Revista Colombiana de Computación
Vol. 26, No.2 Julio – Diciembre 2025, pp. 53-70
e-ISSN: 2539-2115, https://doi.org/10.29375/25392115.5188

REVISTA COLOMBIANA DE COMPUTACIÓN

Grandes modelos lingüísticos en la ingeniería de requerimientos: Una revisión de la literatura

Large language models in requirements engineering: A literature review

Mauro José Pacchiotti[1], Luciana Ballejos[1], Mariel Ale[1]

[1] Universidad Tecnológica Nacional, Facultad Regional Santa Fe – FRSF, Centro de I+D de Ingeniería en Sistemas de Información – CIDISI, Santa Fe, Argentina

mpacchiotti@frsf.utn.edu.ar, lballejos@frsf.utn.edu.ar, male@frsf.utn.edu.ar

(Recibido: 23 Septiembre 2024; aceptado: 27 Diciembre 2025, publicado en línea: 31 Diciembre 2025)

Resumen. Existen diferentes propuestas en la literatura sobre la aplicación de modelos y herramientas de Procesamiento del Lenguaje Natural – PNL para abordar las actividades y desafíos que surgen en el proceso de Ingeniería de Requerimientos – IR. En los últimos años, la Inteligencia Artificial – IA generativa implementada mediante Grandes Modelos de Lenguaje – *LLMs* ha ganado un importante reconocimiento debido a las notables mejoras presentadas en las tareas de PNL. Este artículo presenta un Mapeo Sistemático de la Literatura – MSL que reúne las investigaciones que proponen el uso de *LLM* para mejorar el proceso y resolver problemas presentes en la IR. Los resultados obtenidos muestran propuestas alentadoras dirigidas principalmente a la creación de diferentes tipos de modelos y clasificación de requerimientos. Sin embargo, todavía necesitan más desarrollo y validación empírica para ser ampliamente aceptadas y aplicadas en entornos de desarrollo de software. En consecuencia, esta situación denota la necesidad de seguir investigando y profundizando en posibles aplicaciones de los *LLMs* en IR.

Palabras clave: grandes modelos de lenguaje, ingeniería de requerimientos, IA generativa.

Abstract. There are several proposals in the literature on the application of Natural Language Processing – NLP to address activities and challenges in requirements engineering – RE. In recent years, Generative Artificial Intelligence – AI implemented with Large Language Models – LLM has gained great recognition due to the improvements contributed to NLP tasks. This work proposes a systematic literature review – SLR to collect research that presents some use of LLMs to solve problems and improve the RE process. The results show promising proposals aimed mainly at different model creation and requirements classification tasks. However, these proposals need more development and empirical validation to be widely accepted and applied in software development environments. Therefore, it is necessary to continue researching the applications of LLM in the RE process.

Keywords: large language models, requirements engineering, generative AI.

1 Introducción

La IR es la disciplina que agrupa las actividades durante las primeras etapas del desarrollo de software. Se encarga de lograr la Especificación de Requerimientos de Software – ERS a partir del contacto con el dominio que comprende la organización, usuarios y todo interesado en el desarrollo, así como también las restricciones tecnológicas y organizacionales existentes.

El desarrollo de estas actividades implica, en la mayoría de los casos, el trabajo con interesados que desconocen del proceso de desarrollo de software. Por este motivo es necesario procesar distintos tipos de artefactos expresados en lenguaje natural, tales como documentos de la organización, entrevistas y estándares, entre otros, con el fin de capturar los requerimientos y restricciones que delimitarán y guiarán el desarrollo del producto de software. Esta dependencia de textos en lenguaje natural que incorporan grandes desafíos, tales como la ambigüedad presente en los mismos, hace que desde ya hace tiempo exista un creciente interés y hayan surgido trabajos que estudian el uso de diferentes técnicas de Procesamiento de Lenguaje Natural – PLN para mejorar el desarrollo de las actividades del proceso de IR (Gramajo y otros, 2020; Rahimi y otros, 2020).

Con el avance en los últimos años de la Inteligencia Artificial – IA generativa aplicada con modelos grandes de lenguaje –*LLMs*, por su sigla en inglés *Large Language Models*– se han logrado significativas mejoras en

©2025 Universidad Autónoma de Bucaramanga. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY-NC-SA 4.0) license (https://creativecommons.org/licenses/by-nc-sa/4.0/)

diferentes áreas del PLN, tales como la clasificación, generación, traducción y resumen de textos, así como la capacidad para aplicar estas técnicas a grandes corpus (Hadi y otros, 2025; Raiaan y otros, 2024). Con estas mejoras, los *LLMs* han demostrado tener un gran valor práctico en una variedad de dominios que tienen dependencia del lenguaje natural.

Este trabajo tiene como objetivo buscar, analizar y clasificar los trabajos de investigación en la temática, y luego realizar un análisis acerca de la utilización de los *LLMs* para facilitar las actividades dentro del proceso de IR.

Inicialmente se desarrolla un Mapeo Sistemático de la Literatura – MSL sobre la utilización de *LLMs* para mejorar el desarrollo de las actividades involucradas en el proceso de IR, con el fin de encontrar los trabajos de investigación que abordaron la temática. Para la realización del MSL se siguen los lineamientos del proceso propuesto por Petersen y otros (2008; 2015) para mapeos sistemáticos en el área de ingeniería de software. Luego de obtenidos los resultados, se pretende estructurar u organizar la literatura en esta área de investigación, así como también analizar los resultados para hacerlos más visibles y accesibles, y estudiar su utilidad, además de proponer adaptaciones en caso de ser necesario para diversas situaciones y dominios (Kitchenham & Charters, 2007).

El resto del trabajo se encuentra organizado de la siguiente manera. La sección 2 describe brevemente los principales conceptos referidos a los *LLMs* y la IR en virtud de establecer un marco teórico introductorio. La sección 3 presenta la metodología de investigación aplicada en el MSL. La sección 4 presenta el análisis y discusión de los resultados, mientras que la sección 6 describe conclusiones y futuras líneas de investigación.

2 Marco teórico

2.1 Modelos grandes del lenguaje

Los últimos años han sido testigos de un progreso significativo en el campo del PLN, impulsado en gran medida por el desarrollo de *LLMs*, también citados como modelos fundacionales (Bommasani y otros, 2022). Estos modelos de aprendizaje profundo –Figura 1– que comprenden grandes arquitecturas de redes neuronales, a la vez entrenados en conjuntos de datos masivos de texto y código, han demostrado ser capaces de realizar una amplia gama de tareas, desde la traducción automática hasta la generación de texto creativo.

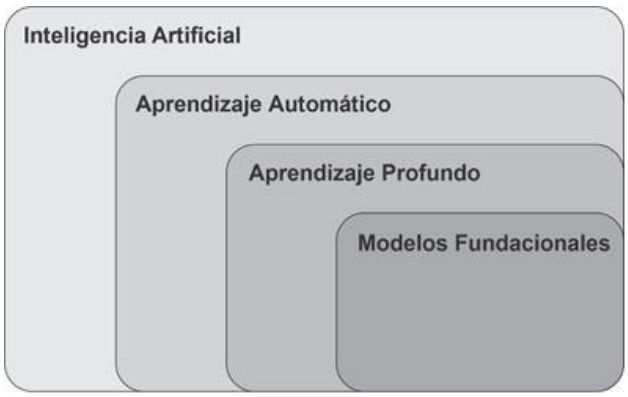

Figura 1. Clasificación de los modelos fundacionales

2.1.1 El Transformer

En 2017, el trabajo de Vaswani y otros (2017) propone el *Transformer* –Figura 2–, un modelo de aprendizaje profundo con una arquitectura dividida en dos estructuras, el codificador y el decodificador, basados en el mecanismo de atención de múltiples cabezas. Hasta ese momento los modelos para PLN se basaban principalmente en redes neuronales recurrentes (*RNN*, por su sigla en inglés *Recurrent Neural Network*) y redes neuronales convolucionales (*CNN*, por su sigla en inglés *Convolutional Neural Network*). Sin embargo, estos modelos tienen una capacidad limitada para manejar las relaciones a largo plazo en el orden de las palabras, y los gradientes se desvanecen o explotan si los modelos incrementan la cantidad de capas. A diferencia de estos modelos tradicionales que procesan el texto secuencialmente, el *Transformer* utiliza un mecanismo de atención que le permite considerar todas las palabras de una oración simultáneamente. Este mecanismo, inspirado en los procesos cognitivos humanos, permite a los modelos de lenguaje "prestar atención" a las partes más relevantes de una

entrada de texto, para comprender mejor su significado y contexto. Así, el modelo puede capturar relaciones de largo alcance entre palabras, lo que resulta en un mejor rendimiento para una variedad de tareas.

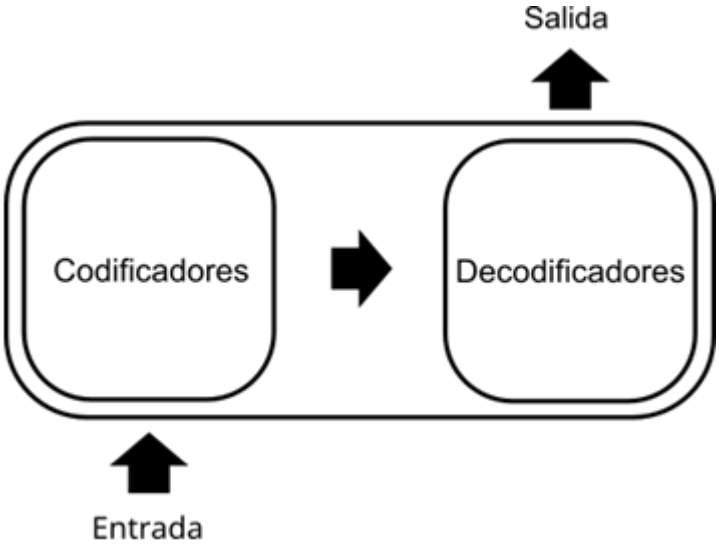

Figura 2. Diagrama simplificado del *Transformer*

2.1.2 *BERT*

A partir de la arquitectura *Transfomer*, surgieron distintos *LLMs* que se basan en ella y lograron mejoras significativas. En 2019 un grupo de investigadores de *Google* presentó *BERT – Bidirectional Encoder Representations from Transformers* (Devlin y otros, 2019) que es considerado como uno de los primeros *LLMs*. Este modelo puede entrenarse de forma auto supervisada, alimentándose con porciones de texto que poseen algunas palabras enmascaradas, para que éste aprenda a predecirlas. Este enfoque de preentrenamiento permite a *BERT* aprender representaciones contextuales de las palabras, lo que significa que puede comprender el significado de una palabra en función del contexto en el que se usa. Además, esta forma de entrenamiento –sin el esfuerzo de preparar conjuntos de datos etiquetados–, permite entrenar el modelo con grandes corpus de texto obtenidos de bibliotecas digitales completas. Este proceso es el que hace posible al modelo almacenar la gramática y las relaciones entre las partes de un texto en lenguaje natural, lo que se denomina modelo de lenguaje.

2.1.3 *GPT*, *T5* y más allá

Desde la aparición de *BERT* y hasta la fecha continúan desarrollándose modelos basados en el *Transformer*, tales como *GPT – Generative Pretraining Transformer* (Brown y otros, 2020) y *T5 – Text-to-Text Transfer Transformer* (Raffel y otros, 2020), que se entrenaron en conjuntos de datos aún más grandes compuestos de texto y código. Estos modelos se destacan por su capacidad para generar texto de alta calidad y realizar una variedad de tareas de PLN, incluyendo traducción automática, resumen y respuesta a preguntas.

La utilización de los *LLMs* se difunde aún más a partir de la aparición de *ChatGPT* en 2022, una aplicación de la empresa *OpenAI* (2025) que permite utilizar IA generativa a través de una interfaz de chat basándose en el modelo *GPT* para generar las respuestas. Esta aplicación permite a usuarios sin conocimientos utilizar los beneficios de un modelo grande de lenguaje, lo que crea nuevos desafíos tales como la ingeniería de *prompts*, que estudia cómo componer textos con los que consultar a estos modelos y obtener los mejores resultados.

2.1.4 Modelos preentrenados y ajuste fino

Tal como se citó anteriormente, los *LLMs* pueden entrenarse de forma semisupervisada, lo que permite utilizar conjuntos masivos de texto y código para el entrenamiento. Esto proporciona una amplia comprensión del lenguaje en general y la capacidad de realizar tareas básicas de PLN (Fengli y otros, 2025). Sin embargo, es posible que un modelo preentrenado no esté optimizado para realizar tareas específicas con la máxima precisión y eficiencia. Hay que tener en cuenta que para el preentrenamiento de los *LLMs* se utilizan grandes bibliotecas y repositorios como

Wikipedia (2024), entre otros. Esto les permite aprender patrones y representaciones del lenguaje en general, ya que estos textos no se especializan en ningún dominio en particular. Al preentrenamiento lo suelen realizar investigadores o grandes empresas de tecnología, ya que para el mismo se necesitan grandes conjuntos de datos y recursos computacionales (Patil & Gudivada, 2024).

Por otro lado, el ajuste fino es el proceso de adaptar un *LLM* preentrenado a una tarea específica. En este proceso, el *LLM* preentrenado se expone a un conjunto de datos más pequeño, que está específicamente diseñado para una tarea en cuestión. De esta forma, el modelo puede ajustar sus parámetros para enfocarse en los aspectos del lenguaje que son más relevantes en la realización de esa tarea. En este proceso se pueden utilizar distintos tipos de entrenamiento, dependiendo de la tarea que se desee realizar. Si la necesidad es mejorar la capacidad generativa en un dominio, en este caso se puede recurrir al proceso semiautomático que se utilizó en el preentrenamiento. Si, en cambio, se desea utilizar el modelo en una tarea de clasificación, seguramente se dispondrá de un conjunto de datos etiquetados con ejemplos de los distintos tipos de texto a clasificar (Lyu y otros, 2024) –Figura 3–.

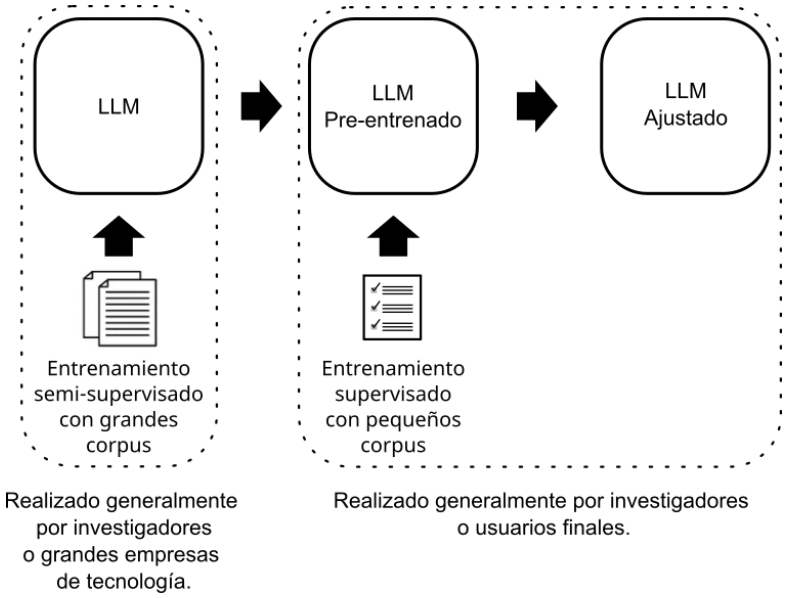

Figura 3. Distintos entrenamientos de un *LLM* para una tarea de clasificación

2.1.5 Prompt engineering

Los *LLMs* son utilizados generalmente por su funcionalidad generativa. Así se trate de una tarea de clasificación, ésta puede obtenerse mediante la generación por parte del modelo de la etiqueta esperada. Para explotar esta capacidad generativa es importante dar al modelo una buena entrada, que conduzca la generación hacia nuestro objetivo (Han y otros, 2025).

La ingeniería de *prompts*, o *prompt engineering*, es una disciplina en constante evolución que se enfoca en diseñar y optimizar las entradas que se le proporcionan a un *LLM* (Prompt Engineering Guide, 2023). En esencia, es la capacidad de formular preguntas o instrucciones de manera precisa para obtener respuestas relevantes y útiles.

En la entrada que se le proporciona al modelo se pueden incluir ejemplos de la tarea que esperamos del mismo. Dependiendo de la cantidad de ejemplos que se le proporcionen al modelo, la ingeniería de *prompts* se puede categorizar en tres enfoques principales (Li, 2023):

- *Zero-shot*: En este enfoque, el modelo genera una respuesta a partir de una entrada que no incluye ejemplos de la tarea a realizar.
- *One-shot*: En esta propuesta se le proporciona al modelo un solo ejemplo de la tarea que se desea realizar. Este enfoque suele ser más efectivo que el *zero-shot*, ya que el modelo tiene una referencia concreta a seguir.
- *Few-shots*: En este caso, se le proporcionan al modelo varios ejemplos de la tarea. El *few-shots* es el enfoque más poderoso, ya que el modelo puede aprender patrones y generalizaciones a partir de múltiples ejemplos.

2.2 El proceso de ingeniería de requerimientos

La IR es la rama de la Ingeniería de Software – IS que se ocupa de gestionar los servicios y restricciones de los sistemas de software (Nuseibeh & Easterbrook, 2000). Según Sommerville (2011), el proceso de IR puede dividirse en cuatro actividades principales: estudio de factibilidad, obtención y análisis de requerimientos, especificación de requerimientos y validación de requerimientos. Estas actividades no están ordenadas de forma secuencial, sino que pueden volver a realizarse de forma iterativa a lo largo del proceso de desarrollo, conforme cambian los requerimientos o se descubren nuevos –**Error! Reference source not found.**–.

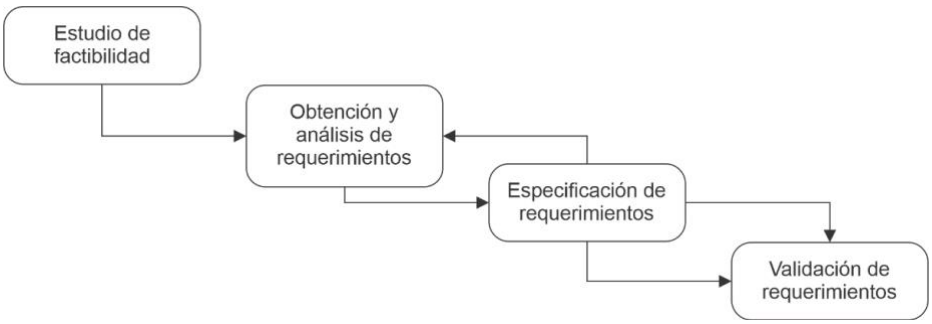

Figura 4. Actividades del proceso de ingeniería de requerimientos

Las actividades del proceso de IR tienen como uno de sus objetivos, el de generar la ERS, artefacto fundamental y transversal del desarrollo de software que se describe en lenguaje natural. Además, el proceso se nutre de diversos artefactos en lenguaje natural, como pueden ser entrevistas realizadas a diferentes interesados, documentos de la organización, minutas de reuniones o manuales de sistemas legados, entre otros. Es esta fuerte dependencia del lenguaje natural en los procesos y artefactos de la IR la que propicia las propuestas de utilizar el PLN y, en este caso, los *LLMs* como herramientas que den soporte y simplifiquen las tareas de este proceso.

3 Metodología

La metodología utilizada en este trabajo se basa principalmente en las propuestas de Petersen y otros (2008; 2015) –Figura 5–, ampliamente aceptadas y empleadas en la comunidad científica para la realización de mapeos y revisiones de literatura en el área de la Ingeniería de Software.

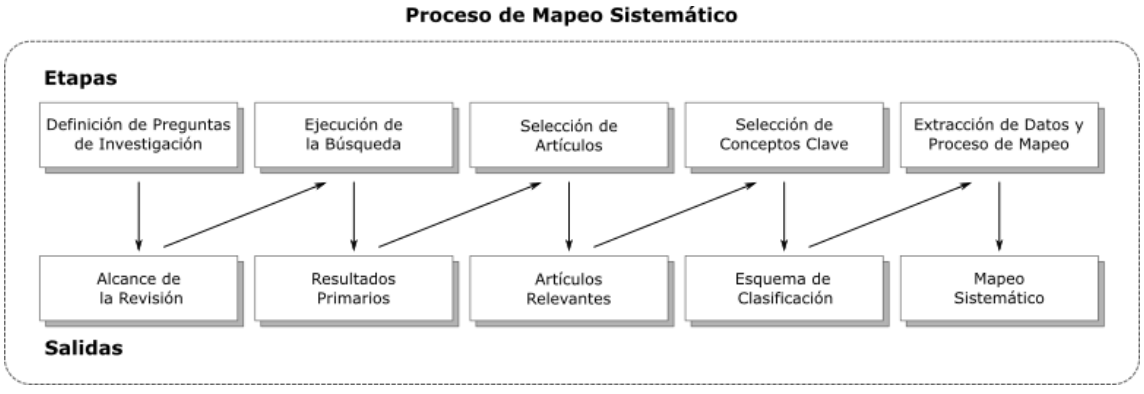

Figura 5. Proceso de mapeo sistemático

3.1 Preguntas de investigación y cadenas de búsqueda

Especificar de forma correcta las preguntas de investigación –Tabla 1– es la parte más importante en un MSL, ya que éstas conducen todo el proceso de búsqueda y selección de artículos mediante el cual se pretende contestar las mismas. La cadena de búsqueda se construyó a partir de los principales conceptos de las preguntas de investigación, uniendo los términos con un conector lógico *AND*, resultando la misma: "*requirements engineering*" *AND* "*large language models*".

Tabla 1. Listado de preguntas de investigación

Pregunta	Motivación
Q1- ¿Cuáles son los *LLMs* más aplicados en el proceso de IR?	Relevar los *LLMs* más utilizados para facilitar el proceso de IR.
Q2- ¿En cuáles etapas del proceso de IR se aplican *LLMs*?	Conocer las etapas y actividades en las que se prueba el uso de *LLMs* como herramienta.

3.2 Bases de datos científicas

Los estudios primarios se identificaron consultando cuatro BD en línea: 1) *IEEE Xplore*; 2) *ScienceDirect*; 3) *SpringerLink*, y 4) *Scopus*. Se seleccionaron estas BD debido a que permiten el acceso institucional gratuito a publicaciones indexadas, son ampliamente utilizadas en trabajos del área y proveen formularios de búsqueda avanzada que permiten utilizar correctamente la cadena de búsqueda generada.

3.3 Temporalidad y tipo de artículos

Se consideró apropiado tomar como punto de partida temporal el año 2019, en el que se presentó el modelo *BERT*. Adicionalmente, se decidió trabajar con producciones que hayan sido sometidas a un proceso de revisión antes de su publicación. Por todo lo expuesto, la búsqueda se restringió sólo a artículos de tipo *journal article* o *conference paper*.

3.4 Resultados primarios

Con los parámetros definidos tras la aplicación de la metodología explicada, se realizó la búsqueda en las cuatro bases de datos seleccionadas en los meses de febrero y marzo de 2024, adaptando la cadena de búsqueda al formato que estas requieren en sus formularios de búsqueda avanzada. La Tabla 2 muestra los resultados obtenidos que fueron, en total, 404 trabajos.

Tabla 2. Resultados primarios de la búsqueda en bases de datos

Base de Datos	Resultados
IEEEXplore	379
Springer Link	8
SCOPUS	17
ScienceDirect	0
Resultados primarios totales	**404**

3.5 Selección de artículos relevantes

En una primera revisión se identificaron y eliminaron los trabajos duplicados, luego, se siguió la propuesta de Mendes y otros (2020), aplicando los criterios de inclusión y exclusión citados en la Tabla 3 para quitar los estudios que no aporten a las respuestas de las preguntas de investigación.

Tabla 3. Criterios de inclusión y exclusión

Criterios de inclusión	Criterios de exclusión
Trabajos que involucren la utilización de al menos un *LLM* para facilitar el desarrollo de alguna de las actividades del proceso de IR	Artículos fuera del período de estudio
Trabajos escritos en formato *journal article* o *conference paper*	El artículo es un editorial o una publicación secundaria (revisión de la literatura)

En esta etapa se logró identificar y eliminar falsos positivos de la búsqueda inicial. La mayor parte se trató de propuestas con *LLMs* e involucrando la IR, pero que se encuentran fuera del alcance de esta revisión. Finalmente, luego del proceso de selección, se identificaron 31 trabajos que proponen alguna aplicación de *LLMs* al proceso

de IR. La Tabla 4 muestra la cantidad de artículos seleccionados por cada base de datos, mientras que la Tabla 5 lista todos los artículos, la base de datos de origen, el año de publicación, autor y referencia.

Tabla 4. Cantidades de artículos seleccionados obtenidos en cada base de datos

Base de datos	Resultados primarios	Eliminados	Trabajos seleccionados
IEEEXplore	379	352	26
Springer Link	8	7	1
SCOPUS	17	9 duplicados + 4	4
SpringerLink	0	0	0
Totales	**404**	**372**	**31**

Tabla 5. Listado de trabajos seleccionados

BD	Año	Título	Autor / referencia
SpringerLink	2023	*Evaluating software user feedback classifier performance on unseen apps, datasets, and metadata*	Devine y otros (2023)
Scopus	2023	*nl2spec: Interactively translating unstructured natural language to temporal logics with large language models*	Cosler y otros (2023)
Scopus	2023	*Agile methodology for the standardization of engineering requirements using large language models*	Tikayat Ray y otros (2023)
Scopus	2024	*ChatModeler: A human-machine collaborative and iterative requirements elicitation and modeling approach via large language models*	Jin y otros (2024)
Scopus	2023	*Comparing general purpose pre-trained word and sentence embeddings for requirements classification*	Cruciani y otros (2023)
IEEE	2023	*Extracting domain models from textual requirements in the era of large language models*	Arulmohan y otros (2023)
IEEE	2023	*Generating requirements elicitation interview scripts with large language models*	Görer & Aydemir (2023)
IEEE	2023	*A transformer-based approach for abstractive summarization of requirements from obligations in software engineering contracts*	Jain y otros (2023)
IEEE	2022	*Automated handling of anaphoric ambiguity in requirements: a multi-solution study*	Ezzini y otros (2022)
IEEE	2023	*Investigating ChatGPT's potential to assist in requirements elicitation processes*	Ronanki y otros (2023)
IEEE	2023	*ECHO: An approach to enhance use case quality exploiting large language models*	De Vito y otros (2023)
IEEE	2020	*A deep context-wise method for coreference detection in natural language requirements*	Wang y otros (2020)
IEEE	2023	*AI-based question answering assistance for analyzing natural-language requirements*	Ezzini y otros (2023)
IEEE	2023	*Automated identification of deontic modalities in software engineering contracts: A domain adaptation-based generative approach*	Rejithkumar y otros (2023)
IEEE	2022	*Automated question answering for improved understanding of compliance requirements: A multi-document study*	Abualhaija y otros (2022)
IEEE	2023	*Zero-shot bilingual app reviews mining with large language models*	Wei y otros (2023)
IEEE	2023	*On the use of GPT-4 for creating goal models: An exploratory study*	Chen y otros (2023)
IEEE	2020	*Extracting and classifying requirements from software engineering contracts*	Sainani y otros (2020)
IEEE	2023	*Empirical evaluation of ChatGPT on requirements information retrieval under zero-shot setting*	Zhang y otros (2023)
IEEE	2023	*MAPE-K loop-based goal model generation using generative AI*	Nakagawa & Honiden (2023)
IEEE	2023	*Requirements modeling aided by ChatGPT: An experience in embedded systems*	Ruan y otros (2023)
IEEE	2023	*TAG: UML activity diagram deeply supervised generation from business textural specification*	Zhu y otros (2023)
IEEE	2023	*A software requirements ecosystem: Linking forum, issue tracker, and FAQs for requirements management*	Tizard y otros (2023)
IEEE	2021	*A pipeline for automating labeling to prediction in classification of NFRs*	Chatterjee y otros (2021)

BD	Año	Título	Autor / referencia
IEEE	2023	*COSMIC-functional size classification of agile software development: Deep learning approach*	Molla y otros (2023)
IEEE	2023	*A data-driven approach for finding requirements relevant feedback from TikTok and YouTube*	Sihag y otros (2023)
IEEE	2021	*Unsupervised topic discovery in user comments*	Stanik y otros (2021)
IEEE	2021	*Transfer learning for mining feature requests and bug reports from tweets and app store reviews*	Restrepo Henao y otros (2021)
IEEE	2022	*Automated extraction of ABAC policies from natural-language documents in healthcare systems*	Xia y otros (2022)
IEEE	2022	*Fine-tuning pre-trained model to extract undesired behaviors from app reviews*	Zhang y otros (2022)
IEEE	2021	*Automatic issue classifier: A transfer learning framework for classifying issue reports*	Nadeem y otros (2021)

En cuanto a la distribución de los trabajos en el tiempo, se calculó la frecuencia de artículos publicados por año dentro del período comprendido para este trabajo, desde el año 2019 y hasta el año 2024 –Figura 6–.

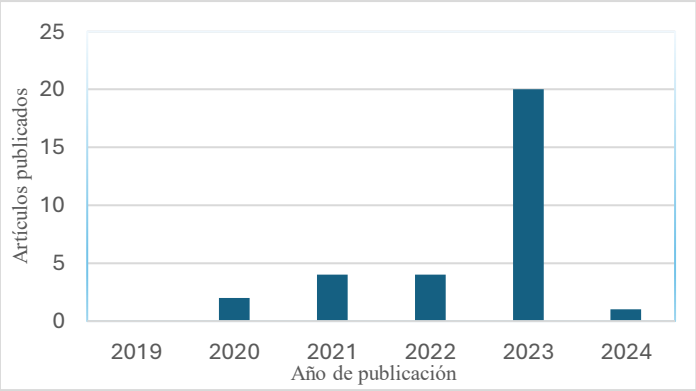

Figura 6. Cantidad de trabajos por año de publicación

Se puede observar que no hay trabajos para el año 2019. Esto puede deberse a que eran muy recientes los *LLMs* y se aplicaban otras técnicas y modelos anteriores de aprendizaje automático. Se observa un incremento significativo en año 2023, con el 65% del total de artículos seleccionados. Este marcado crecimiento se debe sin dudas a la aparición de *ChatGPT* en noviembre de 2022. *ChatGPT* es accesible a un público amplio a través de una interfaz web y una *API* que permite integrar las capacidades del modelo en otras aplicaciones. Esto marca un punto de inflexión en el desarrollo de los *LLMs*, ya que democratiza el acceso a estas tecnologías y las pone al alcance de personas de diversos ámbitos.

Por otro lado, no se debe prestar atención a la baja cantidad de artículos en el año 2024, ya que este trabajo sólo pudo abarcar unas pocas semanas de éste.

Otro análisis sobre el conjunto de artículos comprende la frecuencia de las palabras clave que se visualiza en la Figura 7. La frecuencia de las palabras clave proporciona una visión general de los tópicos más relevantes en los trabajos seleccionados. Mediante esta representación, se puede apreciar la presencia de la IR, los *LLMs* y el PLN como los términos con mayor frecuencia, también se observan los modelos más utilizados y algunas actividades del proceso de IR.

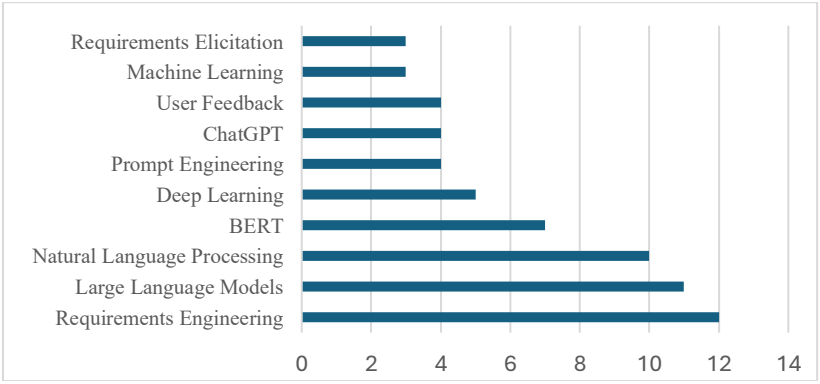

Figura 7. Frecuencia de palabras clave

3.6 Extracción y mapeo de datos

El proceso de extracción de datos consiste en la lectura y análisis completo de los artículos seleccionados, con el objetivo de responder las preguntas de investigación. A continuación, se exponen los resultados obtenidos y se da respuesta a las preguntas de investigación.

Q1- ¿Cuáles son los *LLMs* más aplicados en el proceso de IR?

A partir de los resultados del análisis al conjunto de artículos seleccionados, se puede concluir que los *LLMs* más utilizados en las propuestas que buscan aportar soluciones al proceso de IR son: *BERT* que se utiliza en el 65% de los trabajos y *GPT* que lo hace en el 39% –Figura 8–. Ambos modelos poseen variaciones. Por un lado, *BERT* ha sido modificado y afinado para múltiples tareas en diferentes dominios, por lo que se encontró gran variedad de modelos ajustados y basados en este modelo "base". En cuanto a *GPT*, los trabajos utilizan distintas versiones desde *GPT*-2 a *GPT*-4. Esto puede depender de la versión que había disponible en el momento en que se realizó el trabajo y, también, en el modo de acceso al modelo, ya que algunas versiones de éste sólo permiten el uso mediante el pago del servicio. Es importante citar que la empresa *OpenAI* provee el acceso a los modelos a través de una interface web, lo que agiliza mucho el prototipado y las pruebas, tal como se analizará en una sección posterior.

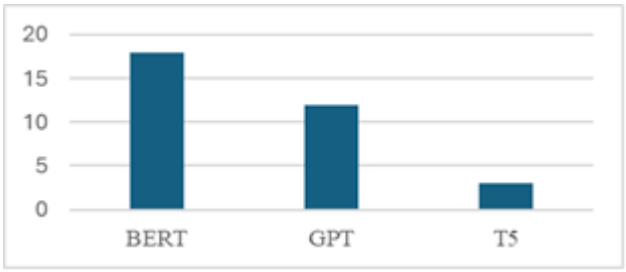

Figura 8. Cantidad de artículos para los modelos más utilizados

Q2- ¿En cuáles etapas del proceso de IR se aplican *LLMs*?

Para ordenar los trabajos con referencia a su aporte en el proceso de IR, se describen los mismos agrupados por la etapa del proceso a la que dedican sus esfuerzos –Tabla 6–, considerando las etapas propuestas por Sommerville (2011) –Figura 3–.

Tabla 6. Aportes del trabajo a etapas del proceso de IR

Etapas de IR	Aplicación propuesta de *LLMs*	Artículos
Obtención y análisis de requerimientos	Análisis / clasificación / agrupamiento de reseñas de usuarios (videos, *posts*, *tweets*, etc.)	Devine y otros (2023) Henao y otros (2021) Nadeem y otros (2021) Sihag y otros (2023) Stanik y otros (2021) Wei y otros (2023) Zhang y otros (2022)
	Calidad de requerimientos (evaluación)	Ezzini y otros (2022) Wang y otros (2020) Ezzini y otros (2023)
	Clasificación de requerimientos	Chatterjee y otros (2021) Cruciani y otros (2023) Sainani y otros (2020) Zhang y otros (2023)
	Extracción de características	Arulmohan y otros (2023) Zhang y otros (2023)
	Extracción / generación de requerimientos	Jain y otros (2023)

Etapas de IR	Aplicación propuesta de *LLMs*	Artículos
Especificación de requerimientos		Ronanki y otros (2023)
		Rejithkumar y otros (2023)
		Xia y otros (2022)
	Estimación de esfuerzo	Molla y otros (2023)
	Técnicas de captura	Görer y Aydemir (2023)
	Estandarización de requerimientos en LN	Tikayat Ray y otros (2023)
	Generación de modelos	Chen y otros (2023)
		Cosler y otros (2023)
		De Vito y otros (2023)
		Jin y otros (2024)
		Nakagawa y Honiden (2023)
		Ruan y otros (2023)
		Zhu y otros (2023)
Validación de requerimientos	Extracción de expresiones para validación	Abualhaija y otros (2022)
Gestión de requerimientos	Trazabilidad	Tizard y otros (2023)

3.6.1 Obtención y análisis de requerimientos

La etapa de obtención y análisis de requerimientos es la que más artículos agrupa. Las técnicas de captura describen las herramientas más utilizadas para la recolección de requerimientos. Dentro de las técnicas utilizadas para obtener requerimientos desde los interesados, las entrevistas se presentan en todos los proyectos. Un solo trabajo se ocupa de las actividades de captura de requerimientos desde interesados, Görer y Aydemir (2023) proponen la generación de guiones de entrevistas aplicando técnicas de *prompt engineering* en un proceso iterativo.

Otras técnicas de captura proponen la detección de documentos que pueden contener requerimientos. El crecimiento en la interacción con los usuarios de distintos productos de software a través de tiendas de aplicaciones, foros o redes sociales incrementó la cantidad de retroalimentación que se obtiene de un producto. Estos textos en lenguaje natural provenientes de los usuarios pueden ser de gran utilidad para detectar distintos tipos de requerimientos. Siete artículos trabajan en la detección de posibles fuentes de requerimientos a partir de la clasificación de artefactos que provienen de los usuarios. Wei y otros (2023) proponen *Mini-BAR*, una herramienta que integra *LLMs* para clasificación, agrupamiento, resumen y *ranking* de reseñas de usuarios, utilizan *ChatGPT* con *zero-shot* aplicando técnicas de *prompt engineering*. Devine y otros (2023) evalúan modelos para clasificación de peticiones de funcionalidades y reportes de error, concluyen que existe transferencia de aprendizaje, por lo que un modelo clasificador entrenado logra buenos rendimientos con conjuntos de datos no vistos.

Stanik y otros (2021) proponen usar un modelo *SBERT* como entrada a un modelo *HDBSCAN* para realizar agrupamiento y modelado de tópicos en comentarios de usuarios en redes sociales, con el objetivo de encontrar información relevante sobre los productos y servicios. Restrepo Henao y otros (2021) proponen analizar reseñas de aplicaciones utilizando *BERT* para *embedding* de texto conectado a distintos clasificadores. Zhang y otros (2022) entrenan un clasificador que utiliza *UBC-BERT*, un ajuste fino de *BERT-BASE*, como *embedding* de texto para identificar entre 21 posibles problemas con aplicaciones móviles. Sihag y otros (2023) ajustan distintos modelos basados en *Transformers* para clasificar la información extraída de videos (*Youtube* y *TikTok*) y luego usan *BERTopic* para aprender sobre los tópicos en los reportes de usuarios. Por último, dentro de esta actividad Nadeem y otros (2021) utilizan un modelo *RoBERTa* ajustado y proponen un clasificador para textos de reportes de incidencias en tres categorías: informe de error, mejora, o pregunta.

Como actividad auxiliar, en el modelado de requerimientos puede citarse también la extracción de características desde textos referidos a los mismos. Dos trabajos, Arulmohan y otros (2023) y Zhang y otros (2023), dedican sus esfuerzos a esta tarea utilizando modelos *GPT* con técnicas de *prompt engineering*.

Los requerimientos expresados en lenguaje natural deben cumplir con ciertos atributos para asegurar su calidad. Tres artículos se dedican a la evaluación de requerimientos, utilizando distintas variantes de *BERT*. Ezzini y otros (2022), por un lado, prueban el desempeño de *SpanBERT* frente a otras alternativas para la detección e interpretación de ambigüedad anafórica y en un segundo trabajo (Ezzini y otros, 2023) plantean *Qassist*, una herramienta para asistir a los analistas en la etapa de análisis de requerimientos. Además, Wang y otros (2020) proponen *DeepCoref*, un ensamble para la detección de correferencia de entidades en requerimientos expresados en lenguaje natural.

Para la actividad de generación y extracción de requerimientos, cuatro trabajos utilizan distintos modelos en sus propuestas. Jain y otros (2023) evalúan *T5*, *GPT* y otros modelos en la extracción y resumen de requerimientos

desde contratos. Ronanki y otros (2023) generan requerimientos con *ChatGPT* y evalúan su calidad. Rejithkumar y otros (2023) ajustan modelos *T5*, *Pegasus* y *GPT* para extraer requerimientos identificando modalidades deónticas en contratos. Por último, Xia y otros (2022) utilizan *BERT* como parte de una propuesta para extraer políticas de control de acceso desde textos en lenguaje natural.

La clasificación de requerimientos es una actividad importante dentro del proceso de IR que aporta claridad, organización y comunicación efectiva. Además, la clasificación de requerimientos no funcionales es una actividad ineludible, ya que permite ordenar las distintas restricciones y medidas de calidad que el producto final de software deberá entregar. Para esta tarea tres trabajos (Chatterjee y otros, 2021; Cruciani y otros, 2023; Sainani y otros, 2020) evalúan distintos modelos para clasificar requerimientos y obtienen los mejores resultados con modelos basados en *BERT*. Un cuarto trabajo, Zhang y otros (2023) proponen utilizar *ChatGPT* con *zero-shot* y técnicas de *prompt engineering*.

La estimación de esfuerzo es una actividad fundamental, que aporta al éxito de un proyecto de desarrollo de software. Permite tener una idea clara del esfuerzo requerido para la implementación de los distintos requerimientos capturados. De esta forma, los diferentes equipos involucrados en el proyecto pueden trabajar de manera más organizada y eficiente. Son distintas y variadas las propuestas para estimar esfuerzo y dependen, en gran medida, del tipo de metodologías y artefactos se están utilizando en el proyecto. Para esta actividad se encontró un trabajo, Molla y otros (2023), que propone distintos clasificadores basados en *BERT* para estimación de esfuerzo mediante puntos de función *COSMIC*.

3.6.2 Especificación de requerimientos

La especificación de requerimientos es el proceso de documentar los requerimientos cumpliendo con ciertas características que aseguran su calidad, tales como la no ambigüedad, la consistencia y la completitud, entre otras. Según Sommerville (2011) los requerimientos pueden ser expresados utilizando distintos lenguajes, de acuerdo con el nivel de detalle con el que se expresen y con el conocimiento técnico de los destinatarios. Así, el autor concluye que los requerimientos de usuario –que deben ser entendidos por interesados del sistema que no cuentan con un conocimiento técnico detallado– deben estar expresados en lenguaje natural, sin tecnicismos ni vocabulario específico. Mientras, los requerimientos de sistema que detallan a los requerimientos de usuario pueden requerir un nivel de detalle mayor que justifique el uso de distintos lenguajes gráficos.

En esta sección se reseñan los trabajos encontrados que hacen uso de *LLMs* para aportar a la especificación de requerimientos, ya sea en lenguaje natural o gráfico. Chen y otros (2023), De Vito y otros (2023), Nakagawa y Honiden (2023), y Ruan y otros (2023) trabajan sobre distintas propuestas para utilizar *ChatGPT* en la generación de diferentes modelos, tales como: modelos de objetivos, casos de uso, y diagramas de problemas. Además, Cosler y otros (2023) presentan *nl2spec*, un marco de trabajo basado en los modelos *Bloom* y *Codex*, para facilitar y automatizar la redacción de especificaciones formales en lógicas temporales. Jin y otros (2024) proponen *Chat modeler* un marco que pretende la obtención de distintos modelos de requerimientos utilizando *chats* con *LLMs*. Finalmente, Zhu y otros (2023) presentan un *pipeline* que utiliza *BERT*, entre otros modelos, para la generación de diagramas de actividad *UML* a partir de especificaciones textuales. En esta etapa, dentro de las actividades de estandarización de requerimientos, Tikayat Ray y otros (2023) analizan el uso de variantes de *BERT* para convertir requerimientos en lenguaje natural en requerimientos semilegibles por máquina.

3.6.3 Validación de requerimientos

La etapa de validación de requerimientos es la que busca asegurar que los requerimientos capturados realmente conducirán al sistema que necesitan los interesados. La validación temprana de requerimientos permite la rápida detección de distintos problemas que, de no ser identificados, serán más difíciles de corregir conforme avance el proceso de desarrollo, ya que es mucho más complejo y costoso corregir errores en etapas de diseño o codificación.

La captura de requerimientos no funcionales que expresen lo que los interesados esperan, posee las complicaciones propias de las expresiones en lenguaje natural, debido a que estos requerimientos generalmente se encuentran en documentos o como parte de otros requerimientos. Un sólo trabajo de los seleccionados se ocupa de esta actividad del proceso, Abualhaija y otros (2022) presentan un *pipeline* para extraer preguntas y encontrar expresiones de validación desde un documento.

3.6.4 Gestión de requerimientos

A medida que transcurre el proceso de IR los requerimientos capturados cambian y, con ellos, los distintos artefactos que los especifican. Esto ocurre porque a medida que se avanza en el proyecto, el mayor entendimiento de las necesidades a cubrir por el sistema causa una evolución de éste. Esta evolución debe estar reflejada en los requerimientos para mantener la consistencia de los distintos artefactos que especifican el sistema.

La trazabilidad es una herramienta fundamental en el proceso de desarrollo de software, ya que permite establecer y mantener un vínculo entre los requerimientos del software, los artefactos de diseño, la implementación, las pruebas y la documentación a lo largo del proyecto. En particular, en el proceso de IR, la trazabilidad facilita el enlace entre las fuentes de requerimientos y los distintos modelos que los especifican, siendo ésta fundamental para mantener actualizados todos los artefactos cuando surge un cambio. De los artículos seleccionados, sólo uno trata el tema, Tizard y otros (2023) proponen trazar relaciones entre entradas de foros, reportes de problemas y *FAQs*, utilizando *USE –Universal Sentence Encoder–* basado en *Transformer* para vectorizar textos y luego comparar similitud.

4 Resultados y discusiones generales

El conjunto de artículos seleccionados se describe en la sección anterior a partir de las preguntas de investigación originalmente definidas, considerando tanto los *LLMs* utilizados como las etapas del proceso de IR involucradas. Asimismo, se analiza la información extraída y resumida del conjunto de trabajos con el fin de observar cómo los modelos se aplican a lo largo del proceso de IR. En este sentido, y tal como reza el objetivo de este trabajo, la intención principal no es únicamente catalogar qué modelos o técnicas se emplean, sino comprender cómo estas aplicaciones inciden en las actividades específicas del proceso de IR. El enfoque adoptado permite identificar de qué manera los *LLMs* contribuyen, potencian o transforman las tareas propias de cada etapa, proporcionando además la identificación de vacíos y oportunidades para investigaciones futuras.

Siguiendo el orden de las etapas del proceso de IR propuesto por Sommerville (2011), sobresale en la etapa Obtención y análisis de requerimientos que un sólo trabajo, Görer y Aydemir (2023), se dedica a la captura de requerimientos desde los interesados, siendo ésta una de las principales actividades en las que se trata de interpretar desde las personas interesadas, las necesidades que el sistema debe cubrir. Las distintas técnicas y herramientas utilizadas frecuentemente para estas tareas, como ser entrevistas, cuestionarios y reuniones –por nombrar las más conocidas–, siempre se ven afectadas y complejizadas por la interacción necesaria con varias personas. Además, estas personas suelen poseer, en general, distintos niveles de conocimiento del problema, necesidades superpuestas y/o en conflicto y se debe lidiar también con la ambigüedad propia del lenguaje natural.

El análisis realizado indica que, precisamente en estas actividades basadas en lenguaje natural y en la interacción humana, los *LLMs* muestran un potencial particularmente alto. Su capacidad para interpretar, resumir y detectar patrones lingüísticos permite asistir al analista en la identificación de necesidades, la reducción de ambigüedad y la organización preliminar de la información obtenida de los interesados. En este sentido, los *LLMs* pueden convertirse en herramientas clave para mitigar problemas tales como la inconsistencia discursiva o la diversidad de vocabulario entre los distintos interesados, dado que ambos aspectos tienen como raíz el uso de lenguaje natural y la interacción entre personas. Por lo expuesto, los resultados obtenidos sugieren que se debería intensificar la investigación orientada al desarrollo de herramientas basadas en *LLMs* que acompañen y asistan estas actividades.

Así mismo, en la etapa de especificación de requerimientos, el trabajo Tikayat y otros (2023) trata con representaciones de requerimientos en lenguaje natural. En este caso lo hacen extrayendo características y reconociendo patrones.

La revisión evidencia que, en esta etapa, los *LLMs* pueden cumplir un rol dual: por un lado, asistir en la redacción y reformulación de requerimientos para mejorar su claridad, consistencia y completitud; y por otro, evaluar automáticamente la calidad lingüística y semántica de los mismos cuando están expresados en lenguaje natural. Esta capacidad resulta especialmente relevante en las etapas tempranas del proceso de IR, donde la mayoría de los requerimientos se representan en lenguaje natural, con las dificultades inherentes a lograr expresiones precisas, completas y no ambiguas. En este sentido, la aplicación de *LLMs* se alinea directamente con la necesidad de producir especificaciones más claras y robustas, ampliando las capacidades tradicionales del analista. Por lo expuesto, se considera pertinente profundizar la investigación orientada al diseño de herramientas basadas en *LLMs* que asistan en la escritura y validación de requerimientos que cumplan con las exigencias de calidad.

Por último, las etapas de validación de requerimientos y de gestión de requerimientos tienen sólo un artículo que se ocupa de éstas. La cobertura es escasa y se puede considerar que en estas áreas aún queda mucho por investigar en cuanto al uso de *LLMs*.

Aun así, con la escasa presencia de estudios, se observa que las capacidades de análisis semántico y detección de inconsistencias de los *LLMs* podrían fortalecer actividades clave de validación y gestión, como las relaciones entre requerimientos, la comparación de versiones y la identificación de conflictos. Esta falta de literatura constituye un hallazgo relevante, pues indica un vacío de investigación y una oportunidad concreta para avanzar en el desarrollo de herramientas basadas en *LLMs* que apoyen esta etapa.

Con el fin de resumir con claridad las tareas donde los *LLMs* ya muestran avances y aquellas en las que la investigación es aún limitada, la Tabla 7 sintetiza las brechas detectadas por etapa del proceso de IR y las oportunidades concretas para futuras líneas de trabajo.

Tabla 7. Aportes y oportunidades de investigación

Etapas de IR	Resumen de aportes	Vacíos / oportunidades de investigación
Obtención y análisis de requerimientos	Uso de *LLMs* para analizar y clasificar textos de usuarios en LN, detectar posibles requerimientos, extraer características y apoyar tareas como reformulación, evaluación de calidad y estimación de esfuerzo.	Falta investigación en interacción directa con interesados (entrevistas asistidas), detección automática de conflictos y apoyo para resolver diferencias entre las partes involucradas.
Especificación de requerimientos	Uso de *LLMs* para reformular y mejorar requerimientos en LN, generación automática de modelos y apoyo en estandarización de especificaciones.	Falta profundizar en generación controlada de expresiones, control de calidad automatizado, métricas asistidas por *LLMs* e integración con enfoques formales o semiformales de redacción.
Validación de requerimientos	Se exploran técnicas con *LLMs* para extraer expresiones de validación desde documentos y apoyar la detección temprana de posibles problemas en requerimientos no funcionales.	Varias oportunidades de investigación en validación automatizada, detección de inconsistencias o conflictos y verificación cruzada entre requerimientos y modelos.
Gestión de requerimientos	Se limita a tareas de trazabilidad basadas en *embeddings* o análisis semántico.	Es necesario profundizar en herramientas que asistan la gestión de cambios, el análisis de impacto y la priorización automática, así como la verificación entre versiones.

En segundo lugar, se evalúan las técnicas utilizadas para aplicar los *LLMs* en los artículos seleccionados y su relación con el tipo de tarea a realizar, además se citan algunas ventajas y desventajas en la aplicación de las distintas técnicas. La Figura 9 muestra la distribución de los artículos respecto de los modelos que implementan y la actividad del proceso de IR afectada. Se puede observar que en las actividades de clasificación se utilizaron preferentemente modelos basados en *BERT*, mientras que para las actividades que involucran generación o extracción de texto fueron más utilizados los modelos de la familia *GPT*.

Este patrón confirma que la elección del modelo no es arbitraria, sino que responde a las necesidades intrínsecas de cada etapa del proceso de IR. Los modelos basados en *BERT*, con su arquitectura orientada a comprensión profunda del contexto, resultan adecuados para clasificación, detección de categorías o análisis de características del texto. En contraste, los modelos generativos como *GPT* muestran un mejor desempeño en tareas donde se requiere producir texto, reformular, completar o sintetizar información, lo que influye directamente en la forma en que estas actividades pueden automatizarse o asistirse dentro del proceso de IR.

Como se puede observar en los artículos, además de existir diferentes modelos, también existen diferentes formas de utilizarlos. Además, no todos los modelos poseen la misma estructura y posibilidades. En cuanto a esto último, es importante citar una diferencia de *BERT* con el resto de los *LLMs*. *BERT* brinda en la salida, además de la secuencia de *tokens* que representan la cadena de palabras propuestas, un *token* llamado *CLS* que representa el texto completo. Esto permite a *BERT* realizar tareas de procesamiento del lenguaje natural que requieren una comprensión profunda del significado de la oración completa, como la clasificación de texto o el análisis de sentimiento, entre otros. Tomando sólo este *token* de la salida, este modelo puede utilizarse como *embedding* de textos, obteniendo una buena representación vectorial de dimensión 768 para los textos ingresados. Al menos cuatro de los artículos seleccionados implementan *BERT* como *embedding* de texto, para luego utilizar esta representación como entrada de otro modelo de clasificación, obteniendo todos buenos resultados. Esto concuerda con lo visto en la Figura 9 acerca de la preferencia de *BERT* como modelo utilizado en tareas de clasificación, entonces puede concluirse que la utilización de *BERT* como una capa de *embedding* dentro de un modelo de clasificación es una buena opción para tener en cuenta.

En general, todos los *LLMs* aplicados en los artículos seleccionados comparten la posibilidad de generar texto como principal característica, más allá de la aplicación que se haga de los mismos. Una familia de éstos, *GPT*, propone el acceso a las distintas versiones de modelos desde una interfaz gráfica, *ChatGPT*, que facilita el acceso a usuarios sin experiencia ni conocimiento de modelos o programación. La posibilidad de acceder de forma rápida y sencilla a los modelos brinda muchas ventajas en entornos de investigación, permitiendo la creación de pruebas y prototipos rápidos y ampliando el acceso a recursos humanos sin capacidades en programación, pero con mucho conocimiento de los dominios específicos de aplicación. Dentro del conjunto de artículos seleccionados se observan, al menos, ocho trabajos que utilizan *ChatGPT*, como parte del proceso de pruebas y prototipado, objeto de la investigación.

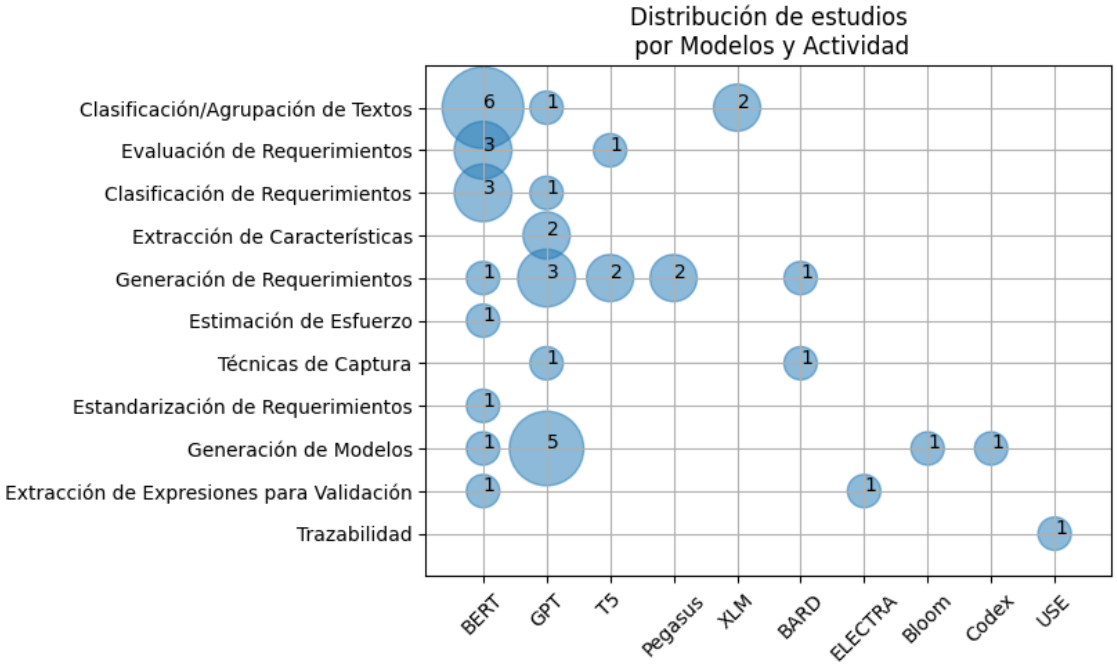

Figura 9. Distribución de artículos por modelo y actividad de IR

Otra característica que presentan algunos *LLMs* es la posibilidad de explotar la capacidad generadora de texto a partir de una conversación con el modelo, dado que son modelos *seq2seq* –toman de entrada una secuencia y devuelven otra–. Además, algunos de ellos permiten ir recordando las iteraciones anteriores con el usuario y tenerlas en cuenta a la hora de devolver una nueva respuesta. Aquí es donde el *prompt engineering* puede ser explotado con mejores resultados, ya sea en una conversación o en una única iteración. Las técnicas de *prompt engineering* hacen posible obtener resultados incluso interactuando con modelos preentrenados sin ajuste fino, además expanden las posibilidades al aplicar técnicas de *one-shot* o *few-shots* (agregando uno o pocos ejemplos a una consulta) o anexando a las consultas contextos de los cuales extraer las respuestas. Estas aplicaciones no pasan desapercibidas en el conjunto de artículos, donde se encuentra que al menos diez trabajos aplican *prompts* para interactuar con el modelo, obteniendo distintos resultados. Si bien existen diferentes propuestas acerca de cómo mejorar los *prompts* de acuerdo con la tarea y el dominio de aplicación, esta posibilidad de conversar con el modelo puede ser investigada para su utilización en los numerosos entornos de interacción que se generan a lo largo del proceso de IR, con el objeto de asistir a los analistas y acortar distancia entre interesados con distintos roles y conocimientos.

5 Limitaciones

Este trabajo adopta una perspectiva general sobre la aplicación de *LLMs* en el proceso de IR, con el objetivo de identificar tendencias, aportes y vacíos de investigación. En consecuencia, no se profundiza en el análisis crítico detallado de cada estudio individual, ni se comparan metodologías o configuraciones específicas de modelos. Asimismo, el alcance se limita al conjunto de artículos seleccionados, por lo que es posible que existan trabajos relevantes no incluidos en la revisión debido a criterios de búsqueda o disponibilidad.

Si bien la rápida evolución de los modelos de IA generativa puede generar futuras actualizaciones en cuanto a versiones o herramientas, los hallazgos identificados se consideran relevantes, ya que abordan aspectos estructurales del proceso de IR. Futuras investigaciones podrían complementar este enfoque mediante análisis más detallados de propuestas específicas, comparaciones entre modelos y evaluaciones empíricas que permitan validar el impacto real de las aplicaciones de *LLMs* en escenarios prácticos del proceso de IR.

6 Conclusiones

Los modelos objeto de este estudio han revolucionado la forma en que interactuamos con la IA con posibilidades como las de “conversar con un modelo”, y esto ha acercado el uso de estas poderosas herramientas a personas con

diversos roles. De este hecho surgen la mayoría de las oportunidades identificadas en este trabajo, que se basan en explotar el potencial de interacción y generación de los *LLMs*.

Claramente, se entrelazan las oportunidades detectadas, tanto desde el punto de vista del proceso de IR como de los *LLMs*. Así se puede concluir que existen potenciales líneas de investigación para proponer, estudiar y evaluar la aplicación de *LLMs*. Algunas de las cuales se listan a continuación:

- Técnicas de captura, que comprendan la interacción, tanto con personas interesadas como con documentos del dominio de información.
- Asistencia en la especificación de requerimientos en lenguaje natural, para asegurar las características de calidad de estos.
- Asistencia en la especificación de requerimientos en lenguajes gráficos a partir de fuentes textuales del dominio de información.
- Gestión de todo el conocimiento almacenado en la ERS, para agilizar el acceso con fines administrativos y de trazabilidad, entre otros.

ORCID ID

Mauro José Pacchiotti https://orcid.org/0000-0002-9162-7890
Luciana Ballejos https://orcid.org/0000-0001-5443-6617
Mariel Ale https://orcid.org/0000-0002-4866-4821

References

Abualhaija, S., Arora, C., Sleimi, A., & Briand, L. C. (2022). Automated question Answering for improved understanding of compliance requirements: A multi-document study. 2022 IEEE 30th International Requirements Engineering Conference (RE), 15-19 August 2022, Melbourne, Australia (págs. 39 - 50). IEEE. https://doi.org/10.1109/RE54965.2022.00011

Arulmohan, S., Meurs, M.-J., & Mosser, S. (2023). Extracting domain models from textual requirements in the era of large language models. 2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C) (págs. 580 - 587). Västerås, Suecia: IEEE Press. https://doi.org/10.1109/MODELS-C59198.2023.00096

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., . . . Liang, P. (12 de Julio de 2022). Onthe opportunities and risks of foundation models. arXiv, 2108.07258v3 [cs.LG], 1-214. https://doi.org/10.48550/arXiv.2108.07258

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., . . . Amodei, D. (2020). Language models are few-shot learners. En H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, & H. Lin (Ed.), Advances in Neural Information Processing Systems, 34th Conference on Neural Information Processing Systems (NeurIPS 2020). 33, págs. 1877 - 1901. Vancouver, Canada: Curran Associates Inc. Obtenido de https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

Chatterjee, R., Ahmed, A., Anish, P. R., Suman, B., Lawhatre, P., & Ghaisas, S. (2021). A pipeline for automating labeling to prediction in classification of NFRs. 2021 IEEE 29th International Requirements Engineering Conference (RE), 20-24 September 2021, Notre Dame, IN, USA (págs. 323 - 323). IEEE. https://doi.org/10.1109/RE51729.2021.00036

Chen, B., Chen, K., Hassani, S., Yang, Y., Amyot, D., Lessard, L., . . . Varró, D. (2023). On the use of GPT-4 for creating goal models: An exploratory study. 2023 IEEE 31st International Requirements Engineering Conference Workshops (REW), 04-05 September 2023, Hannover, Alemania (págs. 262 - 271). IEEE. https://doi.org/10.1109/REW57809.2023.00052

Cosler, M., Hahn, C., Mendoza, D., Schmitt, F., & Trippel, C. (2023). nl2spec: Interactively translating unstructured natural language to temporal logics with large language models. En C. Enea, & A. Lal (Ed.), Computer Aided Verification. CAV 2023. Lecture Notes in Computer Science, vol. 13965 (págs. 383 - 396). Suiza: Springer Nature. https://doi.org/10.1007/978-3-031-37703-7_18

Cruciani, F., Moore, S., & Nugent, C. (2023). Comparing general purpose pre-trained word and sentence embeddings for requirements classification. En A. Ferrari, B. Penzenstadler, I. Hadar, S. Oyedeji, S. Abualhaija, A. Vogelsang, . . . D. Amyot (Ed.), Joint Proceedings of REFSQ-2023 Workshops, Doctoral Symposium, Posters & Tools Track, and Journal Early Feedback Track. Co-located with

REFSQ 2023, April 17. 3378. Barcelona, Cataluña, España: CEUR-WS.org. Obtenido de https://ceur-ws.org/Vol-3378/NLP4RE-paper4.pdf

DAIR.AI. (23 de Abril de 2023). Elements of a Prompt. Obtenido de Prompt Engineering Guide: https://www.promptingguide.ai/introduction/elements

De Vito, G., Palomba, F., Gravino, C., Di Martino, S., & Ferrucci, F. (2023). ECHO: An approach to enhance use case quality exploiting large language models. 2023 49th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 06-08 September 2023 (págs. 53 - 60). Durres, Albania: IEEE. https://doi.org/10.1109/SEAA60479.2023.00017

Devine, P., Koh, Y. S., & Blincoe, K. (2023). Evaluating software user feedback classifier performance on unseen apps, datasets, and metadata. Empirical Software Engineering, 28, 26. https://doi.org/10.1007/s10664-022-10254-y

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. En J. Burstein, C. Doran, & T. Solorio (Ed.), Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). 1, págs. 4171 - 4186. Minneapolis, Minnesota, USA: Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-142

Ezzini, S., Abualhaija, S., Arora, C., & Sabetzadeh, M. (2022). Automated handling of anaphoric ambiguity in requirements: A multi-solution study. 44th IEEE/ACM 44th International Conference on Software Engineering, {ICSE} 2022, May 25-27 (págs. 187 - 199). Pittsburgh, PA, USA: ACM. https://doi.org/10.1145/3510003.3510157

Ezzini, S., Abualhaija, S., Arora, C., & Sabetzadeh, M. (2023). AI-based question answering assistance for analyzing natural-language requirements. 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE), 14-20 May 2023 (págs. 1277 - 1289). Melbourne, Australia: IEEE. https://doi.org/10.1109/ICSE48619.2023.00113

Fengli, X., Hao, Q., Shao, C., Zong, Z., Li, Y., Wang, J., . . . Yong, L. (10 de Octubre de 2025). Toward large reasoning models: A survey of reinforced reasoning with large language models. Patterns, 6(10), 1 - 30, 101370. https://doi.org/10.1016/j.patter.2025.101370

Görer, B., & Aydemir, F. B. (2023). Generating requirements elicitation interview scripts with large language models. 2023 IEEE 31st International Requirements Engineering Conference Workshops (REW). 04-05 September 2023 (págs. 44 - 51). Hannover, Alemania: IEEE. https://doi.org/10.1109/REW57809.2023.00015

Gramajo, M. G., Ballejos, L., & Ale, M. (Julio de 2020). Seizing requirements engineering issues through supervised learning techniques. IEEE Latin America Transactions, 18(07), 1164 - 1184. https://doi.org/10.1109/TLA.2020.9099757

Hadi, M. U., Al-Tashi, Q., Qureshi, R., Shah, A., Muneer, A., Irfan, M., . . . Shah, M. (10 de Febrero de 2025). Large language models: A comprehensive survey of its applications, challenges, limitations, and future prospects. TechRxiv, 1-54. https://doi.org/10.36227/techrxiv.23589741.v8

Han, W., Xiang, W., Cui, X., Cheng, N., Jiang, G., Qian, W., & Zhang, C. (2025). Prompt engineering 101: Prompt engineering guidelines from a linguistic perspective. En M. Sun, J. Liang, X. Han, Z. Liu, Y. He, G. Rao, . . . Z. Tian (Ed.), Chinese Computational Linguistics. CCL 2024. Lecture Notes in Computer Science (LNAI, volume 14761) (págs. 571 - 592). Singapur: Springer Nature. https://doi.org/10.1007/978-981-97-8367-0_34

Jain, C., Anish, P. R., Singh, A., & Ghaisas, S. (2023). A transformer-based approach for abstractive summarization of requirements from obligations in software engineering contracts. 2023 IEEE 31st International Requirements Engineering Conference (RE), 04-08 September 2023 (págs. 169 - 179). Hannover, Alemania: IEEE. https://doi.org/10.1109/RE57278.2023.00025

Jin, D., Jin, Z., Chen, X., & Wang, C. (2024). ChatModeler: A human-machine collaborative and iterative requirements elicitation and modeling approach via large language models. Journal of Computer Research and Development, 61(2), 338 - 350. https://doi.org/10.7544/issn1000-1239.202330746

Kitchenham, B., & Charters, S. M. (2007). Guidelines for performing systematic literature reviews in software engineering. Version 2.3. EBSE Technical Report EBSE-2007-01, Keele University, y University of Durham. Obtenido de https://legacyfileshare.elsevier.com/promis_misc/525444systematicreviewsguide.pdf

Li, Y. (2023). A practical survey on zero-shot prompt design for In-context learning. En R. Mitkov, & G. Angelova (Ed.), Proceedings of the 14th International Conference on Recent Advances in Natural Language Processing (págs. 641 - 647). Varna, Bulgaria: INCOMA Ltd. Obtenido de https://aclanthology.org/2023.ranlp-1.69/

Lyu, Y., Yan, L., Wang, S., Shi, H., Yin, D., Ren, P., . . . Ren, Z. (2024). KnowTuning: Knowledge-aware fine-tuning for Large Language Models. En Y. Al-Onaizan, M. Bansal, & Y.-N. Chen (Ed.), Proceedings of

the 2024 Conference on Empirical Methods in Natural Language Processing (págs. 14535 - 14556). Miami, Florida, USA: Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.emnlp-main.805

Mendes, E., Wohlin, C., Felizardo, K., & Kalinowski, M. (Septiembre de 2020). When to update systematic literature reviews in software engineering. Journal of Systems and Software, 167, 110607. https://doi.org/10.1016/j.jss.2020.110607

Molla, Y. S., Yimer, S. T., & Alemneh, E. (2023). COSMIC - functional size classification of agile software development: Deep learning approach. En 2023 (Ed.), 2023 International Conference on Information and Communication Technology for Development for Africa (ICT4DA), 26-28 October 2023, Bahir Dar, Etiopia (págs. 155 - 159). IEEE. https://doi.org/10.1109/ICT4DA59526.2023.10302232

Nadeem, A., Sarwar, M. U., & Malik, M. Z. (2021). Automatic issue classifier: A transfer learning framework for classifying issue reports. 2021 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), 25-28 October 2021, Wuhan, China (págs. 421 - 426). IEEE. https://doi.org/10.1109/ISSREW53611.2021.00113

Nakagawa, H., & Honiden, S. (2023). MAPE-K loop-based goal model generation using generative AI. En K. Schneider, F. Dalpiaz, & J. Horkoff (Ed.), Proceedings - 2023 IEEE 31st International Requirements Engineering Conference Workshops (REW), 04-05 September 2023, Hannover, Alemania (págs. 247 - 251). IEEE. https://doi.org/10.1109/REW57809.2023.00050

Nuseibeh, B., & Easterbrook, S. (2000). Requirements engineering: a roadmap. ICSE '00: Proceedings of the Conference on The Future of Software Engineering (págs. 35 - 46). Limerick, Ireland: Association for Computing Machinery - ACM. https://doi.org/10.1145/336512.336523

OpenAI. (28 de Abril de 2025). OpenAI. Obtenido de Sitio web de OpenAI: https://openai.com/

Patil, R., & Gudivada, V. (Marzo de 2024). A review of current trends, techniques, and challenges in Large Language Models (LLMs). Applied Sciences, 14(5), 1-42. https://doi.org/10.3390/app14052074

Petersen, K., Feldt, R., Mujtaba, S., & Mattsson, M. (2008). Systematic mapping studies in software engineering. 12th International Conference on Evaluation and Assessment in Software Engineering (EASE), 26 - 27 June 2008 (págs. 68 - 77). Bari, Italy: BCS Learning and Development Ltd. https://doi.org/10.14236/ewic/EASE2008.8

Petersen, K., Vakkalanka, S., & Kuzniarz, L. (Agosto de 2015). Guidelines for conducting systematic mapping studies in software engineering: An update. Information and Software Technology, 64, 1 - 18. https://doi.org/10.1016/j.infsof.2015.03.007

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., . . . Liu, P. J. (Enero de 2020). Exploring the limits of transfer learning with a unified text-to-text transformer. (I. Titov, Ed.) Journal of Machine Learning Research, 21(1), 5485 - 5551, Art. No. 140. Obtenido de https://jmlr.org/papers/volume21/20-074/20-074.pdf

Rahimi, N., Eassa, F., & Elrefaei, L. (2020). An ensemble machine learning technique for functional requirement classification. Symmetry, 12(10), 1601. https://doi.org/10.3390/sym12101601

Raiaan, M. A., Mukta, S. H., Fatema, K., Fahad, N. M., Sakib, S., Mim, M. M., . . . Azam, S. (2024). A review on large language models: Architectures, applications, taxonomies, open issues and challenges. IEEE Access, 12, 26839 - 26874. https://doi.org/10.1109/ACCESS.2024.3365742

Rejithkumar, G., Anish, P. R., & Ghaisas, S. (2023). Automated Identification of Deontic Modalities in Software Engineering Contracts: A Domain Adaptation-Based Generative Approach. 2023 IEEE 31st International Requirements Engineering Conference Workshops (REW), 4-5 September 2023, Hannover, Alemania (págs. 72 - 75). Los Alamitos, CA, USA: IEEE Computer Society. https://doi.org/10.1109/REW57809.2023.00020

Restrepo Henao, P., Fischbach, J., Spies, D., Frattini, J., & Vogelsang, A. (2021). Transfer learning for mining feature requests and bug reports from tweets and app store reviews. 2021 IEEE 29th International Requirements Engineering Conference Workshops (REW), Notre Dame, IN, USA (págs. 80 - 86). Los Alamitos, CA, USA: IEEE Computer Society. https://doi.org/10.1109/REW53955.2021.00019

Ronanki, K., Berger, C., & Horkoff, J. (2023). Investigating ChatGPT's potential to assist in requirements elicitation processes. 2023 49th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 06-08 September 2023 (págs. 354 - 361). Durres, Albania: IEEE. https://doi.org/10.1109/SEAA60479.2023.00061

Ruan, K., Chen, X., & Jin, Z. (2023). Requirements modeling aided by ChatGPT: An experience in embedded systems. 2023 IEEE 31st International Requirements Engineering Conference Workshops (REW), 04-05 September 2023, Hannover, Alemania (págs. 170 - 177). IEEE. https://doi.org/10.1109/REW57809.2023.00035

Sainani, A., Anish, P. R., Joshi, V., & Ghaisas, S. (2020). Extracting and classifying requirements from software engineering contracts. 2020 IEEE 28th International Requirements Engineering Conference

(RE), 31 August 2020 - 04 September, Zurich, Suiza (págs. 147 - 157). IEEE. https://doi.org/10.1109/RE48521.2020.00026

Sihag, M., Li, Z. S., Dash, A., Arony, N. N., Devathasan, K., Ernst, N., . . . Daniela, D. (2023). A data-driven approach for finding requirements relevant feedback from TikTok and YouTube. 2023 IEEE 31st International Requirements Engineering Conference (RE), 04-08 September 2023, Hannover, Alemania (págs. 111 - 122). IEEE. https://doi.org/10.1109/RE57278.2023.00020

Sommerville, I. (2011). Software engineering (9th ed.). Addison-Wesley. Obtenido de https://engineering.futureuniversity.com/BOOKS%20FOR%20IT/Software-Engineering-9th-Edition-by-Ian-Sommerville.pdf

Stanik, C., Pietz, T., & Maalej, W. (2021). Unsupervised topic discovery in user comments. 2021 IEEE 29th International Requirements Engineering Conference (RE), Notre Dame, IN, USA (págs. 150 - 161). Los Alamitos, CA, USA: IEEE Computer Society. https://doi.org/10.1109/RE51729.2021.00021

Tikayat Ray, A., Cole, B. F., Pinon Fischer, O. J., Bhat, A. P., White, R. T., & Mavris, D. N. (2023). Agile methodology for the standardization of engineering requirements using large language models. Systems, 11(7), 352, 1 - 28. https://doi.org/10.3390/systems11070352

Tizard, J., Devine, P., Wang, H., & Blincoe, K. (1 de Abril de 2023). A software requirements ecosystem: linking forum, issue tracker, and FAQs for requirements management. IEEE Transactions on Software Engineering, 49(4), 2381 - 2393. https://doi.org/10.1109/TSE.2022.3219458

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., . . . Polosukhin, I. (2017). Attention is all you need. En I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Ed.), Advances in Neural Information Processing Systems. 31st Conference on Neural Information Processing Systems (NIPS 2017). 30, págs. 1-11. Long Beach, CA, USA: Curran Associates Inc. Obtenido de https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

Wang, Y., Shi, L., Li, M., & Wang, Q. (2020). A deep context-wise method for coreference detection in natural language requirements. 2020 IEEE 28th International Requirements Engineering Conference (RE), 31 August 2020 - 04 September (págs. 180 - 191). Zurich, Suiza: IEEE. https://doi.org/10.1109/RE48521.2020.00029

Wei, J., Courbis, A.-L., Lambolais, T., Xu, B., Bernard, P. L., & Dray, G. (2023). Zero-shot bilingual app reviews mining with large language models. 2023 IEEE 35th International Conference on Tools with Artificial Intelligence (ICTAI), Atlanta, GA, USA (págs. 898 - 904). Los Alamitos, CA, USA: IEEE Computer Society. https://doi.org/10.1109/ICTAI59109.2023.00135

Wikipedia. (28 de Enero de 2024). Wikipedia. Obtenido de Sitio web de Wikipedia: https://www.wikipedia.org/

Xia, Y., Zhai, S., Wang, Q., Hou, H., Wu, Z., & Shen, Q. (2022). Automated extraction of ABAC policies from natural-language documents in healthcare systems. 2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), 06-08 December 2022, Las Vegas, NV, USA (págs. 1289 - 1296). IEEE. https://doi.org/10.1109/BIBM55620.2022.9995559

Zhang, J., Chen, Y., Liu, C., Niu, N., & Wang, Y. (2023). Empirical evaluation of ChatGPT on requirements information retrieval under zero-shot setting. 2023 International Conference on Intelligent Computing and Next Generation Networks（ICNGN), 17-18 November 2023, Hangzhou, China (págs. 1 - 6). IEEE. https://doi.org/10.1109/ICNGN59831.2023.10396810

Zhang, W., Wang, X., Lai, S., Ye, C., & Zhou, H. (2022). Fine-tuning pre-trained model to extract undesired behaviors from app reviews. 2022 IEEE 22nd International Conference on Software Quality, Reliability and Security (QRS), 05-09 December 2022, Guangzhou, China (págs. 1125 - 1134). IEEE. https://doi.org/10.1109/QRS57517.2022.00115

Zhu, R., Li, W., & Jin, C. (2023). TAG: UML activity diagram deeply supervised generation from business textural specification. 2023 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), 21-24 March 2023, Taipa, Macao (págs. 956 - 961). Los Alamitos, CA, USA: IEEE Computer Society. https://doi.org/10.1109/SANER56733.2023.00116