CARRERA DE COMPUTACIÓN

INFORME

1. Datos informativos

1.1. **Módulo:** 1

1.2. **Nivel:** Séptimo

1.3. **Fecha:** 18 de febrero

1.4. **Nombres:** Aupas Antony, Baraja Cristian, Basantes Geovanny

1.5. **Tema:** Explicabilidad de Modelos de IA en Procesamiento de Lenguaje Natural (NLP).

2. Objetivo

Analizar y evaluar técnicas de explicabilidad en modelos de inteligencia artificial aplicados a procesamiento de lenguaje natural (NLP), con el fin de mejorar la transparencia y comprensibilidad de sus decisiones.

3. Contenido

Según Calero Sánchez et al. (2024) mencionan que el Proceso del Lenguaje Natural (PLN) es un campo que ha evoluciona significativamente desde sus inicios en la década de 1950. Surgió como una subdisciplina tanto de la IA como de la lingüística, con el objetivo de proporcionar desafíos relacionados con la generación y comprensión automática del lenguaje humano. Por otro la, a lño largo del tiempo, esta área ha avanzado significativamente, donde es impulsada por el desarrollo de los algoritmos mas sofisticados y el crecimiento del poder computacional.

3.1. Análisis de Requerimientos

3.1.1. Identificación de necesidades

Los modelos de IA en el procesamiento de lenguaje natural (PLN) son muy precisos, pero funcionan como "cajas negras", ya que no se entiende claramente cómo toman decisiones. Esto genera desconfianza, especialmente en aplicaciones críticas como análisis de sentimientos o atención a la cliente automatizada. Por ello, la explicabilidad de estos modelos es crucial para garantizar transparencia y confianza en sus resultados.

3.1.2. Definición de objetivos

El principal objetivo de este proyecto es desarrollar un sistema de explicabilidad para los modelos de PLN. Específicamente, se pretende:

- Implementar técnicas de interpretabilidad, como LIME y SHAP, para explicar de manera clara cómo los modelos de IA toman decisiones.
- Utilizar una base de datos híbrida (SQL para metadatos y NoSQL para texto no estructurado) y desarrollar una interfaz gráfica interactiva que permita visualizar las explicaciones generadas por el modelo.

3.2. Diseño de arquitectura del software

3.2.1. Plan técnico

Según Miller (2019) menciona que para la construcción de este sistema, se implementarán las siguientes tecnologías:

Backend: Se usará Django como framework web para administrar los modelos de procesamiento de lenguaje natural (PLN), integrarlos con las bases de datos y crear una API REST.

Frontend: La interfaz gráfica de usuario se desarrollará con React, permitiendo visualizar de manera clara las explicaciones proporcionadas por los modelos.

Modelos de IA: Se utilizarán modelos preentrenados, como BERT o GPT, junto con técnicas de explicabilidad, como LIME o SHAP, para interpretar los resultados generados por el sistema.

Bases de Datos: Se optará por MongoDB (NoSQL) para almacenar grandes volúmenes de texto no estructurado y PostgreSQL (SQL) para manejar información estructurada, como metadatos y registros de auditoría de las decisiones del modelo.

3.2.2. Interfaces de usuario

Según Django (2025) y React (2025) manifiestan que: La interfaz se desarrollará con el objetivo de simplificar la interacción con el sistema y ofrecer explicaciones claras y visualmente intuitivas de los resultados generados por los modelos. Las interfaces principales incluirán:

Un formulario de entrada que permitirá al usuario cargar o ingresar texto para su análisis.

Un panel de resultados donde se presentarán las explicaciones, incorporando elementos como gráficos de relevancia de términos, visualizaciones de la atención del modelo y descripciones detalladas de las decisiones tomadas.

3.2.3. Bases de datos

Para gestionar el almacenamiento de datos, se implementarán dos tipos de bases de datos:

MongoDB: Se usará para guardar datos textuales no estructurados, como los textos de entrada que serán procesados por el modelo.

PostgreSQL: Se empleará para almacenar metadatos estructurados asociados a los textos y a las explicaciones generadas, incluyendo detalles como información de usuarios, fechas de análisis y registros de auditoría.

3.2.4. Implementación

3.2.4.1. Escribir Código

El desarrollo del sistema se organizará en los siguientes módulos:

Modelo de PLN: Ajuste o entrenamiento de modelos preexistentes, como BERT o GPT, para tareas específicas de procesamiento de lenguaje natural.

API Backend: Construcción de una API REST que facilite la interacción con el modelo y reciba solicitudes de explicación desde la interfaz de usuario.

Frontend: Implementación de la interfaz gráfica utilizando React, enfocada en la visualización de resultados y explicaciones generadas por el modelo.

Bases de Datos: Diseño y creación de las estructuras necesarias en MongoDB y PostgreSQL para almacenar y consultar los datos de manera eficiente.

3.2.4.2. Calidad del Código

Se garantizará la calidad del código mediante el uso de buenas prácticas en programación, como el cumplimiento de estándares PEP 8 en Python y principios de modularidad en JavaScript. Además, se emplearán herramientas de análisis estático de código y revisión de código en equipo para mantener una alta calidad.

3.2.4.3. Trabajo en Equipo

El desarrollo del proyecto se realizará en equipo utilizando Git para el control de versiones y GitHub para la colaboración en el código. El trabajo se organizará en iteraciones ágiles, siguiendo una metodología SCRUM, con reuniones regulares para revisar avances y ajustar el plan de trabajo.

3.2.5. Pruebas: Garantías de Calidad

3.2.5.1. Pruebas Unitarias

Se escribirán pruebas unitarias para verificar que cada componente del sistema funcione correctamente, incluyendo el modelo de PLN, la API y la visualización de las explicaciones.

3.2.5.2. Pruebas de Integración

Se realizarán pruebas de integración para asegurarse de que los diferentes módulos (backend, frontend y base de datos) trabajen de manera conjunta y sin errores. Esto incluirá pruebas del flujo de datos entre el frontend y el modelo.

3.2.5.3. Pruebas de Rendimiento

Según Deepseek (2024) manifiuenta que se evalúa el rendimiento del sistema, específicamente la velocidad de procesamiento del modelo y la eficiencia en las consultas a la base de datos, para asegurar que el sistema pueda manejar grandes volúmenes de datos sin comprometer la experiencia del usuario.

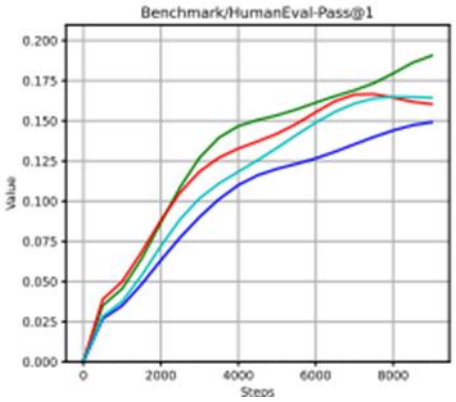

Figura 1 :Prueba de Rendimiento 1

3.2.5.4. Pruebas de Aceptación

Se llevarán a cabo pruebas de aceptación con usuarios finales para verificar si las explicaciones generadas por el modelo son comprensibles y útiles como los mostrados en la Figura 1, ajustando la interfaz según sea necesario.

3.2.6. Despliegue y Mantenimiento

3.2.6.1. Instalación

El sistema será desplegado en un servidor en la nube, utilizando plataformas como AWS o Google Cloud, para asegurar escalabilidad y alta disponibilidad. Se configurarán contenedores Docker para facilitar la instalación y ejecución del sistema en diferentes entornos.

3.2.6.2. Puesta en Marcha

Una vez desplegado, se realizará una puesta en marcha del sistema, verificando que todos los servicios estén operativos y funcionando correctamente. Esto incluirá la configuración de herramientas de monitorización para detectar posibles problemas en tiempo real.

3.2.6.3. Mantenimiento Continuo

El sistema será mantenido de manera continua para asegurar su funcionamiento óptimo. Esto incluirá actualizaciones periódicas de los modelos de IA, ajustes en las bases de datos según sea necesario y mejoras en la interfaz en función del feedback de los usuarios.

4. Conclusiones

En estudio se evidenció que la falta de compresión en modelos de NLP como BERT, GPT y DeepSeek puede afectar su adopción en entornos críticos. La aplicación de técnicas como LIME, SHAP y Attention Visualization permitió interpretar sus decisiones, mejorando la transparencia y la confianza de los usuarios. Además, se observó que la explicabilidad

facilita la detección de sesgos y errores, lo que contribuye a desarrollar modelos más justos y auditables.

El impacto de la explicabilidad en la optimización y regulación

La implementación de una herramienta visual de XAI para NLP demostró que la explicabilidad no solo mejora la interpretabilidad, sino que también permite optimizar el ajuste de los modelos y cumplir con normativas de transparencia y privacidad. Se concluye que la combinación de IA explicable y NLP es clave para el desarrollo de sistemas confiables y auditables, recomendando futuras investigaciones en técnicas más eficientes y especializadas según el contexto de aplicación.

5. Referencias bibliográficas

Calero Sánchez, M., Gónzales Gónzales, J., Sánchez Berriel, I., Burrillo-Putze, G., & Roda García, J. (2024). *El Procesamiento de Lenguaje Natural en la revisión de literatura científica.* doi:10.55633/s3me/REUE030.2024

Deepseek. (26 de Enero de 2024). *DeepSeek-Coder: When the Large Language Model Meets Programming - The Rise of Code Intelligence.* Obtenido de https://arxiv.org/pdf/2401.14196

Django. (2025). *Django hace más fácil construir mejores aplicaciones web más rápidamente y con menos código.* Obtenido de https://www.djangoproject.com/

Miller, T. (2019). *Explanation in artificial intelligence: Insights from the social sciencies.* doi:https://doi.org/10.1016/j.artint.2018.07.007

React. (2025). *Crear interfaces de usuario a partir de componentes*. Obtenido de https://react.dev/