# Optimización de Técnicas NLP para Evaluación de Sentimientos en Textos Largos

**Asignaturas Integradas:**

Análisis y Diseño de Algoritmos (Doc. Yenny Sierra)

Ingeniería de Software (Doc. Oscar Penagos)

**Estudiantes:**

Joseph Yäel Mendieta Avila (ID: 67001618)

## Tabla de Contenidos

**Pregunta Orientadora:**

¿Cómo se puede crear un algoritmo que analice sentimientos en textos largos, evalúe si esos textos son adecuados para redes sociales y determine qué tan bien se recibe un tema en una red social como Reddit?

# Objetivos:

## 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.

## Objetivos Específicos:

1. **Mejorar el Análisis de Sentimientos:**
   - Utilizar herramientas avanzadas como Hugging Face Transformers para evaluar el sentimiento en textos extensos.
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 y la relevancia para el público.
3. **Desarrollar una API:**
   - Crear una API usando FastAPI que facilite la integración del algoritmo 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 en las publicaciones y comentarios.
5. **Probar y Ajustar el Sistema:**
   - Realizar pruebas exhaustivas para asegurar que el sistema funcione correctamente y ajustar el algoritmo según los resultados obtenidos.
6. **Implementar y Monitorear el Sistema:**
   - Lanzar el sistema en un entorno de prueba, supervisar su rendimiento y hacer ajustes finales basados en el feedback recibido.

## 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.

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.

Con el uso principalmente de Python, un lenguaje ampliamente utilizado en el procesamiento de lenguaje natural, se crea un algoritmo que evalúa el sentimiento de un texto y califica la relevancia de un tema para las redes sociales mediante estrellas. Además, de tener un texto que se desea publicar, el algoritmo proporciona una calificación en estrellas y evalúa su idoneidad para su publicación.

El objetivo principal de este proyecto es aplicar los conocimientos adquiridos en algoritmos y desarrollo de software para crear un algoritmo práctico que ayude a comprender qué contenido tiene mayor impacto en las redes sociales y cómo optimizar las publicaciones.

## 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.

En resumen, estos estudios muestran cómo el aprendizaje automático y el procesamiento de lenguaje natural están transformando diversas disciplinas, desde el análisis de sentimientos hasta la seguridad informática y la ingeniería, mejorando la precisión y eficiencia de los procesos en cada área.

## Antecedentes:

El procesamiento de lenguaje natural (NLP) es fundamental para manejar la vasta cantidad de información que se genera en plataformas digitales, especialmente en tareas como el análisis de sentimientos y la clasificación de contenido web. Al combinar NLP con modelos de aprendizaje automático, se han logrado avances notables en la precisión y eficiencia de estas tareas, lo que demuestra su utilidad en diversas aplicaciones.

Anand y Patra (2022) exploran dos métodos para analizar sentimientos: uno convierte la voz en texto antes de evaluarlo, y el otro analiza la voz directamente. Ambos enfoques utilizan clasificadores como Naïve Bayes y árboles de decisión para identificar si los sentimientos son positivos, negativos o neutros. Este tipo de análisis es esencial para las empresas que buscan entender mejor las opiniones de sus clientes y mejorar sus productos o servicios. La efectividad de estas técnicas en el análisis de sentimientos respalda su aplicabilidad en el proyecto, que busca evaluar y optimizar publicaciones en redes sociales.

En el ámbito de la seguridad en línea, Liaquathali y Kadirvelu (2025) se enfocan en la clasificación de páginas web maliciosas mediante el análisis de contenido textual. Al utilizar técnicas de vectorización y modelos de aprendizaje automático, como los bosques aleatorios, este enfoque logra identificar amenazas como phishing y malware con alta precisión. La relevancia de estos métodos en la detección de contenido malicioso subraya su potencial para ser aplicados en diferentes contextos, incluyendo el análisis de texto extenso en redes sociales.

Finalmente, Liu, Chen, Meng y Lukač (2025) presentan un método para identificar frases clave utilizando árboles de dependencia y algoritmos de detección de comunidades. Esto mejora significativamente la extracción de relaciones, lo que es crucial para una comprensión más profunda del texto. La eficacia de esta técnica respalda su relevancia para el proyecto, que también busca identificar y analizar aspectos clave en textos largos.

Estos artículos fueron seleccionados porque demuestran la efectividad del uso de técnicas de NLP y aprendizaje automático en diversas áreas, desde el análisis de sentimientos hasta la seguridad en línea. Dado su éxito en estos campos, estas mismas técnicas serán valiosas para nuestro objetivo de evaluar y optimizar publicaciones en redes sociales, mostrando su aplicabilidad y potencial en diferentes aplicaciones.

## Planteamiento del Problema:

En la era digital, la gran cantidad de contenido textual generado en redes sociales y otras plataformas hace crucial entender el sentimiento y la pertinencia de los textos. Sin embargo, analizar sentimientos en textos largos es un desafío, ya que las emociones pueden variar a lo largo del contenido, y los algoritmos tradicionales suelen fallar al capturar estos matices.

Además, evaluar qué tan adecuado es un texto para ser publicado en redes sociales es igualmente importante, pero carecemos de algoritmos que combinen ambas capacidades: análisis de sentimientos y evaluación de pertinencia. Esta falta de soluciones eficientes limita la capacidad de optimizar el contenido antes de publicarlo, afectando su impacto en la audiencia.

Por ello, este proyecto busca desarrollar un algoritmo  que utilice modelos preentrenados de Hugging Face Transformers, desplegada con FastAPI, para optimizar el análisis de sentimientos en textos largos y evaluar su pertinencia para redes sociales.

**Metodología basada en el ciclo de vida del software:**

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

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.

## Cronograma del Proyecto

### *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.

## Presupuesto del Proyecto

***Herramientas y Tecnologías:***

- Hosting y Dominio (3 meses): $60,000 COP
- Licencias de Software (si es necesario): $0 COP

***Equipo y Materiales:***

- Computadora y Accesorios (uso personal): $0 COP
- Conexión a Internet (3 meses): $240,000 COP
- Papelería y Materiales (para la documentación): $50,000 COP

***Recursos Humanos:***

- Desarrollador/Estudiante (3 meses): $4,200,000 COP
- Consultoría y Asesoría Técnica (5 horas): $500,000 COP

***Otros Gastos:***

- Transporte y Movilidad (10 viajes): $100,000 COP
- Gastos Administrativos: $50,000 COP

***Posibles Gastos Adicionales:***

- Impresión y encuadernación de informes: $20,000 COP (aproximado)
- Costos de envíos: $20,000 COP (aproximado)
- Membresías o suscripciones menores: $30,000 COP (aproximado)
- Otros costos menores: $20,000 COP (aproximado)

***Imprevistos: $100,000 COP***

***TOTAL: $5,320,000 COP***

**Referencias Bibliográficas:**

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

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.2011.05.036