Técnicas de Inteligencia Artificial y Aprendizaje Automático para Pruebas Automáticas de Sistemas Conscientes de Contexto: Un Mapeo de Literatura

Marlen Trevino-Villalobos[1,2], Christian Quesada-López[1]

[1]Posgrado en Computación e Informática
Escuela de Ciencias de la Computación e Informática
Universidad de Costa Rica
San José – Costa Rica

[2]Escuela de Computación – Instituto Tecnológico de Costa Rica
San Carlos – Costa Rica

{marlen.trevino,cristian.quesadalopez}@ucr.ac.cr, mtrevino@itcr.ac.cr

Abstract. *Test automation in context-aware systems has become challenging due to the increasing complexity, adaptability, and heterogeneity of these systems. In this scenario, artificial intelligence and machine learning techniques have emerged as key approaches to support software testing. This article presents a systematic mapping of the literature that analyzes 73 primary studies published between 2009 and March 2025, focusing on the use of artificial intelligence and machine learning techniques for automatic testing in context-aware systems. The study characterizes the techniques employed, the types of tests addressed, the tools used, the evaluation criteria considered, and the main challenges reported. The results show a predominance of graphical interface testing and a recurring adoption of machine learning techniques, particularly deep neural networks and, in recent work, large-scale language models. However, the mapping highlights challenges related to the scalability of the proposed solutions, the incorporation of new functionalities into existing tools, and the evaluation of performance in dynamic contextual scenarios. As a contribution, this work offers a structured overview of the state of the art in the use of artificial intelligence in automated testing of context-aware systems and identifies opportunities for future research, especially in the integration of emerging techniques, such as generative machine learning, with the potential to transform software testing and quality assurance processes.*

Resumen. *La automatización de pruebas en sistemas conscientes del contexto se ha convertido en un desafío debido a la creciente complejidad, adaptabilidad y heterogeneidad de estos sistemas. En este escenario, las técnicas de inteligencia artificial y aprendizaje automático han emergido como enfoques clave para apoyar las pruebas de software. Este artículo presenta un mapeo sistemático de la literatura que analiza 73 estudios primarios publicados entre 2009 y marzo de 2025, enfocados en el uso de técnicas de inteligencia artificial y aprendizaje automático para pruebas automáticas en CAS. El estudio caracteriza las técnicas empleadas, los tipos de pruebas abordados, las herramientas utilizadas, los criterios de evaluación considerados y los principales desafíos reportados. Los*

resultados muestran una predominancia de pruebas de interfaz gráfica y una adopción recurrente de técnicas de aprendizaje automático, en particular redes neuronales profundas y, en trabajos recientes, modelos de lenguaje de gran escala. No obstante, el mapeo evidencia desafíos relacionados con la escalabilidad de las soluciones propuestas, la incorporación de nuevas funcionalidades en las herramientas existentes y la evaluación del rendimiento en escenarios contextuales dinámicos. Como contribución, este trabajo ofrece una visión estructurada del estado del arte sobre el uso de inteligencia artificial en pruebas automáticas de CAS, e identifica oportunidades de investigación futura, especialmente en la integración de técnicas emergentes, como el aprendizaje automático generativo, con potencial para transformar los procesos de prueba y aseguramiento de la calidad del software.

1. Introducción

Las pruebas tradicionales enfrentan grandes retos al evaluar sistemas conscientes del contexto (*Context-Aware Systems*, CAS) debido a su adaptabilidad en tiempo real, la diversidad de entornos y la generación de múltiples escenarios difíciles de analizar manualmente [Mirza et al. 2021, de Souza Doreste and Travassos 2020]. Estas limitaciones pueden comprometer la calidad del software y la experiencia del usuario, afectando la adopción y el éxito comercial de estas aplicaciones [Travis 2019].

Un CAS percibe, interpreta y adapta su comportamiento en función del entorno, dispositivo o usuario [Roda et al. 2018, Seblewongel et al. 2016]. Emplea información contextual como ubicación, hora, actividad del usuario y estado del dispositivo para ofrecer interacciones personalizadas y optimizar recursos [Salomón et al. 2023]. Ejemplos destacados incluyen aplicaciones móviles contextuales, sistemas de IoT, vehículos inteligentes y plataformas de realidad aumentada [de Sousa Santos et al. 2017].

Sin embargo, la heterogeneidad y variabilidad contextual de los CAS dificultan la realización de pruebas exhaustivas [Freitas and Lelli 2022, Matalonga et al. 2021]. Los enfoques tradicionales son costosos, poco escalables y carecen de herramientas específicas para evaluar adecuadamente estos sistemas [Priya and Rajalakshmi 2022, Luo et al. 2020]. Esto resalta la necesidad de metodologías que incorporen la información de contexto y optimicen los procesos de pruebas.

La inteligencia artificial (IA) ofrece un enfoque prometedor al automatizar pruebas y analizar grandes volúmenes de datos, identificando patrones y problemas de calidad [Matalonga et al. 2022]. Además, la IA permite emular contextos dinámicos y proporcionar recomendaciones para mejorar la experiencia del usuario.

Este estudio presenta una revisión sistemática de la literatura sobre el uso de inteligencia artificial en pruebas automáticas para CAS. El objetivo es analizar cómo los modelos de IA han sido utilizados para apoyar los procesos de pruebas en este tipo de sistemas, así como identificar las herramientas existentes, las estrategias de implementación, los criterios de evaluación y los principales retos reportados en la literatura. Para ello, se analizaron 73 estudios primarios publicados entre 2009 y marzo de 2025.

El análisis se organiza en cinco preguntas de investigación (RQ): RQ1: ¿Cuáles son las características de los modelos de IA utilizados en las pruebas automáticas para

sistemas contextuales?, RQ2: ¿Qué características tienen las herramientas que emplean modelos de IA en pruebas automáticas para sistemas contextuales?, RQ3: ¿Cómo se implementan los modelos de IA en el desarrollo de pruebas automáticas para sistemas contextuales?, RQ4: ¿Cuáles son los criterios para evaluar la efectividad de los modelos de IA en pruebas automáticas para sistemas contextuales? y RQ5: ¿Qué retos reporta la literatura sobre el uso de modelos de IA en pruebas automáticas para sistemas contextuales?

Con base en estas preguntas de investigación, este estudio contribuye al estado del arte mediante: (i) la caracterización de los modelos de IA utilizados en pruebas automáticas para CAS; (ii) el análisis de las herramientas que incorporan modelos de IA para apoyar los procesos de prueba en este tipo de sistemas; (iii) la identificación de las estrategias utilizadas para integrar modelos de IA en el desarrollo de pruebas automáticas; (iv) la síntesis de los criterios utilizados en la literatura para evaluar la efectividad de estos modelos; y (v) la identificación de los principales retos y oportunidades de investigación en el uso de IA para pruebas automáticas en CAS.

El artículo se organiza de la siguiente manera: la Sección 2 aborda trabajos relacionados, la Sección 3 detalla el diseño metodológico, la Sección 4 presenta y analiza los resultados, y la Sección 5 expone las conclusiones del estudio.

2. Antecedentes

Los estudios previos que abordan la integración de modelos de IA en procesos de pruebas de software han sentado las bases para comprender este campo de investigación.

Trudova et al. [Trudova et al. 2020] analizaron 34 estudios primarios publicados hasta 2019, destacando el aprendizaje automático como la técnica más utilizada debido a su capacidad para adaptarse a sistemas dinámicos. En una línea similar, Lima et al. [Lima et al. 2020] identificaron los beneficios del aprendizaje automático en las pruebas de caja negra, particularmente en la detección de patrones. Por su parte, Sousa et al. [de Sousa Santos et al. 2017] y Matalonga et al. [Matalonga et al. 2017] señalaron la falta de validación completa en cuanto a la conciencia del contexto en los enfoques existentes. Posteriormente, Matalonga et al. [Matalonga et al. 2021] concluyeron que las soluciones basadas en IA para pruebas suelen ser específicas de ciertos dominios y que aún presentan dificultades para su transferencia a otros contextos. Más adelante, Matalonga et al. [Matalonga et al. 2022] identificaron desafíos relacionados con la selección de conjuntos de prueba y la ausencia de estrategias listas para su adopción generalizada. Finalmente, Islam et al. [Islam et al. 2023] destacaron que la IA facilita la automatización de pruebas mediante la generación de casos de prueba, la predicción de defectos y su aplicación en pruebas de caja blanca.

En conjunto, estos estudios han contribuido a identificar técnicas de IA utilizadas en pruebas de software y a evidenciar sus beneficios y limitaciones. Sin embargo, la mayoría de las revisiones existentes se centran en el uso general de la IA en pruebas o en aspectos específicos como el aprendizaje automático o la generación de casos de prueba, sin profundizar de manera sistemática en su aplicación en CAS ni en las características de las herramientas y modelos utilizados.

En este contexto, la presente revisión amplía el conocimiento existente al analizar 73 artículos publicados entre 2009 y marzo de 2025 sobre el uso de modelos de IA

en pruebas automáticas para CAS. En particular, este estudio incorpora 28 trabajos recientes publicados a partir de 2023, los cuales no han sido considerados en revisiones sistemáticas previas. A diferencia de dichos estudios, esta revisión no solo identifica las técnicas empleadas, sino que también caracteriza de manera integral (i) los modelos de IA utilizados, (ii) las herramientas que los implementan, (iii) las estrategias de integración de estos modelos en los procesos de prueba y (iv) los criterios utilizados para evaluar su efectividad. De esta manera, el estudio proporciona una visión más estructurada del estado del arte y permite identificar tendencias, retos y oportunidades de investigación en la automatización de pruebas para CAS.

3. Metodología

El estudio según los lineamientos de [Petersen et al. 2015, Kitchenham et al. 2009].

3.1. Proceso de búsqueda

La cadena de búsqueda se diseñó según las preguntas de investigación utilizando el modelo PICO, organizando palabras clave por población (P), intervención (I), comparación (C) y resultados esperados (O). La población objetivo son sistemas conscientes del contexto, la intervención abarca pruebas automáticas con IA, sin elementos de comparación, y los resultados esperados incluyen herramientas, implementación del modelo, criterios de evaluación y retos reportados.

La cadena de búsqueda resultante fue: *((“context-based” OR “context-aware”) AND (“system*” OR “application*”)) AND (“artificial intelligence” OR “AI” OR “machine learning” OR “neural network*” OR “deep learning”) AND (“automatic test*” OR “automated test*”).*

Se realizaron búsquedas en Scopus, IEEE Xplore, ACM, Web of science y SpringerLink en marzo de 2025, obteniendo 1310 artículos: 526 en Scopus, 124 en IEEE Xplore, 376 en ACM, 96 en Web of science y 188 en SpringerLink. Tras eliminar 34 duplicados, se analizaron 1276 artículos candidatos.

Aunque se definió una cadena de búsqueda general, esta fue adaptada a la sintaxis específica de cada base de datos. En particular, se ajustaron operadores booleanos, el uso de comodines y los campos de búsqueda (título, resumen y palabras clave) según las restricciones de cada motor. Las versiones específicas de las cadenas de búsqueda utilizadas en cada base de datos se encuentran disponibles en la hoja denominada “Cadena” del documento https://tinyurl.com/2cez2rnp.

3.2. Proceso de selección de estudios

Se establecieron criterios de inclusión y exclusión para identificar los artículos relevantes. La selección se realizó con base en el título, resumen y palabras clave, con lectura completa en caso de dudas. Los criterios de inclusión fueron: (I1) estudios primarios, (I2) relacionados con computación, (I3) que traten herramientas y modelos de IA aplicados a pruebas de software y (I4) escritos en inglés o español. Los criterios de exclusión fueron: (E1) artículos no disponibles en texto completo, (E2) estudios fuera del área de CAS, y (E3) artículos que NO hayan sido publicados en una conferencia o revista científica (como entradas de blog, páginas web, presentaciones, entre otros). Tras la selección, se incluyeron 73 artículos para el análisis y extracción (ver Figura 1).

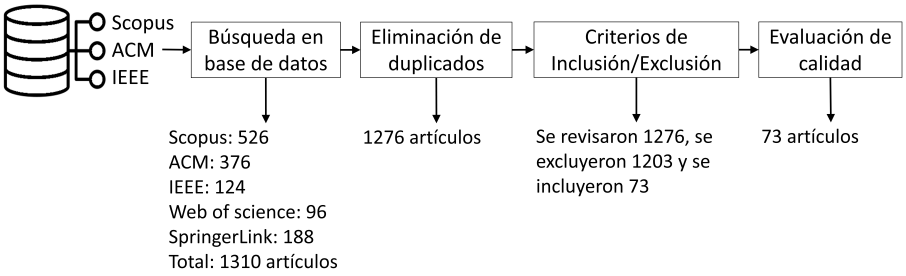

Figura 1. Proceso de selección de estudios.

La evaluación de calidad consideró seis criterios: modelo de IA (Q1), herramientas (Q2), problema investigado (Q3), objetivos (Q4), método de investigación (Q5) y resultados estadísticos (Q6). Estos criterios fueron aplicados de manera secuencial, iniciando con la revisión de título y resumen, seguida de la lectura completa en los casos en que existían dudas sobre la relevancia del estudio. Cada criterio fue calificado de 0 a 3, siendo 3 la máxima calidad. La mediana de Q3, Q5 y Q6 fue de 3, indicando un alto estándar en estos aspectos. En Q1, Q2 y Q4, la mediana superó $\mathbf{2,5}$. Esta diferencia refleja que, aunque la calidad es buena, hubo variaciones en la precisión de la descripción de los modelos y herramientas de IA, así como en la rigurosidad de los métodos utilizados en la investigación. A pesar de estas diferencias menores, los estudios demostraron una buena calidad para asegurar la confiabilidad y la relevancia de los resultados obtenidos en esta revisión sistemática de literatura.

La evaluación de calidad fue realizada por la primera autora y revisada por el segundo investigador en casos de ambigüedad, con el fin de asegurar la consistencia en la asignación de puntuaciones.

3.3. Extracción y análisis de datos

El formulario de extracción consta de seis categorías: información general, modelos de IA (RQ1), herramientas (RQ2), implementación (RQ3), criterios de evaluación (RQ4) y retos (RQ5). El Cuadro 1 detalla los elementos extraídos de los estudios.

Cuadro 1. Datos extraídos de los estudios.

Categoría	Elementos de datos
Información general	Identificador, título, autores, año, resumen, DOI, y base de datos.
Modelos de IA (RQ1)	Nombre, propósito, tipo de sistema contextual, tipo de prueba, nivel de prueba, técnica de prueba y nivel de automatización.
Herramientas (RQ2)	Nombre de la herramienta, arquitectura, funciones, versión, URL del código, lenguaje de programación, resultados reportados y estado de adopción.
Implementación del modelo (RQ3)	Identificación del tipo específico de modelo de IA utilizado, clasificación del modelo, técnica de entrenamiento, tipo de datos utilizados, requerimientos de hardware/software y consumo de recursos.
Criterios para evaluar la efectividad de los modelos (RQ4)	Métricas de evaluación, cobertura de pruebas, identificación de errores, rendimiento del sistema, facilidad de uso, adaptabilidad a diferentes contextos, porcentaje de reducción de tiempo y porcentaje de reducción de costo.
Retos (RQ5)	Desafíos, consideraciones éticas, riesgos de seguridad, perspectivas futuras y retos.

3.4. Amenazas a la validez

Con el fin de minimizar el riesgo de omitir estudios relevantes, se llevó a cabo una búsqueda exploratoria inicial para la construcción de la cadena de búsqueda, la cual fue poste-

riormente validada mediante ensayos preliminares. No obstante, existe la posibilidad de que algunos términos de búsqueda no hayan sido incluidos o que otros no hayan sido los más adecuados, lo que podría haber afectado la exhaustividad de la revisión.

El estudio se basó en cinco bases de datos académicas y consideró publicaciones entre 2009 y marzo de 2025. Si bien este rango temporal fue seleccionado para capturar los avances más recientes en el área, es posible que se hayan excluido contribuciones relevantes anteriores a este período. En este sentido, aunque la revisión se centra en el estado actual del campo, algunos enfoques fundamentales podrían no haber sido considerados.

Asimismo, la restricción de la cadena de búsqueda a términos explícitos relacionados con CAS puede haber limitado la identificación de estudios aplicables que no emplean dicha terminología de manera directa, lo cual representa una amenaza a la validez del estudio.

Por otra parte, el proceso de selección, extracción y categorización de los estudios fue realizado por la primera autora y posteriormente validado por el segundo autor. Aunque la participación de un número reducido de revisores podría introducir sesgos, estos fueron mitigados mediante procesos de validación cruzada y discusión de discrepancias hasta alcanzar consenso.

Finalmente, la naturaleza dinámica del área de estudio implica que continuamente emergen nuevos enfoques y herramientas. Por ello, los resultados de esta revisión deben interpretarse como una representación del estado del arte hasta el momento de su realización, reconociendo la necesidad de futuras actualizaciones.

4. Resultados

En esta sección se presentan los resultados obtenidos de la revisión sistemática de literatura. Los formularios resultantes del protocolo y la lista completa de los estudios primarios incluidos (n = 73), junto con sus metadatos (título, autores, año, DOI y fuente), se encuentra disponible en la hoja denominada “Extracción” del documento https://tinyurl.com/2cez2rnp, con el fin de garantizar la trazabilidad y reproducibilidad de la presente investigación. Los estudios son identificados con la nomenclatura ID:1-ID:73 que es utilizada para el reporte de resultados.

4.1. RQ1: características de los modelos de inteligencia artificial

La Figura 2 muestra un crecimiento anual en uso de IA para apoyar el proceso de pruebas. Según la clasificación propuesta por Graham et al. sobre tipos de pruebas [Graham et al. 2008], las pruebas aplicadas a la GUI son las más comunes (29), lo que refleja un enfoque creciente en la interacción usuario-IA. Las pruebas unitarias ocupan el segundo lugar en la implementación de herramientas de prueba con IA (10), seguidas por las pruebas de regresión (5), juegos (4) y seguridad (4).

Según la taxonomía propuesta por [Vissers-Similon et al. 2024], las técnicas de inteligencia artificial se clasifican en siete categorías. A partir de la clasificación de los estudios presentada en el Cuadro 2, se observa que los modelos más utilizados corresponden a redes neuronales (38), modelos de lenguaje de gran tamaño (LLM) (17) y aprendizaje por refuerzo (8). Entre estos, ChatGPT destaca como la técnica más frecuente, con 16 estudios que reportan su utilización.

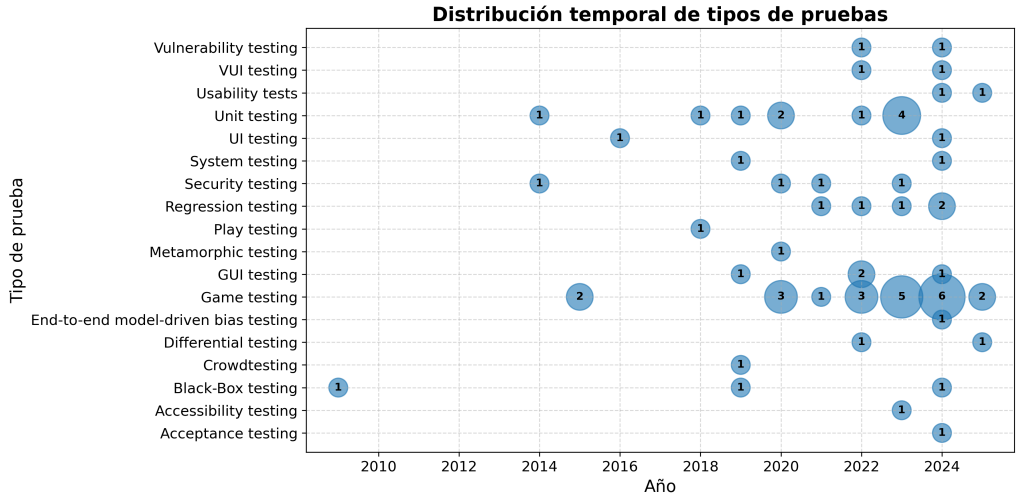

Figura 2. Distribución temporal de tipos de pruebas.

Las redes neuronales sobresalen por su capacidad para aprender representaciones complejas de los datos y realizar predicciones con alto nivel de precisión. Por su parte, el creciente uso de los LLM se relaciona con su habilidad para procesar grandes volúmenes de información textual y generar respuestas coherentes, lo que los convierte en herramientas clave para aplicaciones de procesamiento del lenguaje natural (NLP).

En contraste, la escasa presencia de estudios en la categoría de Generative Machine Learning evidencia una oportunidad para explorar su potencial en la generación automática de contenido y en el desarrollo de modelos más dinámicos. En conjunto, estos resultados sugieren que los enfoques basados en aprendizaje automático se han consolidado como una alternativa relevante para abordar la complejidad y variabilidad del contexto en este tipo de sistemas.

LLM y redes neuronales lideran las pruebas GUI y unitarias (ver Cuadro 3). Se identificaron 14 tipos de CAS, destacando las pruebas GUI en aplicaciones móviles por su énfasis en la usabilidad y experiencia del usuario (ver Cuadro 4). El aprendizaje por refuerzo y los algoritmos genéticos complementan estas pruebas, demostrando la adaptabilidad de la IA a diversos escenarios. Aunque aplicaciones móviles, realidad aumentada, vehículos autónomos y comercio electrónico son dominios comunes, otros sistemas siguen sin ser explorados, lo que abre oportunidades para futuras investigaciones.

El Cuadro 5 muestra que la mayoría de las herramientas se enfocan en la generación de pruebas (32) y la detección de defectos (22), mientras que la ejecución (14) y el monitoreo (5) reciben menor atención. Los propósitos más frecuentes son la generación automática de casos de prueba (16), de entradas de prueba (13) y la detección de errores (11).

En síntesis, las herramientas se concentran principalmente en la generación automática de pruebas, evidenciando una menor atención a otras actividades como la ejecución y el monitoreo, lo que sugiere la necesidad de enfoques más integrales.

Cuadro 2. Técnicas de IA reportadas.

Técnicas de IA agrupadas	Técnicas de IA identificadas	Técnica de IA específica	Código
Agent-based System	Agent-based System	Agent-based System	[ID:4, 18]
	Reinforcement Learning	Deep Q-Network (DQN)	[ID:39, 63]
		DNN	[ID:13, 26]
		Multi-Armed Bandit (MAB)	[ID:37]
		Q-learning	[ID:22, 52]
		Random	[ID:42]
Classic Machine Learning	Neural Network	Balanced Random Forest (BRF)	[ID:36]
		Cluster	[ID:60]
		Contrastive Language–Image Pre-training	[ID:59]
		Decision Tree (DT)	[ID:8, 15, 27, 46]
		Deep learning	[ID:31, 34]
		DLRM-like	[ID:73]
		Fast Gradient Signed Method (FGSM)	[ID:62]
		Faster-RCNN	[ID:24, 50]
		In-context learning	[ID:59]
		LR	[ID:15, 46]
		Multimodal deep learning model	[ID:68]
		NeuroEvolution of Augmenting Topologies	[ID:23]
		Random Forest	[ID:11, 15, 20]
		Redes Neuronales Clásicas	[ID:20]
		Redes Neuronales Convolucionales	[ID:20, 28]
		Redes Neuronales Recurrentes	[ID:20, 28, 72]
		ResNet	[ID:32, 44, 54, 64]
		Standard convolution layer	[ID:34]
		Support Vector Machine	[ID:9, 15]
		VIT	[ID:34]
		YOLOv7-tiny	[ID:17, 33, 70]
Evolutionary Computing	Evolutionary Computing	Smart Mutation	[ID:7]
	Genetic Algorithm	Selection, Crossover, Mutation	[ID:3, 16, 66]
Expert System	Expert System	Automata Learning, Search & Models	[ID:1, 6, 2, 5]
Graph Machine Learning	Graph Machine Learning	Spectral clustering	[ID:21]
Transformer Model	Computer Vision	OCR	[ID:40]
		SIFT	[ID:40]
	LLM	ChatGPT	[ID:38, 41, 47, 48, 49, 51, 53, 55, 57, 58, 59, 61, 65, 67, 69, 70]
		DeepSeekCoder-33BInstruct	[ID:56]
		Gemini	[ID:56]
		LLaMA 2	[ID:55]
		PaLM 2	[ID:55, 56]
		StarCoder2	[ID:56]
	NLP	BERT	[ID:25, 34, 35, 43, 67]
		N-gram model	[ID:45]
		SentenceTransformer	[ID:29]
		Tokenization	[ID:20]
		Word2Vec	[ID:19, 30, 40]

4.2. RQ2: Herramientas reportadas

En los tres primeros meses del año 2025 se identificaron cuatro herramientas basadas en IA para pruebas automáticas en CAS (ID:70-73). En 2024 se documentaron veinticuatro (ID:46–69), en 2023 catorce (ID:32-45) y en 2022 nueve (ID:23-31). En 2021 se reportaron tres (ID:20-22), en 2020 se registraron seis (ID:14-19). En 2019 se encontraron cuatro (ID:10-13) y en 2018 dos (ID:7-9). Finalmente, se documentó una herramienta en 2016 (ID:6), dos en 2015 (ID:4-5), dos en 2014 (ID:2-3) y una en 2009 (ID:1).

Las tecnologías utilizadas en pruebas automáticas apoyadas con IA varían en frecuencia y tipo. El framework más empleado es UI Automator (ID: 6, 14, 22, 34, 38, 45, 51, 53, 61, 68, 70), mientras que entre las librerías destacan TensorFlow (ID:17, 20, 27, 29, 31, 62, 73) y PyTorch (ID:13, 29, 32, 39, 63, 64, 73). Otras herramientas incluyen Sklearn (ID:20,46), StanfordCoreNLP (ID:25, 67), Keras (ID:27), Pattern (ID:69), Google Play Scraper (ID:46), Whisker (ID:23), HMCBL (ID:43), Soot (ID:28), GUI testing

Cuadro 3. Tipos de pruebas según modelo de IA.

Tipo de prueba	Modelo de IA	Código
Acceptance Testing	LLM	[ID:47]
Accessibility Testing	NLP	[ID:35]
Black-Box testing	Expert System	[ID:1]
	Neural network	[ID:33, 62]
Crowdtesting	Neural network	[ID:10]
Differential testing	Neural network	[ID:29]
	NLP	[ID:73]
End-to-end model-driven bias testing	LLM	[ID:69]
Game testing	Neural network	[ID:13, 23, 26, 60]
	Reinforcement Learning	[ID:13, 26]
GUI testing	Agent-Based System	[ID:4]
	Computer vision	[ID:40]
	Expert System	[ID:5]
	LLM	[ID:38, 49, 53, 55, 57, 59]
	Neural network	[ID:14, 15, 24, 30–32, 34, 44, 50, 54, 59, 68, 70, 72]
	NLP	[ID:15, 19, 30, 34, 40]
	Reinforcement Learning	[ID:22, 37, 39, 42, 52]
Metamorphic Testing	Neural network	[ID:17]
Play Testing	Evolutionary computing	[ID:7]
Regression testing	Graph Machine Learning	[ID:21]
	LLM	[ID:51, 58]
	Neural network	[ID:27]
	NLP	[ID:43]
Security testing	Expert System	[ID:2]
	Neural network	[ID:9, 20, 36]
	NLP	[ID:20]
System testing	NLP	[ID:12, 63]
	Reinforcement Learning	[ID:63]
UI testing	Expert System	[ID:6]
	LLM	[ID:61]
Unit testing	Agent-Based System	[ID:18]
	Genetic algorithm	[ID:3, 16]
	LLM	[ID:41, 48, 56, 65]
	Neural network	[ID:8, 11, 64]
Usability test	LLM	[ID:71]
	Neural network	[ID:46]
VUI testing	LLM	[ID:67]
	NLP	[ID:25, 67]
Vulnerability testing	Genetic algorithm	[ID:66]
	Neural network	[ID:28]

tool (ID:15), FBLearner (ID:8) y FlowDroid (ID:28). Sin embargo, 32 estudios no especifican la tecnología utilizada (ID:4, 5, 10, 12, 21, 24, 26, 30, 33, 35-37, 40-42, 44, 47-50, 52, 54-60, 65, 66, 71, 72), lo que resalta la necesidad de mejorar la documentación y estandarización en el campo.

De las herramientas reportadas, 51 no han sido adoptadas (ID:1-7, 9-19, 23, 25, 27, 32, 35, 37, 38, 40-42, 46, 48-52, 55-57, 59-63, 65-73). Diez están en prueba piloto (ID:20, 24, 26, 28, 29, 30, 31, 43, 44, 47). Ocho se encuentran en adopción inicial (ID:21, 22, 34, 39, 45, 53, 54, 64). Finalmente, cuatro han alcanzado la adopción completa (ID: 8, 33, 36, 58). Esto refleja un creciente interés en la temática, pero también desafíos en la implementación efectiva.

4.3. RQ3: Implementación de los modelos de inteligencia artificial

En las pruebas automáticas de CAS, los datasets reflejan una preferencia por datos reales, utilizados en 67 estudios (ID:1-11, 13-15, 17-34, 36-56, 58-64, 66-68, 70-73), resaltando su relevancia práctica. Solo un estudio (ID: 35) combinó datos reales y sintéticos, mientras que cinco (ID:12, 16, 57, 65, 69) no especificaron el tipo de datos, afectando la

Cuadro 4. Tipos de pruebas según tipo de sistema contextual.

Tipo de prueba	Tipo de sistema contextual	Código
Acceptance Testing	FinTech software	[ID:47]
Accessibility Testing	Automatic speech recognition	[ID:35]
Black-Box testing	Context-aware system	[ID:1]
	Deep learning software	[ID:62]
	Games	[ID:33]
Crowdtesting	Mobile Apps	[ID:10]
Differential testing	Deep learning software	[ID:29, 73]
End-to-end model-driven testing	Large Language Model (LLM)	[ID:69]
Game testing	Games	[ID:13, 23, 26, 60]
GUI Testing	Applications in the Internet of Robotic Things	[ID:72]
	Augmented Reality	[ID:54]
	Context-aware system	[ID:31]
	Internet of Vehicles App	[ID:32]
	Mobile Apps	[ID:4, 5, 14, 15, 19, 22, 24, 30, 34, 37–40, 42, 44, 45, 49, 50, 52, 53, 55, 57, 59, 68, 70]
Metamorphic Testing	Object Detection System	[ID:17]
Play testing	Games	[ID:7]
Regression testing	Deep learning software	[ID:27]
	Large Language Model (LLM)	[ID:58]
	Machine learning software	[ID:21]
	Mobile Apps	[ID:51]
	Semantic Applications	[ID:43]
Security testing	Business Flow Tampering for Digital Content Service	[ID:36]
	E-commerce platforms	[ID:20]
	Mobile Apps	[ID:2, 9]
System testing	Chatbots	[ID:12]
	Sistemas aéreos no tripulados	[ID:63]
UI testing	Mobile Apps	[ID:6, 61]
Unit testing	Context-aware system	[ID:65]
	Deep Neural Networks	[ID:64]
	Intelligent avionics systems	[ID:18]
	Large Language Model (LLM)	[ID:56]
	Microservice applications	[ID:11]
	Mobile Apps	[ID:8, 41]
	Robots	[ID:3]
	Semantic Applications	[ID:48]
	Sistema Viajero Multimodo (MMTS)	[ID:16]
Usability tests	Mobile Apps	[ID:46, 71]
VUI testing	Voice-based virtual assistants	[ID:25, 67]
Vulnerability testing	Mobile Apps	[ID:28]
	Smart contracts	[ID:66]

replicabilidad.

La mayoría de los datasets (29) fueron creados para cada estudio (ID:20-23, 26, 28, 30, 31, 33, 34, 36, 37, 39, 40-42, 45-47, 49, 50-52, 54, 56, 59, 60, 66, 68), evidenciando la necesidad de adaptar los datos a contextos específicos. Otros datasets utilizados incluyen Rico (ID:24, 38, 44, 50, 53, 55, 70, 72), Themis's Dataset (ID: 49, 51), Baidu crowd-sourcing platform (ID:50, 10) y RoBERTa (ID:67, 25).

Cuadro 5. Número de herramientas según su propósito.

Clasificación	Propósito	Frecuencia	Código
Generación automática de pruebas	Generación automática de casos de prueba	16	[ID:3, 7, 8 18, 21, 22, 41, 42, 47, 52, 57, 58, 62, 63, 65, 69]
	Generación automática de entradas de prueba	13	[ID:4, 6, 17, 23, 25, 34, 36, 38, 48, 53, 64, 67, 70]
	Generación automática de patrones de prueba	1	[ID:2]
	Herramientas que automatizan la creación de casos y escenarios de prueba para garantizar una cobertura adecuada	1	[ID:12]
	Generación automática de casos de prueba	1	[ID:19]
Detección de defectos	Detección de errores	11	[ID:5, 11, 13, 24, 26, 27, 32, 43, 45, 46, 73]
	Detección de vulnerabilidades	3	[ID:9, 28, 66]
	Detectar violaciones del Principio de Minimización de Datos	2	[ID:29, 39]
	Predecir automáticamente la categoría de vulnerabilidad	1	[ID:20]
	Detectar y clasificar problemas	1	[ID:10]
	Detectar errores de implementación en el código	1	[ID:16]
	Predecir automáticamente la calidad de la ubicación de objetos en escenas de AR del mundo real	1	[ID:54]
	Evaluación de código real	1	[ID:56]
	Detección automática de inconsistencias en los datos	1	[ID:59]
Ejecución de pruebas	Detectar los widgets de la interfaz gráfica de usuario	1	[ID:33]
	Ejecución automática de pruebas de GUI	8	[ID:14, 15, 30, 31, 44, 49, 61, 72]
	Generación automática de modelos de comportamiento para pruebas	1	[ID:1]
	Navegación automatizada de pantallas	1	[ID:68]
	Priorizar los eventos de UI	1	[ID:37]
	Probar automáticamente la accesibilidad de los sistemas	1	[ID:35]
	Reproducción automática de errores	1	[ID:51]
Evaluación y monitoreo	Detección y localización de violaciones a principios de diseño	2	[ID:50, 71]
	Generación de informes sobre violaciones a principios de diseño	1	[ID:55]
	Inspeccionar el contenido que muestran las cámaras	1	[ID:60]
	Explotar la consistencia en las apariencias de los widgets	1	[ID:40]

Se identificaron datasets empleados solo una vez, como ArduPilot (ID:63), MoTIF (ID:61), COCO (ID:17), Qasper y Assignment (ID:48), ANDROR2+ (ID:51), ReCDroid (ID:51), HUMANEVAL (ID:58), Common Voice (ID:35), VSTDR-2023 (ID:32), CodeSearch-Net (ID:43) y SFData (ID:27). A pesar del crecimiento en el desarrollo de datasets específicos, estos hallazgos evidencian la necesidad de desarrollar enfoques más generalizables y datasets abiertos que faciliten la comparación y reproducibilidad de los estudios en este campo.

El análisis evidencia que los modelos preentrenados (12) son los más frecuentes, dado que ChatGPT y BERT destacan en las pruebas automáticas de CAS. Estos modelos incluyen ID:9, 14, 17, 25, 29, 30, 46, 59, 64, 65, 67 y 70. Los modelos supervisados (9) ocupan el segundo lugar, siendo comunes en pruebas de GUI. Entre ellos se encuentran

ID:20, 27, 30, 31, 32, 34, 60, 63 y 72. Los modelos no supervisados (4) tienen menor presencia debido a su mayor complejidad. En esta categoría están ID:19, 21, 39 y 56. Los modelos por refuerzo (3) también presentan una menor frecuencia, estos incluyen ID: 13, 22 y 52. Finalmente, el aprendizaje auto supervisado (1) es una técnica emergente con gran potencial, representada por ID: 68.

4.4. RQ4: Criterios para evaluar la efectividad de los modelos de inteligencia artificial

El análisis revela una amplia variedad de métricas para evaluar pruebas de software, reflejando la complejidad del campo. Las más reportadas incluyen Code coverage (4 veces: ID:37, 47, 49 , 52), Number of bugs detected (2 veces: ID:49, 58), Activity coverage (3 veces: ID:8, 22, 57) y Method coverage (2 veces: ID: 22, 68), resaltando la importancia de garantizar una evaluación integral. Otras métricas, como LOC coverage (ID:52), Number of incorrect/correct cache misses/hits (ID:48), Business Scenario Coverage (BSC) (ID:47), Crash-revealing capability (ID:37), Triggered page/activity number (ID:38), Screen coverage (ID:57), Round-Trip Correctness (RTC) (ID:56), Bug detection rate (ID: 53), Execution time, Number of inconsistencies (ID:73), Tolerance level (ID: 69), Line/Branch Coverage (ID:26) y Feature coverage (ID:57), fueron reportadas solo una vez, evidenciando la necesidad de métricas adaptadas a contextos específicos.

En modelos de IA, Recall (22 veces: ID:10, 14, 15, 17, 19, 21, 24, 27, 28, 32-34, 39, 40, 46, 50, 54, 55, 59, 66-68), Precision (21 veces: ID:14, 15, 17, 19, 24, 27, 28, 32-34, 39, 40, 43, 46, 50, 54, 55, 59, 66, 67, 68) y F1-Score (13 veces: ID: 21, 24, 27, 32, 39, 40, 44, 46, 54, 55, 59, 66, 68) son las métricas más utilizadas por su capacidad para medir exactitud y exhaustividad. False Positive y False Negative identifican errores, mientras que Accuracy ofrece una visión general del rendimiento. Otras métricas, como Reproducibility (ID: 51), Character Error Rate, Overhead (ID: 32), Match Error Rate (MER), Word Information Lost y Word Error Rate (WER) (ID: 35), aparecen ocasionalmente, abordando necesidades específicas. La diversidad de métricas refleja la complejidad de las pruebas automáticas, resaltando la necesidad de enfoques adaptados y colaborativos para mejorar la evaluación de las pruebas automáticas apoyadas con IA en CAS.

4.5. RQ5: Retos reportados

El análisis de los estudios incluidos revela diversos desafíos asociados con el uso de modelos de IA en pruebas automáticas para CAS. Uno de los retos más frecuentes es la incorporación de nuevas funcionalidades en las herramientas de prueba, reportado en 25 estudios (ID:2, 5, 7–10, 13–15, 17–19, 22, 24, 27, 32, 38, 45, 46, 55, 61–64, 66). Este resultado refleja la necesidad de desarrollar soluciones más completas que permitan abordar la creciente complejidad de los sistemas contextuales.

Otro desafío relevante corresponde a la mejora del rendimiento de los modelos de IA, identificado en 24 estudios (ID:4, 5, 7, 10, 17, 19–24, 27, 32, 37, 39, 44, 46, 49, 50, 51, 55, 58, 70, 71). A pesar de los avances recientes, optimizar el rendimiento y la eficiencia computacional continúa siendo una preocupación central en el desarrollo de herramientas de prueba basadas en IA.

Asimismo, la migración de pruebas a diferentes plataformas o dominios constituye un reto importante, señalado en 20 estudios (ID:4, 7, 13, 14, 18–20, 25, 35, 38, 40, 42,

45, 54, 57, 60, 64, 70, 71, 73). Esto evidencia la dificultad de adaptar los modelos y herramientas existentes a distintos entornos tecnológicos, lo que limita su reutilización y escalabilidad.

Otros desafíos identificados incluyen la exploración de nuevos modelos de IA (ID:9, 13, 14, 18, 25, 27, 39, 40, 42, 46, 61–64, 67, 69), la optimización de parámetros de los modelos (ID:5, 7, 10, 17, 25, 47–49, 60, 62, 67, 71) y la definición de modelos de calidad y métricas adecuadas para evaluar la efectividad de las pruebas (ID:5, 7, 14, 20, 48, 52, 62, 63, 67). Además, aspectos como la mejora de la facilidad de uso de las herramientas (ID:4, 19, 44, 55, 61, 63, 71), el desarrollo de nuevas plataformas de prueba (ID:2, 4, 5, 8, 14, 19, 50, 61–63) y la realización de estudios de ablación para comprender mejor el comportamiento de los modelos (ID:7, 14, 16, 48, 62) también fueron mencionados en la literatura.

En conjunto, estos retos evidencian que, aunque la inteligencia artificial ha demostrado un gran potencial para apoyar las pruebas automáticas en CAS, aún existen importantes desafíos relacionados con la escalabilidad, la generalización de los modelos y la evaluación sistemática de su efectividad. Estos hallazgos abren oportunidades para futuras investigaciones orientadas al desarrollo de herramientas más adaptables, la integración de modelos de IA emergentes y la creación de marcos de evaluación estandarizados para las pruebas automáticas en CAS.

5. Conclusiones y trabajo futuro

Este estudio presenta una revisión sistemática de la literatura sobre el uso de inteligencia artificial en pruebas automáticas para CAS. A partir del análisis de 73 estudios publicados entre 2009 y marzo de 2025, se identifican tendencias, herramientas, modelos de IA, criterios de evaluación y principales retos de la automatización de pruebas en CAS.

Los resultados evidencian un crecimiento sostenido en la investigación del área durante los últimos años. Las pruebas enfocadas en aplicaciones móviles predominan en la literatura, lo que refleja la importancia de la interacción usuario–sistema en los CAS. En cuanto a las técnicas utilizadas, los modelos basados en redes neuronales y los modelos de lenguaje de gran tamaño se posicionan como las alternativas más utilizadas, debido a su capacidad para procesar grandes volúmenes de datos y adaptarse a escenarios dinámicos.

El estudio también revela que la mayoría de las herramientas propuestas se orientan a la generación automática de casos de prueba y a la detección de defectos, mientras que aspectos como la ejecución automática de pruebas, el monitoreo continuo y la evaluación integral de los sistemas han recibido menor atención. Asimismo, se observa una baja adopción industrial de las herramientas identificadas, lo que sugiere la existencia de barreras relacionadas con la escalabilidad, la integración en entornos reales y la estandarización de metodologías.

En relación con la evaluación de los modelos de IA, se identificó una gran diversidad de métricas utilizadas en los estudios analizados. Aunque métricas como Precision, Recall, F1-score y cobertura de código son las más frecuentes, la falta de criterios de evaluación estandarizados dificulta la comparación de resultados entre diferentes propuestas.

A partir del análisis realizado, se identifican varios retos abiertos en el área. Entre los más relevantes se encuentran la mejora del rendimiento de los modelos de IA, la adap-

tación de herramientas de prueba a múltiples plataformas y dominios, la optimización de parámetros de los modelos y la definición de métricas de evaluación más consistentes. Asimismo, muchos estudios dependen de datasets creados específicamente para cada investigación, lo que limita la reproducibilidad y comparabilidad de los resultados.

Desde una perspectiva práctica, este estudio permite derivar varias lecciones aprendidas. En primer lugar, la efectividad de las pruebas en CAS depende en gran medida de la capacidad de gestionar adecuadamente la variabilidad del contexto, por lo que se recomienda a la industria adoptar enfoques que integren la generación dinámica de escenarios contextuales en lugar de limitarse a configuraciones predefinidas. En segundo lugar, la incorporación de modelos de inteligencia artificial, especialmente LLM, muestra un alto potencial para automatizar tareas como la generación de pruebas y la detección de defectos; sin embargo, su uso debe acompañarse de mecanismos de validación y control que mitiguen problemas como la falta de explicabilidad o la generación de resultados inconsistentes. Finalmente, se recomienda promover el desarrollo de herramientas interoperables y datasets abiertos que permitan validar soluciones en entornos reales, contribuyendo así a cerrar la brecha entre la investigación académica y su adopción en la industria.

Como líneas de trabajo futuro, se identifican varias oportunidades relevantes. Primero, resulta necesario ampliar la investigación hacia tipos de sistemas contextuales menos explorados, como sistemas de realidad mixta, entornos de automatización industrial y aplicaciones de Internet de las Cosas. Segundo, se requiere el desarrollo de datasets abiertos y estandarizados que faciliten la reproducibilidad de los estudios. Por último, se podría explorar la integración de técnicas emergentes de IA, incluyendo modelos generativos y enfoques híbridos que combinen diferentes técnicas de aprendizaje automático, con el objetivo de mejorar la eficiencia y efectividad de las pruebas automáticas en CAS.

6. Agradecimientos

Esta investigación fue parcialmente financiada por el Proyecto No. 834-C5218 ECCI-CITIC-UCR y el Programa de Posgrado en Computación e Informática de la Universidad de Costa Rica (UCR), así como por el proyecto IAutoTest: Integración de Técnicas de Inteligencia Artificial para la Automatización de Pruebas en Sistemas de Software Basados en el Contexto del Instituto Tecnológico de Costa Rica (ITCR). Durante la preparación de este trabajo se utilizaron herramientas basadas en inteligencia artificial para apoyar la mejora del lenguaje. Los autores revisaron, validaron y editaron todas las sugerencias generadas, con el fin de garantizar que el manuscrito refleje sus propias contribuciones científicas, interpretaciones y aportes. Los autores asumen plena responsabilidad por el contenido del artículo.

Referencias

de Sousa Santos, I., de Castro Andrade, R. M., Rocha, L. S., Matalonga, S., de Oliveira, K. M., and Travassos, G. H. (2017). Test case design for context-aware applications: Are we there yet? *Information and Software Technology*, 88:1–16.

de Souza Doreste, A. C. and Travassos, G. H. (2020). Towards supporting the specification of context-aware software system test cases. In *CIbSE*, pages 356–363.

Freitas, L. and Lelli, V. (2022). Using machine learning on testing iot applications: A systematic mapping. In *Brazilian Symposium on Multimedia and the Web*.

Graham, D., Veenendaal, E. v., Evans, I., and Black, R. (2008). *Foundations of software testing: ISTQB certification*. Intl Thomson Business Pr.

Islam, M., Khan, F., Alam, S., and Hasan, M. (2023). Artificial intelligence in software testing: A systematic review. In *TENCON 2023-2023 IEEE Region 10 Conference*.

Lima, R., da Cruz, A. M. R., and Ribeiro, J. (2020). Artificial intelligence applied to software testing: A literature review. In *15th Iberian Conference on Information Systems and Technologies*, pages 1–6. IEEE.

Luo, C., Goncalves, J., Velloso, E., and Kostakos, V. (2020). A survey of context simulation for testing mobile context-aware applications. *ACM Computing Surveys*, 53:1–39.

Matalonga, S., Amalfitano, D., Doreste, A., Fasolino, A. R., and Travassos, G. H. (2021). Alternatives for testing of context-aware contemporary software systems in industrial settings: Results from a rapid review. *arXiv preprint arXiv:2104.01343*.

Matalonga, S., Amalfitano, D., Doreste, A., Fasolino, A. R., and Travassos, G. H. (2022). Alternatives for testing of context-aware software systems in non-academic settings: results from a rapid review. *Information and Software Technology*, 149:106937.

Matalonga, S., Rodrigues, F., and Travassos, G. H. (2017). Characterizing testing methods for context-aware software systems: Results from a quasi-systematic literature review. *Journal of Systems and Software*, 131:1–21.

Mirza, A. M., Khan, M. N. A., Wagan, R. A., Laghari, M. B., Ashraf, M., Akram, M., and Bilal, M. (2021). Contextdrive: Towards a functional scenario-based testing framework for context-aware applications. *IEEE Access*, 9:80478–80490.

Priya, S. S. and Rajalakshmi, B. (2022). Testing context aware application and its research challenges. In *2022 International Conference on Smart Technologies and Systems for Next Generation Computing (ICSTSN)*, pages 1–7. IEEE.

Roda, C., Navarro, E., Zdun, U., López-Jaquero, V., and Simhandl, G. (2018). Past and future of software architectures for context-aware systems: A systematic mapping study. *Journal of Systems and Software*, 146:310–355.

Salomón, S., Duque, R., Montaña, J. L., and Tenés, L. (2023). Towards automatic evaluation of the quality-in-use in context-aware software systems. *Journal of Ambient Intelligence and Humanized Computing*, 14(8):10321–10346.

Seblewongel, E., Berhanyikun, A., Mekuria, S., Abel, D., and Koceski, S. (2016). Context-aware technologies and systems. In *International Conference on Applied Internet and Information Technologies*.

Travis, G. (2019). How the boeing 737 max disaster looks to a software developer. *IEEE Spectrum*, 18.

Trudova, A., Dolezel, M., and Buchalcevova, A. (2020). Artificial intelligence in software test automation: A systematic.

Vissers-Similon, E., Dounas, T., and De Walsche, J. (2024). Classification of artificial intelligence techniques for early architectural design stages. *International Journal of Architectural Computing*, page 14780771241260857.