Revista Facultad de Ingeniería (Rev. Fac. Ing.) Vol. 33, No. 69.
L-ISSN: 0121-1129, e-ISSN: 2357-5328.

DOI: 10.19503/01211129.v33.n69.2024.18076

# SMART PRODUCT BACKLOG: CLASIFICACIÓN AUTOMÁTICA DE HISTORIAS DE USUARIO USANDO MODELOS DE LENGUAJE DE GRAN ESCALA

## Smart Product Backlog: Automatic Classification of User Stories Using Large Language Models (LLM)

**Mauricio Gaona-Cuevas**
Ph.D. Universidad del Valle (Cali-Valle del Cauca, Colombia).
mauricio.gaona@correounivalle.edu.co

**Víctor Bucheli-Guerrero**
Ph.D. Universidad del Valle (Cali-Valle del Cauca, Colombia).
victor.bucheli@correounivalle.edu.co

**Fredy Vera-Rivera**
Ph.D. Universidad Francisco de Paula Santander (Cúcuta-Norte de Santander, Colombia).
fredyhumbertovera@ufps.edu.co

**Received / Recibido:** 13/06/2024
**Accepted / Aceptado:** 29/09/2024

## RESUMEN

En los procesos de desarrollo ágil de software, específicamente de las aplicaciones inteligentes que aprovechan la inteligencia artificial (IA), el Smart Product Backlog (SPB) es un artefacto que incluye funcionalidades implementables tanto con IA como sin esta. En este contexto, existe un trabajo notable en el desarrollo de modelos de Procesamiento del Lenguaje Natural (NLP) en los que, aquellos de gran escala (LLM por sus siglas en inglés), han demostrado un rendimiento excepcional.
Sin embargo, surgió la pregunta respecto a si dichos modelos podían utilizarse en tareas de clasificación automática, sin necesidad de una anotación previa, permitiendo la extracción directa del Smart Product Backlog (SPB). En este estudio, se comparó la eficacia de las técnicas de ajuste con los métodos de prompting para esclarecer el potencial de los modelos ChatGPT-4o, Gemini Pro1.5 y ChaGPT-Mini; se construyó un set de datos con historias de usuario, clasificadas manualmente por un grupo de expertos, que permitió realizar el ensamble de experimentos y, a su vez, construir las tablas de contingencia, respectivas; y se evaluaron estadísticamente las métricas de desempeño de la clasificación de cada LLM y se utilizaron métricas de rendimiento, como la exactitud, la sensibilidad y el F1-Score, para determinar la efectividad de cada modelo. Este enfoque comparativo buscó destacar las fortalezas y limitaciones de cada LLM en el contexto de estructurar la asistencia en la construcción del SPB de manera eficiente y precisa. El análisis demostró que ChatGPT-Mini tiene limitaciones en el balance entre precisión y sensibilidad. Además, aunque Gemini Pro1.5 mostró superioridad en la puntuación de exactitud, y ChatGPT también exhibió un rendimiento aceptable, ninguno es lo suficientemente robusto para construir una herramienta completamente automatizada para la clasificación de historias de usuario. Por lo tanto, se identifica la necesidad de desarrollar un clasificador especializado que permita la construcción de una herramienta automatizada para recomendar historias de usuario viables para el desarrollo con IA, apoyando así la toma de decisiones en proyectos de software ágiles.

Esta edición se financió con recursos del Patrimonio Autónomo Fondo Nacional de Financiamiento para la Ciencia, la Tecnología y la Innovación, Francisco José de Caldas, Minciencias

**How to cite:** M. Gaona-Cuevas, V. Bucheli-Guerrero, F. Vera-Rivera, "Smart Product Backlog: Automatic Classification of User Stories Using Large Language Models (LLM)", *Revista Facultad de Ingeniería*, vol. 33, no. 69, e18076, 2024. https://doi.org/10.19053/01211129.v33.n69.2024. 18076

**Palabras clave:** backlog de producto inteligente; clasificación de historias de usuario; especificación de requerimientos software; identificador inteligente de historias de usuario; inteligencia artificial; modelos de lenguaje a gran escala.

## ABSTRACT

In agile software development processes, specifically within intelligent applications that leverage artificial intelligence (AI), *Smart* Product Backlog (SPB) serves as an artifact that includes both AI-implementable functionalities and those that do not use AI. Significant work has been done in the development of Natural Language Processing (NLP) models, and Large Language Models (LLMs) have demonstrated exceptional performance. However, whether LLMs can be used in automatic classification tasks without prior annotation, thereby allowing direct extraction from the Smart Product Backlog (SPB) remains an unanswered question. In this study, we compared the effectiveness of fine-tuning techniques with "prompting" methods to determine the potential of models such as ChatGPT-4o, Gemini Pro 1.5, and ChaGPT-Mini. A dataset was constructed with user stories manually classified by a group of experts, which enabled assembling experiments and creating the respective contingency tables. The classification performance metrics of each LLM were statistically evaluated; accuracy, sensitivity, and F1-Score were used to assess the effectiveness of each model. This comparative approach aimed to highlight the strengths and limitations of each LLM in efficiently and accurately assisting in the construction of the SPB. This comparative analysis demonstrates that ChatGPT-Mini has limitations in balancing precision and sensitivity. Although Gemini Pro 1.5 was superior in accuracy scores and ChatGPT performed well, neither is robust enough to build a fully automated tool for user story classification. Therefore, we identified the need to develop a specialized classifier that enables the construction of an automated tool to recommend viable user stories for AI development, thereby supporting decision-making in agile software projects.

**Keywords:** artificial intelligence; large scale language models; smart product backlog; smart user story identifier; Software Requirements Specification; user story classification.

## 1. INTRODUCCIÓN

Las metodologías ágiles de desarrollo de software emplean historias de usuario (HU) para detallar las funcionalidades de la aplicación, enfocándose en la perspectiva del usuario final [1]. Estas HU no solo describen la funcionalidad esperada, sino que también destacan el valor que cada requisito añade a la aplicación. El Product Backlog (PB) contiene las funcionalidades definidas para la aplicación que se representan a través de HU. [2].

El PB, esa lista ordenada que guía el desarrollo de software, se transformó con la llegada de la Inteligencia Artificial (IA) [3]. Ahora, las aplicaciones aspiran a ser inteligentes, y su backlog debe reflejarlo. En este contexto, surgió el Smart Product Backlog (SPB) que incorpora no solo las funcionalidades tradicionales, sino también aquellas que aprovechan el poder de la IA, convirtiéndose en la brújula de los equipos, permitiéndoles visualizar y planificar cómo la IA puede llevar su proyecto a otro nivel.

La construcción manual del SPB, aunque posible, presenta obstáculos significativos. El proceso es lento, costoso y requiere conocimientos especializados en IA. Su inherente subjetividad aumenta el riesgo de errores, lo que puede impactar negativamente el proyecto en términos de tiempo, presupuesto y calidad final, tal como lo señalaron Kaur y Kaur [4]. Por ello, resulta crucial contar con herramientas que asistan en la toma de decisiones durante su construcción, facilitando el desarrollo de aplicaciones inteligentes y maximizando el potencial de la IA [5].

Para clasificar eficazmente las HU en el desarrollo de *software* y determinar su idoneidad para la implementación con IA, es crucial comprender sus funcionalidades y evaluar su potencial de mejora mediante técnicas de IA. Las HU que se benefician de esta suelen incluir, entre otras características, personalización automatizada, análisis predictivo y aprendizaje automático basado en datos de usuarios. Por su parte, las HU transaccionales se centran en la gestión de datos, la interacción directa con los usuarios y el procesamiento de transacciones estándar.

Reconociendo esta distinción, esta investigación buscó aprovechar el poder de los Modelos de Lenguaje de Gran Escala (LLM por sus siglas en inglés) para clasificar automáticamente HU, explorando su capacidad para identificar patrones en las descripciones de estas que permitieran reconocer aquellas que eran adecuadas para su implementación con IA.

En esta línea, se propuso responder a la siguiente pregunta: ¿Los Modelos de Lenguaje de Gran Escala (LLM) pueden utilizarse en tareas de clasificación automática, sin necesidad de una anotación previa, permitiendo la extracción directa del Smart Product Backlog?, comparando la eficacia de las técnicas de ajuste con los métodos de *prompting* para esclarecer el potencial de los modelos Chat GPT4o, Gemini Pro-1.5 y GPT4o-Mini.

Para dar respuesta, se construyó un set de datos con historias de usuario clasificadas manualmente por un grupo de expertos, que permitió realizar un ensamble de experimentos y, a su vez, construir las tablas de contingencia respectivas; y se evaluaron las métricas estándar tales como accuracy, precisión, recall, F1-Score.

Este enfoque comparativo permitió destacar las fortalezas y limitaciones de cada LLM en el contexto de desarrollar un asistente de la construcción del SPB en inglés de manera eficiente y precisa. Dicho análisis demostró que ChatGPT4o-Mini tiene limitaciones en el balance entre precisión y sensibilidad, Gemini Pro-1.5 mostró superioridad en la puntuación de exactitud, y ChatGPT4o también exhibió un rendimiento aceptable. A su vez, los resultados subrayaron su potencial para construir una herramienta automatizada para recomendar HU viables para el desarrollo con IA, apoyando la toma de decisiones en proyectos de software ágiles. Sin embargo, no fueron prometedores y muestran la necesidad de construir un modelo para la recomendación automática de HU desarrollables con IA.

Este documento está estructurado en seis secciones. La primera corresponde a la introducción, los trabajos relacionados a la segunda, la metodología a la tercera, los resultados a la cuarta, la discusión a la quinta y, finalmente, las conclusiones a la sexta.

## 2. TRABAJOS RELACIONADOS

En los últimos años, los LLM han emergido como herramientas poderosas en diversas áreas, incluyendo el desarrollo de *software*. Este estado del arte revisó las investigaciones recientes que exploraron su potencial para mejorar la calidad de las historias de usuario, la automatización en su generación, su aplicación en contextos de clasificación, redacción y extracción de estas, y la confiabilidad de los modelos (Tabla 1).

Zhang *et al.* [6] demostraron que los LLM pueden mejorar la calidad de las historias de usuario en entornos de desarrollo ágil, y sugirieron que su uso ayudaría a los equipos de desarrollo a redactar HU más precisas y alineadas con los requisitos del cliente, lo que resultaría en un mejor producto final.

Rahman y Zhu [7] especially Large Language Models (LLM exploraron la capacidad de los LLM para automatizar la generación de HU a partir de documentos de requisitos. Sus hallazgos indicaron que, con el entrenamiento adecuado, los LLM tienen la capacidad de generar historias de usuario que cumplen con los estándares esperados en la industria del *software*.

Chuor et al. examinaron cómo se utilizaban modelos LLM y técnicas de aprendizaje automático para la clasificación automática de historias de usuario en entornos de desarrollo ágil. La investigación mostró que el uso de LLM incidía en mejorar la precisión en la categorización de dichas historias, ayudando a los

equipos a priorizar mejor las tareas y optimizar los procesos de desarrollo [8].

Carlos Dos Santos et al. generaron automáticamente historias de usuario basados en dos enfoques distintos: la representación de N-gramas con heurística lingüística y el modelo GPT-3. El trabajo se evaluó utilizando diversos corpus de HU y calculando las métricas BLEU, ROUGE-N y BERTScore para cada conjunto de estas, generadas por ambos enfoques. Concluyeron que, aunque los modelos GPT se destacaban en la producción de historias de usuario más completas, los de N-gramas exhibían un nivel mayor de sensibilidad semántica [3]

Junyuan Hong et al. analizaron los desafíos relacionados con la confianza en los LLM, particularmente cuando se aplicaban técnicas de compresión para mejorar la eficiencia computacional. La investigación es relevante en el contexto del uso de LLM para la clasificación de historias de usuario, ya que aseguró que la compresión no degradaba la calidad de las decisiones automatizadas que estos modelos tomaban [9].

*Tabla 1. Resumen del trabajo relacionado*

| Autor | Tema |
|---|---|
| Zhang *et al*. [6] | Mejora de la calidad de historias de usuario con LLM. |
| Rahman y Zhu [7] especially Large Language Models (LLM | Automatización en la generación de historias de usuario con LLM. |
| Chuor *et al*. [8] | Clasificación automática de historias de usuario con LLM. |
| Dos Santos Carlos *et al*. [3] | Generador automático de historias de usuario basados en representación de N-gramas con heurísticas lingüísticas y el modelo GPT-3. |
| Junyuan Hong [9] | Uso de LLM para la clasificación de historias de usuario analizando la confianza en los LLM, particularmente cuando se aplican técnicas de compresión. |
| Lichao Sun [10] | Marco de trabajo para mejorar la confiabilidad y la trazabilidad en LLM. |
| Kaur *et al*. [4] | Automatización en la identificación y clasificación de requerimientos de *software* con IA. |
| Kumar B. *et al*. [11] | Clasificación de requisitos no funcionales (NFR) según su relevancia para las historias de usuario con aprendizaje automático. |
| Liu J. *et al*. [12] | Generación de conocimiento impulsada por indicaciones para el razonamiento de sentido común. |

Lichao Sun et al. propusieron un marco de trabajo para mejorar la confiabilidad y la trazabilidad en los LLM, lo cual es esencial para asegurar que los modelos sean utilizados de manera efectiva y segura en la clasificación de HU y otras aplicaciones críticas [10].

Kaur Kamaljit y Kaur Parminder resaltaron el potencial de la IA para automatizar la identificación y clasificación de requerimientos de software en funcionales y no funcionales, particularmente en proyectos de gran envergadura y complejidad. Su estudio, que abarcó 60 investigaciones sobre técnicas automáticas, concluyó que el aprendizaje por transferencia basado en redes neuronales convolucionales (CNN) y las máquinas de soporte vectorial (SVM) ofrecía los resultados más prometedores. Sin embargo, los autores advirtieron que la aplicabilidad de estas técnicas en entornos reales y complejos requería mayor validación. Este hallazgo subraya la necesidad de una colaboración más estrecha entre la ingeniería de requisitos y la IA para superar los desafíos que enfrentan los sistemas automatizados en la práctica [4]

Kumar B. et al. señalaron la importancia de los requisitos no funcionales (NFR) en la ingeniería de requisitos del desarrollo de software ágil; y propusieron un enfoque basado en aprendizaje automático para clasificar los NFR según su relevancia para las HU. Su estudio comparativo demostró que los clasificadores Random Forest y XGBoost alcanzaron la mayor precisión (96,39 %), superando a otros modelos como K-vecinos más cercanos, análisis discriminante lineal, clasificador de vectores de soporte, regresión logística y Naive Bayes [11]

Por último, Liu J. et al. exploraron la generación de conocimiento impulsada por indicaciones para el razonamiento de sentido común y analizaron cómo la incorporación de conocimiento externo mejoraría el razonamiento de sentido común. El método propuesto, la generación de conocimiento impulsada por indicaciones, no requería supervisión específica de la tarea para la integración del conocimiento. Se demostró que dicho método mejoraba el rendimiento de los modelos de última generación en cuatro tareas de razonamiento de sentido común [12].

## 3. METODOLOGÍA

En esta sección se describen las preguntas de investigación que guiaron el estudio, se proporciona una descripción y comparación exhaustiva de los LLM aplicados para clasificar y se detallan las métricas de evaluación utilizadas para valorar el rendimiento del modelo.

### *3.1. Preguntas de investigación*

Otras investigaciones han empleado técnicas matemáticas, estadísticas y predictivas para clasificar automáticamente historias de usuario implementables con inteligencia artificial. Este estudio evaluó tres modelos de LLM para determinar su eficacia en la automatización de la clasificación.

Para esto, se planteó varias preguntas clave:

P1: ¿Pueden utilizarse los LLM en tareas de clasificación automática, sin necesidad de una anotación previa, permitiendo así la extracción directa del Smart Product Backlog?

P2: ¿Qué tan precisas son las clasificaciones de los LLM (IA, no IA) en historias de usuario?

P3: ¿Cuáles LLM muestran la menor tasa de error en la clasificación binaria de historias de usuario?

Las respuestas a estas preguntas se encuentran en los resultados y conclusiones de este documento.

### *3.2. Protocolo de datos*

Para elegir un conjunto de datos apropiado y evaluar los modelos de LLM capaces de identificar automáticamente si una historia de usuario era implementable con IA, se utilizó un dataset que incluyó proyectos de software junto con sus correspondientes historias de usuario en inglés. Este recopiló 22 proyectos y 1680 HU que reflejaban los requerimientos de dichos proyectos. Como resultado, se seleccionó el repositorio de Mendeley [13] debido a la diversidad de los proyectos y a la buena calidad de las descripciones de las HU.

#### *3.2.1. Preparación del dataset por expertos*

El proceso de preparación del dataset para evaluar los LLM se llevó a cabo de manera sistemática, involucrando un equipo de expertos en desarrollo de software con experiencia en metodologías ágiles e IA. Se les brindó una comprensión clara del objetivo del proyecto y la importancia de clasificar con precisión las HU en función de la viabilidad técnica de ser implementadas con IA.

Los criterios de clasificación se definieron colaborativamente, asegurando consistencia y objetividad. Cada historia de usuario fue evaluada y etiquetada como susceptible de implementación con IA (1) o no (0), alcanzando un consenso en casos de ambigüedad para garantizar la precisión del *dataset* final.

La selección de los tres expertos se realizó por experiencia profesional o investigativa en el tema: uno era experto en desarrollo de *software*, con más de 20 años de experiencia, quien aportó una sólida comprensión de metodologías ágiles, arquitectura de sistemas y gestión de historias de usuario, siendo esencial para identificar dependencias técnicas y optimizar la priorización. Otro era especialista en inteligencia artificial con más de 10 años de experiencia, quien se centró en técnicas avanzadas de machine learning y procesamiento de lenguaje natural (NLP), mejorando la clasificación automatizada de HU. Por último, un profesional con más de 10 años de experiencia en desarrollo de software e inteligencia artificial que combinaba su conocimiento en programación y en diversas técnicas de IA, permitiendo una integración efectiva de soluciones inteligentes en el proceso de priorización y gestión ágil de HU.

Cada una de estas historias tiene una estructura que se define por el usuario, la descripción de la funcionalidad y por ser lo más explícita posible. Un ejemplo de esto es: “As a Broker user, I want to Upload and Validate the error message to have accurate text”. En este caso, el grupo de expertos clasificó la HU como No-IA.

Del total de registros se seleccionó una muestra que correspondió al tamaño de un muestreo simple con nivel de confianza 95 % y margen de error 6 %. Esto resultó en una muestra de 262 historias de usuario que se seleccionaron totalmente al azar. Su clasificación manual derivó en un conjunto de datos de 262 HU en dos clases balanceadas: 131 HU-IA y 131 HU-NoIA.

### *3.3. Flujo de proceso de la investigación*

El proceso inició con las historias de usuario del repositorio de Mendeley. De estas, se extrajo una muestra representativa para el análisis que fue sometida a una clasificación manual por parte de expertos, quienes determinaron si cada HU era susceptible de ser implementada con IA (HU-IA) o no (HU-NoIA). Posteriormente, se formularon prompts para cada LLM y se aplicaron a las HU seleccionadas con cada modelo. El prompt usado fue:

A partir de ahora actuará como un experto clasificador de historias de usuario susceptibles de implementarse con técnicas de inteligencia artificial. Por favor, clasifique las HU asignando el valor de 1 si se recomienda su implementación con IA y 0 si no. Para la clasificación, puede basarse en los criterios que considere más adecuados según la información que tenga disponible al respecto. El resultado debe presentarse en una tabla con tres columnas:

Columna 1. Número de la HU

Columna 2. Descripción de la HU

Columna 3. Clasificación (1 o 0)

A continuación, encontrará las HU a clasificar: [Inserta aquí las Historias de Usuario a clasificar]

En el prompt se decidió no dar los criterios de clasificación para evitar sesgos. Luego de esto, con los resultados de las clasificaciones obtenidas, fueron almacenados para cada HU y cada LLM, y este proceso se repitió diez veces para obtener múltiples corridas, teniendo en cuenta que los LLM no son sistemas determinísticos.

Con los datos recopilados de estas clasificaciones, se construyeron tablas de verdad por cada corrida, y se calcularon métricas de desempeño para cada LLM en cada una de las corridas. Finalmente, se obtuvo la mediana, el promedio y la desviación estándar de las métricas de evaluación para cada LLM para evaluar el desempeño general de cada uno, y se seleccionó el que tuvo mejor desempeño.

Figura 1 presenta el flujo de datos, desde las fuentes de información Mendeley de historias de usuario hasta la clasificación de los LLM HU-IA y HU-NoIA. En este sentido, es posible determinar la capacidad predictiva de los modelos y su menor error.

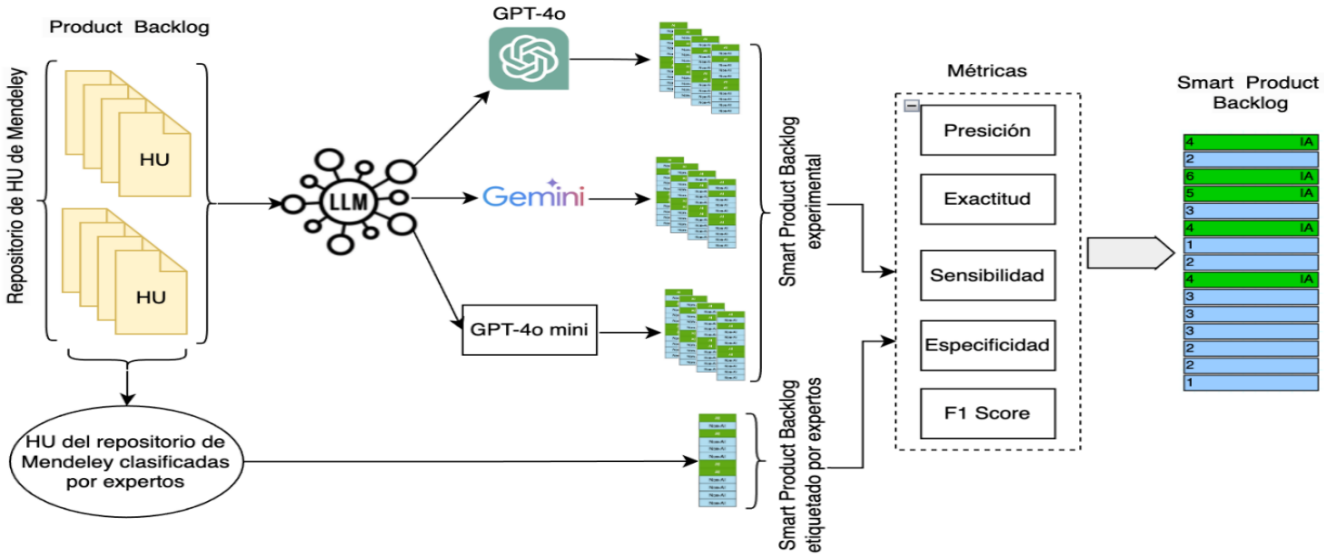


**Figura 1.** *Clasificador inteligente del Product Backlog*

## 4. RESULTADOS

En los procesos de desarrollo ágil de software, específicamente de las aplicaciones inteligentes que aprovechan la inteligencia artificial (IA), el Smart Product Backlog (SPB) es un artefacto que incluye tanto funcionalidades implementables con IA como otras que no la utilizan.

Actualmente, es generalizado el uso de LLM para desarrollar tareas específicas en las que han demostrado buen rendimiento. En este estudio, se comparó la eficacia de las técnicas de ajuste con los métodos de *prompting* para esclarecer el potencial de los modelos ChatGPT-4o, Gemini Pro-1.5 y ChatGPT4o-Mini.

Así, en el contexto de desarrollo ágil de software, los resultados obtenidos con el modelo Gemini Pro-1.5 ofrecen una perspectiva interesante (Tabla 2).

**Tabla 2.** *Resultados de la clasificación con Gemini Pro-1.5*

| Mediana | | | | | |
|---|---|---|---|---|---|
| **Clase** | **Exactitud** | **Precisión** | **Sensibilidad** | **Especificidad** | **F1-Score** |
| 0 - HU-NoIA | 0,637 | 0,805 | 0,601 | 0,717 | 0,694 |
| 1 - HU-IA | 0,637 | 0,450 | 0,717 | 0,601 | 0,556 |
| **Promedio** | | | | | |
| **Clase** | **Exactitud** | **Precisión** | **Sensibilidad** | **Especificidad** | **F1-Score** |
| 0 - HU-NoIA | 0,646 | 0,803 | 0,615 | 0,707 | 0,695 |
| 1 - HU-IA | 0,646 | 0,489 | 0,707 | 0,615 | 0,575 |
| **Desviación** | | | | | |
| **Clase** | **Exactitud** | **Precisión** | **Sensibilidad** | **Especificidad** | **F1-Score** |
| 0 - HU-NoIA | 0,056 | 0,025 | 0,053 | 0,054 | 0,036 |
| 1 - HU-IA | 0,056 | 0,107 | 0,054 | 0,053 | 0,089 |

De una parte, al observar los valores medianos y promedios para la clase HU-IA, se destacó una alta precisión (~0.803), lo que indica que el modelo es confiable al identificar correctamente las historias clasificadas como IA. Sin embargo, la sensibilidad (~0.615) mostró que había margen para mejorar en la identificación completa de todas las HU que deberían ser clasificadas como IA. La especificidad (~0.707) sugirió que el modelo manejó razonablemente bien los falsos negativos, pero también que podría beneficiarse de ajustes.

Para la clase HU-NoIA, el desempeño del modelo fue menos favorable. La precisión media (~0.489) y el F1-Score (~0.575) reflejaron un desafío en la reducción de falsos positivos, lo que implica que el modelo clasificó incorrectamente algunas HU como no implementables con IA. La sensibilidad y especificidad fueron decentes (~0.707 y ~0.615 respectivamente), pero la variabilidad en la precisión, como lo indicó la desviación estándar más alta (~0.107), sugirió que el modelo no era tan consistente en esta clase, lo que afectó la confiabilidad general del clasificador.

En suma, el análisis de la consistencia del modelo Gemini mostró que, aunque el modelo tuvo un rendimiento estable para la clase HU-IA, la inconsistencia en la clase HU-NoIA era una preocupación.

Esto significa que es necesario construir un modelo propio optimizado y entrenado para la tarea específica en cuestión. Así pues, aunque Gemini tiene un rendimiento aceptable, las mejoras en la precisión y consistencia para las HU clasificadas como no implementables con IA podrían ser clave para seleccionar un modelo equilibrado y confiable en un entorno de desarrollo ágil.

En el caso de ChatGPT4o, este ha ganado una popularidad considerable debido a su capacidad para generar texto coherente y realizar tareas generales con un alto nivel de competencia. Su uso se ha extendido en diversas aplicaciones, desde la redacción de contenido hasta la asistencia virtual, gracias a su habilidad para manejar una amplia variedad de contextos y solicitudes. Sin embargo, aunque su desempeño en tareas generales es destacado, enfrenta desafíos cuando se trata de tareas específicas y altamente especializadas. En estos casos, su precisión y efectividad pueden disminuir, revelando limitaciones en su capacidad para entender y procesar información con la profundidad y exactitud necesarias para aplicaciones más técnicas o complejas.

De acuerdo con lo anterior, los resultados obtenidos para ChatGPT4o revelaron un desempeño general bajo (ver Tabla 3). Las métricas medianas y promedios para la clase HU-NoIA representa que las historias de usuario clasificadas mostraron valores apenas superiores al 50 %. La exactitud y la especificidad fueron consistentemente bajas, con una precisión media de 0,531 para la clase HU-NoIA y 0,466 para la HU-IA, lo que sugirió una alta tasa de falsos positivos.

El F1-Score, que combina precisión y sensibilidad, también reflejó este bajo rendimiento, indicando que el modelo tuvo dificultades significativas para distinguir entre HU implementables con IA y aquellas que no lo eran. Adicionalmente, la baja desviación estándar observada en todas las métricas significó que el modelo de ChatGPT4o fue consistentemente deficiente en su desempeño, lo que no presentó fluctuaciones importantes entre las diferentes corridas. Este nivel de consistencia en un rendimiento subraya, nuevamente, la necesidad de construir un modelo propio optimizado y entrenado para la tarea específica en cuestión.

**Tabla 3.** *Resultados de la clasificación con ChatGPT4o*

| **Mediana** | | | | | |
|---|---|---|---|---|---|
| **Clase** | **Exactitud** | **Precisión** | **Sensibilidad** | **Especificidad** | **F1-Score** |
| 0 - HU-NoIA | 0,492 | 0,534 | 0,493 | 0,492 | 0,506 |
| 1 - HU-IA | 0,492 | 0,477 | 0,492 | 0,493 | 0,482 |
| **Promedio** | | | | | |
| **Clase** | **Exactitud** | **Precisión** | **Sensibilidad** | **Especificidad** | **F1-Score** |
| 0 - HU-NoIA | 0,498 | 0,531 | 0,498 | 0,498 | 0,513 |
| 1 - HU-IA | 0,498 | 0,466 | 0,498 | 0,498 | 0,481 |
| **Desviación** | | | | | |
| **Clase** | **Exactitud** | **Precisión** | **Sensibilidad** | **Especificidad** | **F1-Score** |
| 0 - HU-NoIA | 0,026 | 0,042 | 0,024 | 0,029 | 0,030 |
| 1 - HU-IA | 0,026 | 0,037 | 0,029 | 0,024 | 0,030 |

Por último, en el caso de ChatGPT4o -Mini, se trata de un modelo de lenguaje de menor escala. Este estudio analizó su eficacia para la clasificación automática de historias de usuario en un Smart Product Backlog (ver Tabla 4). Los resultados indicaron que, para la clase HU-NoIA, tuvo un rendimiento aceptable con una mediana de precisión de 0,752 y un F1-Score promedio de 0,648. Sin embargo, para la clase HU-IA, el rendimiento fue notablemente inferior con una mediana de precisión de 0,395 y un F1-Score promedio de 0,477, lo que sugiere dificultades en la clasificación correcta de esta clase.

**Tabla 4.** *Resultados de la clasificación con ChatGPT-Mini*

| **Mediana** | | | | | |
|---|---|---|---|---|---|
| **Clase** | **Exactitud** | **Precisión** | **Sensibilidad** | **Especificidad** | **F1-Score** |
| 0 - HU-NoIA | 0,571 | 0,752 | 0,557 | 0,603 | 0,636 |
| 1 - HU-IA | 0,571 | 0,395 | 0,603 | 0,557 | 0,477 |
| **Promedio** | | | | | |
| **Clase** | **Exactitud** | **Precisión** | **Sensibilidad** | **Especificidad** | **F1-Score** |
| 0 - HU-NoIA | 0,579 | 0,768 | 0,560 | 0,623 | 0,648 |
| 1 - HU-IA | 0,579 | 0,388 | 0,623 | 0,560 | 0,477 |
| **Desviación** | | | | | |
| **Clase** | **Exactitud** | **Precisión** | **Sensibilidad** | **Especificidad** | **F1-Score** |
| 0 - HU-NoIA | 0,044 | 0,054 | 0,031 | 0,080 | 0,038 |
| 1 - HU-IA | 0,044 | 0,052 | 0,080 | 0,031 | 0,059 |

El análisis de la desviación estándar reveló que, aunque el modelo fue relativamente consistente, hubo más variabilidad en la especificidad y sensibilidad de la clase HU-IA, lo que podría indicar problemas en la identificación precisa de casos implementables con IA. Esta diferencia de rendimiento entre las clases mostró un posible sesgo hacia la clase no implementables con IA. En general, aunque ChatGPT4o-Mini ofreció un rendimiento aceptable para la clase no implementables con IA, sus dificultades con la clase implementables con IA subrayaron la necesidad de ajustar el modelo o explorar otras técnicas.

En este trabajo se evaluó si los Modelos de Lenguaje de Gran Escala (LLM) podían ser utilizados en tareas de clasificación automática de historias de usuario (HU) sin necesidad de una anotación previa. Sin embargo, su eficacia depende de varios factores, como la precisión del modelo en la interpretación del contexto de las HU y la calidad de los *prompts* utilizados para guiar el proceso de clasificación. Aunque los LLM pueden ofrecer un punto de partida valioso, es crucial el trabajo presentado, ya que evaluó el rendimiento comparando los resultados de las clasificaciones. Comparando los resultados de las clasificaciones e identificando algunas limitaciones en la clasificación, por lo tanto, se responde la pregunta de investigación P1.

Respecto a la respuesta a la pregunta de investigación P2: ¿Qué tan precisas son las clasificaciones de los LLM (IA, no IA) en HU? Las clasificaciones de los LLM en tareas de clasificación binaria de HU como IA o no IA mostraron un rendimiento moderado. Si bien estos modelos fueron capaces de distinguir entre categorías en términos generales, la precisión podría variar significativamente dependiendo del LLM específico utilizado y de la calidad del *prompt*. En los resultados obtenidos, las métricas de precisión y exactitud de algunos LLM apenas superaron la predicción aleatoria, lo que indica que, aunque son útiles para ciertas tareas, su desempeño en clasificaciones específicas como estas puede no ser suficientemente

robusto para aplicaciones críticas sin un mayor refinamiento o ajuste.

Respecto a la pregunta P3: ¿Cuáles LLM muestran la menor tasa de error en la clasificación binaria de HU? De los modelos evaluados en este contexto, la tasa de error varió entre los diferentes LLM. Según los resultados del estudio, aunque todos los modelos presentaron desafíos en la clasificación binaria de historias de usuario, ChatGPT-4o, Gemini Pro-1.5 y ChatGPT4o-Mini fueron comparados para identificar cuál de ellos ofrecía la menor tasa de error. Respecto a esto, el análisis preliminar indicó que ninguno se destacó significativamente en términos de precisión, lo que indica que se requiere una evaluación más profunda o la combinación de técnicas de ajuste para mejorar el rendimiento en esta tarea específica.

## 5. DISCUSIÓN

Se revisaron diversos artículos sobre la clasificación automática de historias de usuario utilizando Modelos de Lenguaje de Gran Escala. La revisión de literatura demostró que, aunque los LLM han evidenciado potencial en la mejora y automatización de HU, su aplicación específica en la clasificación automática aún enfrenta desafíos significativos, ya que su precisión y confiabilidad pueden variar, especialmente cuando se aplican a tareas más especializadas como la identificación de historias viables para implementación con inteligencia artificial.

Los resultados obtenidos en el estudio no fueron del todo satisfactorios. Si bien modelos como ChatGPT40, Gemini Pro-1.5 y ChatGPT4o-Mini mostraron un rendimiento aceptable en ciertas métricas, la inconsistencia y la falta de precisión en la clasificación de HU-IA y HU-NoIA revelaron limitaciones importantes. La variabilidad en los resultados, especialmente en términos de sensibilidad y especificidad, indicó que estos modelos no eran lo suficientemente robustos para ser utilizados como herramientas completamente automatizadas en un entorno ágil sin intervención humana. Esto significó que, aunque los LLM podían proporcionar una base inicial, no eran adecuados para tareas que requirieran alta precisión y consistencia.

Dada la limitada efectividad de los modelos generales evaluados, sería más beneficioso desarrollar un modelo personalizado que esté específicamente optimizado para la clasificación de historias de usuario en el contexto de aplicaciones con inteligencia artificial. Este podría ser entrenado con un dataset más relevante y utilizando técnicas de ajuste fino, lo que permitiría abordar las debilidades observadas en los modelos generales y mejorar la precisión en la identificación de historias que son viables para la implementación con IA. Este enfoque podría ofrecer una herramienta más fiable y eficiente para apoyar la toma de decisiones en proyectos de desarrollo de *software* ágiles.

## 6. CONCLUSIONES

En este estudio se evaluó el uso de Modelos de Lenguaje de Gran Escala (LLM) para la clasificación automática de historias de usuario (HU) en el contexto del Smart Product Backlog. Los resultados obtenidos demuestran que, aunque los LLM como ChatGPT-4o, Gemini Pro-1.5 y ChatGPT4o-Mini ofrecen una clasificación inicial útil, su desempeño en tareas específicas como la diferenciación entre HU implementables con IA y las que no lo son, es limitado. Las métricas de desempeño que incluyen precisión, sensibilidad y especificidad indican que estos modelos no superan de manera consistente las predicciones aleatorias, lo que pone en duda su aplicabilidad en entornos que requieren alta exactitud sin intervención humana.

Además, la variabilidad en los resultados sugiere la necesidad de enfoques más sofisticados, como la incorporación de técnicas de ajuste fino o el uso de otros modelos o métodos híbridos que combinen LLM. Estos hallazgos subrayan la importancia de no depender exclusivamente de LLM para tareas críticas de clasificación, sino de emplearlos como una herramienta complementaria en un marco más amplio de análisis y toma de decisiones.

Como trabajo futuro, se propone explorar la integración de otras técnicas de aprendizaje de máquina como clasificación logística, árboles de decisión, Random Forest, Support Vector Machine o Deep Learning, y el uso completo del dataset Mendeley para mejorar el rendimiento de los modelos.

Además, sería valioso investigar el desarrollo de métodos híbridos que combinen la capacidad de comprensión contextual de los LLM con modelos específicos de dominio a los cuales se les incluyan criterios para realizar la clasificación de las HU y optimizar la precisión en tareas de clasificación.

## ATRIBUCIÓN DE AUTORÍA

**Mauricio Gaona Cuevas:** Conceptualización, metodología, preparación de datos, metodología, validación, escritura del artículo original, revisión y edición.

**Victor Bucheli Guerrero:** Conceptualización, preparación de datos, metodología, software, validación,revisión y edición.

**Fredy H. Vera Rivera:** preparación de datos, metodología, escritura, validación, revisión y edición.

## REFERENCIAS

[1] K. Beck, M. Fowler, Planning Extreme Programming. Addison Wesley, 2001.

[2] T. Sedano, P. Ralph, C. Peraire, "The Product Backlog," in *International Conference on Software Engineering*, IEEE Computer Society, Montreal, Canada, 2019, pp. 200-211. https://doi.org/10.1109/ICSE.2019.00036

[3] C. A. Dos Santos, K. Bouchard, F. Petrillo, "AI-Driven User Story Generation," in *International Conference on Artificial Intelligence, Computer, Data Sciences, and Applications (ACDSA)*, Victoria, Seychelles, 2024. https://doi.org/10.1109/ACDSA59508.2024.10467677

[4] K. Kaur and P. Kaur, "The application of AI techniques in requirements classification: a systematic mapping," *Artificial Intelligence Review*., vol. 57(3), pp. 1-48, 2024. https://doi.org/10.1007/S10462-023-10667-1

[5] S. Arulmohan, M. J. Meurs, S. Mosser, "Extracting Domain Models from Textual Requirements in the Era of Large Language Models," in *ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, Suecia, 2023, pp. 580-587. https://doi.org/10.1109/MODELS-C59198.2023.00096

[6] Z. Zhang, M. Rayhan, T. Herda, M. Goisauf, P. Abrahamsson, "LLM-Based Agents for Automating the Enhancement of User Story Quality: An Early Report," in *Agile Processes in Software Engineering and Extreme Programming*, Germany, 2024, pp. 117-126. https://doi.org/10.1007/978-3-031-61154-4_8

[7] T. Rahman, Y. Zhu, "Automated User Story Generation with Test Case Specification Using Large Language Model," in *Arxiv-Software Engineering*, 2024. https://arxiv.org/abs/2404.01558v1

[8] P. Chuor, A. Ittoo, S. Heng, "User Story Classification with Machine Learning and LLMs," in *Lecture Notes in Computer Science*. Berlin, Germany: Springer Science and Business Media, 2024, pp. 161-175. https://doi.org/10.1007/978-981-97-5492-2_13

[9] J. Hong *et al.*, "Decoding Compressed Trust: Scrutinizing the Trustworthiness of Efficient LLMs Under Compression," in *Arxiv-Computation and Language*, 2024. https://arxiv.org/abs/2403.15447v3.

[10] L. Sun et al., "TrustLLM: Trustworthiness in Large Language Models," *Arxiv-Computation and Language*, 2024. https://arxiv.org/abs/2401.05561v4

[11] B. Kumar, U. K. Tiwari, D. C. Dobhal, "Classification of NFR based Importance Level of User Story in Agile Software Development", in *9th International Conference on Signal Processing, Communications and Computing*, India, 2023, pp. 264-268. https://ieeexplore.ieee.org/document/10441284

[12] J. Liu *et al.*, "Rainier: Reinforced Knowledge Introspector for Commonsense Question Answering," in *Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates, 2022, pp. 8938-8958. https://doi.org/10.18653/v1/2022.emnlp-main.611

[13] F. Dalpiaz, "Requirements data sets (user stories)", *Mendeley Data*, vol. 1, e8, 2018. https://doi.org/10.17632/7ZBK8ZSD8Y.1

## ANEXO 1. MÉTRICAS USADAS PARA LA EVALUACIÓN

| Métrica | Definición | Fórmula |
|---|---|---|
| Exactitud | La exactitud es una métrica comúnmente utilizada para evaluar el desempeño de un modelo de clasificación. Representa la proporción de predicciones correctas realizadas por el modelo. | Exactitud = (Verdaderos Positivos + Verdaderos Negativos) / (Muestras Totales) |
| Pérdida | La pérdida es una medida del error entre los valores predichos y los valores reales. Se utiliza para optimizar el proceso de entrenamiento de un modelo de aprendizaje automático. En la clasificación binaria utilizando regresión logística, la función de pérdida es, a menudo, la entropía cruzada. y_pred se refiere a los valores predichos por el modelo de aprendizaje automático e y_true representa los valores reales o verdaderos para los datos de entrada dados. | Pérdida = -(suma(y_true * log(y_pred)) + (1 - y_true) * log(1 - y_pred)) |
| Precisión | La precisión mide la proporción de predicciones positivas que son realmente correctas. Se calcula como el número de Verdaderos Positivos dividido por el número total de predicciones positivas (Verdaderos Positivos + Falsos Positivos). | Precisión = Verdaderos Positivos / (Verdaderos Positivos + Falsos Positivos) |
| Recall | El recall mide la proporción de positivos reales que son correctamente identificados por el modelo. Se calcula como el número de Verdaderos Positivos dividido por el número total de positivos reales (Verdaderos Positivos + Falsos Negativos). | Recall = Verdaderos Positivos / (Verdaderos Positivos + Falsos Negativos) |
| F1-Score | El F1-score es una media armónica de precisión y recall, proporcionando una medida balanceada de ambos aspectos del desempeño de un modelo. | F1-Score = 2 * (Precisión * Recall) / (Precisión + Recall) |
| Área Bajo la Curva ROC (AUC) | El AUC se calcula a partir de la curva ROC, que es una gráfica de la tasa de verdaderos positivos (Recall) contra la tasa de falsos positivos (FP = Falsos Positivos / (Falsos Positivos + Verdaderos Negativos)) para diferentes umbrales de decisión. No tiene una fórmula directa simple, pero puede calcularse utilizando métodos como la regla del trapecio. | |
| Especificidad | La especificidad mide la proporción de negativos reales que son correctamente identificados por el modelo. Se calcula como el número de Verdaderos Negativos dividido por el número total de negativos reales (Verdaderos Negativos + Falsos Positivos). | Especificidad = Verdaderos Negativos / (Verdaderos Negativos + Falsos Positivos) |