Ambiguity in Requirements Engineering: A Systematic Mapping of AI-Based Approaches

Juan Ignacio Torres[1][0000-0002-9399-7561], Leandro Antonelli[2,3][0000-0003-1388-0337] and Pablo Thomas[1][0000-0001-9861-987X]

[1] III-LIDI, Facultad de Informática, Universidad Nacional de La Plata, Buenos Aires, Argentina
[2] LIFIA, Facultad de Informática, Universidad Nacional de La Plata, Buenos Aires, Argentina
[3] CAETI - Facultad de Tecnología Informática - Universidad Abierta Interamericana.
{jitorres, pthomas}@lidi.info.unlp.edu.ar
lanto@lifia.info.unlp.edu.ar

Abstract. Ambiguity in requirements engineering is a pervasive quality defect that can lead to misunderstandings, rework, and increased development costs. In recent years, Artificial Intelligence (AI) techniques have been increasingly explored to support the detection and analysis of ambiguity in natural language requirements. However, existing research is dispersed across different ambiguity types, AI techniques, and Requirements Engineering (RE) activities, making it difficult to obtain a consolidated view of the state of the art. This paper presents a Systematic Mapping Study (SMS) that examines AI-based approaches for addressing ambiguity in requirements engineering. Following established guidelines for systematic studies in software engineering, the literature was systematically identified, selected, and analyzed. The study provides a structured overview of existing approaches, identifies research trends and limitations, and highlights directions for future research in AI-supported requirements quality analysis.

Keywords: Ambiguity detection, Requirements Engineering, Artificial Intelligence, Systematic Mapping Study.

1 Introduction

Software systems are increasingly complex, and their successful development depends critically on the quality of their requirements. Since most requirements are expressed in natural language, they are inherently prone to quality defects, among which ambiguity is one of the most pervasive and consequential. A requirement is ambiguous when it can be interpreted in more than one way by different readers, leading to misunderstandings, incorrect design decisions, rework, and cost overruns. Research has consistently shown that the cost of correcting requirements defects increases significantly in later development stages, making early detection a critical concern. [1] [2]

Over the past two decades, the research community has increasingly explored the application of Artificial Intelligence (AI) techniques to address ambiguity in require-

Proceedings of the 29th Workshop on Requirements Engineering (WER2026),
August 11-13, 2026, La Plata, Argentina.
https://doi.org/10.29327/1884517.29-6

ments. Early efforts relied on rule-based Natural Language Processing (NLP) approaches, while more recent work has adopted classical machine learning, deep learning, and large language models (LLMs) based on the transformer architecture. Despite this growing body of research, the literature remains fragmented across different types of ambiguity, AI techniques, and Requirements Engineering activities, and there is a need for a structured overview that maps existing contributions and identifies research gaps. [3]

To address this need, this paper presents a Systematic Mapping Study (SMS) on AI-based approaches for handling ambiguity in software requirements, following established guidelines for systematic studies in software engineering and using the taxonomy of Berry and Kamsties as the conceptual framework. [4]

The remainder of this paper is organized as follows. Section 2 presents the background and related concepts. Section 3 describes the planning of the SMS. Section 4 reports the execution and selection results. Section 5 presents the synthesis of results. Section 6 discusses the threats to validity of the study. Section 7 interprets the findings, identifies research gaps, and outlines directions for future work. Finally, Section 8 presents the conclusions.

2 Background and Related Work

This section summarizes the main concepts related to ambiguity in software requirements and AI techniques used to address it.

2.1 Ambiguity in Software Requirements

Ambiguity is a fundamental challenge in natural language requirements. Berry and Kamsties proposed a comprehensive taxonomy that distinguishes between linguistic ambiguity and software engineering ambiguity, along with related phenomena.

Linguistic ambiguity includes four types: lexical ambiguity, when a word has multiple meanings; syntactic ambiguity, when grammatical structure allows more than one parsing; semantic ambiguity, when sentence meaning is unclear despite unambiguous words and syntax; and pragmatic ambiguity, when interpretation depends on context or implicit assumptions. Software engineering ambiguity covers requirements document ambiguity, application domain ambiguity, system domain ambiguity, and development domain ambiguity, all arising from the specific context of software development. The taxonomy also recognizes related phenomena such as vagueness, generality, and imprecision, which, while distinct from ambiguity in a strict sense, contribute to interpretation problems and are frequently addressed by the same techniques.

2.2 AI Techniques for Handling Ambiguity in Requirements

AI approaches for handling ambiguity in requirements have evolved from rule-based NLP techniques to machine learning, deep learning, and, more recently, transformer-based language models. Early approaches relied on linguistic rules and pattern matching, while later work introduced data-driven methods such as Support Vector Machines

and neural networks. Recent advances in large language models (LLMs) enable context-aware analysis and generation capabilities, opening new possibilities for ambiguity detection and mitigation.

2.3 Related Work

Several secondary studies have addressed ambiguity and quality in Requirements Engineering, as well as the application of Artificial Intelligence techniques to support requirements analysis. However, these studies differ in scope and do not provide a consolidated view specifically focused on AI-based approaches for handling ambiguity in software requirements.

A limited number of works directly focus on ambiguity. Existing reviews on ambiguity in requirements highlight that research has predominantly concentrated on detection using Natural Language Processing techniques, with limited attention to reduction or prevention and a lack of unified classification frameworks [5] [6]. Similarly, broader studies on ambiguity in Natural Language Processing provide valuable insights into disambiguation techniques but are not tailored to the specific context of Requirements Engineering [7].

Other secondary studies analyze Artificial Intelligence in Requirements Engineering more broadly, including machine learning, Natural Language Processing, and large language models [8] [9] [10]. These studies show that AI is increasingly applied to requirements analysis, formalization, and quality assessment, but ambiguity is usually treated as one quality attribute among many [11]. In contrast, this paper presents a Systematic Mapping Study specifically focused on AI-based approaches for handling ambiguity in software requirements.

3 Planning of the Systematic Mapping Study

The protocol follows the guidelines proposed by Kitchenham and Charters for conducting systematic literature reviews in Software Engineering, and the guidelines for systematic mapping studies proposed by Petersen et al. and later updated by Petersen, Vakkalanka, and Kuzniarz. These guidelines inform the definition of the research questions, the search strategy, the inclusion and exclusion criteria, the study selection procedure, and the data extraction and classification strategy. [12] [13] [14]

3.1 Research Questions

To achieve the objective of the SMS, the following main research question is defined:

RQ: What is the state of the art regarding AI-based approaches for handling ambiguity in software requirements?

This main research question is decomposed into a set of secondary research questions (RQ1-RQ6), summarized in Table 1, that support a classification-oriented mapping of the existing literature. These questions are designed to characterize the types of ambiguity addressed, the AI techniques employed, the Requirements Engineering activities involved, and the nature and maturity of the reported research contributions.

Table 1. Secondary research questions and their motivation.

Research Question	Motivation
RQ1: Which types of ambiguity, according to the taxonomy of Berry and Kamsties, are addressed by AI-based approaches in software requirements?	To map the coverage of ambiguity categories and identify which types are most frequently addressed and which remain unexplored.
RQ2: Which AI techniques are used to detect, prevent, mitigate, or analyze ambiguity in software requirements?	To identify the AI technique families applied in this context and observe technological trends, including the emergence of generative AI approaches.
RQ3: Which types of requirements artifacts are analyzed for ambiguity, and at what level of granularity?	To characterize the inputs of the proposed approaches, distinguishing between different requirements artifacts and units of analysis.
RQ4: In which Requirements Engineering activities are AI-based approaches applied to handle ambiguity in software requirements?	To understand at what stages of the Requirements Engineering process AI techniques are used to address ambiguity.
RQ5: What types of research contributions are proposed by the identified studies?	To classify the nature of the contributions, such as tools, methods, frameworks, metrics, or datasets.
RQ6: What is the type and maturity of the empirical evidence reported in the studies?	To assess the level of empirical validation and the readiness of the proposed approaches for practical adoption.

3.2 Search Strategy

The search strategy was designed to identify primary studies that address ambiguity in software requirements through the application of Artificial Intelligence techniques. To ensure comprehensive coverage of the Software Engineering and Requirements Engineering literature, the search was conducted in digital libraries that are widely recognized and frequently used by the research community.

Accordingly, the selected digital libraries were IEEE Xplore, the ACM Digital Library, Springer Nature Link, Scopus and WERpapers. These sources index the main journals, conferences, and workshops in Software Engineering, Requirements Engineering, and Artificial Intelligence.

The search period spans from 2004 to 2025. The year 2004 was selected as the starting point because it corresponds to the publication of the taxonomy of ambiguity in software requirements proposed by Berry and Kamsties, which is adopted in this study as the baseline conceptual framework.

The complete search strings adapted for each digital library are provided in the supplementary appendix. [15]

The search string was constructed by combining three main conceptual dimensions derived from the research questions: i*) software requirements*, ii) *ambiguity and related phenomena*, and iii) *Artificial Intelligence techniques*. For each dimension, a set of synonyms and related terms was identified to improve recall while preserving relevance to Requirements Engineering.

Based on the identified conceptual dimensions, the following generic search string was defined using Boolean operators:

("software requirement" OR "requirements engineering" OR SRS OR "user stor*" OR "use case*" OR "acceptance criteria" OR backlog) AND*

(ambigu OR vagueness OR vague OR generality OR imprecise) AND ("artificial intelligence" OR AI OR "natural language processing" OR NLP OR "machine learning" OR "deep learning" OR "large language model*" OR LLM OR transformer* OR "language model*" OR "retrieval augmented generation" OR RAG)*

The generic search string was adapted to the syntax and constraints of each selected digital library. Despite these adaptations, the same conceptual structure of the search string was preserved across all databases to ensure consistency in the search process.

3.3 Inclusion and Exclusion Criteria

This section defines the inclusion and exclusion criteria applied to select the primary studies considered in this Systematic Mapping Study. These criteria were defined to ensure that only relevant, high-quality, and peer-reviewed studies addressing ambiguity in software requirements through Artificial Intelligence techniques were included. The inclusion and exclusion criteria are summarized in Table 2.

Table 2. Inclusion and Exclusion criteria.

Inclusion Criteria	Exclusion Criteria
IC1*: The study is a primary research paper published in a peer-reviewed venue, including journals, conferences, or workshops.*	**EC1***: Studies that address ambiguity in artifacts other than software requirements (e.g., source code, architecture models, test cases) without a clear link to Requirements Engineering.*
IC2*: The study explicitly addresses ambiguity in software requirements or related phenomena (e.g., vagueness, generality, imprecision) within a Requirements Engineering context.*	**EC2***: Studies focused exclusively on linguistic ambiguity or generic natural language processing tasks without an explicit connection to software requirements.*
IC3*: The study proposes, applies, or evaluates Artificial Intelligence-based techniques (e.g., NLP, machine learning, deep learning, or large language models) to detect, analyze, prevent, or mitigate ambiguity in software requirements.*	**EC3***: Studies in which neither Artificial Intelligence nor computational Natural Language Processing techniques are used as part of the proposed approach (e.g., purely manual inspections or approaches based exclusively on keyword lists without linguistic processing).*
IC4*: The study analyzes one or more requirements artifacts (e.g., SRS, user stories, use cases, scenarios, acceptance criteria) as part of the ambiguity handling process.*	**EC4***: Secondary studies, such as systematic literature reviews, systematic mapping studies, surveys, editorials, keynote papers, tutorials, and textbooks.*
IC5*: The full text of the study is available and written in English.*	**EC5***: Gray literature, including technical reports, theses, dissertations, blog posts, and non-peer-reviewed materials.*
	EC6*: Studies where ambiguity is mentioned only incidentally and is not a central aspect of the research contribution.*

3.4 Study Selection Procedure

The study selection process was conducted following a structured and systematic procedure to ensure consistency and transparency. The selection was performed in multiple stages, based on the inclusion and exclusion criteria defined in section 3.3.

First, an initial screening based on titles was carried out to remove clearly irrelevant studies. This screening was primarily conducted by one author. Studies that could not be clearly excluded based on the title alone were retained for the next stage.

Second, an abstract screening was performed on the remaining studies to further assess their relevance. The inclusion and exclusion criteria were applied to each abstract, and studies that could not be clearly excluded at this stage were retained for full-text review.

Third, a full-text review of the remaining studies was performed to assess their relevance in detail. During this stage, the inclusion and exclusion criteria were applied rigorously. When uncertainties or borderline cases arose, the inclusion decision was discussed among the authors until a consensus was reached.

3.5 Data Extraction and Classification Scheme

This section describes the data extraction process and the classification scheme used to systematically characterize the primary studies included in this Systematic Mapping Study.

Each primary study was analyzed and classified according to a set of predefined dimensions covering ambiguity types, Artificial Intelligence techniques, Requirements Engineering context, and research characteristics.

For each selected primary study, relevant data were extracted using a structured data extraction form. The extracted information was recorded in a spreadsheet to support systematic analysis and traceability. When a study addressed multiple categories within a given classification dimension, all applicable categories were recorded.

Data extraction was primarily performed by one author. To reduce the risk of individual bias, a sample of the extracted classifications was independently reviewed by a second author. Any disagreements or borderline cases identified during this process were discussed among all authors until consensus was reached.

The classification scheme adopted in this study is summarized in Table 3.

Table 3. Classification scheme and dimensions used for data extraction.

Classification Dimension	Categories / Values
D1*: Type of Ambiguity Addressed*	Linguistic ambiguity (lexical, syntactic, semantic, pragmatic); Software engineering ambiguity (requirements document, application domain, system domain, development domain); Related phenomena (vagueness, generality, imprecision).
D2*: AI Technique Used*	Rule-based or heuristic NLP (e.g., part-of-speech tagging, pattern matching, handcrafted linguistic rules); Classical machine learning (e.g., Support Vector Machines, Decision Trees, Conditional Random Fields); Deep learning (e.g., neural networks, Convolutional Neural Networks, Recurrent Neural Networks); Transformer-based language model approaches (e.g., BERT, T5, GPT, LLaMA, including fine-tuning, prompt-based, and retrieval-augmented approaches).
D3*: Requirements Artifact Analyzed*	Software Requirements Specification (SRS); User stories; Use cases; Scenarios; Acceptance criteria; Other requirements-related artifacts.
D4*: Requirements Engineering Activity*	Requirements elicitation; Requirements analysis; Requirements specification; Requirements validation and inspection; Requirements change or quality management.
D5*: Type of Research Contribution*	Tool or prototype; Method or technique; Framework or process; Metric or quality model; Dataset or benchmark.
D6*: Research Type and Evaluation Method*	Solution proposal; Validation research; Evaluation research.

The classification scheme defined in this section provided the basis for the synthesis of results presented in Section 5. The explicit mapping between classification dimensions and research questions enabled a systematic analysis of the literature and supported the identification of research trends and gaps discussed in Section 7.

The full list of primary studies included in this mapping is available in the supplementary appendix. [15]

4 Execution of the Systematic Mapping Study

4.1 Search Execution

The automated search was conducted in the selected digital libraries, namely IEEE Xplore, the ACM Digital Library, Springer Nature Link, and Scopus, using the search strings described in Section 3.2. For WERpapers, studies were identified through targeted web searches and manual inspection of proceedings.

The initial search returned a total of 844 records across all digital libraries, before any duplicate removal or screening was performed. Table 4 summarizes the number of studies retrieved from each digital library.

Table 4. Number of studies retrieved per digital library.

Digital Library	Retrieved studies
IEEE Xplore	123
ACM Digital Library	71
Springer Nature Link	176
Scopus	461
WERpapers	13
Total	844

4.2 Study Selection Results

After duplicate removal, 675 unique studies were retained for screening. The selection was performed in four stages, as illustrated in the PRISMA-style flow diagram (Fig. 1).

Title screening excluded 564 clearly out-of-scope studies, retaining 111 for abstract review. Abstract screening excluded a further 63 studies, retaining 48 for full-text review.

Full-text assessment excluded 7 additional studies, primarily due to the absence of a clear focus on software requirements, lack of AI techniques, insufficient methodological detail, or treatment of ambiguity as peripheral to the research contribution. Consequently, 41 primary studies were selected for inclusion in the Systematic Mapping Study.

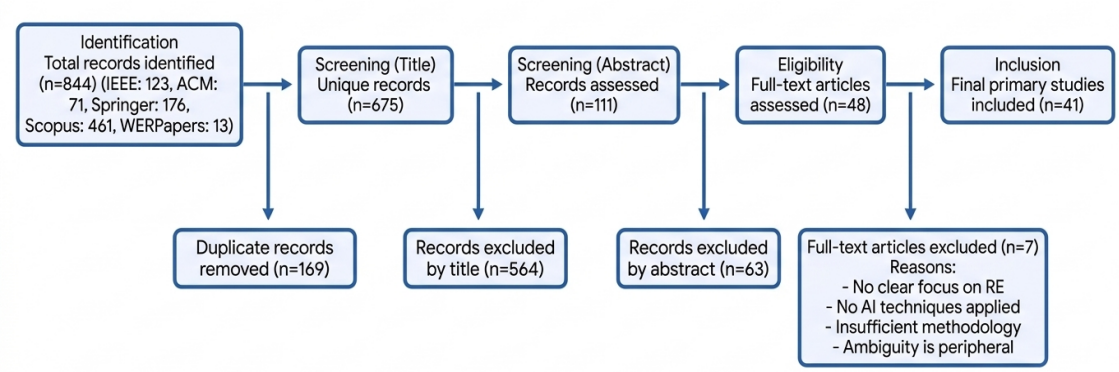

Fig. 1. PRISMA-style flow diagram illustrating the study selection process of the Systematic Mapping Study.

5 Synthesis of Results

5.1 RQ1 — Types of Ambiguity Addressed

The mapping reveals a clear concentration of research efforts around linguistic ambiguity, which is addressed in 36 of the 41 studies (88%), making it by far the most prevalent category in the literature. Among its subcategories, lexical ambiguity is the most frequently addressed type, appearing in 23 studies (56%), followed by syntactic ambiguity in 18 studies (44%), semantic ambiguity in 15 studies (37%), and pragmatic ambiguity in 11 studies (27%).

Related phenomena, including vagueness, generality, and imprecision, are addressed in 19 studies (46%), reflecting their conceptual proximity to ambiguity and their practical relevance in requirements quality assessment. Vagueness is the most commonly addressed phenomenon within this category (14 studies, 34%), while generality appears in only 4 studies (10%). Software engineering ambiguity is addressed in 10 studies (24%), with requirements document ambiguity and application domain ambiguity each appearing in 5 studies (12%). System domain ambiguity is addressed in only 2 studies (5%), and development domain ambiguity is not addressed by any of the identified studies, representing a notable gap in the literature.

5.2 RQ2 — AI Techniques Used

The analysis of AI techniques reveals a dominant role for rule-based and heuristic NLP approaches, which are employed in 29 of the 41 studies (71%). Classical machine learning approaches appear in 11 studies (27%), and deep learning approaches in 10 studies (24%), indicating a gradual but consistent adoption of data-driven techniques as annotated datasets became more available. Transformer-based language model approaches, the most recent development in the field, are present in 6 studies (15%), signaling an emerging trend toward both encoder-based models and generative AI methods that had not yet reached full maturity within the period covered by this study.

5.3 RQ3 — Requirements Artifacts Analyzed

Software Requirements Specifications (SRS) constitute the dominant artifact type in the corpus, appearing in 29 studies (71%). This concentration reflects both the historical centrality of SRS documents in Requirements Engineering research and the availability of benchmark datasets derived from SRS artifacts. Other requirements-related artifacts, such as elicited requirements, domain corpora, and terminology lists, appear in 10 studies (24%), representing a heterogeneous category that encompasses contributions focused on requirements elicitation and domain-level analysis.

In contrast, agile and user-centric artifact types remain substantially underrepresented. User stories appear in only 2 studies (5%), use cases in 1 study (2%), and scenarios in 2 studies (5%). No study in the corpus explicitly addresses acceptance criteria as a primary artifact.

5.4 RQ4 — Requirements Engineering Activities

The mapping shows that AI-based ambiguity handling is most strongly associated with requirements validation and inspection, which is targeted by 29 of the 41 studies (71%), and requirements analysis, addressed by 26 studies (63%). The concentration in these two activities reflects the predominant framing of ambiguity as a quality defect to be detected after requirements have been formulated, rather than prevented during their construction.

Requirements elicitation is addressed in 10 studies (24%), primarily in contributions focused on cross-domain terminology and stakeholder communication. Requirements specification is targeted by 4 studies (10%), and requirements change and quality management by 9 studies (22%). The relatively limited coverage of elicitation and specification activities suggests that AI-based approaches have been applied primarily to post-hoc quality assessment rather than to proactive support during the earlier, more constructive phases of the Requirements Engineering process.

5.5 RQ5 — Types of Research Contributions

The identified studies propose a diverse range of research contributions. Tools and prototypes represent the most common contribution type, appearing in 28 studies (68%), followed closely by methods and techniques in 27 studies (66%). The high frequency of both categories reflects a tendency in the field toward the joint development of computational approaches and their implementation as working software artifacts. Frameworks and processes are proposed in 5 studies (12%), and quality models or metrics in 4 studies (10%).

Datasets and benchmarks are reported in only 3 studies (7%), representing a notable scarcity given the central role of labeled data in the training and evaluation of machine learning and deep learning approaches. The limited availability of publicly accessible, standardized datasets for requirements ambiguity has been a persistent challenge in the field and may partially explain the continued prevalence of rule-based approaches, which do not require large annotated corpora.

5.6 RQ6 — Research Type and Empirical Evidence

The classification of studies by research type reveals that evaluation research is the predominant category, encompassing 31 of the 41 studies (76%). These contributions report empirical evaluations of proposed approaches, typically using precision, recall, and F1-score as performance measures, often on annotated datasets of requirements documents. Validation research, in which approaches are assessed in more controlled or preliminary settings, accounts for 7 studies (17%). Solution proposals, which present conceptual contributions without substantial empirical validation, appear in only 3 studies (7%).

The dominance of evaluation research suggests a relatively mature evaluation practice within this subfield. However, the evaluations conducted are predominantly performed on academic or curated datasets, and large-scale industrial evaluations involving real development projects remain comparatively rare. This gap between research evaluation and practical adoption represents an important dimension for future investigation.

6 Threats to validity

6.1 Construct validity

Construct validity concerns whether the study accurately captures the concept it intends to measure. In this work, ambiguity in software requirements was operationalized using the taxonomy proposed by Berry and Kamsties as the baseline conceptual framework. Although this taxonomy is widely recognized in Requirements Engineering research, ambiguity is inherently multifaceted and may be interpreted differently across studies. Some primary studies also address related phenomena such as vagueness, uncertainty, or requirement smells, which do not always correspond directly to formal ambiguity categories. To mitigate this threat, the classification scheme explicitly included related phenomena and allowed multiple categories to be assigned to a single study when appropriate.

Another potential threat arises from the classification of AI techniques. Given the rapid evolution of the field, boundaries between rule-based NLP, machine learning, deep learning, and large language model–based approaches may be blurred, especially in hybrid methods. The classification criteria were defined prior to analysis and applied consistently across all studies.

6.2 Internal validity

Internal validity refers to potential biases introduced during the study selection and data extraction processes. Data extraction was primarily performed by one researcher. To mitigate the risk of individual bias, a sample of the extracted classifications was independently reviewed by a second author, and any disagreements or borderline cases were discussed among all authors until consensus was reached.

6.3 External validity

External validity concerns the generalizability of the findings. This study focused on peer-reviewed publications in selected digital libraries widely recognized in Software Engineering research. While these sources provide broad coverage, relevant studies published in other venues or in gray literature — such as technical reports, theses, or industrial white papers — may not have been included. Excluding gray literature may limit the representation of industrial practices, where ambiguity handling techniques may be applied but not formally published. This limitation is particularly relevant for transformer-based language model approaches, where industrial adoption may precede peer-reviewed publication.

Additionally, the search period began in 2004, corresponding to the publication of the Berry and Kamsties taxonomy. Earlier studies addressing ambiguity in requirements may therefore be underrepresented. Language restrictions to English publications may also have excluded relevant research conducted in other languages, including Portuguese and Spanish; this decision was made to ensure consistency in the application of inclusion criteria and to avoid potential misclassification arising from translation.

6.4 Reliability

Reliability refers to the reproducibility of the study. To enhance transparency, the review protocol explicitly defined the search strategy, selection procedure, classification scheme, and data extraction approach.

Nevertheless, exact replication may be affected by database indexing changes and by interpretive judgment in some classification decisions. Despite these limitations, the systematic and structured approach adopted in this work, combined with the explicit documentation of all methodological decisions, is expected to yield consistent results under comparable conditions.

7 Discussion and Research Gaps

This section interprets the results of the Systematic Mapping Study, identifies major trends in the literature, highlights research gaps, and outlines directions for future work in AI-based approaches for handling ambiguity in software requirements.

7.1 Discussion of Key Findings

The results confirm a strong concentration on linguistic ambiguity detection — particularly at the lexical and syntactic levels — applied predominantly to SRS documents through rule-based NLP techniques. Notably, ambiguity is treated primarily as a linguistic phenomenon rather than a socio-technical one, and most approaches focus on post-hoc detection rather than prevention or resolution. These trends directly motivate the research gaps identified below.

7.2 Identified Research Gaps

Six significant gaps emerge from the analysis. These gaps are presented in order of relevance, from those with the broadest impact on research validity to those reflecting misalignments with current industrial practice.

Limited availability of standardized datasets and benchmarks. Only a small portion of studies provides publicly available datasets or evaluation benchmarks. This lack of shared resources is arguably the most structurally limiting gap in the field, as it prevents systematic comparison across approaches and makes it difficult to assess whether reported improvements represent genuine advances or artifacts of dataset-specific conditions.

Insufficient attention to agile requirements artifacts. Despite the prevalence of agile development, ambiguity detection techniques are rarely tailored to user stories, acceptance criteria, or backlog items. This mismatch represents one of the most consequential gaps between academic research and industrial practice, given that agile artifacts are now the dominant form of requirements documentation in many organizations.

Limited focus on non-linguistic ambiguity. Most studies address ambiguity at the textual level, while ambiguities arising from domain knowledge gaps, stakeholder perspectives, or contextual assumptions remain relatively underexplored. These forms of ambiguity are often the most challenging in real-world projects.

Scarcity of prevention-oriented approaches. Research overwhelmingly concentrates on post hoc detection. Few studies investigate techniques to prevent ambiguity during elicitation or specification, such as controlled language support, interactive authoring tools, or stakeholder negotiation mechanisms.

Underrepresentation of large-scale industrial validation. Although evaluation research is common, relatively few studies report deployment in real industrial settings over extended periods. Consequently, the practical effectiveness and cost-benefit trade-offs of many techniques remain unclear.

Emerging but immature use of large language models. Transformer-based models show significant potential due to their contextual understanding capabilities, but their use in requirements ambiguity analysis is still in early stages. Specific open challenges include susceptibility to hallucination in low-resource domain-specific settings, limited explainability of model outputs in safety-critical contexts, and the absence of RE-specific benchmarks for evaluating LLM performance on ambiguity tasks.

8 Conclusions and Future Work

This paper presented a Systematic Mapping Study examining the state of the art regarding AI-based approaches for handling ambiguity in software requirements. Following established methodological guidelines, a total of 41 primary studies published between 2004 and 2025 were identified, selected, and systematically classified across six dimensions covering ambiguity types, AI techniques, requirements artifacts, RE activities, contribution types, and empirical evidence.

The results show that research in this area has grown considerably over the past two decades, evolving from early rule-based NLP approaches toward machine learning,

deep learning, and more recently large language model–based techniques. Linguistic ambiguity, particularly at the lexical and syntactic levels, dominates the literature, while software engineering ambiguity and pragmatic forms remain comparatively underexplored. The Software Requirements Specification is by far the most studied artifact, whereas agile artifacts such as user stories and acceptance criteria receive minimal attention. Most contributions target requirements validation and inspection, reflecting a predominant focus on post-hoc defect detection rather than proactive prevention during elicitation or specification.

Several significant gaps were identified. The scarcity of publicly available datasets and benchmarks remains the most structurally limiting challenge, hindering systematic comparison across approaches. The underrepresentation of agile artifacts, non-linguistic ambiguity, and large-scale industrial validation further highlight the distance between current research and the realities of modern software development practice. The emerging use of large language models shows promise but requires dedicated research on hallucination mitigation, explainability, and RE-specific evaluation benchmarks.

This study provides a structured overview of the field and a solid foundation for future research.

Based on the identified gaps, several directions for future work can be outlined.

First, future research should consider socio-technical aspects of ambiguity, including stakeholder diversity and domain knowledge evolution.

Second, there is a need for approaches that support ambiguity prevention during requirements elicitation and authoring. Interactive tools that provide real-time feedback while writing requirements, possibly powered by AI assistants, could significantly reduce downstream defects.

Third, future studies should focus more explicitly on agile development contexts. Adapting ambiguity detection techniques to short, informal artifacts such as user stories would improve their practical relevance.

Fourth, and most urgently, the creation of shared datasets, annotated corpora, and benchmark tasks would enable systematic comparison of approaches and accelerate progress in the field. Without such resources, the cumulative progress of the research community remains difficult to assess.

Fifth, large language models offer new opportunities for both detection and explanation of ambiguity, as well as for generating candidate revisions. However, their adoption in requirements engineering requires targeted research on hallucination mitigation, domain-specific fine-tuning strategies, and alignment with engineering traceability standards — challenges that are distinct from general NLP benchmarks.

Finally, more longitudinal industrial studies are needed to evaluate the real-world effectiveness and adoption of AI-based ambiguity detection techniques.

References

1. Boehm, B.W.: Software Engineering Economics. Prentice-Hall, Englewood Cliffs (1981)
2. Davis, A.M., Overmyer, S.P., Jordan, K., Caruso, J., Dandashi, F., Dinh, A., Kincaid, G., Ledeboer, G., Reynolds, P., Sitaram, P., Ta, A., Theofanos, M.: Identifying and Measuring Quality in a Software Requirements Specification. In: Proceedings of the First International

Software Metrics Symposium, pp. 141–152. IEEE (1993). https://doi.org/10.1109/METRIC.1993.263792
3. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention Is All You Need. In: Advances in Neural Information Processing Systems (NeurIPS), pp. 5998–6008 (2017). https://doi.org/10.48550/arXiv.1706.03762
4. Berry, D.M., Kamsties, E.: Ambiguity in Requirements Specification. In: do Prado Leite, J.C.S., Doorn, J.H. (eds.) Perspectives on Software Requirements, pp. 7–44. Springer, Boston (2004). https://doi.org/10.1007/978-1-4615-0465-8_2
5. Bano, M.: Addressing the Challenges of Requirements Ambiguity: A Review of Empirical Literature. In: 2015 IEEE Fifth International Workshop on Empirical Requirements Engineering (EmpiRE), pp. 21–24. IEEE (2015). https://doi.org/10.1109/EmpiRE.2015.7431303
6. Gupta, A.K., Deraman, A., Siddiqui, S.T.: A Survey of Software Requirements Specification Ambiguity. ARPN Journal of Engineering and Applied Sciences 14(17), 3046–3061 (2019)
7. Yadav, A., Patel, A., Shah, M.: A Comprehensive Review on Resolving Ambiguities in Natural Language Processing. AI Open 2, 85–92 (2021). https://doi.org/10.1016/j.aiopen.2021.05.001
8. Hemmat, A., Sharbaf, M., Kolahdouz-Rahimi, S., Lano, K., Tehrani, S.Y.: Research Directions for Using LLM in Software Requirement Engineering: A Systematic Review. Frontiers in Computer Science 7, 1519437 (2025). https://doi.org/10.3389/fcomp.2025.1519437
9. Zamani, K., Zowghi, D., Arora, C.: Machine Learning in Requirements Engineering: A Mapping Study. In: Proceedings of the IEEE International Conference on Requirements Engineering Workshops (REW), pp. 116–125 (2021). https://doi.org/10.1109/REW53955.2021.00023
10. Kolahdouz-Rahimi, S., Lano, K., Lin, C.: Requirement Formalisation Using Natural Language Processing and Machine Learning: A Systematic Review. In: Proceedings of the International Conference on Model-Driven Engineering and Software Development (MODELSWARD), pp. 237–244 (2023). https://doi.org/10.5220/0011789700003402
11. Montgomery, L., Fucci, D., Bouraffa, A., Scholz, L., Maalej, W.: Empirical Research on Requirements Quality: A Systematic Mapping Study. Requirements Engineering 27(2), 183–209 (2022). https://doi.org/10.1007/s00766-021-00367-z
12. Kitchenham, B., Charters, S.: Guidelines for Performing Systematic Literature Reviews in Software Engineering. EBSE Technical Report EBSE-2007-01, Keele University and Durham University (2007)
13. Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M.: Systematic Mapping Studies in Software Engineering. In: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering (EASE), pp. 68–77. BCS Learning & Development Ltd. (2008). https://doi.org/10.14236/ewic/EASE2008.8
14. Petersen, K., Vakkalanka, S., Kuzniarz, L.: Guidelines for Conducting Systematic Mapping Studies in Software Engineering: An Update. Information and Software Technology 64, 1–18 (2015). https://doi.org/10.1016/j.infsof.2015.03.007
15. Torres, J.I., Antonelli, L., Thomas, P.: Appendix: Ambiguity in Requirements Engineering: A Systematic Mapping of AI-Based Approaches. Figshare (2026). https://doi.org/10.6084/m9.figshare.31884781