EDUCACIÓN, CREATIVIDAD E INTELIGENCIA ARTIFICIAL: NUEVOS HORIZONTES PARA EL APRENDIZAJE. ACTAS DEL VIII CONGRESO INTERNACIONAL SOBRE APRENDIZAJE, INNOVACIÓN Y COOPERACIÓN, CINAIC 2025

María Luisa Sein-Echaluce Lacleta, Ángel Fidalgo Blanco y Francisco José García Peñalvo (coords.)

1º Edición. Zaragoza, 2025

Edita: Servicio de Publicaciones. Universidad de Zaragoza.

Servicio de Publicaciones Universidad Zaragoza

EBOOK ISBN 978-84-10169-60-9

DOI 10.26754/uz.978-84-10169-60-9

Referencia a esta obra:

Sein-Echaluce Lacleta, M.L., Fidalgo Blanco, A. & García-Peñalvo, F.J. (coords.) (2025). *Educación, Creatividad e Inteligencia Artificial: nuevos horizontes para el Aprendizaje. Actas del VIII Congreso Internacional sobre Aprendizaje, Innovación y Cooperación. CINAIC 2025 (11-13 de Junio de 2025, Madrid, España)*. Zaragoza. Servicio de Publicaciones Universidad de Zaragoza. DOI 10.26754/uz.978-84-10169-60-9

Nuevos métodos de evaluación continua asistidos por Modelos de Lenguaje e Inteligencia Artificial Generativa

New Continuous Assessment Methods Assisted by Language Models and Generative Artificial Intelligence

Pablo Manuel Vigara-Gallego, Angel Garcia-Beltrán, Ascensión Lopez-Vargas, Javier Rodriguez-Vidal
pm.vigara@upm.es, agarcia@etsii.upm.es, a.lvargas@upm.es, javier.rodriguez.vidal@upm.es

Departamento de Automática, Ingeniería Eléctrica y Electrónica e Informática Industrial
Universidad Politécnica de Madrid
Madrid, España

Resumen- El auge de los Grandes Modelos de Lenguaje (LLMs) ha transformado el ámbito educativo, ofreciendo nuevas oportunidades tanto para estudiantes como para docentes. Este estudio explora el uso de la inteligencia artificial generativa (IAG) para evaluar ejercicios de programación y proporcionar retroalimentación personalizada a los alumnos tras la entrega de sus trabajos en una plataforma web segura, diseñada para garantizar la integridad académica. La aplicación implementa estrategias de aprendizaje por pocos ejemplos (*few-shot learning*) y rúbricas estructuradas para mejorar la precisión de la calificación. Los hallazgos sugieren que, si bien los LLMs aún requieren supervisión humana, pueden generar evaluaciones precisas y detalladas, reduciendo la carga docente y proporcionando retroalimentación útil para los estudiantes. La integración de estas tecnologías en la educación permite mejorar la escalabilidad y accesibilidad de la evaluación en cursos de programación, optimizando el proceso de aprendizaje y garantizando estándares de calidad en la formación académica.

Palabras clave: Evaluación Automatizada, Grandes Modelos de Lenguaje, Inteligencia Artificial Generativa, Retroalimentación personalizada, Tecnología Educativa.

Abstract- The rise of Large Language Models (LLMs) has transformed the educational landscape, offering new opportunities for both students and educators. This study explores the use of generative artificial intelligence (GAI) to assess programming exercises and provide personalized feedback to students after submitting their work on a secure web platform designed to ensure academic integrity. The application implements few-shot learning strategies and structured rubrics to improve grading accuracy. Findings suggest that while LLMs still require human supervision, they can generate precise and detailed assessments, reducing the workload for educators and providing valuable feedback to students. The integration of these technologies in education enhances the scalability and accessibility of assessments in programming courses, optimizing the learning process and ensuring high-quality academic standards.

Keywords: Automated assessment, Large Language Models, Generative AI, Personalzed feedback, Educational technology.

1. INTRODUCCIÓN

La evaluación de código en entornos educativos representa un desafío constante para los docentes, quienes deben garantizar una calificación precisa, justa y eficiente, especialmente en cursos con un alto número de estudiantes. Tradicionalmente, este proceso ha requerido una corrección manual que implica una carga de trabajo significativa, así como la posibilidad de subjetividad en la evaluación. La irrupción de los Modelos de Lenguaje de Gran Escala (LLMs) y la Inteligencia Artificial Generativa ha abierto nuevas oportunidades para automatizar y optimizar este proceso, permitiendo evaluaciones más rápidas, consistentes y con retroalimentación detallada para los estudiantes (Smolic et al., 2024 y Balse et al., 2023).

Este trabajo presenta el desarrollo inicial de una aplicación web diseñada para la evaluación automatizada de ejercicios de programación, utilizando LLMs avanzados como GPT-3.5 (OpenAI, 2023), GPT-4o (OpenAI, 2024) y GPT-4o-mini. La herramienta integra estrategias de aprendizaje por pocos ejemplos (*few-shot learning*) y rúbricas estructuradas para mejorar la precisión de la calificación y proporcionar retroalimentación cualitativa. Además, se ha implementado en un entorno seguro basado en Laravel 11, con mecanismos de supervisión para garantizar la integridad académica.

Las primeras pruebas realizadas han comparado las evaluaciones generadas por estos modelos con las calificaciones de docentes humanos, analizando su correlación y precisión en la detección de errores de código. Si bien los resultados sugieren que los LLMs aún requieren supervisión humana, han demostrado un alto potencial para reducir la carga docente y mejorar la escalabilidad del proceso de evaluación. En este estudio se presentan los primeros avances logrados y se proponen mejoras para su futura implementación a mayor escala en entornos educativos.

2. CONTEXTO Y DESCRIPCIÓN

A. Necesidad del desarrollo de la aplicación

Garantizar evaluaciones justas, objetivas y con retroalimentación detallada en ejercicios de programación representa un desafío en entornos educativos, especialmente en cursos con muchos estudiantes. El proceso, generalmente manual y demandante, implica una gran carga de trabajo para los docentes, lo que dificulta ofrecer comentarios personalizados y oportunos. Sin embargo, estos comentarios son más valiosos para los alumnos que una simple calificación numérica sin contexto. En la actualidad, las tendencias en

didáctica de la programación abogan por reducir la relevancia del examen final, promoviendo en su lugar un proceso formativo continuo que favorezca el desarrollo de competencias sólidas en el estudiantado. No obstante, esta orientación ha sido objeto de cuestionamientos debido a preocupaciones relacionadas con la integridad académica y a las dificultades que enfrentan los docentes para constatar la autoría legítima de los trabajos presentados (Chowdhury, 2019).

Con la creciente adopción de la Inteligencia Artificial Generativa y los Modelos de Lenguaje de Gran Escala, surge la oportunidad de automatizar la evaluación de código sin comprometer la calidad del aprendizaje. Implementar un sistema basado en LLMs permite agilizar la corrección de ejercicios, reducir la carga docente y mejorar la precisión y coherencia de las evaluaciones. Este desarrollo responde a la necesidad de integrar tecnologías avanzadas en la educación para optimizar la enseñanza de la programación y mejorar la experiencia de los estudiantes. Propuestas que, aunque actualmente pueden parecer avanzadas o futuristas, tienen el potencial de integrarse en la práctica docente en los próximos años, siguiendo una trayectoria similar a la adopción de aulas virtuales y entornos de aprendizaje en línea.

B. Objetivos

El principal objetivo de este trabajo es diseñar e implementar una aplicación web que utilice Modelos de Lenguaje de Gran Escala para la evaluación automatizada de ejercicios de programación, garantizando retroalimentación precisa y útil para los estudiantes. Para lograr esto, se plantean los siguientes objetivos específicos:

1. Desarrollar un sistema capaz de calificar ejercicios de programación de manera automática utilizando modelos grandes de lenguaje comerciales como ChatGPT, Gemini, Llama…
2. Integrar estrategias de aprendizaje por pocos ejemplos (*few-shot learning*) y rúbricas estructuradas para mejorar la precisión de la evaluación y garantizar la calidad de esta.
3. Comparar las calificaciones generadas por los modelos con las evaluaciones realizadas por docentes reales.
4. Implementar un entorno seguro basado en tecnologías web modernas que garantice la integridad académica mediante supervisión de la actividad del usuario.

C. Contexto y público objetivo

Este trabajo se desarrolla en el ámbito de la enseñanza universitaria, específicamente en cursos de programación dentro de diferentes carreras de ingeniería y computación. El público objetivo de esta aplicación está compuesto por:

- Estudiantes de ingeniería y ciencias de la computación, quienes recibirán una retroalimentación más detallada y objetiva sobre sus ejercicios de programación.
- Docentes y evaluadores, quienes podrán reducir la carga de trabajo y disponer de una herramienta de apoyo que facilite la revisión de código y que les asista en las labores de corrección y evaluación de los trabajos entregados.
- Instituciones educativas, que podrán implementar soluciones escalables para mejorar la enseñanza de la programación y optimizar el uso de recursos en la evaluación, así como capacitar de tiempo extra a los docentes para realmente mejorar el proceso de enseñanza-aprendizaje.

El desarrollo y prueba de la aplicación se ha llevado a cabo en un entorno universitario con estudiantes de ingeniería, quienes han participado en la fase inicial de validación del sistema.

D. Metodología

El trabajo se ha llevado a cabo mediante un enfoque experimental y diseñado desde el contacto directo con los alumnos que serán los primeros usuarios y beneficiados del producto. Para ello se han desarrollado las siguientes etapas:

1. Definición del modelo de evaluación: Tras una revisión del estado del arte en las mejores prácticas de evaluación se concluyó que era necesario el uso de rúbricas estructuradas y criterios de corrección precisos para utilizar por el sistema de evaluación y que así pueda mejorarse la explicabilidad de los modelos y comprender las decisiones. De esta forma se garantizarían también los derechos de los alumnos a la hora de comprender sus resultados (Chowdhury, 2019).
2. Desarrollo de la aplicación: Se implementó una plataforma web desde un servidor de la universidad que dispone de un compilador mediante un *framework* de desarrollo basado en Laravel 11 en el que se integran diferentes modelos de lenguaje comerciales para poder usarlos dentro del sistema de información. Esta tecnología podría desplegarse como módulos de cualquier Sistema de Gestión del Aprendizaje (LMS) aunque para el total manejo de todas las funciones del producto se decidió una conexión por API para compartir los datos y mantenerlos actualizados.
3. Pruebas con estudiantes: Se llevaron a cabo pruebas con prácticas, exámenes y ejercicios de cursos anteriores propuestos en diferentes grados, por diferentes profesores, resueltos por diferentes alumnos y corregidos como referencia por diferentes profesores con la intención de garantizar la aleatoriedad de los resultados y de las pruebas. Una vez se cargaron de forma artificial todas estas entregas se pudieron comparar los resultados automáticos con los originales que habían estado ocultos (Kiesler et al., 2023).
4. Análisis de los resultados: Con los datos generados de nuevo, se pudieron comparar de forma sencilla con los resultados originales de la evaluación realiza por profesores reales.
5. Mejoras y ajustes: En base a los resultados obtenidos previamente, se identificaron ajustes para optimizar el sistema antes de pruebas reales con alumnos en una escala controlada y posteriormente a gran escala de forma industrial.

E. Metodología

Para el uso efectivo de la inteligencia artificial generativa en esta tarea, es necesario seguir las técnicas recomendades que aquí se han implementado:

1. Aprendizaje por pocos ejemplos (*few-shot learning*): Previo al proceso de evaluación, se proporcionan ejemplos representativos con el fin de mejorar la comprensión del modelo evaluativo y la precisión en la valoración. Asimismo, estos ejemplos permiten calibrar el nivel de exigencia y establecer referencias claras que orienten el desarrollo del resto de la evaluación. Normalmente la

generación de una respuesta y está condicionada por un contexto $C = \{(x_1, y_1), \ldots, (x_k, y_k)\}$ que incluye k ejemplos, por lo que con una entrada x el proceso se puede definir como:

$$LLM(y|C, x) = \prod_{t=1}^{T} p(y_t|C, (x_i, y_i)_{\forall i<t})$$

2. Rubricas estructuradas: Se diseñaron rubricas por profesores para la evaluación y también rubricas automáticas diseñadas por el sistema para poder ajustar las calificaciones y así estandarizar el proceso de evaluación (Hung & Hoi, 2010).
3. Comparación con evaluaciones humanas: Se realizaron análisis estadísticos para evaluar la fiabilidad de las calificaciones generadas por los modelos en relación con la evaluación docente (Anishka et al., 2024).

3. Resultados

En esta sección se describen los efectos de la implementación de la aplicación web para la evaluación automatizada de ejercicios de programación, la forma de cuantificar dicho impacto y los resultados obtenidos tras varias pruebas piloto con estudiantes de ingeniería.

A. Impacto y forma de evaluación

El impacto principal del sistema radica en la reducción de la carga docente y la mejora de la objetividad en la corrección de ejercicios de programación. Para evaluar este impacto, se han seguido los siguientes pasos:

1. Comparación entre calificaciones de la IA y evaluaciones humanas. Se recopilaron las notas dadas por docentes y las generadas por diferentes modelos para un conjunto significativo de ejercicios. Se estudio la distribución de ambas calificaciones para analizar la consistencia y precisión de la IA frente al criterio docente.

   Como se aprecia en la Figura 1 la similitud de las distribuciones de los datos los diferentes métodos:

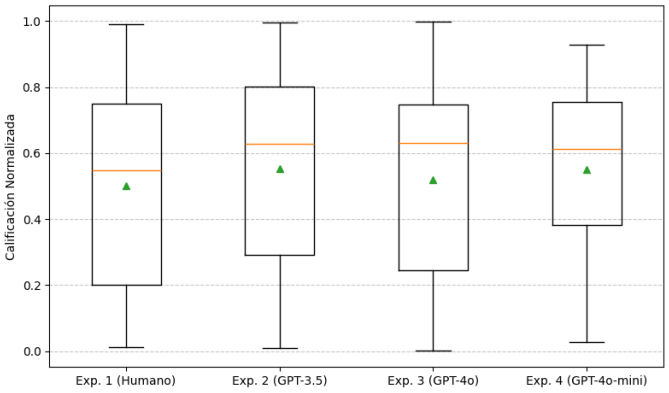

Figura 1. Comparativa de distribuciones de calificaciones dependiendo del agente corrector

2. Medición de la satisfacción de estudiantes y docentes. Se llevaron a cabo grupos de enfoque breves para obtener la percepción de los estudiantes respecto a la utilidad y claridad de la retroalimentación proporcionada por la aplicación. Los docentes valoraron el tiempo invertido en comparación con la corrección tradicional y la calidad de las evaluaciones generadas.
3. Análisis estadístico de la distribución de calificaciones. Se calculó la desviación estándar y se realizaron análisis de varianza para ver si las notas otorgadas por la IA seguían un patrón coherente con las evaluaciones manuales.

B. Impacto y forma de evaluación

1. Alta correlación con la evaluación humana. Las notas generadas por los modelos comerciales *GPT* de *OpenAI* mostraron una correlación significativa con las de los docentes. En la mayoría de los casos, se observó que los modelos penalizaron de forma similar los errores de lógica y la falta de cumplimiento de requisitos, aunque tendían a ser ligeramente más indulgentes con aspectos relacionados con la organización y la nomenclatura del código (Figura 2).

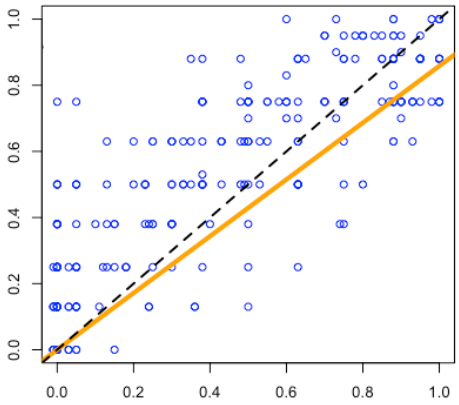

Figura 2. Correlación entre notas reales (y) y notas estimadas por el modelo (x) (R^2=0,87, y=0,90x)

2. Reducción en el tiempo de corrección. Los docentes participantes reportaron una disminución sustancial en el tiempo dedicado a evaluar los ejercicios de programación. El sistema automatizado proporcionó una primera calificación con retroalimentación que los profesores únicamente revisaron y ajustaron mínimamente en caso de discrepancias.
3. Retroalimentación más detallada para el estudiantado. La mayoría de los estudiantes resaltó la utilidad de recibir comentarios específicos sobre los puntos de mejora y la posible optimización de su código. Esta información, generada mediante rúbricas estructuradas y *few-shot learning*, incrementó la motivación por corregir y profundizar en los errores.
4. Escalabilidad y menor costo de implementación. El sistema demostró ser escalable: tras la implementación inicial, se realizaron pocas adaptaciones para corregir incidencias relacionadas con ejercicios muy específicos o con código no compilable. El coste asociado a la ejecución de modelos de menor tamaño (GPT-4o-mini) fue considerablemente más bajo que en versiones de mayor tamaño, con resultados muy similares en precisión.

En conjunto, los resultados evidencian que la aplicación web logra agilizar y mejorar la evaluación continua de ejercicios de programación, manteniendo una alta concordancia con las valoraciones humanas y brindando retroalimentación útil para los estudiantes. Las mejoras propuestas en la sección de conclusiones permitirán refinar el sistema e integrarlo a mayor escala en el proceso de enseñanza-aprendizaje.

4. Conclusiones

La aplicación propuesta para la evaluación continua de ejercicios de programación ha demostrado su viabilidad y efectividad en las fases de prueba. A continuación, se presentan los principales aspectos relacionados con la sostenibilidad, la transferibilidad del proyecto a otros contextos y recomendaciones para su aplicación.

A. Sostenibilidad del trabajo

El enfoque adoptado resulta sostenible tanto a nivel institucional como operativo. Por un lado, el uso de LLMs se ha mostrado eficiente, especialmente en versiones de menor tamaño (por ejemplo, GPT-4o-mini), lo que disminuye los costos de procesamiento. Por otro lado, la automatización en la corrección de código reduce la carga docente, liberando recursos y permitiendo a los profesores dedicar más tiempo a la enseñanza y al diseño de actividades formativas. Además, la arquitectura propuesta en Laravel 11 es lo suficientemente flexible para escalar y alojar futuras mejoras o integraciones con otros sistemas de gestión del aprendizaje.

B. Transferibilidad a otros contextos

El modelo de evaluación desarrollado no se limita a un único lenguaje de programación ni a un entorno específico. Con ligeros ajustes en la definición de rúbricas y criterios de calificación, puede adaptarse a otras asignaturas de ingeniería o incluso a disciplinas que requieran la revisión de trabajos extensos, como ensayos o proyectos escritos. De igual modo, la infraestructura de la plataforma, basada en principios de seguridad y seguimiento de la actividad del usuario, es aplicable en otros contextos educativos que demanden verificación de integridad académica. Esto supone un alto potencial de adopción por parte de diferentes instituciones y programas formativos con necesidades de evaluación a gran escala.

C. Recomendaciones de aplicación

1. Incorporar supervisión docente permanente: A pesar de la alta precisión de los modelos, se recomienda que los profesores revisen los resultados y ajusten las calificaciones cuando sea necesario. De este modo, se garantiza la confiabilidad final del proceso de evaluación y se aprovecha el componente humano para casos complejos o ambiguos.
2. Crear rúbricas personalizadas por asignatura: Para asegurar la coherencia en la calificación, es fundamental diseñar rúbricas específicas que tengan en cuenta los objetivos de cada curso, así como los diferentes niveles de dificultad de los ejercicios de programación.
3. Ampliar el conjunto de ejemplos para *few-shot learning*: Cuantos más ejemplos con correcciones detalladas se proporcionen al modelo, mayor será la precisión de sus evaluaciones. Esto es especialmente relevante en contextos con una gran diversidad de ejercicios y estilos de codificación.
4. Formación inicial del profesorado: Para que la implementación sea exitosa, es importante capacitar a los docentes en el uso de la plataforma y la interpretación de los resultados generados por la IA.

En suma, la propuesta aporta una estrategia sostenible y escalable para la evaluación continua, respaldada por tecnología de vanguardia y sólidas prácticas de integridad académica. Con la adopción de estas recomendaciones, la herramienta tiene el potencial de consolidarse como un recurso clave en la formación de futuros profesionales en ámbitos de ingeniería, ciencias de la computación y otras áreas.

AGRADECIMIENTOS

Este trabajo ha sido financiado por la Universidad Politécnica de Madrid a través de la Convocatoria 2024-2025 de Proyectos de Innovación Educativa (Implementación de Asistentes Virtuales Basados en Inteligencia Artificial para le Evaluación Personalizada y Rápida en Asignaturas de Programación, código de proyecto IE25.0501)

REFERENCIAS

Anishka, Atharva Mehta, Nipun Gupta, Aarav Balachandran, Dhruv Kumar, and Pankaj Jalote. 2018. Can ChatGPT Play the Role of a Teaching Assistant in an Introductory Programming Course?. In Proceedings of Innovation and Technology in Computer Science Education (ITiCSE 2024). ACM, New York, NY, USA, 8 pages.

Balse, R., Valaboju, B., Singhal, S., Madathil Warriem, J., & Prasad, P. (2023). *Investigating the potential of GPT-3 in providing feedback for programming assessments.* In *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1 (ITiCSE 2023)* (pp. 292–298). Association for Computing Machinery.

Chowdhury, F. (2019). *Application of rubrics in the classroom: A vital tool for improvement in assessment, feedback, and learning.* International Education Studies, **12**(1), 61–68.

Hung, C. M., & Hoi, L. T. (2010). *Maximizing the benefits of the use of rubrics to promote assessment for learning in inquiry study.* Educational Practice and Theory, **32**(2), 5–21.

Kiesler, N., Lohr, D., & Keuning, H. (2023). *Exploring the potential of large language models to generate formative programming feedback.* In *2023 IEEE Frontiers in Education Conference (FIE)* (pp. 1–5). IEEE Computer Society.

OpenAI. (2023, August 22). *GPT-3.5 Turbo fine-tuning and API updates: Developers can now bring their own data to customize GPT-3.5 Turbo for their use cases.* Retrieved January 4, 2025, from https://openai.com

OpenAI. (2024, May 13). *Hello GPT-4o.* Retrieved January 4, 2025, from https://openai.com

Smolic, E., Pavelic, M., Boras, B., Mekterovic, I., & Jagust, T. (2024). *LLM generative AI and students' exam code evaluation: Qualitative and quantitative analysis.* In *Proceedings of the 2024 47th MIPRO ICT and Electronics Convention (MIPRO)* (pp. 1505–1510). Opatija, Croatia, May 20–24, 2024. University of Zagreb, Faculty of Electrical Engineering and Computing, Department of Applied Computing; Šibenik University of Applied Sciences, IEEE.