OPTIMIZACIÓN DE TÉCNICAS NLP PARA EVALUACIÓN DE SENTIMIENTOS EN TEXTOS LARGOS

Joseph Yäel Mendieta Avila (CÓDIGO: 67001618)

Universidad Católica de Colombia
Facultad de Ingeniería
Análisis y Diseño de Algoritmos
Ingeniería de Software
Bogotá D.C.
2024

Optimización de Técnicas NPL para Evaluación de Sentimientos en Textos Largos

Joseph Yäel Mendieta Avila (CÓDIGO: 67001618)

Optimización de Técnicas NPL para Evaluación de Sentimientos en Textos Largos

Yenny Paola Sierra Bonilla
Docente

Oscar Fernando Penagos Espinel
Docente

Universidad Católica de Colombia
Facultad de Ingeniería
Análisis y Diseño de Algoritmos
Ingeniería de Software

Bogotá D.C.

CONTENIDO

GLOSARIO

MODELO MULTISWITCHNET: se refiere a un sistema de red que utiliza múltiples switches para gestionar la comunicación entre dispositivos en una red de computadoras

PIPELINE: (o "tubería" en español) es un concepto utilizado en diferentes disciplinas de la computación e ingeniería para describir un proceso en el cual una serie de tareas se ejecutan en etapas secuenciales, de manera que la salida de una etapa se convierte en la entrada de la siguiente.

EMBEDDING: Es una representación numérica de datos en un espacio vectorial de menor dimensión que captura sus características relevantes.

TRANSFORMACIONES SECUENCIALES: Procesan datos de manera ordenada, transformando cada elemento de la secuencia en función de los elementos anteriores, lo cual es útil para capturar dependencias en secuencias de datos, como en el procesamiento de texto.

PROCESAMIENTO PARALELO: Ejecuta múltiples operaciones simultáneamente para acelerar el procesamiento de datos grandes o complejos, aprovechando varias unidades de procesamiento al mismo tiempo. En modelos de lenguaje, permite que el modelo analice relaciones entre palabras sin seguir un orden secuencial estricto.

RESUMEN

El proyecto "Optimización de Técnicas NLP para Evaluación de Sentimientos en Textos Largos" desarrolla un algoritmo que analiza el sentimiento en textos extensos, evalúa su adecuación para redes sociales y mide la aceptación de temas en plataformas como Reddit. Integrando conocimientos de Análisis y Diseño de Algoritmos e Ingeniería de Software, este proyecto utiliza herramientas avanzadas de procesamiento de lenguaje natural (NLP), aprendizaje automático y se apoya en una arquitectura serverless para ofrecer escalabilidad y eficiencia en el análisis.

Los objetivos específicos incluyen mejorar el análisis de sentimientos, establecer criterios de relevancia para redes sociales, desarrollar una API escalable usando FastAPI y analizar interacciones en Reddit. La metodología sigue el ciclo de vida de desarrollo de software desde el análisis de requisitos hasta el despliegue y mantenimiento. En la fase de implementación, se emplean modelos preentrenados de Hugging Face Transformers y técnicas de preprocesamiento para optimizar la precisión y eficiencia del análisis, mientras que se implementa un historial de análisis para evaluar cambios en el sentimiento a lo largo del tiempo.

Con un presupuesto de 5,320,000 COP destinado a recursos humanos y técnicos, se espera que el algoritmo optimice el impacto de publicaciones en redes sociales, proporcionando una herramienta práctica y escalable para la evaluación y mejora de contenido digital.

PALABRAS CLAVE: Análisis de sentimientos, procesamiento de lenguaje natural, redes sociales, optimización de publicaciones, aprendizaje automático.

1. INTRODUCCIÓN

El análisis de sentimientos es crucial en la comprensión de las opiniones expresadas en redes sociales y otras plataformas digitales. Dada la gran cantidad de texto generado a diario, especialmente en publicaciones extensas, contar con algoritmos efectivos para captar emociones y opiniones resulta esencial. Para abordar estos desafíos, se requiere el uso de modelos avanzados como BERT (Bidirectional Encoder Representations from Transformers), un modelo de lenguaje basado en la arquitectura Transformer, que ha demostrado una alta precisión en el análisis contextual profundo de textos. Su enfoque bidireccional permite analizar el contexto de cada palabra, lo que resulta esencial para captar matices en textos largos, donde las emociones pueden variar.

El proyecto denominado “Optimización de Técnicas NLP para Evaluación de Sentimientos en Textos Largos” combina conocimientos de *Análisis y Diseño de Algoritmos* e *Ingeniería de Software*. Estas disciplinas son fundamentales para desarrollar software eficiente y resolver los desafíos que implica analizar sentimientos en textos extensos. En este proyecto, se integra nlptown/bert-base-multilingual-uncased-sentiment, un modelo preentrenado basado en BERT, el cual clasifica los textos en una escala de 1 a 5 estrellas, proporcionando un análisis detallado de sentimientos en múltiples idiomas y facilitando la evaluación de la idoneidad del contenido en redes sociales.

Mediante el uso de Python y herramientas como Hugging Face Transformers, el proyecto desarrolla un algoritmo que no solo evalúa el sentimiento de un texto y califica su relevancia para las redes sociales, sino que también optimiza su adecuación en función del impacto potencial en plataformas como Reddit. Así, este trabajo se orienta a crear una herramienta práctica que permita comprender qué contenido genera mayor impacto, optimizando las publicaciones y mejorando la efectividad de las estrategias en redes sociales.

2. OBJETIVOS

2.1 OBJETIVO GENERAL

Desarrollar un algoritmo que permita analizar sentimientos en textos largos, evaluar su adecuación para redes sociales y analizar la recepción de un tema en Reddit.

2.2 OBJETIVOS ESPECÍFICOS

1. Mejorar el análisis de sentimientos:
    - Utilizar herramientas avanzadas como Hugging Face Transformers para evaluar el sentimiento en textos extensos, aplicando modelos basados en BERT para una comprensión profunda y contextual de las emociones expresadas.
2. Evaluar la adecuación para redes sociales:
    - Definir los criterios para determinar si un texto es adecuado para redes sociales, considerando aspectos como el tono, la relevancia para el público y el impacto potencial del contenido en plataformas como Reddit.
3. Desarrollar una API:
    - Crear una API usando FastAPI que facilite la integración del algoritmo de análisis de sentimientos y evaluación de adecuación con otras aplicaciones y plataformas.
4. Medir la recepción en redes sociales:
    - Implementar métodos para analizar cómo se recibe un tema en Reddit, observando las interacciones y el sentimiento reflejado en publicaciones y comentarios para mejorar la estrategia de contenido.
5. Probar y ajustar el sistema:
    - Realizar pruebas exhaustivas para asegurar el correcto funcionamiento del sistema, ajustando el algoritmo según los resultados obtenidos y las métricas de desempeño.
6. Implementar y monitorear el sistema:
    - Lanzar el sistema en un entorno de prueba, supervisar su rendimiento en tiempo real y realizar ajustes finales basados en el feedback recibido para optimizar su efectividad.

3 PLANTEAMIENTO DEL PROBLEMA

3.1 DEFINICIÓN DEL PROBLEMA

En la era digital, las redes sociales y plataformas digitales generan una gran cantidad de contenido textual a diario, especialmente en formato de publicaciones largas. Comprender el sentimiento y la pertinencia de estos textos es crucial para optimizar su impacto. Sin embargo, analizar sentimientos en textos extensos presenta varios desafíos, ya que las emociones pueden variar a lo largo del contenido, y los algoritmos tradicionales suelen fallar al captar estos matices complejos. Además, evaluar si un texto es adecuado para redes sociales requiere criterios específicos que integren el tono y relevancia del mensaje. La falta de una solución que combine efectivamente el análisis de sentimientos con la evaluación de idoneidad para redes limita el potencial de las publicaciones en términos de alcance y eficacia.

3.2 JUSTIFICACIÓN

Resolver el problema de analizar sentimientos en textos largos y determinar su adecuación para redes sociales es esencial para mejorar la efectividad de las estrategias de contenido. La solución propuesta utiliza modelos avanzados de procesamiento de lenguaje natural (NLP), como nlptown/bert-base-multilingual-uncased-sentiment, basado en la arquitectura BERT. Este modelo permite realizar un análisis contextual profundo en múltiples idiomas y clasificar los textos en una escala de sentimiento. Además, se integran herramientas como Hugging Face Transformers y FastAPI, que permiten el desarrollo de una API eficiente y escalable, facilitando la implementación del algoritmo en diversas plataformas y aplicaciones.

4. MARCO TEÓRICO

El procesamiento de lenguaje natural (NLP) ha avanzado mucho en los últimos años, y se ha vuelto crucial para tareas como el análisis de sentimientos en redes sociales, la detección de ataques informáticos y la simulación de procesos en ingeniería. Gran parte de estos avances se deben a la integración de aprendizaje automático, lo que ha permitido mejorar la precisión y eficiencia al interpretar datos textuales y multimodales.

Por ejemplo, en el análisis de sentimientos, Prakash y Vijay (2024) desarrollaron un modelo llamado MultiSwitchNet que está diseñado para analizar textos en idiomas Dravidian mezclados con inglés (code-switching). Este modelo utiliza XLM-RoBERTa y añade una capa especial para tener en cuenta aspectos socio-pragmáticos, logrando un rendimiento superior en comparación con otros modelos. Esto demuestra que la inclusión de factores sociales y pragmáticos en el análisis de sentimientos puede mejorar la precisión al trabajar con textos complejos y multilingües.

Por otro lado, Liu et al. (2024) crearon un modelo llamado MAF para analizar comentarios en redes sociales que combinan datos de texto, imágenes y sonido. En lugar de usar una arquitectura Transformer tradicional, que puede ser compleja y consumir muchos recursos, optaron por una estructura más sencilla que aún así logra un rendimiento comparable con los modelos más avanzados. Este enfoque destaca la importancia de considerar todas las modalidades de datos para hacer un análisis de sentimientos más completo.

En el campo de la seguridad informática, Sreeram y Vuppala (2017) desarrollaron un modelo inspirado en algoritmos bioinspirados para detectar ataques DDoS en la capa de aplicación. Estos ataques son comunes y difíciles de detectar, por lo que las técnicas basadas en anomalías, como la que ellos proponen, son clave para una detección rápida y eficaz.

En cuanto a la ingeniería, Marques et al. (2024) revisaron el uso de aprendizaje automático en la modelización de láminas metálicas. La precisión en la simulación de estos procesos de fabricación depende mucho del modelo constitutivo que se utilice, y las técnicas de aprendizaje automático han permitido mejorar tanto la identificación de parámetros como la creación de modelos más precisos.

Por último, Weinreich y Buchgeher (2011) subrayan la importancia de una arquitectura de software bien diseñada durante todo el ciclo de vida del desarrollo de software. Su enfoque propone un modelo que integra actividades de diseño, implementación y análisis, garantizando la consistencia de la información y evitando redundancias a lo largo del desarrollo del software.

5. MATERIALES Y MÉTODOS

5.1 MATERIALES

Para el desarrollo de este proyecto se utilizaron diversos materiales de hardware, software y bases de datos. En cuanto a hardware, se empleó una computadora personal con procesador Intel i5 o superior, 8GB de RAM y al menos 256GB de almacenamiento, para implementar, probar y ejecutar el algoritmo de análisis de sentimientos en textos largos. El software utilizado incluyó Python, como lenguaje principal de programación, y la librería Hugging Face Transformers, que facilitó la integración del modelo nlptown/bert-base-multilingual-uncased-sentiment, basado en BERT, el cual permite realizar el análisis de sentimientos en múltiples idiomas. Además, se usó FastAPI para desarrollar una API que permite la interacción del algoritmo con otras aplicaciones de redes sociales, y Google Colab, una plataforma en la nube para entrenar y ajustar el modelo, aprovechando sus recursos computacionales. Para el control de versiones y colaboración en equipo se empleó GitHub.
Asimismo, se utilizó un conjunto de datos de reseñas multilingües, que contiene textos y comentarios de redes sociales en diferentes idiomas, necesario para entrenar y ajustar el modelo de análisis de sentimientos. Finalmente, se consultaron referencias teóricas y documentación relevante, incluyendo la documentación de los modelos de Hugging Face y guías de FastAPI, para asegurar una adecuada integración y funcionalidad del sistema. También se incluyeron estudios y artículos sobre análisis de sentimientos, procesamiento de lenguaje natural y arquitectura de software como apoyo teórico para el proyecto.

5.2 METODOLOGÍA

1. Análisis de Requisitos:

En esta primera fase, identificamos claramente el problema: necesitamos un algoritmo que analice sentimientos en textos largos y evalúe si son adecuados para redes sociales. Aquí se definen los requisitos del proyecto, tanto funcionales (como la precisión del análisis de sentimientos) como no funcionales (como la rapidez de procesamiento). Esta etapa también implica interactuar con las partes interesadas para asegurarnos de que el sistema cubra sus necesidades.

2. Diseño del Sistema:

Con los requisitos en mano, diseñamos la arquitectura del sistema. Esto incluye la selección de las herramientas y tecnologías más adecuadas, como modelos de Hugging Face Transformers para el análisis de sentimientos y FastAPI para crear la API. También se define cómo se integrarán estos componentes, y se planea cómo será el flujo de datos desde la entrada de texto hasta la evaluación final. En esta fase, también diseñamos la estructura del algoritmo de Machine Learning, incluyendo la selección de técnicas de preprocesamiento de texto, y el pipeline de entrenamiento y evaluación del modelo.

3. Implementación:

Aquí empieza la construcción del sistema, siguiendo el diseño definido. El desarrollo se lleva a cabo de manera iterativa, construyendo el algoritmo en etapas. Se usan plataformas como Google Colab para entrenar modelos, realizar pruebas y ajustes. En paralelo, se implementa la API que permitirá la interacción con el sistema. La implementación también incluye el manejo de la base de datos, la cual almacena las publicaciones procesadas y sus resultados.

4. Pruebas:

Cuando el sistema está implementado, es momento de probarlo. Se realizan pruebas unitarias para asegurarnos de que cada componente funcione correctamente. Luego, se llevan a cabo pruebas de integración para verificar que los componentes se comuniquen de manera efectiva. Además, se aplican pruebas con conjuntos de datos que no se usaron en el entrenamiento, evaluando la precisión y otras métricas del modelo. Esta etapa también incluye pruebas de rendimiento para asegurarnos de que el sistema puede manejar grandes volúmenes de datos y seguir siendo rápido y eficiente.

5. Despliegue:

En lugar de solo enfocarnos en llevar el sistema a producción, distribuimos el algoritmo en diferentes entornos de prueba. Se comparte con un grupo seleccionado de usuarios o colaboradores, quienes lo usan en situaciones reales y proporcionan retroalimentación. Esto permite ajustar detalles según el comportamiento observado en el mundo real, pero sin 10 exponer el sistema a todos los usuarios de inmediato. Además, se crea una documentación clara para facilitar futuras implementaciones y se establecen guías para un posible despliegue más amplio en una etapa posterior, asegurando que todos los aspectos estén listos cuando llegue el momento de expandir su uso.

6. Mantenimiento y Evolución:

Después del despliegue, el sistema entra en la fase de mantenimiento. Esto incluye corregir cualquier error que surja, realizar ajustes para mejorar el rendimiento y actualizar el sistema a medida que se obtienen más datos o cambian las necesidades del usuario. También se pueden integrar nuevas funcionalidades o mejorar las existentes para hacer el sistema más robusto y adaptable.

6 DESARROLLO DEL PROYECTO

Este proyecto emplea un modelo avanzado de procesamiento de lenguaje natural (NLP) basado en la arquitectura **BERT** (Bidirectional Encoder Representations from Transformers) para analizar sentimientos en textos extensos, clasificar su tono en una escala de 1 a 5 estrellas y evaluar su pertinencia para redes sociales. El modelo, nlptown/bert-base-multilingual-uncased-sentiment, se optimiza para múltiples idiomas, lo que permite un análisis de sentimiento preciso en diversos contextos culturales. Esta adaptación del análisis de sentimientos muestra cómo el aprendizaje automático y NLP han transformado múltiples áreas, desde la evaluación de opiniones en redes sociales hasta aplicaciones complejas en seguridad informática y simulaciones en ingeniería.

6.1 ANÁLISIS DEL DESARROLLO DEL PROYECTO

El desarrollo del proyecto explora a fondo la arquitectura de BERT, un modelo que revolucionó el procesamiento de lenguaje natural al introducir un enfoque bidireccional para la comprensión de palabras, lo cual es crucial en textos largos y complejos. Al basarse en una arquitectura **Transformer** (Vaswani et al., 2017), BERT emplea un mecanismo de autoatención para cada palabra en el texto, permitiendo que esta "preste atención" a otras palabras relevantes independientemente de su ubicación en la secuencia. Este mecanismo logra una alta precisión, especialmente en secuencias donde el contexto cambia o se vuelve complejo de interpretar.

Funcionamiento Interno del Modelo

1. **Inicialización del Clasificador**:
    - La línea de código clasificador = pipeline('sentiment-analysis', model='nlptown/bert-base-multilingual-uncased-sentiment') llama a un *pipeline* de Hugging Face que utiliza el modelo preentrenado nlptown/bert-base-multilingual-uncased-sentiment. Este pipeline crea un flujo automático desde la entrada de texto hasta la clasificación de sentimiento.
2. **Recepción y Preprocesamiento de Texto**:
    - Cuando un texto ingresa al modelo, el *pipeline* convierte las palabras en *tokens*, unidades de procesamiento que representan palabras o fragmentos de palabras en forma numérica. Este proceso asegura que las palabras sean compatibles con el modelo, sin importar el idioma.

3. **Proceso de Atención y Codificación**:
    - Los tokens pasan a través de 12 capas de autoatención en BERT-base, donde cada palabra "observa" otras palabras en la oración, incluso las lejanas. Esto se hace mediante matrices de atención cuadráticas O(N2·d)O(N^2 \cdot d)O(N2·d), donde NNN es la longitud de la secuencia y ddd la dimensión del embedding (768 en BERT-base). Este proceso permite que el modelo capture las relaciones contextuales y el tono general del texto.
4. **Algoritmo Utilizado**:
    - El modelo aplica un enfoque basado en **transformaciones secuenciales** y **procesamiento paralelo**, en lugar de algoritmos voraces o divide y vencerás, maximizando la comprensión bidireccional de BERT.
5. **Clasificación Final**:
    - Tras codificar el contexto completo, el modelo pasa las representaciones de cada token por una capa densa, que produce probabilidades asociadas a cada clase de sentimiento (de 1 a 5 estrellas). La clase con mayor probabilidad se elige la clasificación final, que representa el sentimiento predominante en el texto.

Limitaciones y Justificación de la Arquitectura Base

La arquitectura de BERT-base, con 12 capas, fue diseñada para encontrar un equilibrio entre profundidad y rendimiento. Las versiones más grandes, como BERT-large, aumentan la precisión pero también incrementan significativamente el tiempo de procesamiento, lo cual puede ser limitante en aplicaciones en tiempo real. La elección de BERT-base en este proyecto garantiza un análisis preciso sin comprometer la eficiencia, especialmente para textos de longitud media a larga.

Importancia de la Arquitectura de Software

Como destaca Weinreich y Buchgeher (2011), una arquitectura de software bien diseñada garantiza la escalabilidad y la adaptabilidad del sistema. Integrar BERT y FastAPI asegura que el modelo sea accesible, escalable y adaptable a futuras expansiones, mejorando la precisión del contenido en redes sociales y optimizando la relevancia del análisis de sentimientos en cada contexto.

Arquitectura Serverless

Para mejorar la escalabilidad y eficiencia del sistema, este proyecto se implementa sobre una arquitectura serverless, empleando FastAPI para crear una API que facilita el análisis de sentimientos y que puede ser desplegada en plataformas

como AWS Lambda o Google Cloud Functions. Esta arquitectura permite que la aplicación se ejecute sin necesidad de un servidor dedicado, lo que facilita el procesamiento bajo demanda y reduce costos, ya que se paga únicamente por las ejecuciones. Además, esta estructura permite adaptarse a variaciones en la demanda, ideal para analizar el sentimiento en tiempo real en redes sociales.

Conexión a la Base de Datos SQLite para el Historial

El proyecto incluye una base de datos SQLite para almacenar el historial de análisis de sentimientos, con el objetivo de facilitar análisis longitudinales y observar patrones de cambio en el sentimiento a lo largo del tiempo. Esta base de datos permite consultar el historial de análisis y realizar un seguimiento de los resultados obtenidos en cada análisis.

Estructura de la Base de Datos

La tabla de historial en SQLite tiene los siguientes campos:

- **id**: identificador único para cada registro.
- **texto**: el texto analizado.
- **clasificación**: el resultado de la clasificación de sentimiento en una escala de 1 a 5 estrellas.
- **sentimiento**: una codificación numérica del sentimiento (positiva, negativa o neutra) derivada de la clasificación.

Objetivo del Almacenamiento de Datos

El almacenamiento de estos datos permite no solo revisar el historial de análisis, sino también realizar un análisis en el tiempo sobre los sentimientos reflejados en los textos procesados. Esto ayuda a identificar tendencias o patrones emocionales, que pueden ser útiles para optimizar estrategias de contenido en redes sociales y mejorar la adecuación de publicaciones a las expectativas de la audiencia.

6.2 CRONOGRAMA

Mes 1: Identificación del Problema, Requisitos y Diseño

Identificación del Problema y Necesidades (Semana 1):

- Análisis detallado del problema a resolver.
- Identificación de las necesidades y desafíos específicos del proyecto.

Definición de Requisitos y Objetivos (Semana 2):

- Definición de requisitos funcionales y no funcionales del sistema.
- Establecimiento de objetivos claros y alcanzables que guiarán el desarrollo.

Diseño de la Solución (Semanas 3 y 4):

- Diseño de la arquitectura general del sistema.
- Selección de herramientas y tecnologías adecuadas (Hugging Face Transformers, FastAPI).
- Planificación de los módulos y funciones que compondrán el sistema.

Mes 2: Desarrollo Iterativo

Desarrollo Iterativo (Semanas 5 a 10):

- Desarrollo de la API utilizando FastAPI.
- Integración del modelo preentrenado de Hugging Face Transformers.
- Implementación de funciones clave para el análisis de sentimientos y evaluación de pertinencia.
- Realización de pruebas unitarias para asegurar el funcionamiento correcto de cada módulo.

Mes 3: Pruebas, Despliegue y Entrega Final

Pruebas y Validación (Semanas 11 y 12):

- Pruebas exhaustivas del sistema para verificar su funcionamiento completo.
- Validación de los resultados con casos de prueba específicos.
- Realización de ajustes y mejoras según los resultados obtenidos de las pruebas.

Despliegue y Monitoreo (Semana 13):

- Implementación del sistema en un entorno de prueba, donde se observa su rendimiento en situaciones reales.

- Monitoreo continuo del sistema para detectar y resolver problemas en tiempo real.
- Recopilación de feedback de los usuarios de prueba y realización de ajustes finales basados en esa retroalimentación.

Documentación y Entrega Final (Semana 13):

- Preparación de la documentación completa del proyecto, que incluye el diseño, desarrollo y resultados del sistema.
- Elaboración del informe final y preparación para la presentación del proyecto, asegurando que todos los aspectos estén claramente comunicados

CONCLUSIONES

1. **Desarrollo de un algoritmo de análisis de sentimientos en textos largos**: Se logró desarrollar un algoritmo basado en el modelo nlptown/bert-base-multilingual-uncased-sentiment, que permite evaluar sentimientos en textos extensos con una precisión óptima. La aplicación de BERT asegura que el análisis capture los matices emocionales en diversos contextos, contribuyendo a una comprensión más profunda del contenido en redes sociales.

2. **Evaluación de la adecuación para redes sociales**: Se establecieron criterios específicos que permiten determinar si un texto es adecuado para redes sociales, tomando en cuenta el tono y la relevancia del mensaje para el público objetivo. Esto ayuda a maximizar el impacto del contenido al adaptarlo según el contexto y el público de la plataforma.

3. **Desarrollo de una API con FastAPI**: La implementación de una API en FastAPI facilitó la integración del algoritmo de análisis de sentimientos con diversas aplicaciones y plataformas, asegurando accesibilidad y escalabilidad en el uso del sistema, lo cual permite su integración en proyectos futuros de análisis de datos.

4. **Medición de la recepción en redes sociales (Reddit)**: Al implementar métodos para analizar la recepción de temas en Reddit, el sistema pudo identificar la reacción de los usuarios a diferentes temas, contribuyendo a una mejor estrategia de contenido.

5. **Pruebas exhaustivas y ajustes en el sistema**: Las pruebas realizadas aseguraron el correcto funcionamiento del sistema, ajustando el algoritmo para optimizar su rendimiento en términos de precisión y rapidez en el análisis de sentimientos en textos largos.

6. **Implementación y monitoreo del sistema**: El despliegue del sistema en un entorno de prueba permitió identificar áreas de mejora y asegurar su funcionalidad en condiciones reales, logrando un sistema robusto y efectivo para el análisis de sentimientos en redes sociales.

BIBLIOGRAFÍA

Anand, S., & Patra, S. R. (2022). Voice and text based sentiment analysis using natural language processing. Lecture Notes in Networks and Systems, Springer, Singapore. https://doi.org/10.1007/978-981-16-8763-1_42

Liaquathali, S., & Kadirvelu, V. (2025). WCA: Integration of natural language processing methods and machine learning model for effective analysis of web content to classify malicious webpages. Journal of Advanced Research in Applied Sciences and Engineering Technology, 47(1), 105-122. https://doi.org/10.37934/araset.47.1.105122

Liu, S., Chen, X., Meng, J., & Lukač, N. (2025). Improved relation extraction through key phrase identification using community detection on dependency trees. Computer Speech & Language, 89, 101706. https://doi.org/10.1016/j.csl.2024.101706

Liu, Z., Yang, T., Chen, W., Chen, J., Li, Q., & Zhang, J. (2024). Sentiment analysis of social media comments based on multimodal attention fusion network. Applied Soft Computing, 112011. https://doi.org/10.1016/j.asoc.2024.112011

Marques, A. E., Parreira, T. G., Pereira, A. F. G., Ribeiro, B. M., & Prates, P. A. (2024). Machine learning applications in sheet metal constitutive modelling: A review. International Journal of Solids and Structures, 113024. https://doi.org/10.1016/j.ijsolstr.2024.113024

Mendieta, J. Y. (2024). *Análisis de Textos* [Repositorio en GitHub]. GitHub. https://github.com/josephmendieta/analisisTextos.git

Prakash, J., & Vijay, A. A. (2024). A novel socio-pragmatic framework for sentiment analysis in Dravidian–English code-switched texts. Knowledge-Based Systems, 112248. https://doi.org/10.1016/j.knosys.2024.112248

Sreeram, I., & Vuppala, V. P. (2017). HTTP flood attack detection in application layer using machine learning metrics and bio-inspired bat algorithm. Applied Computing and Informatics, 13(3), 13-24. https://doi.org/10.1016/j.aci.2017.10.003

Weinreich, R., & Buchgeher, G. (2011). Towards supporting the software architecture life cycle. Journal of Systems and Software, 84(10), 1704-1716. https://doi.org/10.1016/j.jss.2011.05.0