas a la forma en que las personas se comunican[61].

Integración con Otras Tecnologías: Una de las áreas clave en el desarrollo futuro del PLN es su integración con otras tecnologías, como el aprendizaje automático y la visión por computadora. Esto habilitara nuevas formas de interacción, como la generación de texto y reconocimiento de vos, lo que proporcionara un experiencia más fluida y eficiente[61].

Aplicaciones Potenciales en Diferentes Industrias: El PLN tiene el potencial de transformar sectores clave como la atención médica, el comercio electrónico y la asistencia virtual. Las empresas que adopten el PLN podrán mejorar la automatización de tareas y la interacción con los usuarios, ofreciendo una experiencia más fluida y eficiente[61]

2.3.3.4 Fundamentos psicolingüísticos para el análisis de personalidad y emociones en textos en español.

Desde la perspectiva de la psicolingüística, disciplina que estudia la relación entre los procesos mentales y el lenguaje, se plantea que la personalidad humana es un constructo complejo que se manifiesta no solo en los comportamientos observables, sino que también en el uso del lenguaje. Según Quevedo-Aguado e Iraegui-Torralbo sostienen que las características individuales y los rasgos de la personalidad pueden ser identificados y analizados a través de la expresión lingüística. Esto implica que el lenguaje no solo es un medio de comunicación, sino también un reflejo de las emociones y características psicológicas propias de cada persona[62].

En su investigación, los autores plantean una clasificación taxonómica para el estudio de la personalidad en el idioma español desde una perspectiva psicolingüística. Argumentan que ciertos rasgos de la personalidad, pueden ser identificados por medio del análisis léxico y discursivos en el lenguaje utilizado. Esta taxonomía está sustentada en una observación y clasificación sistemática de patrones léxicos, semánticos y pragmáticos que caracterizan la expresión verbal de dichos rasgos[62].

La importancia radica en que establece un vínculo directo entre los rasgos de personalidad y su expresión en el lenguaje cotidiano, lo cual es fundamental para el análisis de textos escritos en contextos sociales diversos. Esta conexión permite analizar textos con mayor profundidad, identificando no solo patrones lingüísticos, sino que también indicios de rasgos psicológicos. Así, el lenguaje se convierte en una herramienta diagnóstica para comprender las emociones relacionada con determinados perfiles psicológicos[62].

En el desarrollo de esta investigación, se basó en el enfoque de Quevedo-Aguado e Iraegui-Torralbo para vincular rasgos de personalidad con el lenguaje. Esta perspectiva respaldó tanto la decisión de agrupar emociones en categorías generales como la posibilidad de etiquetar múltiples emociones por comentario. Además, de la lista personalizada de stopwords en español, que preserva vocabulario emocionalmente relevante, fortaleciendo el análisis desde una manera computacional y psicológica.

CAPÍTULO 3

2.4. Requerimientos

2.4.1. Requerimientos Funcionales

Preparación de Datos	
Código	**Descripción**
RF-01	El código debe permitir la carga y lectura del dataset en formato Excel en Google Colab.
RF-02	El código debe realizar clasificación multietiqueta, permitiendo asignar hasta 3 emociones por comentario.
RF-03	El código debe filtrar y eliminar etiquetas vacías de la lista de emociones en cada comentario.
RF-04	El código debe convertir todos los comentarios a minúsculas.
RF-05	El código debe limpiar los textos eliminando caracteres especiales, signos de puntuación, números, emojis.
RF-06	El código debe filtrar stopwords (palabras vacías).
RF-7	El código debe tokenizar los comentarios utilizando SpaCy, para fragmentar el texto en unidades significativas.
RF-8	El código debe manejar el desbalanceo de clases mediante la agrupación de emociones similares en 7 categorías.

Tabla 3: Tabla de requerimientos funcionales: Preparación de datos: Elaboración propia

Vectorización	
Código	**Descripción**
RF-9	El código debe implementar la técnica de FaxtText para convertir palabras en representaciones vectoriales.
RF-10	El código implementa la técnica de Word2Vec para convertir palabras en representaciones vectoriales.

RF-11	El código debe calcular el promedio de los vectores de las palabras de cada comentario para representar cada texto con un solo valor.
RF-12	El código debe generar secuencia de vectores y aplicar relleno (padding) para el uso de los modelos de Deep Learning.

Tabla 4: Tabla de requerimientos funcionales Vectorización : Elaboración propia

Entrenamiento de modelos	
Código	**Descripción**
RF-13	El código debe dividir el conjunto de datos en subconjuntos de entrenamiento y validación.
RF-14	El código debe entrenar modelos de Machine Learning (SVM, Regresión Logística y Naïve Bayes Gaussiano)
RF-15	El código debe entrenar modelos de Deep Learning (BiLSTM, GRU, LSTM).

Tabla 5: Tabla de requerimientos funcionales Entrenamiento de modelos: Elaboración propia

Evaluación y visualización de modelos	
Código	**Descripción**
RF-16	El código debe Calcular métricas de precisión, recall, F1-score y área bajo la curva (AUC) para evaluar el rendimiento de los modelos.
RF-17	El código debe generar y visualizar la matriz de confusión.
RF-18	Comparar los resultados de los modelos de Machine Learning y Deep Learning mediante tablas, con el fin de determinar cuál presenta un mejor desempeño en la clasificación de emociones.

Tabla 6: Tabla de requerimientos funcionales Evaluación y visualización de modelos: Elaboración propia.

Predicción a través de interfaz básica	
Código	**Descripción**
RF-19	El código debe permitir ingresar un comentario mediante una interfaz básica y mostrar hasta 3 emociones predichas utilizando el modelo entrenado.

Tabla 7: Tabla de requerimientos funcionales: Predicción a través de interfaz básica: Elaboración propia

2.4.2. Requerimientos no Funcionales

Tipo de requerimiento	Código	Especificación de requisitos
Requerimiento de Usabilidad	RN-F1	El código debe ser comprensible, con comentarios explicativos que faciliten la comprensión del flujo del trabajo.
Requerimiento de Mantenibilidad	RN-F2	El código debe estar estructurado de manera que sea fácilmente mantenible, permitiendo modificaciones futuras en los modelos , parámetros o técnicas sin complicaciones.
Requerimiento de Seguridad	RN-F3	El código no debe procesar ni almacenar datos sensibles. Todo el análisis debe realizarse con datos públicos proporcionados por el dataset.
Requerimiento de Portabilidad	RN-F4	El código debe poder ejecutarse correctamente en diferentes entornos (Google Colab, Jupyter Notebook) sin necesidad de realizar cambios significativos en la configuración.

Tabla 8: Tabla de requerimientos no funcionales: Elaboración propia

2.5. Diseño de la Propuesta

La arquitectura general del proyecto sigue un flujo de trabajo estructurado para procesar y analizar emociones en textos en español. El proceso comienza con el mensaje, el cual pasa por la fase del preprocesamiento, donde el texto es limpiado con el fin de eliminar elementos que son innecesarios y no aportan valor semántico en el análisis. Posteriormente, se realiza la extracción de características, donde el texto es transformado en representaciones vectoriales utilizando técnicas de Word Embedding, como Word2Vec o FastText. Estas representaciones permiten capturar el significado semántico del texto de forma numérica.

Los vectores generados sirven como entrada para el entrenamiento de modelos de Machine Learning (SVM, Regresión Logística y Naïve Bayes Gaussiano) y Deep Learning(BILSMT, LSTM, GRU). En esta etapa se realiza la división del conjunto de datos en 80% entrenamiento y 20% pruebas, se configuran los hiperparámetros de cada modelo, se realizan pruebas y se evalúa el rendimiento utilizando métricas como precisión, recall, F1-score y AUC. Finalmente, el modelo entrenado realiza la clasificación del texto, identificando las emociones presentes en cada mensaje.**:**

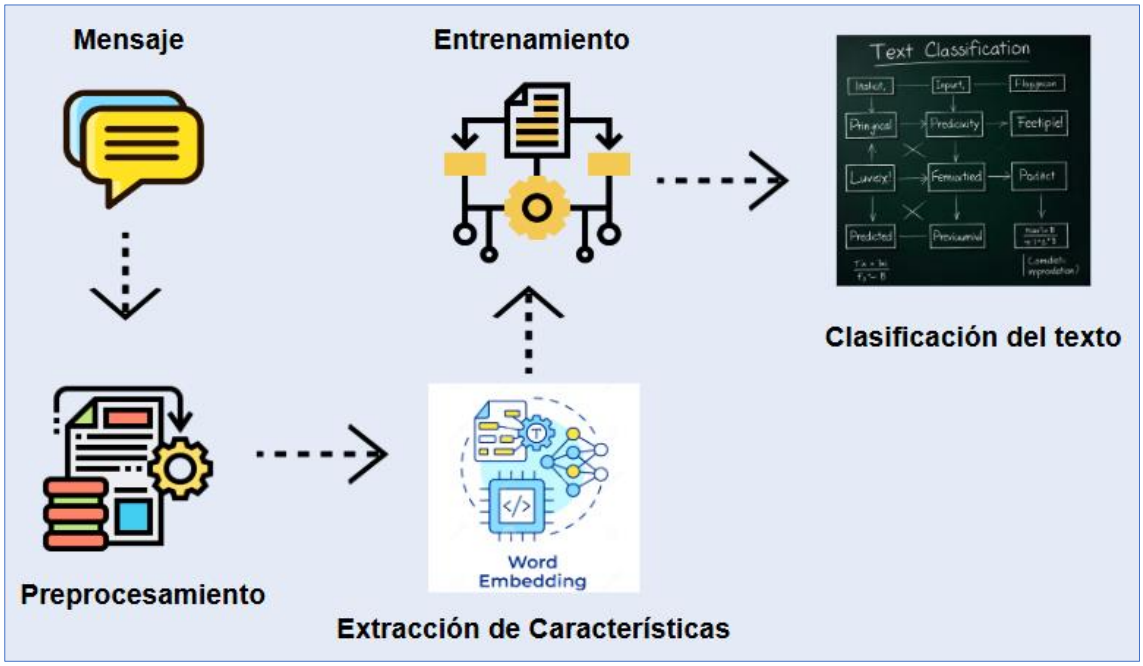


Figura 10: Arquitectura general del proyecto: Elaboración propia

2.5.1 Preprocesamiento

En la Fase del Preprocesamiento, el mensaje pasa por diversos procesos con el objetivo de transformar el texto original en una versión limpia y estructurada para ser analizadas por los modelos. Primero, se realiza la conversión del texto a minúsculas, para que el formato este uniforme y evitar que una misma palabra sea interpretada de manera distinta. Luego, se eliminan los números, ya que no aportan un valor semántico relevante para la detección de emociones.

En el siguiente paso , se eliminan los emojis y los signos de puntuación, elementos que pueden generar ruido, es decir, información irrelevante. Una vez completadas estas etapas se procede con la tokenización, que consiste en dividir el mensaje en palabras o tokens individuales. Finalmente, se eliminan los stopwords o palabras vacías, aquellas que no aportan significado útil al análisis, al finalizar todos los procesos se obtiene un mensaje limpio. Es importante señalar que durante este proceso se conservaron las tildes en las vocales y la letra “ñ”, ya que eliminarlas podría generar inconvenientes en la comprensión del contexto del mensaje, especialmente en el idioma español.

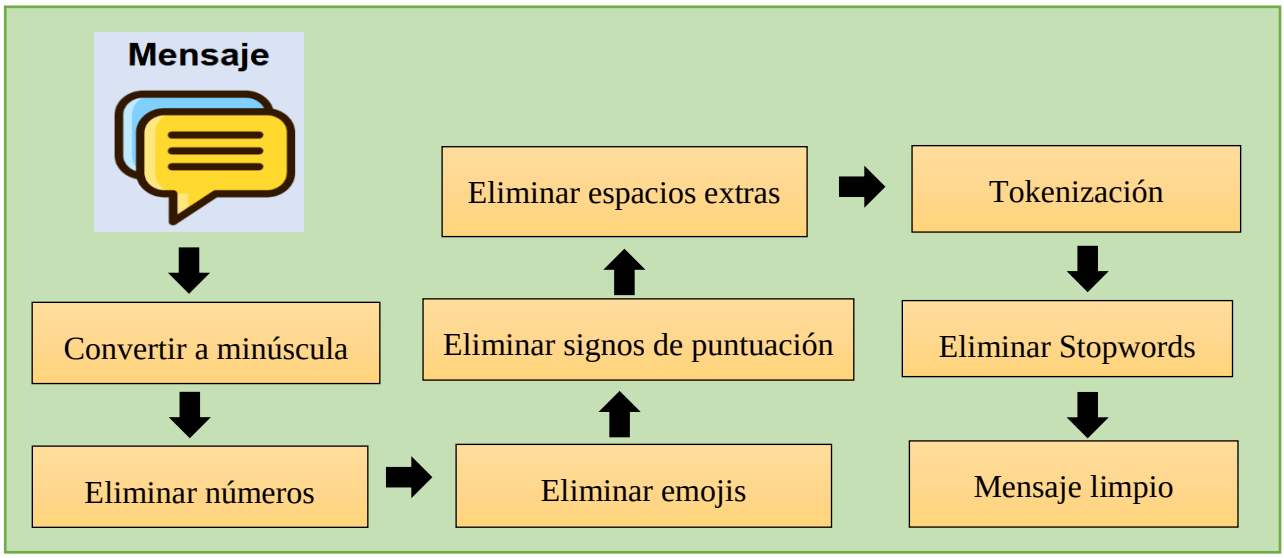


Figura 11: Preprocesamiento: Elaboración propia

2.5.2 Word Embedding

La Fase de Extracción de características, se realizó mediante dos técnicas de Word Embedding: FastText y Word2Vec. Estos modelos permiten representar las palabras en vectores numéricos, conservando sus relaciones semánticas. Se aplicó el modelo Skip-Gram, que permite predecir el contexto de una palabra y capturar relaciones semánticas más profundas. También se realizó el ajuste de hiperparámetros, configurando valores como la dimensión del vector, la ventada del contexto, el número de épocas, entre otros.

Debido a que cada tipo de algoritmo requiere un formato de entrada diferentes, se procedió de la siguiente manera:

- Para machine Learning, se calculó el vector promedio de cada comentario, simplificando así cada texto en una única representación vectorial. Esto generó una matriz en 2D, donde cada fila representa un comentario vectorizado.
- Para Deep Learning, se construyó la secuencia de embedding por palabra, lo que permite preservar el orden de las palabras en el texto. Además, para que todas las secuencias tuvieran la misma longitud, se aplicó un relleno de secuencias (padding). El resultado es una matriz en 3D, adecuada para la entrada en redes neuronales recurrentes

De esta manera, los datos fueron adaptados según el tipo de algoritmos a utilizar, asegurando una correcta representación para cada enfoque.

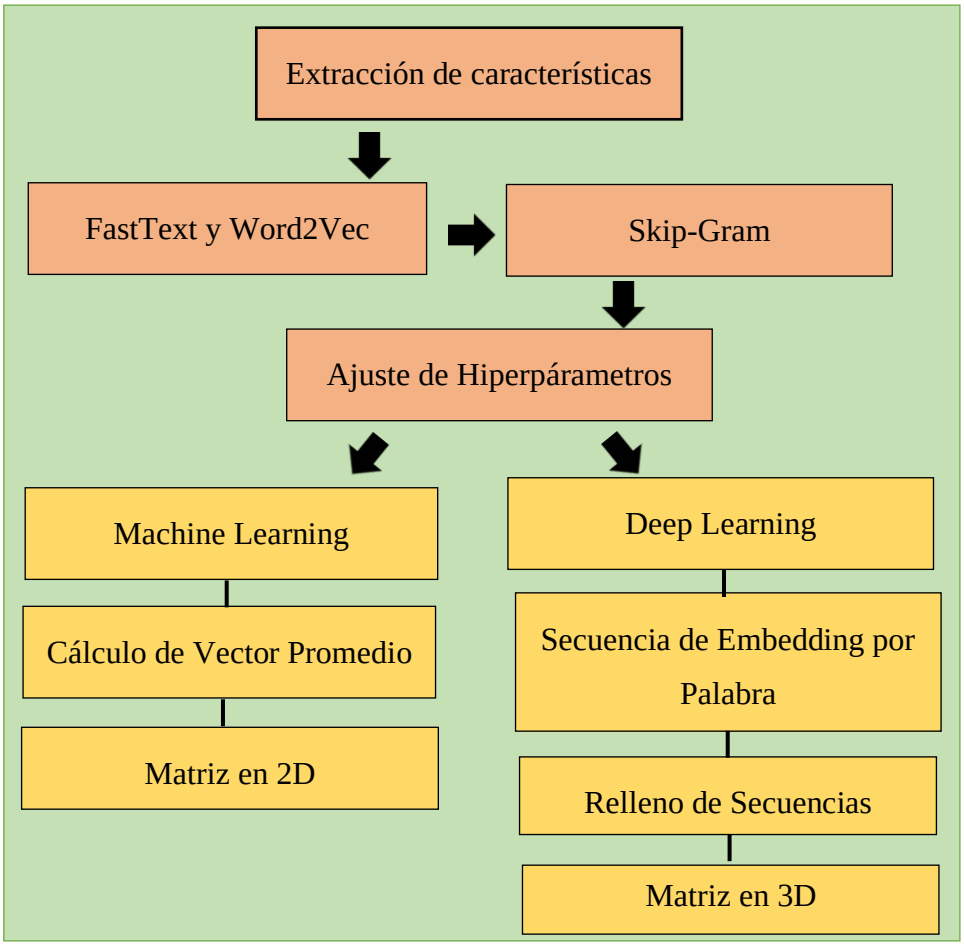


Figura 12: Extracción de características: Elaboración propia.

2.6 Desarrollo y pruebas

Fase1: Preparación de los Datos

Durante la primera fase se realizaron tres tareas fundamentales para preparar el conjunto de datos y dejarlo listo para su uso en modelos de clasificación multietiqueta basados en técnicas de Word Embedding (FastText y Word2vec). A continuación, se describen las actividades realizadas:

Tarea 1 carga y lectura del dataset: El entorno de trabajo empleado fue Google Colab, el cual permite una integración fluida con Google Drive y el aprovechamiento de recursos computacionales en la nube. El conjunto de datos se almaceno en formato Excel **(.xlsx).**

Para acceder al archivo desde Google drive, se utilizó la biblioteca **google.colab.drive.** Posteriormente, mediante la librería **pandas**, se realizó la

lectura del archivo, la selección de las columnas relevantes y la manipulación inicial de los datos. Además, las columnas seleccionadas fueron renombradas para mejorar la claridad y consistencia de la información.

Tarea 2 Tratamiento y normalización de etiquetas: Debido a que las emociones estaban escritas en diferentes formas (variaciones en el uso de mayúsculas, tildes o espacios), el modelo hubiese interpretado una misma emoción como etiquetas distintas. Para evitar este inconveniente, se realizó un proceso de normalización que consistió en la eliminación de tildes, la conversión a minúsculas y la eliminación de espacios en blanco innecesarios. Este procedimiento se llevó a cabo mediante la función **normalize_emotions(),** utilizando la biblioteca **unicodedata** para estandarizar los caracteres.

Tarea 3 Agrupación de emociones: Debido a que se maneja un Dataset pequeño y un total de 11 emociones las cuales son: egocentrismo, soberbia, sensualidad, disgusto/desagrado, antipatía, rechazo, enfado/enojo, odio, aceptación, orgullo, independencia. Para que no haya un desbalance muy drástico y reducir la complejidad del conjunto de etiquetas. Se implemento una estrategia de agrupación de emociones. Mediante la función **group_emotions()**, se definieron categorías más generales combinando las emociones similares en términos de significado o contexto emocional común**.**

La agrupación quedo establecida de la siguiente manera **:**

- **egocentrismo_soberbia**
- **disgusto_antipatia_rechazo**
- **enfado_enojo_odio**
- **aceptacion, orgullo, independencia, sensualidad (como etiquetas individuales)**

Por último, se utilizó la clase **MultiLabelBinarizer** de la biblioteca de **Scikit-learn** para transformar las etiquetas agrupadas en un formato binario multietiqueta. Este procedimiento generó una matriz binaria en la que cada fila representa un

comentario del dataset, y cada columna corresponde a una categoría de emoción, codificada como 1(presente) o 0 (ausente).

Fase 2 : Preprocesamiento

Tarea 1 limpieza de datos: En esta etapa se realizó un proceso de limpieza de texto, con el fin de reducir el ruido, eliminar elementos innecesarios, y preparar los datos de entrada para el entrenamiento de los modelos. Se implementaron diversas funciones de preprocesamiento utilizando bibliotecas de Python como **re y string.** A continuación, se describe el funcionamiento de cada una:

convert_to_lower: Convierte el texto en minúsculas mediante **text.lower()**, para garantizar uniformidad y evitar distinciones entre palabras mayúsculas y minúsculas.

remove_numbers: Elimina los números mediante una expresión regular **(\d+),** ya que generalmente no aportan significado emocional relevante al análisis del texto.

remove_emojis: Usa una expresión regular que abarca rangos Unicode específicos asociados con emojis y pictogramas **(como U+1F600 – U+1F64F, U+1F300 – U+1F5FF, entre otros).**

remove_punctuation: Elimina signos de puntuación como puntos, comas, signos de interrogación, entre otros, mediante el método **str.translate()** utilizando **string.punctuation.**

remove_extra_space: Reduce múltiples espacios a uno solo utilizando: **''.join(text.split()),** dejando el texto más limpio y legible**.**

preprocess_text: Función principal que aplica secuencialmente todas las funciones anteriores. Se ejecuta directamente sobre la columna **COMENTARIOS** del DataFrame, utilizando **apply().** El resultado del texto limpio se almacena en una nueva columna llamada **COMENTARIOS_LIMPIOS.**

Es importante destacar que, en el proceso de limpieza, no se eliminaron las tildes de las vocales ni la letra “ñ”. Esto se debe a que estos elementos son fundamentales

para la correcta interpretación y diferenciación de palabras en español, ya que su presencia o ausencia puede cambiar completamente el significado de una palabra.

Tarea 2 Tokenización: En esta etapa, cada comentario se divide en unidades léxicas básicas, denominadas tokens. Para realizar este proceso, se utilizó la biblioteca **Spacy**, específicamente el modelo preentrenado **es_core_news_md,** que contiene vocabulario, reglas gramaticales y lematización entrenados sobre corpus en idioma español. La carga del modelo se realiza con: **nlp = spacy.load("es_core_news_md").**

La función de tokenización se aplica sobre la columna COMENTARIOS_LIMPIOS del DataFrame. El resultado se guarda en una nueva columna llamada COMENTARIOS_TOKENIZADOS. Además, se asegura que no haya valores nulos en la columna de tokens. Este proceso es crucial ya que divide el texto en componentes significativos que serán utilizados como entradas para los modelos de Word Embedding (Word2Vec, FastText) y Deep Learning (LSTM, BiLSTM, GRU).

Tarea 3 eliminación de stopwords: En esta etapa se eliminan las palabras irrelevantes que no aportan valor semántico al texto. Para ello, se combinaron técnicas tradicionales con análisis gramatical, utilizando 2 herramientas especializadas para el idioma español:

- **NLTK (Natural Language Toolkit):** Se empleó el corpus de stopwords en español **(stopwords.words('spanish'))** para eliminar palabras de alta frecuencia, pero bajo contenido semántico.
- **spaCy**: Se utilizo el modelo es_core_news_md, que incluye capacidades de POS tagging (etiquetado gramatical). Esto permite identificar y filtrar categorías gramaticales específicas.

El proceso seguido fue el siguiente:

Definición de palabras claves relevantes: Se construyo un conjunto de términos emocionales significativos, los cuales se eliminan del conjunto original de

stopwords, creando así una lista personalizada más adecuada para la tarea de clasificación de emociones.

Conversión del texto a objeto SpaCy: Cada comentario limpio fue procesado con **nlp(text),** lo que permitió representar el contenido como un objeto **Doc** y acceder a los atributos gramaticales de cada token.

Aplicación de filtros específicos: Se conservan únicamente aquellos tokens que cumplieran simultáneamente con los siguientes criterios:

- No pertenecer al conjunto de stopwords personalizado
- No ser signos de puntuación **(**verificado mediante **token.is_punct == False).**
- No pertenecen a las siguientes categorías gramaticales, específicamente:
- ADP: preposiciones
- CONJ: conjunciones coordinadas
- SCONJ: conjunciones subordinadas.
- Tener longitud mayor a un carácter, eliminando tokens de una sola letra.

Este enfoque, que integra eliminación de stopwords tradicional con filtrado basado en **(POS tagging),** permite mejor depuración más precisa y contextualizada del texto.

Fase 3 : Vectorización de Palabras

Tarea 1 técnica de Word embedding: Para generar representaciones vectoriales densas, que capturen relaciones semánticas entre palabras en idioma español, se entrenaron dos modelos de Word Embedding utilizando la biblioteca **Gensim** :

FastText **(from gensim.models import FastText):** Este modelo permite representar cada palabra como una suma de n-gramas de caracteres, lo cual permite mejorar su capacidad para manejar variaciones morfológicas y palabras pocos frecuentes.

Word2Vec **(from gensim.models import Word2Vec):** Esta técnica representa cada palabra como un vector denso en un espacio semántico, permitiendo capturar relaciones contextuales entre palabras basadas en su coocurrencia.

Además, ambos modelos fueron entrenados utilizando el algoritmo **Skip-gram (sg=1),** debido a que es más eficaz en conjuntos de datos pequeños o cuando se

quiere dar mayor importancia a la representación de las palabras que no son tan frecuentes.

Se definieron los siguientes hiperparámetros para el entrenamiento de los modelos:

Word embedding			
Hiperparámetros	**FastText**	**Word2Vec**	**Descripción**
vector_size	200	200	Dimensión del espacio vectorial.
window	5	5	Tamaño de la ventana de contexto (palabras vecinas consideradas).
min_count	1	1	palabras con frecuencia menor a este valor son descartadas
sg	1	1	algoritmo Skip-Gram, más eficiente en corpus pequeños
epochs	35	35	Numero de épocas de entrenamiento.
negative	15	15	Número de muestras negativas.
min_n	2	-	Longitud mínima de los n-gramas de caracteres.
max_n	4	-	Longitud máxima de los n-gramas de caracteres.
workers	4	4	Numero de hilos utilizados en paralelo para acelerar el proceso de entrenamiento.

Tabla 9: Tabla de Hiperparámetros de Word Embedding: Elaboración propia

Tarea 2 cálculo de vector promedio para FastText y Word2Vec para Machine Learning: Para generar representaciones vectoriales, se aplicó el método del vector promedio, el cual consiste en promediar los vectores de cada palabra presente en un comentario. Inicialmente, se utilizó el método .**split()** para convertir los comentarios en listas de palabras. Posteriormente, se definió la función **obtener_vector_promedio_fasttext(),** la cual toma como entrada un comentario tokenizado y calcula su representación vectorial promedio, utilizando los embedding generados por el modelo FastText.

La función fue aplicada a cada comentario del conjunto de datos mediante el método **apply()** de **Pandas**, operando sobre la columna **COMENTARIOS_SIN_STOPWORD**. Los vectores promedio generados fueron almacenados en una nueva columna de DataFrame llamada **VECTOR_PROMEDIO_FASTTEXT.** Para el caso de Word2Vec, se utilizó el mismo procedimiento, modificando únicamente el nombre de la función, el modelo de embedding utilizado y el nombre de la columna **(VECTOR_PROMEDIO_WORD2VEC).**

Tarea 3 secuencia de vectores de palabra para FastText y Word2Vec aplicados en modelos de Deep Learning: A diferencia de los modelos de Machine Learning, que requieren representar cada comentario como un único vector promedio, los algoritmos de Deep Learning como LSTM, BiLSTM o GRU requieren como entrada una secuencia de vectores, uno por cada palabra del comentario. Para ello, se parte de la columna COMENTARIOS_SIN_STOPWORDS, la cual contiene listas de palabras ya tokenizadas por comentario. Esto es fundamental, ya que permite mapear cada palabra a su respectivo vector usando el modelo de Word Embedding entrenado.

Se definió la función **obtener_embedding_palabra(),** la cual recorre cada palabra del comentario y extrae su embedding desde el modelo FastText (mediante **fasttext_model.wv[palabra]).** El resultado intermedio es una lista llamada **X_ft_secuencias**, donde cada comentario ha sido convertido con una secuencia de vectores de dimensión 200 (de acuerdo con el tamaño de embedding definido al entrenar).
Se aplicó la función **pad_sequences() de Keras** para igualar la longitud de todas las secuencias, agregando relleno al final **(padding='post').** Esto es un requisito para que las redes neuronales puedan procesar los datos en batch, ya que esperan entradas de dimensiones uniformes. Para definir la longitud máxima de las secuencias (maxlen), se realizó un análisis estadístico de la longitud de los comentarios del dataset. Se observo que el 75% de los comentarios tenían 26 palabras o menos y el máximo era de 60. Con base en esto se fijó maxlen=40 como un valor intermedio que permite preservar la información complete en la mayoría

de los casos, evitando el truncamiento excesivo de información y manteniendo equilibrio entre contexto semántica y eficiencia computacional.

El resultado final es un arreglo tridimensional llamado **X_ft_secuencias_pad** con la forma **(n_comentarios, secuencia_maxima, dimensión),** que representa el conjunto de entradas listas para alimentar a un modelo de Deep Learning. Para el modelo de Word2Vec se repite el mismo procedimiento, cambiando el modelo (**model_w2v** en lugar de **fasttext_model**) y los nombres de las variables correspondientes.

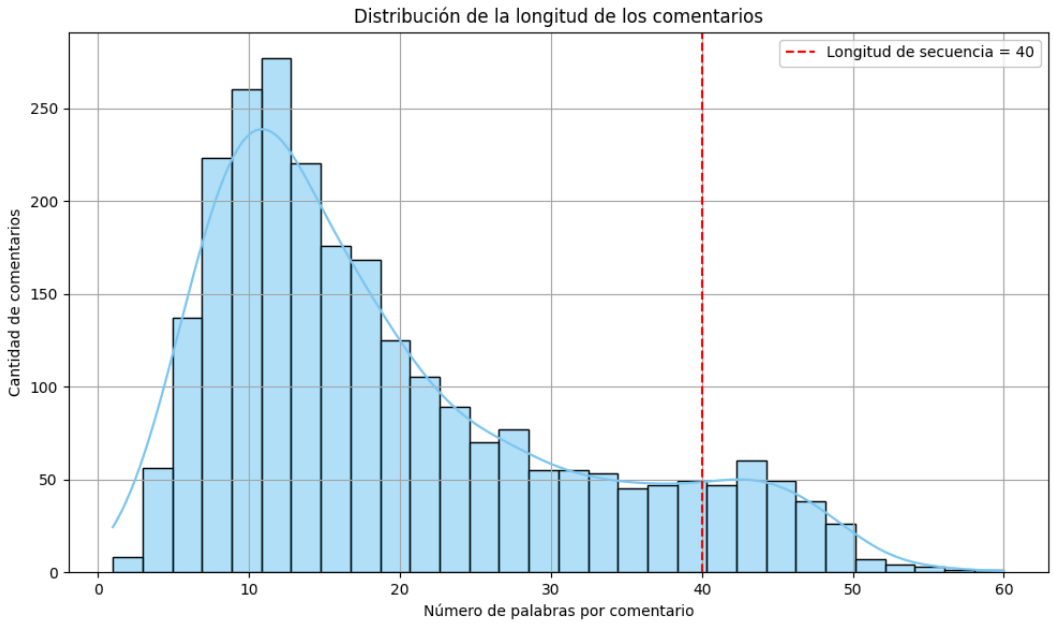


Figura 13: Distribución de la longitud de los comentarios. Base para la selección de la longitud máxima de secuencia: Elaboración propia.

Fase 4 : Entrenamiento

Tarea1 División de datos para el entrenamiento para Machine Learning :

Se utilizó la columna VECTOR_PROMEDIO_FASTTEXT del DataFrame, que fue obtenido calculando el promedio de los embedding, resultando en vectores de dimensión 200 por observación. Además, antes de construir la matriz de entrada se realizó un filtrado del conjunto de datos para eliminar aquellas observaciones cuyo vector promedio este vacío o nulo, para poder tener validación en los datos.

Posteriormente, se construyó la matriz de entrada **X_ft** extrayendo los vectores de dicha columna y convirtiéndolos en un arreglo **NumPy** con forma **(n_samples, 200),** donde **n_samples** representa la cantidad de comentarios y 200 la dimensión de embeddings. Las etiquetas asociadas a cada comentario se extrajeron de la columna EMOCIONES (listas de emociones por comentario) y se transformaron mediante la clase MultiLabelBinarizer, generando una matriz binaria **y_bin** de forma (n_samples, n_clases), adecuada para tareas de clasificación multietiqueta.

Finalmente, se realizó la división del conjunto de datos utilizando la función **train_test_split** de la biblioteca **scikit-learn**, asignando el 80% de los datos para entrenamiento y el 20% restante para prueba. Esto produjo las matrices **X_train_ft y X_test_ft** (vectores de entrada) y **y_train_ft / y_test_ft** (etiquetas). Se realizó el mismo procedimiento a la técnica de Word2Vec, ajustando únicamente las variables involucradas.

Tarea 2 División de datos con secuencias de vectores para Deep Learning:
Una vez generadas las secuencias de vectores, se realiza la división del conjunto de datos, para ello se utilizó la función **train_test_split** de la biblioteca **scikit-learn,** asignando el 80% de las muestras para el entrenamiento y el 20% para prueba. La entrada **(X_ft_padded)** corresponde a una matriz tridimensional con forma **(n_samples, secuencia_maxima, 200),** y las etiquetas (y) fueron previamente transformadas mediante codificación multietiqueta utilizando **MultiLabelBinarizer.** Como resultado, se obtuvieron las siguientes matrices: X**_train_ft_padded y X_test_ft_padded** para las secuencias de entrada y **y_train_ft y y_test_ft** para etiquetas codificadas. Se realizó el mismo procedimiento a la técnica de Word2Vec, ajustando únicamente las variables involucradas.

Tarea 3 hiperparámetros de los modelos de Machine Learning para FastText y Word2Vec:

MODELO SVM		
HIPERPARAMETRO	**VALOR**	**VALOR DESCRIPCIÓN**
C	0.01	Inverso de la regularización

kernel	linear	Función núcleo que transforma los datos.
class_weight	balanced	Ajusta automáticamente los pesos de la clase según su frecuencia
tol	1e-5	Tolerancia para el criterio de convergencia.
max_iter	5000	Número máximo de iteraciones permitidas para la optimización.
probability	True	Habilita la estimación de probabilidades para cada clase

Tabla 10: Tabla de Hiperparámetros del Modelo SVM: Elaboración propia

MODELO REGRESIÓN LOGÍSTICA		
HIPERPARAMETRO	**VALOR**	**VALOR DESCRIPCIÓN**
C	0.001	Inverso de la regularización
solver	liblinear	Algoritmo para la optimización
class_weight	balanced	Ajusta automáticamente los pesos de la clase según su frecuencia
tol	1e-5	Tolerancia para el criterio de convergencia.
max_iter	5000	Número máximo de iteraciones permitidas para la optimización.
fit_intercept	True	Permite que el modelo aprenda un término independente
penalty	l2	Tipo de regularización aplica
multi_class	ovr	Estrategia "one-vs-rest" para clasificación multietiqueta

Tabla 11: Tabla de Hiperparámetros del Modelo de Regresión Logística: Elaboración propia

MODELO NAÏVE BAYES GAUSSIANO (GAUSSIANNB)		
HIPERPARAMETRO	**VALOR**	**VALOR DESCRIPCIÓN**
var_smoothing	1e-2	Suma un pequeño valor a la varianza para evitar errores matemáticos y hacer más estables los cálculos del modelo

Tabla 12: Tabla de Hiperparámetros del Modelo de Naïve Bayes Gaussiano: Elaboración propia

Tarea 4 hiperparameros de los modelos de Deep Learning para FastText y Word2Vec:

MODELO BILSMT		
HIPERPARAMETRO	**VALOR**	**VALOR DESCRIPCIÓN**
Unidades LSTM	128	Número de neuronas en la capa BiLSTM.
return_sequences	False	Devuelve sola la última salida, útil para clasificación.
Dropout (1ra capa)	0.3	Proviene sobreajuste desactivando neuronas aleatoriamente
Dropout (2da capa)	0.2	Dropout adicional antes de la salida para regularización
Capa Densa Intermedia	128	Numero de neuronas en la capa, que permite cambiar la información aprendida.
Activation capa intermedia	tanh	Función de activación que introduce no linealidades, ayudando que el modelo aprenda patrones complejos.
activation (capa de salida)	sigmoid	Genera probabilidades independientes por clase.
Optimizer	Nadam	Optimizador que combina la adaptabilidad de Adam, con la velocidad de Nesterov.
learning_rate	0.001	Controla la magnitud de los ajustes realizados en los pesos del modelo durante el entrenamiento.

Función de pérdida	binary_crossentropy	Para clasificación multietiqueta, calcula el error por clase.
patience	5	Detiene el entrenamiento si no hay mejora en la métrica de validación tras 5 épocas consecutivas.
Épocas(epochs)	35	Iteraciones máximas de entrenamiento
batch_size	32	Numero de muestras utilizadas para actualizar los pesos en cada iteración.
Umbral de predicción	0.3	Valor mínimo de probabilidad para considerar que una clase ha sido predicha.

Tabla 13: Tabla de Hiperparámetros del Modelo del BILSMT: Elaboración propia

MODELO LSTM		
HIPERPARAMETRO	**VALOR**	**VALOR DESCRIPCIÓN**
Unidades LSTM	64	Numero de neuronas en la capa LSTM.
return_sequences	False	Devuelve sola la última salida, útil para clasificación.
Dropout (1ra capa)	0.3	Proviene sobreajuste desactivando neuronas aleatoriamente
Dropout (2da capa)	0.3	Dropout adicional antes de la salida para regularización
Capa Densa Intermedia	128	Numero de neuronas en la capa, que permite combinar la información aprendida.
activation (capa de salida)	sigmoid	Genera probabilidades independientes por clase.
Optimizer	Nadam	Optimizador que combina la adaptabilidad de Adam, con la velocidad de Nesterov.

learning_rate	0.001	Controla la magnitud de los ajustes realizados en los pesos del modelo durante el entrenamiento.
Función de pérdida	binary_cros sentropy	Para clasificación multietiqueta, calcula el error por clase.
patience	5	Se detiene si no hay mejora tras 5 épocas.
Épocas(epochs)	30	Iteraciones máximas de entrenamiento
batch_size	16	Numero de muestras utilizadas para actualizar los pesos en cada iteración.
Umbral de predicción	0.3	Probabilidad mínima para predecir clase.

Tabla 14: Tabla de Hiperparámetros del Modelo del LSMT: Elaboración propia

MODELO GRU		
HIPERPARAMETRO	**VALOR**	**VALOR DESCRIPCIÓN**
Unidades LSTM	128	Numero de neuronas en la capa BiLSTM.
return_sequences	False	Devuelve sola la última salida, útil para clasificación.
Dropout (1ra capa)	0.2	Proviene sobreajuste desactivando neuronas aleatoriamente
Dropout (2da capa)	0.2	Dropout adicional antes de la salida para regularización
Capa Densa Intermedia	64	Numero de neuronas en la capa, que permite combinar la información aprendida.
activation (capa de salida)	sigmoid	Genera probabilidades independientes por clase.
Optimizer	Nadam	Optimizador que combina la adaptabilidad de Adam, con la velocidad de Nesterov.
learning_rate	0.001	Controla la magnitud de los ajustes realizados en los pesos del modelo durante el entrenamiento.

Función de pérdida	binary_cros sentropy	Para clasificación multietiqueta, calcula el error por clase.
patience	5	Se detiene si no hay mejora tras 5 épocas.
Épocas(epochs)	35	Iteraciones máximas de entrenamiento
batch_size	32	Numero de muestras utilizadas para actualizar los pesos en cada iteración.
Umbral de predicción	0.2	Probabilidad mínima para predecir clase.

Tabla 15: Tabla de Hiperparámetros del Modelo del GRU: Elaboración propia

Fase 5 : Evaluación

Tarea 1 Métricas: Para evaluar el rendimiento de los modelos , se utilizó el método **classification_report** de la librería **sklearn.metrics**, el cual genera un resumen detallado de las métricas de clasificación para cada una de las emociones. El análisis se realizó con las siguientes métricas de evaluación para cada emoción:

Precisión: Nos permite determinar cuan confiable es el modelo al predecir una emoción especifica, ayuda a saber si está etiquetando emociones erróneamente en comentarios que no las contenían (falsos positivos), esto es importante cuando una mala clasificación puede malinterpretar el estado emocional real de un comentario.

Recall (Sensibilidad o Exhaustividad): Nos indica la capacidad que tiene el modelo para detectar todos los comentarios que realmente contienen una emoción. Por ejemplo: si hay una cantidad de comentarios con una emoción, pero el modelo solo detecta algunos, el recall será bajo, esto implicaría que el modelo no los está reconociendo, representando una pérdida de información emocional.

F1-Score: Debido a que las emociones utilizadas son complicadas de identificar, esta métrica nos permite medir el equilibrio, entre que tan bien detecta la emoción y que tan preciso es al predecirla, especialmente en multietiqueta, ayuda a tener una visión más general más justa del rendimiento real por emoción.

Support: Muestra cuantos ejemplos hay de cada emoción, ayuda a interpretar el contexto de cada emoción, si tiene poco soporte (support), un buen resultado no puede ser tan confiable, también permite ver el desbalance de datos, lo cual influye en el desempeño del modelo.

Macro avg: Evalúa el rendimiento del modelo de forma equilibrada considerando a todas las emociones por igual, sin importar cuantos ejemplos tenga cada una. Representa el modelo de forma justa con emociones pocos frecuentes, no solo se enfoca en las más comunes.

Weighted avg: Refleja el rendimiento general del modelo, es decir, este promedio da una evaluación más realista del comportamiento global del modelo, considerando que algunas emociones están más representadas que otras.

Tarea 2 AUC(Área bajo la curva) Y ROC: Para complementar el análisis del rendimiento de los modelos, se implementó un gráfico de curvas ROC por clase, así como el cálculo del AUC promedio del modelo, de manera que podamos evaluar su capacidad para distinguir las diferentes emociones.
Primero, se binarizaron las etiquetas utilizando la función **label_binarize**, de la librería **sklearn,** lo cual es necesario ya que la curva ROC se basa en comparaciones binarias(una emoción contra las demás). Después, se obtuvieron las probabilidades de predicción para cada clase utilizando **predict_proba** del modelo entrenado.

Posteriormente, se calculó la tasa de verdaderos positivos (TPR) y la tasa de falsos positivos (FPR) para cada clase, y se graficó la curva ROC correspondiente. Cada curva fue acompañada de su valor AUC, el cual representa la capacidad del modelo para distinguir entre clases, un valor más cercano a 1 indica un mejor rendimiento. Finalmente, se realizó el cálculo AUC promedio macro utilizando **roc_auc_score** con el parámetro **average='macro'.** Este valor nos proporciona tener una visión general del rendimiento de modelo, considerando todas las emociones por igual, independientemente del número de ejemplos por clase.

Tarea 2 Matriz de confusión multietiqueta: Aunque la matriz de confusión no es una métrica como tal, si es una herramienta útil para analizar el rendimiento real del modelo. Permite visualizar el número de verdaderos positivos (TP), falsos positivos (FP), verdaderos negativos (TN) y falsos negativos (FN) por cada emoción.

Para obtener la matriz de confusión multietiqueta, se utilizó la función **multilabel_confusion_matrix** de la librería **sklearn.metrics**, la cual genera una matriz de confusión binaria para cada una de las emociones evaluadas. A continuación, se procesan los valores TN, FP, FN y TP y se resumen en un solo arreglo para facilitar la visualización mediate un heatmap, utilizando las bibliotecas **seaborn** y **matplotlib.**

2.7. Resultados

Machine Learning								
Técnica	**FaxtText**				**Word2Vec**			
Algoritmo	Precisión	Recall	F1-Score	AUC	Precisión	Recall	F1-Score	AUC
SVM	0.48	0.73	0.57	0.78	0.50	0.71	0.58	0.79
Regresión logística	0.45	0.76	0.56	0.77	0.45	0.73	0.56	0.76
GaussianNB	0.46	0.67	0.54	0.75	0.48	0.69	0.56	0.77

Tabla 16: Tabla de Métricas de Machine Learning: Elaboración propia

Comparación entre FastText y Word2Vec

Entre ambas técnicas, Word2Vec mostró un rendimiento ligeramente superior a FastText en algunos casos, sobre todo en el algoritmo SVM, donde alcanzó un F1-Score de 0.58 y un AUC de 0.79. Sin embargo, en términos generales ambos proporcionan resultados similares. Esto indica que, independiente del embedding utilizado, ambos lograron capturar cierta información semántica útil para la clasificación multietiqueta. Cabe resaltar que su rendimiento influyó por el tamaño del conjunto de datos, limitando su aprendizaje.

Interpretación de métricas

Precisión: Los modelos evaluados mostraron una precisión entre 0.45 y 0.50, lo que significa que aproximadamente la mitad de las emociones predichas fueron correctas. Aunque este valor es razonable, puede verse afectado por la presencia de múltiples emociones en un mismo texto, además de la ambigüedad propia del lenguaje emocional, lo que dificulta al modelo identificar con precisión cada emoción presente.

Recall: Se obtuvieron valores entre 0.67 y 0.76, lo que refleja una buena capacidad de los modelos para detectar la mayoría de las emociones reales presentes en los comentarios. Este comportamiento puede atribuirse a la naturaleza multietiqueta del problema, en el cual los modelos tienden a predecir varias etiquetas por comentario, lo cual favorece al recall, aunque a veces afectando a la precisión.

F1-Score: Con valores entre los 0.54 y 0.58, representa un equilibrio razonable entre la precisión y la capacidad del modelo para identificar emociones (recall) , estos valores reflejan la dificultad del modelo para balancear correctamente los aciertos sin incurrir en errores, especialmente ante clases desbalanceadas y un conjunto de datos limitado, factores que dificultan el los resultados.

AUC (Área bajo la curva ROC ponderada): Con valores entre 0.75 y 0.79, esta métrica indica un desempeño aceptable en la capacidad de los modelos para distinguir entre diferentes clases emocionales, incluso bajo condiciones poco ideales. Estos resultados son prometedores, considerando los desafíos que implica trabajar con textos en español y complejidad semántica del lenguaje emocional.

Comparación entre algoritmos: En general los resultados obtenidos mediante los algoritmos de Machine Learning, indican que el modelo SVM con Word2Vec obtuvo el mejor rendimiento, alcanzando una precisión de 0.50, un recall de 0.71, un F1-Score de 0.58 y un AUC de 0.79. Este modelo demostró ser el más eficaz para identificar múltiples emociones por comentario, lo que significa que Word2Vec proporciono una representación semántica útil para la discriminación entre clases.

El modelo de Regresión Logística presentó un rendimiento similar al de SVM cuando se utilizó con FastText. Sin embargo, con la técnica de Word2Vec su

rendimiento disminuyo levemente, lo que puede deberse a que este algoritmo depende principalmente de relaciones lineales entre variables, una limitación que podría impedirle aprovechar por completo la riqueza semántica de los vectores generados por Word2Vec.

El modelo Naïve Bayes Gaussiano fue el que obtuvo el menor rendimiento, especialmente en el F1-Score, con un valor de 0.54. Se optó por utilizar la variante Gaussiana debido a que los embedding generan representaciones continuas, lo cual no es compatible con la versión Multinomial, que está orientada al tratamiento de conteos discretos o conteo de palabras. No obstante, su rendimiento fue inferior a los demás, probablemente porque el modelo asume independencia entre las características, una condición que no se cumple en los embedding donde las dimensiones suelen estar altamente correlacionadas, afectando la capacidad predictiva del modelo.

En resumen, los resultados fueron similares y se mantuvieron estables, aunque no hubo grandes diferencias entre las técnicas y los algoritmos utilizados, considerando la complejidad de trabajar con texto en español y con múltiples emociones por comentario, es decir, mostraron un rendimiento razonable al identificar emociones.

Deep Learning								
Técnica	**FaxtText**				**Word2Vec**			
Algoritmo	Precisión	Recall	F1Score	AUC	Precisión	Recall	F1-Score	AUC
BiLSTM	0.52	0.70	0.59	0.80	0.50	0.70	0.58	0.79
LSTM	0.51	0.68	0.57	0.78	0.50	0.66	0.55	0.77
GRU	0.40	0.80	0.52	0.73	0.37	0.86	0.52	0.69

Tabla 17: Tabla de Métricas de Deep Learning: Elaboración propia

Comparación entre FastText y Word2Vec

En los modelos de Deep Learning, se observaron resultados generales similares entre los embedding, aunque FaxtText mostro una leve ventaja en términos de precisión, alcanzando un valor de 0.52 y un AUC de 0.80 con el modelo BiLSTM.

Por otro lado, Word2Vec logró un rendimiento muy cercano, especialmente en F1-Score, sin embargo, no supero a FastText en AUC. Esta similitud de rendimiento sugiere que las arquitecturas de redes neuronales profundas son capaces de adaptarse a distintas representaciones semánticas, siempre que exista coherencia contextual en los vectores generados.

Interpretación de métricas

Precisión: Los modelos evaluados mostraron una precisión entre 0.40 y 0.52, lo cual indica que aproximadamente la mitad de las emociones predichas fueron correctas. Esta métrica puede verse afectada por la complejidad del lenguaje emocional , y el hecho de que muchos comentarios tengan múltiples emociones, lo que tiende a dificultar una predicción precisa.

Recall: Se obtuvieron valores entre 0.66 y 0.86, lo cual demuestra que los modelos son buenos para detectar la mayoría de las emociones verdaderas presentes en los comentarios. Este desempeño está relacionado con las redes neuronales recurrentes, para mantener el contexto a lo largo de la secuencia, favoreciendo así la detención de múltiples etiquetas.

F1-Score: Los modelos alcanzaron valores entre 0.52 a 0.59 , lo cual indica un equilibrio razonable entre precisión y recall. Esta métrica se convierte en un indicador esencial en problemas multietiqueta. Aunque los resultados no son sobresalientes, son aceptables dadas las características del problema. Los modelos lograron capturar patrones relevantes sin inclinarse excesivamente hacia falsos positivos o negativos.

AUC (Área bajo la curva ROC ponderada): El AUC de los modelos vario entre 0.69 y 0.80, el más alto lo obtuvo BiLSTM con FastText, lo que reflejan una capacidad adecuada para diferenciar entre las emociones, a pesar de las dificultades

asociadas a la ambigüedad entre categorías. Un AUC superior a 0.70 se considera positivo, lo que refuerza la efectividad de los modelos.

Comparación entre algoritmos

Entre los modelos de Deep Learning, el que obtuvo el mejor rendimiento fue BiLSTM , con un F1-Score de 0.59 con FastText y 0.58 con Word2Vec, además de un AUC de 0.80 y 0.79, respectivamente. Este desempeño se puede atribuir a su estructura bidireccional, que le permite procesar secuencias de texto en ambas direcciones (hacia adelante y hacia atrás), es un aspecto importante en tareas de detección de emociones, especialmente cuando involucra ambigüedad emocional y etiquetas múltiples.

El modelo LSTM alcanzó un F1-Score de 0.57 con FastText y 0.55 con Word2Vec, junto a un recall de 0.68 y 0.66, respectivamente. A pesar de no contar con la bidireccionalidad, LSTM demostró ser una buena alternativa capaz de capturar emociones presentes en los textos.

El modelo que tuvo el rendimiento más bajo fue GRU, con un F1-Score de 0.52 en ambas técnicas y un AUC de 0.73 (FastText) y 0.69 (Word2Vec). Aunque GRU es un modelo más eficiente en términos computacionales, sus resultados indican que tiene una menor capacidad para captar relaciones emocionales complejas presentes en los comentarios, en comparación con los modelos LSTM y BiLSTM.

Machine Learning								
Técnica	**FaxtText**				**Word2Vec**			
Algoritmo	Precision	Recall	F1-Score	AUC	Precision	Recall	F1-Score	AUC
SVM	0.48	0.73	0.57	0.78	0.50	0.71	0.58	0.79
Regresión logística	0.45	0.76	0.56	0.77	0.45	0.73	0.56	0.76
Gaussian NB	0.46	0.67	0.54	0.75	0.48	0.69	0.56	0.77
Deep Learning								
Técnica	**FaxtText**				**Word2Vec**			
Algoritmo	Precision	Recall	F1-Score	AUC	Precision	Recall	F1-Score	AUC
BiLSTM	0.52	0.70	0.59	0.80	0.50	0.70	0.58	0.79
LSTM	0.51	0.68	0.57	0.78	0.50	0.66	0.55	0.77
GRU	0.40	0.80	0.52	0.73	0.37	0.86	0.52	0.69

Tabla 18: Tabla de Comparación de Métricas de Machine y Deep Learning: Elaboración propia

Mejor combinación embedding y algoritmo.

De acuerdo a los resultados obtenidos, podemos observar que la mejor combinación de técnica de Word Embedding y modelo fue FaxtText con el modelo BiLSTM,

alcanzando una Precisión de 0.52, Recall de 0.70, F1-Score de 0.59 y un AUC de 0.80. Esta configuración logró un balance entre la precisión y la capacidad del modelo para recuperar correctamente las etiquetas emocionales (recall), lo que es importante destacar en problemas de multietiqueta como la clasificación de emociones

Comparación general entre técnicas (FastText vs Word2Vec)

Al realizar la comparación entre las técnicas de Word Embedding utilizadas, FastText superó ligeramente a Word2Vec, en la mayoría de los modelos evaluados. Destacó especialmente en BiLSTM y LSTM, donde alcanzó los mejores resultados en F1-Score y AUC. Por otro lado, Word2Vec tuvo un mejor desempeño en algunos modelos de Machine Learning, como SVM, donde alcanzó un F1-Score de 0.71, superior al 0.57 de FastText. Sin embargo, obtuvo un recall más bajo, lo que indica menor capacidad para detectar múltiples emociones. En general, ambos métodos demostraron ser efectivos, pero FastText mostró un rendimiento más consistente en modelos de Deep Learning, lo que sugiere que representa mejor la semántica contextual en este tipo de tareas.

Comparación Machine Learning vs Deep Learning

En términos generales, los modelos de Deep Learning, obtuvieron un mejor rendimiento frente a los algoritmos tradicionales de Machine Learning. Esta diferencia se evidenció especialmente en las métricas como el recall y el área bajo la curva (AUC), métricas que son fundamentales en esta investigación debido al enfoque de multietiqueta empleado, ya que reflejan la capacidad del modelo para identificar correctamente la mayor cantidad de emociones posibles por comentarios.

Modelos como BiLSTM y LSTM, que son redes neuronales recurrentes, aprovecharon su capacidad para capturar la secuencia y el contexto de las palabras en los textos, lo cual es esencial en tareas de Procesamiento de Lenguaje Natural (PLN). Debido a eso, ofrecieron una mejor detección emocional que los modelos clásicos como SVM o regresión logística.

Sin embargo, los resultados obtenidos reflejan un rendimiento moderado, principalmente por limitaciones del conjunto de datos empleado, que presenta características que complicaron el proceso de aprendizaje automático, como el tamaño relativamente pequeño, la presencia de múltiples emociones por comentario, el desbalance entre clases, la ambigüedad de las palabras, es decir, la calidad de los datos, tales factores representan un desafío significativo para los algoritmos, afectando la capacidad de aprendizaje y generalización de los modelos, especialmente bajo el enfoque multietiqueta, donde las diferencias entre emociones no siempre son claras o están bien delimitadas, lo que dificulta que el modelo los distinga con precisión.

Finalmente, a pesar de estas dificultades, las métricas de precisión, recall, F1-Score y AUC, se mantuvieron dentro de un rango estable, lo cual es un indicativo positivo considerando la complejidad del problema abordado. Esto sugiere que los modelos poseen una capacidad razonable para reconocer diversas emociones en textos, con un desempeño aceptable pero que se puede continuar fortaleciendo los modelos con mejora de datos y en la arquitectura.

Matriz de Confusión

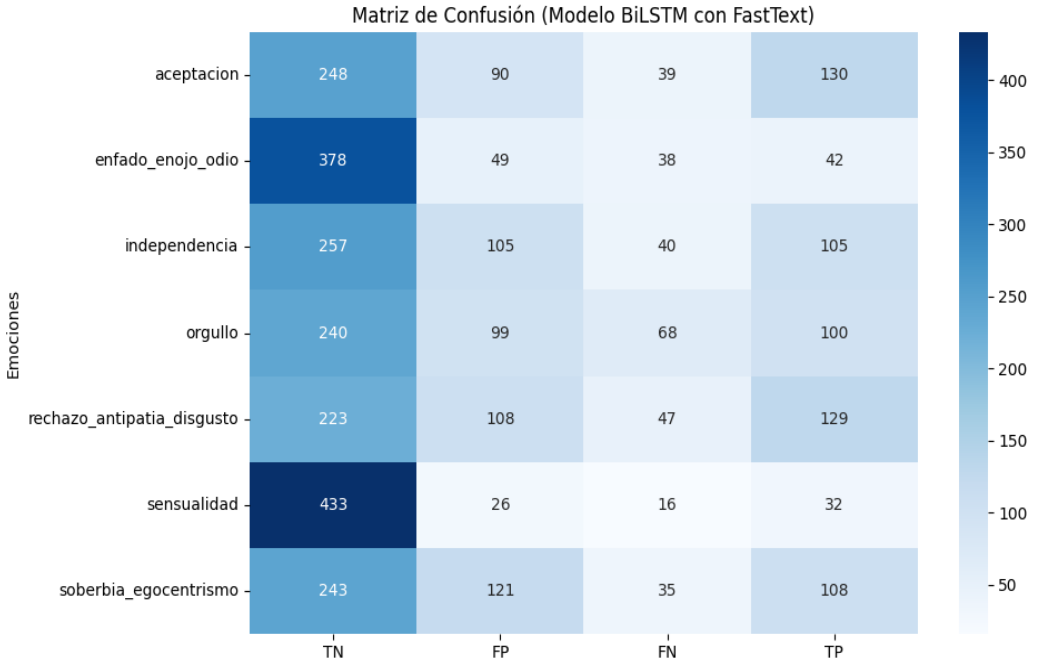


Figura 14: Matriz de confusión del modelo BiLSTM con FastText: Elaboración propia

Esta matriz de confusión corresponde al modelo BiLSTM con la técnica de Word Embedding FastText. Se observa un buen desempeño en la clasificación de emociones como aceptación, independencia y rechazo_antipatía_disgusto, reflejando un número considerable de verdaderos positivos. No obstante, también se evidencian falsos positivos y falsos negativos, en particular se detectan patrones de confusión entre “orgullo” y “soberbia_egocentrismo”, lo que indica la dificultad del modelo para diferenciar entre emociones con significados cercanos. En general, este modelo fue el que obtuvo mejores resultados entre todos los modelos evaluados.

CONCLUSIONES

- Mediante el estudio de las técnicas de Word Embedding, se comprendió que Word2Vec y FastText desempeñan un papel fundamental en el Procesamiento del lenguaje natural, ya que permiten representar las palabras en un espacio semántico numérico que captura relaciones contextuales. Este tipo de representaciones facilita que los modelos computacionales mejoren en la interpretación del contenido del texto, siendo útil para la tarea de clasificación de emociones. Además, se identificó que FastText presenta ventaja frente a Word2Vec, al incorporar subpalabras durante su entrenamiento, lo que permite manejar mejor la complejidad del idioma español.

- En las pruebas experimentales, demostró que las técnicas FastText y Word2Vec, utilizando representaciones secuenciales en lugar de promedios, mejoran la capacidad de los modelos para captar el significado y las emociones presentes en los textos, especialmente cuando hay varias emociones a la vez. Además, se observó que los modelos de Deep Learning superaron en rendimiento a los algoritmos tradicionales de Machine Learning debido a su habilidad para procesar secuencias y aprender dependencias contextuales entre palabras, es decir, gracias a su capacidad de manejar el orden y la relación entre las palabras.

- El desempeño de los modelos no solo depende de su arquitectura, sino también de cómo se procesa el texto, en el cual influyen muchos factores como el preprocesamiento, el balance de clases, la longitud de las secuencias, el tamaño del conjunto de datos y la eliminación de stopwords. Cada uno de estos aspectos resulta fundamental, especialmente en el idioma español por su complejidad lingüística, debido a que una misma palabra puede tener varios significados, y donde hay mucha variedad en la forma que se expresa una emoción. En conclusión, una preparación adecuada y

adaptada a las características del idioma, mejora el rendimiento de modelo en la clasificación de las emociones.

- Cuando se trata de clasificación multietiqueta, la selección adecuada de las métricas de avaluación es importante. En este estudio, se descartó como métrica principal la exactitud (accuracy), ya que un modelo puede obtener resultados altos prediciendo solo las emociones mayoritarias, sin identificar correctamente las menos frecuentes, lo cual puede ser engañoso. Por ello, se priorizaron métricas como F1-score y el área bajo la curva (AUC), ya que permiten evaluar tanto la capacidad del modelo para detector múltiples emociones, como su habilidad para diferenciar entre etiquetas positivas y negativas en diferentes umbrales de probabilidad, esto proporciona una visión más realista del desempeño del modelo.

- En los resultados obtenidos, el modelo BiLSTM combinado con la técnica FastText alcanzó un F1-score del 59%, lo que indica que logró un equilibrio aceptable entre precisión y sensibilidad (Real) al predecir emociones múltiples. Además, obtuvo un AUC (Área bajo la curva) del 80%, que refleja una buena capacidad para distinguir correctamente entre las clases emocionales. Estas métricas son equilibradas, considerando la complejidad del idioma español y el tamaño reducido del conjunto de datos.

RECOMENDACIONES

- Uno de los factores que influyen en el desempeño de los modelos de clasificación es la cantidad, calidad, y diversidad del conjunto de datos. En este estudio se trabajó con un dataset pequeño, lo cual limito el aprendizaje y la capacidad para captar patrones complejos de emociones. Por ello, se recomienda trabajar con un volumen de datos más amplio y representativo, que se ajuste a las necesidades del modelo, lo necesario para que tenga un mejor rendimiento.

- En el presente estudio se utilizaron únicamente las técnicas de Word2Vec y FastText para representar semánticamente los textos. Sin embargo, para trabajos futuros se recomienda considerar la técnica Glove, además que también existen técnicas más modernas y potentes, como los basados en arquitecturas de tipo Transformer, tales como BERT, RoBERTa o su versión entrenada para el idioma español, BETO, donde no se descarta que podrían proporcionar mejores resultados.

- Los modelos utilizados en machine Learning fueron Naïve Bayes Gaussiano, SVM y Regresión Logística, los cuales son conocidos por ser algoritmos tradicionales. Sin embargo, hay variedad de métodos más avanzados y con mejores características, como XGBoost, LightGBM y CatBoost. Se recomienda considerar estos enfoques, seleccionando el modelo más adecuado en función al conjunto de datos y el contexto de la investigación.

- En la etapa del preprocesamiento, es importante fortalecer el proceso de limpieza y preparación del texto, adaptándolo a las particularidades del idioma español, tomando en consideracion que hay herramientas como lematizadores, correctores ortográficos y diccionarios de sinónimos y de emojis. Además, se recomienda aplicar la eliminación de stopwords de manera selectiva y con criterio, por motivo que algunas palabras pueden

aportar información relevante en el contexto emocional, Por lo tanto, se sugiere utilizar listas personalizadas de stopwords, como en esta investigación ampliándola

- Se recomienda aplicar técnicas de optimización de hiperparámetros durante los procesos de entrenamiento, con el fin de encontrar las combinaciones más adecuadas de configuraciones. Estrategias como búsqueda en cuadricula (Grid Search), búsqueda aleatoria (Random Search) u optimización bayesiana (Bayesian Optimization) pueden resultar útiles para ajustar parámetros clave. Sin embargo, es importante considerar que implica un mayor tiempo de ejecución y de recursos computacionales.

REFERENCIAS

[1] «Aprendizaje automático aplicado al análisis de sentimientos». Accedido: 24 de marzo de 2025. [En línea]. Disponible en: https://portal.amelica.org/ameli/journal/339/3391369008/html/#redalyc_3391369008_ref1

[2] D. Boullier y A. Lohard, «Opinion mining et Sentiment analysis», *Prensa OpenEdition*, mar. 2012, doi: https://doi.org/10.4000/books.oep.198.

[3] Yoav Goldber, «A Primer on Neural Network Models for Natural Language Processing», *Journal of Artificial Intelligence*, noviembre de 2016. Accedido: 24 de marzo de 2025. [En línea]. Disponible en: https://www.jair.org/index.php/jair/article/view/11030/26198

[4] J. P. Tessore, «Modelado e implementación de algoritmos inteligentes de análisis de opinión», Universidad Nacional de La Plata, 2023. Accedido: 24 de marzo de 2025. [En línea]. Disponible en: https://sedici.unlp.edu.ar/bitstream/handle/10915/176789/Documento_completo.pdf-PDFA.pdf?sequence=1&isAllowed=y

[5] D. Fausto y A. Mamani, «DEEP NEURAL NETWORK APPROACHES FOR SPANISH SENTIMENT ANALYSIS OF SHORT TEXTS», UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA, AREQUIPA-PERÚ, 2019. Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://repositorio.unsa.edu.pe/server/api/core/bitstreams/0d820bdb-f9fc-42de-a830-88dc03b8da23/content

[6] J. Scotto, «Análisis de sentimientos de opiniones en redes sociales usando técnicas de procesamiento del lenguaje natural», Universidad de Belgrano, Buenos Aires - Argentina, 2021. Accedido: 23 de mayo de 2025. [En línea]. Disponible en: http://repositorio.ub.edu.ar/bitstream/handle/123456789/9534/Scotto.pdf?sequence=1&isAllowed=y

[7] J. Herrero Lanos, «Análisis de sentimientos en Twitter mediante técnicas de Deep Learning», Trabajo Fin De Grado, Universidad de Valladolid, 2022. Accedido: 24 de marzo de 2025. [En línea]. Disponible en: https://uvadoc.uva.es/bitstream/handle/10324/57316/TFG-G5815.pdf?sequence=1&isAllowed=y

[8] S. Palmero Muñoz, «Trabajo fin de grado Aprendizaje automático aplicado a Sentiment Analysis en castellano», TRABAJO FIN DE GRADO, UNIVERSIDAD AUTÓNOMA DE MADRID, 2019. Accedido: 24 de marzo de 2025. [En línea]. Disponible en: https://repositorio.uam.es/bitstream/handle/10486/688906/palmero_mu%C3%B1oz_santiago_tfg.pdf?sequence=1&isAllowed=y

[9] D. García Soriano, «Análisis de emociones en textos escritos mediante técnicas de aprendizaje automático», Trabajo Fin de Máster, Universidad de Sevilla, Sevilla, 2021. Accedido: 4 de julio de 2024. [En línea]. Disponible en: https://idus.us.es/bitstream/handle/11441/115423/TFM-1999-GARCIA%20SORIANO.pdf?sequence=1&isAllowed=y

[10] J. D. Granados Figueroa, «Aplicación de técnicas de Machine Learning para hacer análisis de polaridad de sentimientos en texto para detectar tendencias de opinión en plataformas online», Proyecto de grado, UNIVERSIDAD SANTO TOMÁS, BOGOTA, 2020. Accedido: 24 de marzo de 2025. [En línea]. Disponible en: https://biblus.us.es/bibing/proyectos/abreproy/71999/fichero/TFM-1999+GARC%C3%8DA+SORIANO%2C+DESIREE.pdf

[11] G. Lasso Mendoza, «Plan-de-Creación-de-Oportunidades-2021-2025», Quito, Ecuador, 2021. Accedido: 24 de marzo de 2025. [En línea]. Disponible en: https://www.planificacion.gob.ec/wp-content/uploads/2021/09/Plan-de-Creacio%CC%81n-de-Oportunidades-2021-2025-Aprobado.pdf

[12] R. Hernández Sampieri, C. Fernández Collado, y M. del P. Baptista Lucio, *Metodologia de la investigación*, Sexta edición. 2014. Accedido: 25 de

marzo de 2025. [En línea]. Disponible en: https://esup.edu.pe/wpcontent/uploads/2020/12/2.%20Hernandez,%20Fernandez%20y%20Baptista-Metodolog%C3%ADa%20Investigacion%20Cientifica%206ta%20ed.pdf

[13] Beltrán Néstor Camilo y E. C. Rodríguez, «Procesamiento del lenguaje natural (PLN) - GPT-3, y su aplicación en la Ingeniería de Software», *Academia TIA*, Bogotá-Colombia, 2021.

[14] A. Troncoso Lora, «INTELIGENCIA ARTIFICIAL: PASADO, PRESENTE Y FUTURO», *Encuentros Multidisciplinares*, 2022. Accedido: 30 de marzo de 2025. [En línea]. Disponible en: http://www.encuentros-multidisciplinares.org/revista-70/alicia-troncoso..pdf

[15] S. Acedo Pérez, «APLICACIÓN PRÁCTICA DE LA INTELIGENCIA ARTIFICIAL EN LA GESTIÓN EMPRESARIAL», TRABAJO FIN DE GRADO, Madrid, 2023. Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://burjcdigital.urjc.es/bitstream/handle/10115/24590/2023-24-FCEE-N-2246-2246042-s.acedo.2019-MEMORIA.pdf?sequence=-1&isAllowed=y

[16] A. Castro Solano, M. L. Lupano Perugini, y G. De La Iglesia, *LA PSICOLOGÍA DE INTERNET Explorando la mente digital*. Paidós, 2024. Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://static0planetadelibroscommx.cdnstatics.com/libros_contenido_extra/58/57814_TPC_La%20psicologia%20de%20internet.pdf

[17] C. E. Litardo Caicedo, «Ley Orgánica de Protección de Datos Personales», Quito,Ecuador, abr. 2021.

[18] Reglamento General de Protección de Datos (GDPR), «Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo», Diario Oficial de la Unión Europea. Accedido: 23 de mayo de 2025. [En línea]. Disponible en: https://eur-lex.europa.eu/eli/reg/2016/679/oj#d1e2016-1-1

[19] Organisation for Economic Co-operation and Development (OECD), «Recommandation du Conseil sur l'intelligence artificielle». Accedido: 23 de mayo de 2025. [En línea]. Disponible en: https://legalinstruments.oecd.org/fr/instruments/oecd-legal-0449

[20] Comisión Europea (European Commission), «Regulatory framework on artificial intelligence». Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://digital-strategy.ec.europa.eu/es/policies/regulatory-framework-ai

[21] European Commission, «Recomendación sobre la ética de la inteligencia artificial | UNESCO». Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://www.unesco.org/es/articles/recomendacion-sobre-la-etica-de-la-inteligencia-artificial

[22] J. Carlos y S. Sande, «ANÁLISIS DE SENTIMIENTOS EN TWITTER», 2018, Accedido: 2 de julio de 2024. [En línea]. Disponible en: https://openaccess.uoc.edu/bitstream/10609/81435/6/jsobrinosTFM0618memoria.pdf

[23] Alejandro Raiter y Virginia Jaichenco, «Psicolingüística - Manual». Accedido: 15 de junio de 2025. [En línea]. Disponible en: https://www.researchgate.net/publication/308735345_Psicolinguistica_-_Manual

[24] M. H. González García, N. Y. Zambrano Villamil, y C. A. Molina Rodríguez, «"La macroestructura semántica como estrategia didáctica en la comprensión de textos expositivos"», *rev.educien*, vol. Núm. 22, 22 de septiembre de 2019. doi: https://doi.org/10.19053/0120-7105.eyc.2019.22.e10055.

[25] P. A. Pauli, «Análisis de sentimiento», 2019.

[26] Enciclopedia significados, «"Sentimientos"». Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://www.significados.com/sentimiento/

[27] M. Miranda-Leon y R. Toala-Dueñas, «Detección de emociones en discursos utilizando machine learning», *593 Digital Publisher CEIT*, vol. 9, n.º 4, pp. 72-101, jul. 2024, doi: 10.33386/593dp.2024.4.2367.

[28] Google, «Google Colaboratory». Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://colab.research.google.com

[29] Python Software Foundation, «Python Official Website». Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://www.python.org/

[30] H. M. Lell y R. Escudero Giménez, «Los conjuntos de datos de la Corte Suprema de Justicia de la Nación. A propósito de las estadísticas y los datasets como forma de justicia abierta», Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://repositorio.uca.edu.ar/bitstream/123456789/18717/1/conjuntos-datos-corte.pdf

[31] hilarispublisher, «Modelo computacional». Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://spanish.hilarispublisher.com/scholarly/computational-model-journals-articles-ppts-list-376.html

[32] J. M. De La Serna, *Ciberpsicología:Relacion mente e internet*, Primera edición. 2015. Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://books.google.es/books?hl=es&lr=&id=qLqYDwAAQBAJ&oi=fnd&pg=PT15&dq=Ciberpsicolog%C3%ADa&ots=fa6I--x4Oc&sig=FInaGwR4WslFaJtqs2hTC_ix1vI#v=onepage&q=Ciberpsicolog%C3%ADa&f=false

[33] L. Español y A. T. Orellana, «EVALUACIÓN DE TÉCNICAS DE PROCESAMIENTO DE LENGUAJE NATURAL PARA TEXTOS CORTOS EN», 2018. Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://dspace.uazuay.edu.ec/handle/datos/8501

[34] A. Navarro y C. Dirigido, «T ´ ECNICAS DE MACHINE LEARNING PARA PROCESAMIENTO DEL LENGUAJE NATURAL». Accedido: 30 de marzo de 2025. [En línea]. Disponible en:

https://digibuo.uniovi.es/dspace/bitstream/handle/10651/63977/TFG_AlejandraNavarroCastillo.pdf?sequence=4&isAllowed=y

[35] Real Academia Española, «“Inteligencia” Diccionario de la lengua española». Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://dle.rae.es/inteligencia#2DxmhCT

[36] J. A. Muñoz, «Vista de Entendiendo el poder de la Inteligencia Artificial», *TEPEXI Boletín Científico de la Escuela Superior Tepeji del Río*, vol. Vol. 10, 2023. doi: ttps://doi.org/10.29057/estr.v10i20.10807.

[37] A. Taboada Villamarín, «Vista de Big data en ciencias sociales. Una introducción a la automatización de análisis de datos de texto mediante procesamiento de lenguaje natural y aprendizaje automático», *Revista CENTRA de Ciencias Sociales*, vol. 3, España, 2024. doi: https://doi.org/10.54790/rccs.51.

[38] E. M. Rojas, «Machine Learning: análisis de lenguajes de programación y herramientas para desarrollo», Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://www.proquest.com/openview/c7e24c997199215aa26a39107dd2fe98/1?pq-origsite=gscholar&cbl=1006393

[39] M. E. Larrondo, N. M. Grandi, M. E. Larrondo, y N. M. Grandi, «Inteligencia Artificial, algoritmos y libertad de expresión», *Universitas-XXI, Revista de Ciencias Sociales y Humanas*, vol. 34, n.º 34, pp. 177-194, feb. 2021, doi: 10.17163/UNI.N34.2021.08.

[40] J. I. Ramos Alvarez, «Análisis de evaluación automática de documentos académicos a través de la aplicación de técnicas de incrustación (Word Embedding)». Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://dspace.utpl.edu.ec/visorHub/?handle=123456789_52178

[41] Chanika Ruchini, «Introduction to Word Embeddings.», Análisis Vidhya. Accedido: 31 de mayo de 2025. [En línea]. Disponible en: https://medium.com/analytics-vidhya/introduction-to-word-embeddings-c2ba135dce2f

[42] D. De, J. Ramos, C. Directores, : Alveiro, y R. Gomez, «CLASIFICACIÓN ÚNICA DE OCUPACIONES PARA COLOMBIA UTILIZANDO PROCESAMIENTO DEL LENGUAJE NATURAL». Accedido: 31 de marzo de 2025. [En línea]. Disponible en: https://repository.unab.edu.co/bitstream/handle/20.500.12749/22679/2023_Tesis_Deimer_Ramos.pdf?sequence=1&isAllowed=y

[43] GeeksforGeeks, «Word Embeddings Using FastText», geeksforgeeks. Accedido: 31 de marzo de 2025. [En línea]. Disponible en: https://www.geeksforgeeks.org/word-embeddings-using-fasttext/

[44] L. P. Rouhiainen, *INTELIGENCIA ARTIFICIAL 101 COSAS QUE DEBES SABER HOY SOBRE NUESTRO FUTURO INTELIGENCIA ARTIFICIAL*. España: Planeta, 2018. [En línea]. Disponible en: www.planetadelibros.com

[45] R. Vargas Lopez, P. Wong Ramírez, A. Zamora Esquivel, y W. Aguilar Calvo, «Máquina de Soporte Vectorial y Multiplicadores de Lagrange derivados respecto a vectores de entrenamiento». Accedido: 31 de mayo de 2025. [En línea]. Disponible en: https://www.researchgate.net/publication/351900988_Maquina_de_Soporte_Vectorial_y_Multiplicadores_de_Lagrange_derivados_respecto_a_vectores_de_entrenamiento

[46] Sonia Jessica, «How Does Logistic Regression Work?» Accedido: 31 de mayo de 2025. [En línea]. Disponible en: https://www.kdnuggets.com/2022/07/logistic-regression-work.html

[47] IBM, «¿Qué son los clasificadores Naive Bayes?» Accedido: 31 de mayo de 2025. [En línea]. Disponible en: https://www.ibm.com/es-es/think/topics/naive-bayes

[48] SAS Institute, «Deep Learning: Qué es y por qué es importante». Accedido: 31 de marzo de 2025. [En línea]. Disponible en: https://www.sas.com/es_es/insights/analytics/deep-learning.html

[49] X. Basogain Olabe, *REDES NEURONALES ARTIFICIALES Y SUS APLICACIONES* .

Accedido: 1 de junio de 2025. [En línea]. Disponible en: https://ocw.ehu.eus/pluginfile.php/40137/mod_resource/content/1/redes_neuro/contenidos/pdf/libro-del-curso.pdf

[50] Y. Gutiérrez, «Introducción a las Redes Neuronales Recurrentes (RNN)», imaster.academy. Accedido: 9 de junio de 2025. [En línea]. Disponible en: https://imaster.academy/contenidos-tematicos/talentotech/TalentoTech/M3unidades/Inteligencia%20artificial/Innovador/Unidad1/assets/files/L2_1_Introduccion-aRedesNeuronalesRecurrentesRNN.pdf

[51] F. Liu, K. Hou, y Y. Dong, «Deep parallel contextual analysis framework based emotion prediction in community wellness communications on social media», *Heliyon*, vol. 10, n.º 11, p. e31626, jun. 2024, doi: 10.1016/J.HELIYON.2024.E31626.

[52] JUAN DE LUCIO, WILDER BERMÚDEZ, y NAJAT BAZAH, «Comparativa de arquitecturas de redes neuronales para modelización de variables económicas».

[53] mathworks, «Introducción a la memoria a corto-largo plazo (LSTM)», MathWorks. Accedido: 1 de junio de 2025. [En línea]. Disponible en: https://la.mathworks.com/discovery/lstm.html

[54] E. L. Merchán Pérez, «APLICACIÓN DE MODELOS TRANSFORMERS PARA CLASIFICAR TEXTOS EN IDIOMA ESPAÑOL », UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA, 2024. Accedido: 1 de junio de 2025. [En línea]. Disponible en: https://repositorio.upse.edu.ec/bitstream/46000/11875/1/UPSE-TTI-2024-0035.pdf

[55] S. B. Alemán Viteri, «Análisis de sentimientos para Twitter con Vader y TextBlob», *Revista ODIGOS*, vol. Vol. 2, QUITO-ECUADOR, 10 de octubre de 2021. doi: https://doi.org/10.35290/ro.v2n3.2021.494.

[56] Matplotlib Developers, «Using Matplotlib», Matplotlib. Accedido: 1 de junio de 2025. [En línea]. Disponible en: https://matplotlib.org/stable/users/index.html

[57] M. Waskom, «seaborn: statistical data visualization», *J Open Source Softw*, vol. 6, n.º 60, p. 3021, abr. 2021, doi: 10.21105/JOSS.03021.

[58] Streamlit, «Streamlit documentation», Streamlit Docs. Accedido: 1 de junio de 2025. [En línea]. Disponible en: https://docs.streamlit.io/

[59] «Análisis de emociones en texto – Procesamiento del Lenguaje Natural». Accedido: 23 de mayo de 2025. [En línea]. Disponible en: https://prompt.uno/procesamiento-del-lenguaje-natural/analisis-de-emociones-en-texto/

[60] iArtificial, «Del texto al entendimiento: IA y el poder del procesamiento del lenguaje natural», iArtificial.blog. Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://iartificial.blog/aprendizaje/del-texto-al-entendimiento-ia-y-el-poder-del-procesamiento-del-lenguaje-natural/

[61] Toolify.ia, «Desafíos y aplicaciones del Procesamiento del Lenguaje Natural». Accedido: 30 de marzo de 2025. [En línea]. Disponible en: https://www.toolify.ai/es/ai-news-es/desafos-y-aplicaciones-del-procesamiento-del-lenguaje-natural-1899980

[62] «Aproximación psicolingüística al estudio de la personalidad en español: una propuesta taxonómica», Accedido: 11 de junio de 2025. [En línea]. Disponible en: https://www.researchgate.net/publication/28057339_Aproximacion_psicolinguistica_al_estudio_de_la_personalidad_en_espanol_una_propuesta_taxonomica

ANEXOS

Anexo 1: Librerías utilizadas para la implementación de la interfaz

Para la implementación de la interfaz se utilizó principalmente la librería de Streamlit para la construcción de la interfaz web. Además, se emplearon otras librerías para el procesamiento del texto y manejo del modelo, incluyendo NLTK, TensorFlow, Gensim, SpaCy, Sciklearn. Como parte del procesamiento de lenguaje natural, se descargó el modelo en español es_core_news_md de SpaCy para apoyar el análisis y la limpieza del texto.

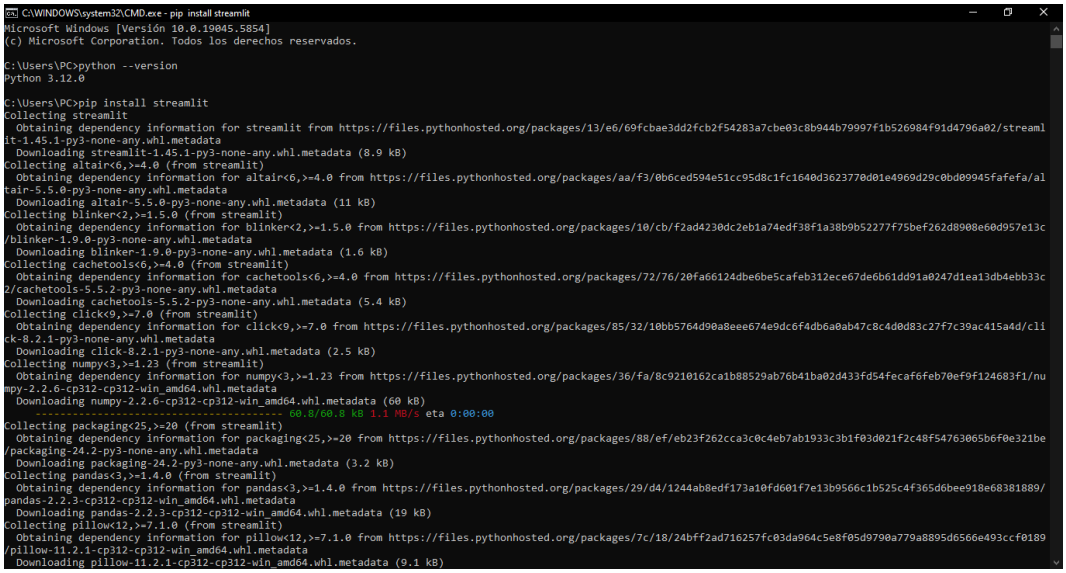


Figura 15: Instalación de la librería Streamlit desde la línea de comandos

Anexo 2: Código de la interfaz

A continuación, se presenta el código completo de la interfaz desarrollada en Streamlit, la cual permite al usuario ingresar un comentario en español y obtener como salida las emociones detectadas por el modelo BiLSTM, entrenado previamente con vectores FastText.

Esta interfaz realiza el preprocesamiento del texto ingresado, convierte las palabras en vectores utilizando FastText, y luego utiliza el modelo para predecir hasta tres emociones presentes en el comentario.

```
import streamlit as st
import unicodedata
import numpy as np
import pickle
import nltk
from gensim.models.fasttext import FastText as FT_gensim
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing.sequence import pad_sequences
from nltk.corpus import stopwords

# Descargar stopwords si es la primera vez
nltk.download('stopwords')

# Cargar recursos
fasttext_model = FT_gensim.load("fasttext_model.model")
model_biLSTM = load_model("model_biLSTM.h5")

# Cargar MultiLabelBinarizer
with open("mlb_emociones.pkl", "rb") as f:
    mlb = pickle.load(f)

stop_words = set(stopwords.words('spanish'))
unique_classes = mlb.classes_

# Funciones auxiliares
def normalizar_emociones(emociones):
    normalized = []
    for em in emociones:
        em = unicodedata.normalize('NFKD', em).encode('ASCII', 'ignore').decode('utf-8')
        normalized.append(em.lower().strip())
    return normalized


def preprocesar_texto(texto):
...     texto_tokenizado = texto.lower().split()
...     return [palabra for palabra in texto_tokenizado if palabra not in stop_words]
...
... def obtener_embedding_palabra(texto_tokenizado):
...     embeddings = []
...     for palabra in texto_tokenizado:
...         try:
...             emb = fasttext_model.wv[palabra]
...             embeddings.append(emb)
...         except KeyError:
...             embeddings.append(np.zeros(200))
...     return np.array(embeddings)
...
... def clasificar_emociones(texto):
...     tokens = preprocesar_texto(texto)
...     secuencia = obtener_embedding_palabra(tokens)
...     secuencia_pad = pad_sequences([secuencia], maxlen=40, padding='post', dtype='float32')
...     pred = model_biLSTM.predict(secuencia_pad)
...     pred_bin = (pred > 0.4).astype(int)
...     emociones = [unique_classes[i] for i, val in enumerate(pred_bin[0]) if val == 1]
...     if not emociones:
...         return ["No se detectaron emociones."]
...     return emociones[:3]
...
... # Interfaz Streamlit
... st.set_page_config(page_title="Clasificación de Emociones", layout="centered")
... st.title("🧠 Clasificador de Emociones en Comentarios")
...
... texto_usuario = st.text_area("Escribe un comentario para analizar:", height=150)
...
... if st.button("🔍 Clasificar Emoción"):
...     if texto_usuario.strip() == "":
...         st.warning("Por favor, escribe un comentario.")
...     else:
...         emociones = clasificar_emociones(texto_usuario)
...         st.success("🎯 Emociones detectadas:")
```

Figura 16: Código completo de la interfaz Streamlit para la clasificación de emociones

Anexo 3: Interfaz desarrollada en Streamlit

A continuación, se presentan capturas de pantalla de la interfaz desarrollada con Streamlit para la clasificación multietiqueta de emociones en comentarios en español.

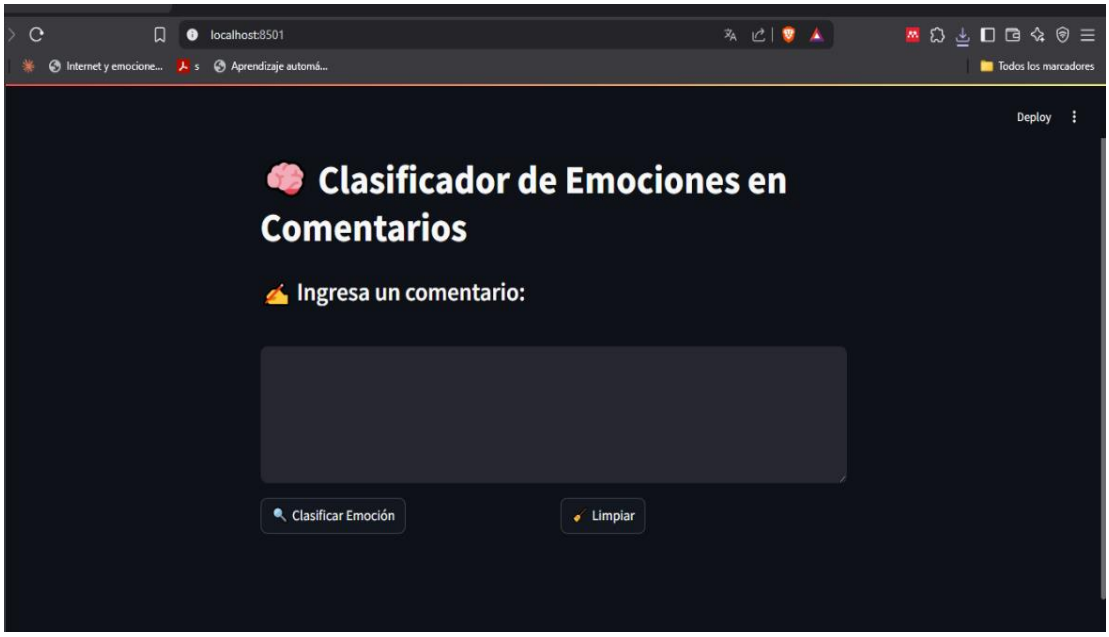


Figura 17: Interfaz del clasificador de emociones en Streamlit

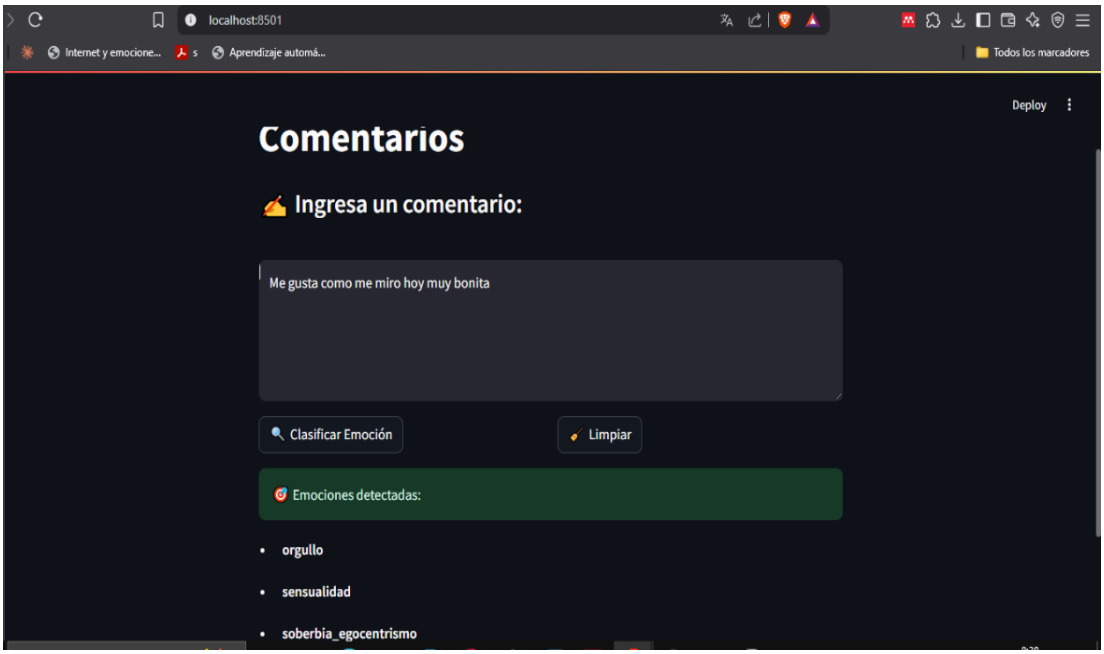


Figura 18: Emociones detectadas tras el análisis del comentario ingresado.