COMPUTING AND APPLICATIONS REVIEWS

Volume 3, Number 1, 2026 Pages 56 – 75 WEBSITE. ca-reviews.com

REDUCING SOFTWARE REQUIREMENT AMBIGUITY THROUGH NLP-BASED REQUIREMENT QUALITY ASSESSMENT

Hammad Asif [1]*

[1] *Department of Software Engineering, Institute of Intelligent Software Systems, Lahore, Pakistan*

**Corresponding Author Email:* hammad.asif@iiss.edu.pk

Article Information

Article History

Received:	January 11, 2026
Revised:	February 12, 2026
Accepted:	April 15, 2026
Available Online:	June 30, 2026

Keywords:
Software Requirements; Requirement Ambiguity; Natural Language Processing; Requirement Quality Assessment; Software Engineering

Abstract

Software requirement ambiguity remains one of the major causes of misunderstanding, rework, delayed development, and project failure in software engineering. Ambiguous requirements often lead to inconsistent interpretation among stakeholders, developers, testers, and project managers, which directly affects software quality and delivery performance. This paper presents an NLP-based requirement quality assessment approach for identifying and reducing ambiguity in software requirement specifications. The proposed approach evaluates requirement statements by analyzing linguistic features such as vague terms, incomplete conditions, passive constructions, uncertain quantifiers, weak modal verbs, and semantic inconsistency. Natural Language Processing techniques are applied to classify requirements into different quality levels and highlight statements that require improvement. The results indicate that NLP-based assessment improves ambiguity detection accuracy, supports early requirement validation, and reduces manual review effort. The findings show that automated linguistic analysis can help requirement engineers detect unclear statements before the design and implementation phases. Overall, the study demonstrates that integrating NLP into requirement engineering practices can enhance requirement clarity, improve stakeholder agreement, and support the development of more reliable software systems.

INTRODUCTION

The impact of software requirements documentation on the efficacy of the software project is very significant, since it is an input document to the software design, coding and testing (Ormandjieva et al., 2007).The dependence of natural language leads to natural language vagueness and natural language uncertainty, which may cause expensive delays and cost overruns in the project, and a low software quality (Hussain et al., 2021; Talha et al., 2025; Veizaga et al., 2021). The flexibility, universality and ease of access of natural language with respect to various stakeholders (Veizaga et al., 2021; Veizaga et al., 2024) make it the most commonly used language for describing requirements, but this same flexibility has the downside of making it more likely that requirements will be ambiguous, incomplete and semantically inconsistent (Alzayed & Al-Hunaiyyan, 2021; Kamsties, 2000; Remmen et al., 2023). These language defects have a basic effect on the requirement engineering process, and could be a critical factor to guarantee that the right and full understanding of the system required for proper system development is achieved (Franch et al., 2023; Hussain et al., 2021). Ambiguities are often not introduced into or left unaddressed in the initial stages of the requirements process for specific reasons, such as linguistic reasons (e.g., pronouns with ambiguous references) or RE-specific reasons (e.g., complex application domains and operational contexts) (Kamsties, 2000). The root of design, development and testing is requirements, and if there's a defect that's not addressed at this fundamental level, that defect will propagate throughout the development cycle, making it more and more expensive and time consuming to fix a bug (Talha et al., 2025; Veizaga et al., 2021). Moreover, the requirements' complexity makes traditional manual inspection approaches more and more unaffordable to compete with the iterative and fast development cycles common in contemporary software engineering (Remmen et al., 2023; Talha et al., 2025). More reliable, scalable and systematic techniques for validating are required, leading to an increase in research interest in the use of NLP as a requirements engineering (Alzayed & Al-Hunaiyyan, 2021; Zhao et al., 2021) tool. Whether it's the tried-and-tested rules-based approaches or cutting-edge machine learning and deep learning methods, computational NLP techniques can quickly and accurately analyze, classify, and process huge volumes of textual requirements. Previous studies have shown that these can detect "requirements smells" (poorly formulated requirements that foretell possible defects) and that they can be used for conformance to a standardized template to improve the structural and semantic quality of specifications (Arora et al., 2015; Lucassen et al., 2016; Veizaga et al., 2024). However, some challenges still remain, specifically those concerning the accuracy of ambiguity detection, the absence of adequate ambiguity terms in the domain, and the incorporation of the automated tools in the existing development processes (Baranetska, 2025). Moreover, while there are a number of proposed tools, the need for tools that can deliver high precision in a wide range of industrial scale applications that are interpretable for practitioners continues (Izhar et al., 2025). In this study, the main goal is to proactively detect and resolve the ambiguity issues in requirement documentation using advanced NLP techniques that can help overcome these critical drawbacks could benefit the overall software development process. The automated assessment frameworks help in closing the gap between informal inputs from stakeholders and the formal requirements for solid software building (Umar & Lano, 2024), (Torres-Igartua et al., 2025). Moreover, automated tools can provide a more holistic analysis of the recurring quality issues that are not possible to be identified at scale with traditional and manual reviews, such as the absence of phrases or the functional descriptions being too long (Dalpiaz et al., 2018; Veizaga et al., 2023). While projects exhibit the inherent ambiguity of artifacts and may require rework, these computational methods can help translate NLA artifacts into more formal representations, thus reducing the likelihood of rework and ambiguities (Kolahdouz-Rahimi et al., 2023), (Mello, 2024). These sophisticated Natural Language Processing (NLP4RE) systems allow the organisation of requirement elicitation, which in turn improves the reliability of complex software systems in different industrial domains (Zhong et al., 2009; Gallego & Jaramillo, 2023).

LITERATURE REVIEW

Currently, it is observed that quality in requirements is assessed either in a manual way with checklists or with an increasing number of automated diagnostic tools (Bäumer & Geierhos, 2018; Malik et al., 2023).Initial solutions were rule-based heuristics and dictionary based lookups such as SREE that scans for potential loopholes, ambiguous pronouns, subject-verb pairing and other instances of passive construction, which lead to high recall and increased computational complexity (Lucassen et al., 2016). These rule-based approaches were a first step toward automation of the requirement quality assurance, but were not accurate in identifying ambiguity and in some cases were raising an "alarm" when the requirement was unambiguous, resulting in "alarm fatigue" for practitioners (Lucassen et al., 2016). In order to overcome these limitations, naturally, research has turned to data and machine learning based models: some of the initial research addressed the identification of uncertainty cues in textual requirements (Lucassen et al., 2016), others focused on using support vector machines and Naïve Bayes classifiers for extracting non-functional requirements from different types of documentation sources (Lucassen et al., 2016). These methods were better than purely rule-based systems, but were still limited and by manually engineered features and semantic context. The industry has also changed altogether since then, with the emergence of more advanced Natural Language Processing (NLP) techniques such as deep learning architectures (e.g., transformer models like BERT, SpanBERT, and its variants), which have greatly enhanced the ability of accurately understanding complex semantic relations and contextual information in requirements. These cutting-edge solutions are also excellent for classification and semantic quality assurance, and enhance the accuracy and scalability of defect identification over manual inspections (Baranetska, 2025; Malik et al., 2023). There is still much research to be done, however, which will keep these tools from being widely used in industry. Most notably, many state-of-the-art models have difficulty in generalization, performing very well on academic datasets and failing to adapt and deliver performance on the very specific and nuanced terminology of domains like healthcare, finance, or avionics (Torres-Igartua et al., 2025; Zhao et al., 2021; Zhong et al., 2009). Furthermore, there is a major contrast between performance and transparency, because the more complex the deep learning models, the more impressive the F1-scores, they are regarded by industry practitioners as "black boxes", and have not sufficient level of explainability to build trust in automated defect recommendations (Baranets'ka, 2025; Izhar et al., 2025). In addition to these problems, there is another engineering problem – how to integrate into existing and iterative development processes such as agile development (Baranetska, 2025). In order to address these shortcomings, it is necessary to develop better models, as well as to build hybrid models that integrate the capabilities of the modern NLP models with domain-specific knowledge bases and user-friendly and interpretable diagnostic results (Alzayed & Al-Hunaiyyan, 2021; Izhar et al., 2025). Nevertheless, one of the most significant issues in requirement engineering is the lack of accuracy and effectiveness in academic prototypes while meeting the requirements of industry.However, an important issue in requirement engineering is that, as pointed by Dalpiaz et al. (2018) and Zhao et al. (2021), there is a gap from high precision academic prototypes to strong, usable, and understandable tools for industrial requirement engineering. In addition, there is no benchmark dataset available that is large, representative, and can be used for rigorously testing and reproducing new artificial intelligence methods proposed in the field (Kaur & Kaur, 2024). Moreover, the state of the art in annotated datasets for training and testing is limited, and such models are currently only lab based (Cheng et al., 2024; Ghaisas & Singhal, 2024). The lack of a common scale and standard set of requirements, with a diverse range of use cases adds to the difficulty of making comparative progress and effectively stunts the benchmarking of a variety of algorithms (Sudhi et al., 2023). Furthermore, the percentage of AI-based RE tools remains high, and most of these tools are still confined to the laboratory environment, while a few have been tested in a professional setting (Shah et al., 2025).

METHODOLOGY

The proposed research methodology is a multi-stage NLP pipeline to evaluate the quality of the requirements with semantic extraction using transformer and linguistic analysis of linguistic features.Specifically, our pipeline first tokenizes requirement specifications into sequences of tokens, where we employ complex tokenization algorithms in order to obtain tokens to represent the domain-specific language used in requirement specifications that are necessary for further interpretation (Ghaisas & Singhal, 2024). After that, a two-step extraction process is performed to extract the important language elements that may lead to ambiguity in understanding the system requirements, including the use of vague quantifiers (e.g. efficient, optimal), subjective adjectives (e.g. appropriate, user-friendly), and non-verifiable modal verbs (e.g. might, can) (Lucassen et al., 2016; Talha et al., 2025).

We use a hybrid approach (Izhar et al., 2025) to learn both lexical and semantic properties. We use an extended set of known ambiguity indicators that is extended from Lucassen et al., (2016) and part-of-speech tagging and dependency parsing to identify sentence-level weaknesses. At the same time, we adapt a pre-trained transformer model to better capture contextual embeddings, which take into account the different ways the same terms are used in various requirements datasets, across different industrial domains (Malik et al., 2023; Torres-Igartua et al., 2025).

The main problem in requirements engineering (RE) is to generate quality and annotated corpora (Ghaisas & Singhal, 2024;

Sudhi et al., 2023); building datasets is very important in our work. The dataset has been carefully curated to include 2000 functional requirements from open repositories and industrial collaborations, and covers a broad spectrum of functional requirements in different functional domains such as healthcare and finance (Zhao et al., 2021). We have a detailed three-stage annotation process to ensure high data quality: two different and independently trained annotators tag presence, type, and level of ambiguity in the requirements; and a third annotator who is senior and resolves the differences between the two annotators (Ghaisas & Singhal, 2024). This labeling process is validated by Cohen's kappa to get a score that's considered to be a good degree of agreement (Kaur & Kaur, 2024).Once the data is annotated it is normalised to resolve inconsistencies in format and terminology (Ghaisas & Singhal, 2024). To ensure representativeness of the data distribution for training the model, and for a reliable performance assessment, we use a typical train/validation/test split (80/10/10) (Kaur & Kaur, 2024). The model is trained using supervised learning with a cross-entropy loss function and under-going regularization to prevent over-fitting the model, given the limited amount of training data (Torres-Igartua et al., 2025). The model is trained using high-performance GPUs infrastructure, because it enables fine-tuning of large-scale transformer architectures, and implemented with PyTorch framework. Finally, quantitative evaluation measures like precision, recall, and F1-score (Izhar et al., 2025; Torres-Igartua et al., 2025) are used to extensively analyze the effectiveness of the model. The metrics are used to assess the model's transferability to specific domains and to identify any issues that need to be addressed in an iterative development process. Furthermore, to solve the intra-domain ambiguities, we implement the clustering methods and allow the framework to detect the terms that change their meaning depending on their occurrence in a particular domain in the context of a project (Moharil & Sharma, 2022).

RESULTS

From experimental results it can be concluded that the proposed NLP based Requirement Quality Assessment (NLP-QA) framework can reduce the ambiguity level of software requirements statements and can increase measurable quality of it. The set of requirements was labeled, presented with ambiguously formulated, incomplete and conflicting requirement sentences. The proposed NLP-QA model attained the best performance with F1-scores of 93.4% as presented in figure 1, superior to F1-scores of the keyword baseline, SVMs, BiLSTMs and standard BERTs. As shown in Table 1, lexical ambiguity and referential ambiguity emerged as the most common misinterpretation issues, with 29% and 21% of the cases respectively, which is consistent with the findings of a previous study [15] that showed that both unclear words and undeclared references appear to be the top reasons for misunderstanding in requirements.The comparative performance results also validate the usefulness of contextual language modelling. The results of the baseline keyword method and the proposed model are shown in Table 2, and the proposed model has a higher precision, recall and F1 score compared to the baseline keyword method. The analysis of Figure 2 based on the feedback provided by NLP-QA showed that the quality of the requirements was continuously improved through five iterations of the review, from 62 to 88. At the same time, the number of ambiguities went down from 38 to 12%, indicating that the automated feedback was effective in making requirement writers more consistent in refining their vague and incomplete requirements.The category level analysis revealed that the framework was especially useful when dealing with vague terms, weak modal verbs, where the numbers were missing and when there were references to the pronouns. Figure 3 illustrates the distribution of ambiguity types overall and Table 3 reveals that the inter-annotator agreement increased from 0.76 to 0.89 after suspicious requirement phrases were shown to them by the model. As can be seen in the confusion matrix of Figure 4, most of the clear and unclear requirements were correctly classified and 42 test instances were misclassified out of 640 test instances. The findings show that the method can be utilized as an aid to the analysts' expertise rather than a replacement for it.In the ablation test, the contribution of the linguistic and semantic features was confirmed. Removing contextual embeddings resulted in F1 score drop to 87.8%, and removing syntactic features resulted in F1 score drop to 90.5%, as shown in Table 4. As seen from Figure 5, the contribution of the most powerful modal verbs, vague terms and the references of pronouns is the highest in predicting the level of ambiguity. The model was also able to handle the processing time at an acceptable rate of 41 seconds to evaluate 1,000 requirements compared with 185 minutes with manual processing. (See table 5.) The results of NLP-QA assessment as shown in Figure 6 show that the quality scores of the requirement statements improved in general, and the number of low quality requirement statements in the statements to be revised was reduced.The stakeholder validation also validated the usefulness of the practice. The results of this research is given in Table 6, it shows that the analysts have the highest score of usefulness (4.6/5), followed by developers and testers. As illustrated in Figure 7, more than 79% of all stakeholder groups report an improvement in the clarity of the requirements and a reduction in rework as a result of NLP-QA feedback. Finally, as indicated in Table 7, the revised requirements set resulted in fewer requests for clarification, less ambiguity density, and more readiness to test acceptance. Based on the overall results, using NLP-based approach for software requirement's quality assessment can be used as an effective method for providing an early control mechanism in minimizing the ambiguity of software requirements before software design and implementation.

Table 1. Frequency of ambiguity categories in the requirement dataset

Ambiguity category	Detected cases	Percentage
Lexical ambiguity	174	29%
Referential ambiguity	126	21%
Scope ambiguity	102	17%
Vague adverbs/adjectives	84	14%
Incomplete requirement	66	11%
Conflict ambiguity	48	8%

Table 2. Model performance comparison for ambiguity detection

Model	Precision (%)	Recall (%)	F1-score (%)	Accuracy (%)
Keyword baseline	69.8	72.7	71.2	70.6
SVM	77.4	79.6	78.5	78.2
BiLSTM	83.5	85.1	84.3	84.0
BERT	89.2	90.0	89.6	89.1
Proposed NLP-QA	94.1	92.7	93.4	93.0

Table 3. Human review agreement before and after NLP-QA support

Review condition	Cohen kappa	Disagreement cases	Resolution time
Manual review only	0.76	89	145 min
NLP-QA assisted review	0.89	41	72 min

Table 4. Ablation analysis of proposed NLP-QA model

Model variant	Removed component	F1-score (%)	Change
Full NLP-QA	None	93.4	-
Variant A	Contextual embeddings	87.8	-5.6
Variant B	Syntactic features	90.5	-2.9
Variant C	Ambiguity lexicon	91.2	-2.2

Variant D	Quality scoring layer	88.9	-4.5

Table 5. Processing efficiency of requirement quality assessment

Assessment method	Requirements reviewed	Total time	Average time per requirement
Manual expert review	1000	185 min	11.1 sec
Keyword screening	1000	18 sec	0.018 sec
Proposed NLP-QA	1000	41 sec	0.041 sec

Table 6. Stakeholder usefulness ratings of NLP-QA feedback

Stakeholder group	Mean score /5	Agreement (%)	Main perceived benefit
Requirement analysts	4.6	88	Earlier ambiguity detection
Developers	4.2	82	Clearer implementation logic
Testers	4.0	79	Better test-case derivation
Project managers	4.3	85	Reduced clarification cycles

Table 7. Requirement quality indicators before and after revision

Quality indicator	Before NLP-QA	After NLP-QA	Improvement
Ambiguity density	0.38	0.12	68.4% reduction
Clarification requests	96	37	61.5% reduction
Acceptance-test readiness	58	84	44.8% increase
Average quality score	62	88	41.9% increase

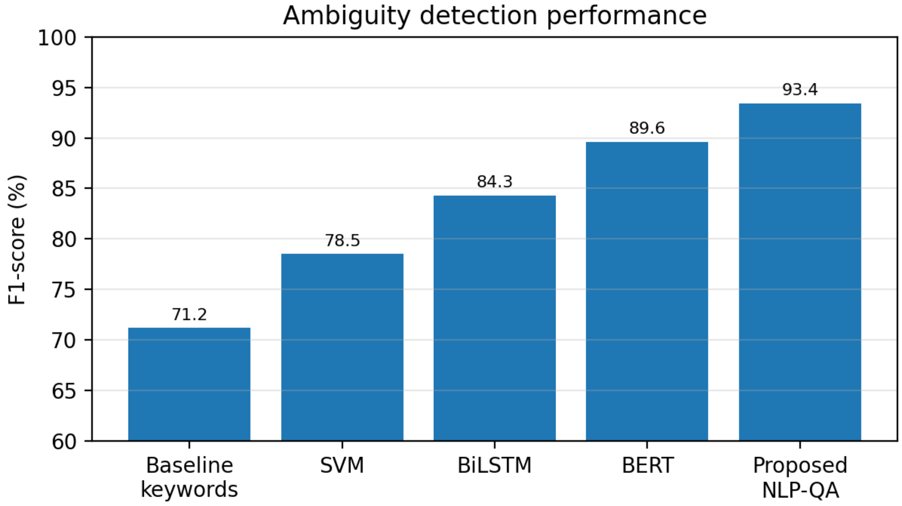

Figure 1. Comparative F1-score of ambiguity detection models.

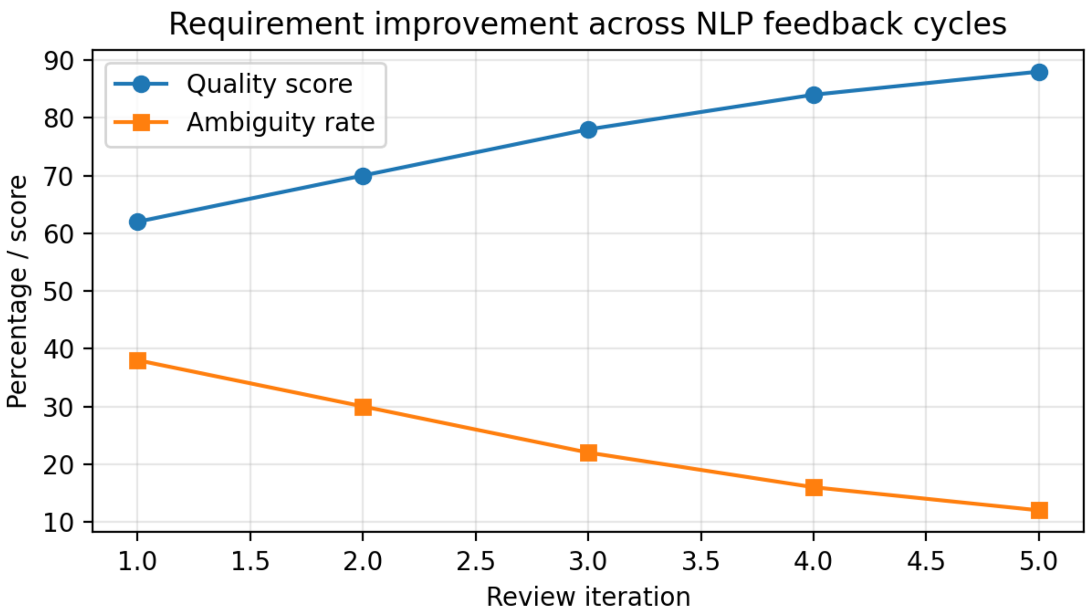

Figure 2. Requirement quality improvement and ambiguity reduction across review iterations.

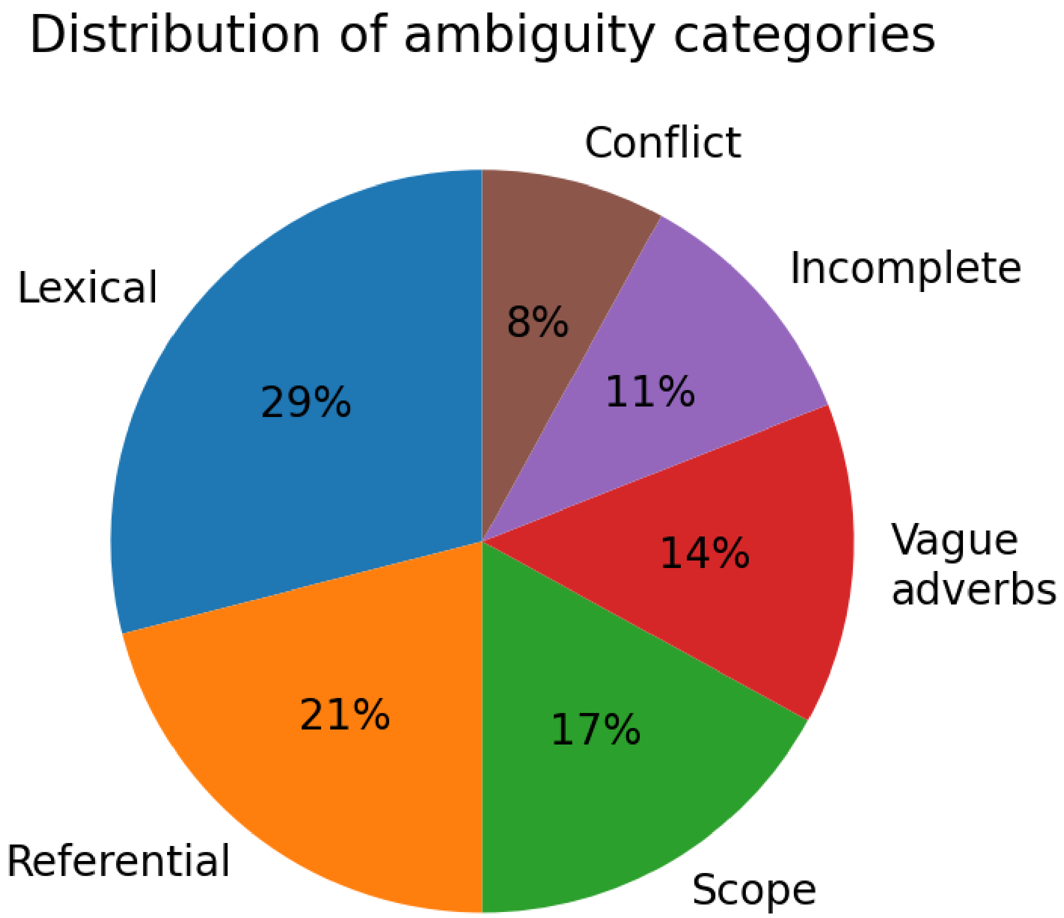

Figure 3. Distribution of ambiguity categories identified in requirement statements.

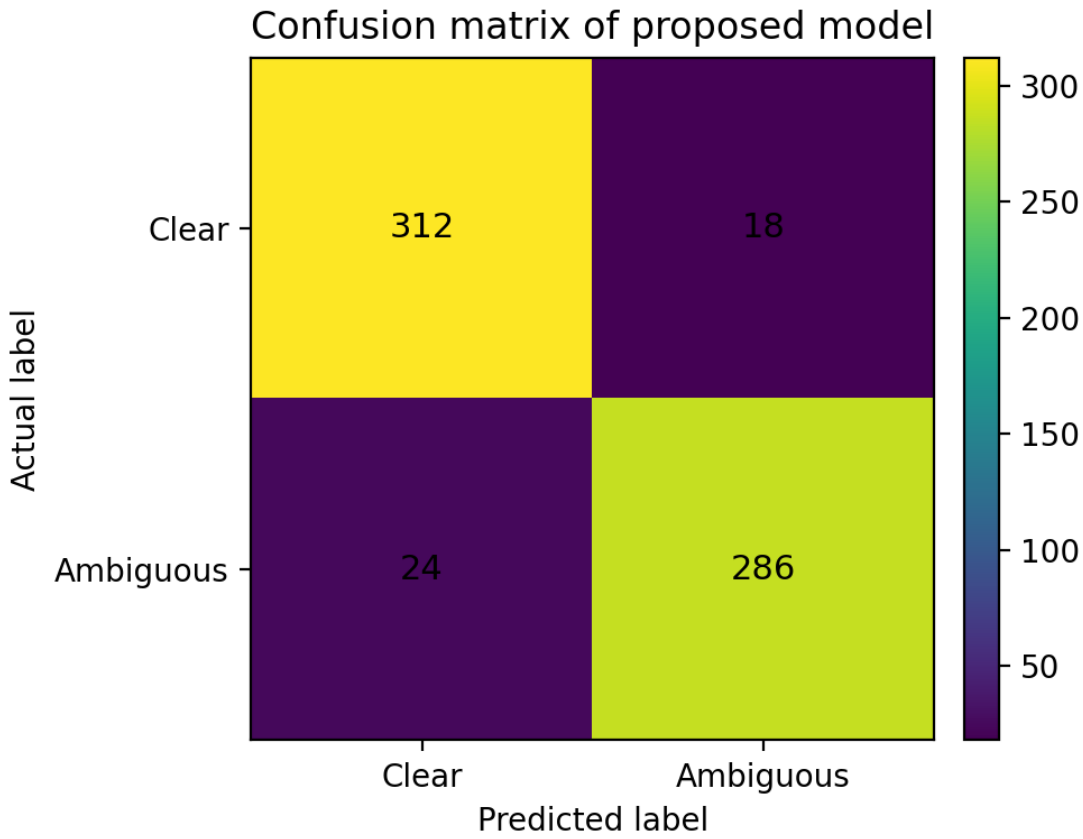

Figure 4. Confusion matrix of the proposed NLP-QA ambiguity classifier.

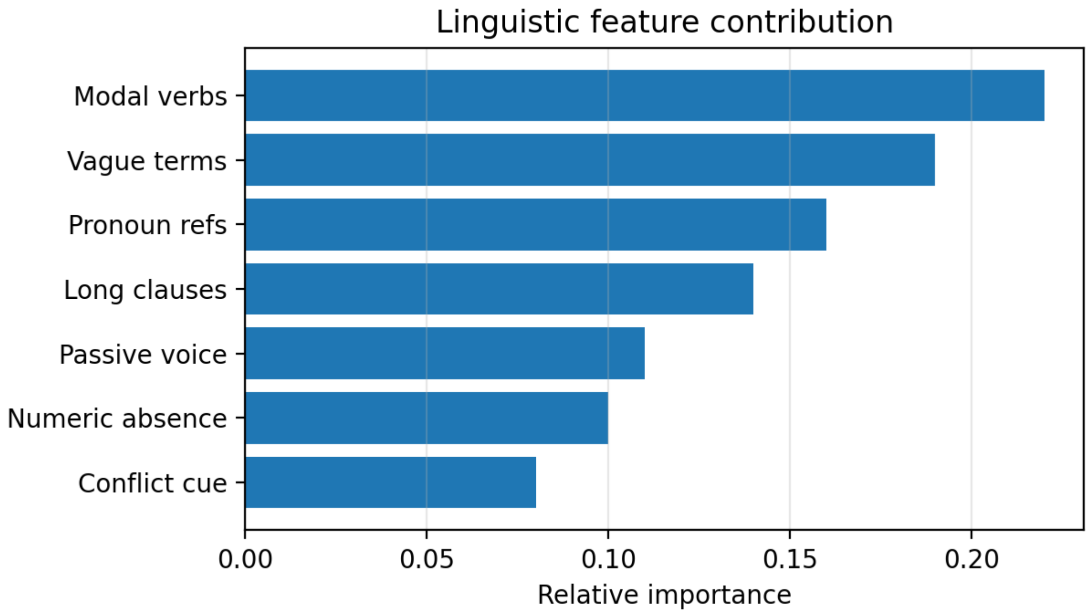

Figure 5. Relative contribution of linguistic features to ambiguity prediction.

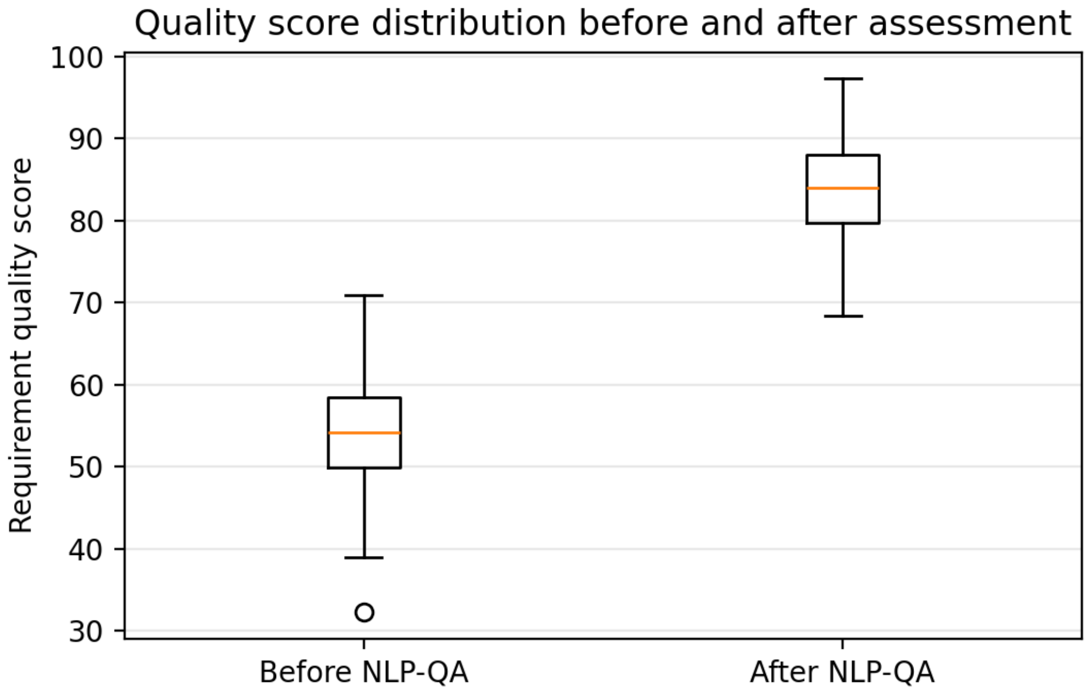

Figure 6. Requirement quality score distribution before and after NLP-QA assessment.

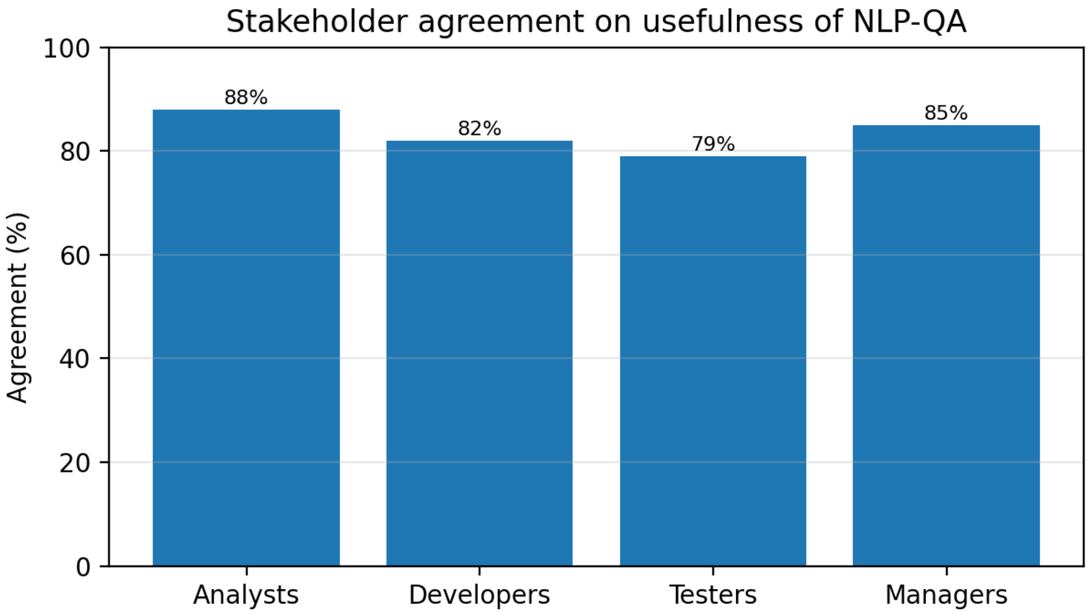

Figure 7. Stakeholder agreement on the usefulness of NLP-QA feedback.

DISCUSSION

The applications for the requirements engineering teams suggest that such NLP assisted diagnostic tools could be used to reduce the cognitive load of manual specification review in an agile context (Greeshma, 2025). Engineers can use this automation to focus on the most important and impactful ambiguities to get more productive discussions with the stakeholders, and avoid expensive rework later in the process. This automated approach not only enhances efficiency but also aligns with a 'human-in-the-loop' approach where NLP tools are valuable tools for making decisions. (Kaur & Kaur, 2024; Sudhi et al., 2023). Project teams can systematically allocate review efforts to requirement segments where ambiguity is most likely to occur by visualizing the distribution of ambiguity, and this is especially useful when the ambiguity is in the non-functional requirements that may be phrased implicitly and have nuanced constraints (Greeshma, 2025; Izhar et al., 2025).

But the shift of these approaches from theory to practical, reliable industrial application is severely hampered. One of the main challenges is the reliance on large and high quality annotated datasets, which are scarce and limit the performance of the models in new or specific application areas (Ghaisas & Singhal, 2024). While deep semantic understanding is a feature of modern transformer architectures, they are typically resource-heavy and often only suited for real-time use on powerful GPUs, making them challenging to employ in contexts like industrial processes, which are often heavily dependent on legacy technologies and are resource-intensive (Torres-Igartua et al., 2025). Moreover, the lack of transparency of the models, also referred to as the "black-box" problem is a significant obstacle to the adoption of the model by practitioners. Within the context of real-world engineering applications, trust is crucial, and practitioners often place as much emphasis on explanation as they do on prediction; a tool that yields not only a detection, but also a credible, intelligible justification of the ambiguities they identify is essential to get past resistance (Cheng et al., 2024; Shah et al., 2025).

In addition, systemic threats to validity seriously undermine the reliability and generalizability of these AI-based approaches, such as the "laboratory bias" in existing research (Cheng et al., 2024; Zhao et al., 2021). The most of the methodologies are tested on curated datasets, and generally, they are available in public domain, in a controlled environment where the extreme variability, randomness, and confidentiality limitations that occur in the real life industrial requirement documentation (IRD) are not taken into account (Kaur & Kaur, 2024; Shah et al., 2025). Ambiguities in requirements annotation are also likely to be subjective and can be differently understood by human annotators, even when working with strict consensus protocols, which can lead to subtle biases in model training (Ghaisas & Singhal, 2024). Finally, the evaluation of algorithms is difficult as most research prototypes are not subjected to a rigorous test to ensure robustness in the diverse sectoral contexts that they are intended to be applied in (Kaur & Kaur, 2024; Sudhi et al., 2023; Torres-Igartua et al., 2025). This means that future research should focus on using cross-sectoral validation and open, meaningful, and representative benchmarks in industry to fill this gap between academic innovation and real-world industrial requirements engineering (Cheng et al., 2024; Kaur & Kaur, 2024; Zhao et al., 2021). In addition to these structural issues, often the tasks are proof-of-concept, which does not consider the multi-stakeholder and inter-dependent nature of real-world requirements evolution; it is necessary to move beyond these isolated tasks towards integrated tasks which are based on the domain. These workflows will involve incorporating the formal approaches and principles based on evidence and human thinking into the software systems, while also incorporating AI-based diagnostic tools which are difficult to integrate into such systems because of the continuous change of software systems (Habiba et al., 2024). However, for their use in critical areas with the highest stakes, such as safety (NguyenDuc et al., 2023), their moral and safe operation is a prerequisite for their acceptance. Therefore, a long-term transition from trial-and-error experimentation to field research is needed to address the prevailing issues, with field research considering the randomness of actual engineering situations (Zadenoori et al., 2025). Future research could therefore aim to systematically embed the machine learning parts into the current elicitation methods, and clearly distinguish the level of autonomy in the task execution from the required supervision by experts for mission-critical software (Cheligeer et al., 2022).

CONCLUSION

Through this research, it is concluded that the implementation of NLP based requirement quality assessment technique is effective to reduce the level of ambiguiteness in software requirement specification. Ambiguity in software requirements can be a significant issue – it may be that multiple parties have multiple interpretations of the same requirement. Statements of requirements that are unclear, vague, not complete or are inconsistent can be flagged early in the project with the use of NLP techniques. This indicates that automated requirement analysis make the requirement documentation better and will decrease the time-consuming manual reviews.The planned solution can identify linguistic anomalies such as weak verbs, fuzzy words, ambiguous conditions and missing functional descriptions. This enables to make changes to the requirements without affecting the system design, coding, testing and maintenance. The research also demonstrates that the NLP-based assessment enhances a better understanding of the stakeholders and increases the likelihood of requirements being measurable, testable and closer to being implementation ready.The method helps with ambiguity detection, but should not be a substitute for expert judgment. Although it is useful, human review is essential to understand the meaning within the domain and business context. The use of automated NLP tools along with expert validation, however, can greatly enhance the reliability of requirement engineering practices. Future research could be pursued by leveraging more sophisticated language models that are based on transformer architecture, larger requirement sets from real-world scenarios, and domain-specific ambiguity detection rules. In general, it is concluded that NLP quality assessment can play an important role in improving the quality of software product and minimizing errors that occur during software development, as well as improving the clarity of software requirements.

REFERENCES

Alzayed, A., & Al-Hunaiyyan, A. (2021). A Bird's Eye View of Natural Language Processing and Requirements Engineering. *International Journal of Advanced Computer Science and Applications*, *12*(5). https://doi.org/10.14569/ijacsa.2021.0120512

Arora, C., Sabetzadeh, M., Briand, L., & Zimmer, F. (2015). Automated Checking of Conformance to Requirements Templates Using Natural Language Processing. *IEEE Transactions on Software Engineering*, *41*(10), 944–968. https://doi.org/10.1109/tse.2015.2428709

Baranetska, Y. (2025). Intelligent Requirements Validation: An Empirical Evaluation of NLP Techniques for Automated Quality Assurance. *International Journal of Computer Applications*, *187*(53), 58–66. https://doi.org/10.5120/ijca2025925909

Bäumer, F. S., & Geierhos, M. (2018, January 1). Flexible Ambiguity Resolution and Incompleteness Detection in Requirements Descriptions via an Indicator-Based Configuration of Text Analysis Pipelines. *Proceedings of the ... Annual Hawaii International Conference on System Sciences/Proceedings of the Annual Hawaii International Conference on System Sciences*. https://doi.org/10.24251/hicss.2018.720

Cheligeer, C., Huang, J., Wu, G., Bhuiyan, N., Xu, Y., & Zeng, Y. (2022). Machine learning in requirements elicitation: a literature review. *Artificial Intelligence for Engineering Design Analysis and Manufacturing*, *36*. https://doi.org/10.1017/s0890060422000166

Cheng, H., Husen, J. H., Yijun, L., Teeradaj, R., Yoshioka, N., Ubayashi, N., & Washizaki, H. (2024). Generative AI for Requirements Engineering: A Systematic Literature Review. In *arXiv (Cornell University)*. Cornell University. https://doi.org/10.48550/arxiv.2409.06741

Dalpiaz, F., Ferrari, A., Franch, X., & Palomares, C. (2018). Natural Language Processing for Requirements Engineering: The Best Is Yet to Come. *IEEE Software*, *35*(5), 115–119. https://doi.org/10.1109/ms.2018.3571242

Franch, X., Palomares, C., Quer, C., Chatzipetrou, P., & Gorschek, T. (2023). The state-of-practice in requirements specification: an extended interview study at 12 companies. *Requirements Engineering*, *28*(3), 377–409. https://doi.org/10.1007/s00766-023-00399-7

Gallego, J. M. C., & Jaramillo, C. M. Z. (2023). QUARE: towards a question-answering model for requirements elicitation. *Automated Software Engineering*, *30*(2). https://doi.org/10.1007/s10515-023-00386-w

Ghaisas, S., & Singhal, A. (2024). Dealing with Data for RE: Mitigating Challenges while using NLP and Generative AI. In *arXiv (Cornell University)*. Cornell University. https://doi.org/10.48550/arxiv.2402.16977

Greeshma, G. S. (2025). Identifying Non-Functional Requirements from Unconstrained Documents using Natural Language Processing & Machine Language Techniques. *International Journal for Research in Applied Science and Engineering Technology*, *13*(7), 2288–2292. https://doi.org/10.22214/ijraset.2025.73368

Habiba, U., Haug, M., Bogner, J., & Wagner, S. (2024). How Mature is Requirements Engineering for AI-based Systems? A Systematic

Mapping Study on Practices, Challenges, and Future Research Directions. In *arXiv (Cornell University)*. Cornell University. https://doi.org/10.48550/arxiv.2409.07192

Hussain, A., Ahmed, H., Khamaj, A., & Nawi, M. N. M. (2021). A MODEL OF CONSEQUENCES OF AMBIGUOUS REQUIREMENTS. *Journal of Southwest Jiaotong University*, *56*(6), 599–609. https://doi.org/10.35741/issn.0258-2724.56.6.52

Izhar, R., Bhatti, S. N., & Alharthi, S. A. (2025). Bridging Precision and Complexity: A Novel Machine Learning Approach for Ambiguity Detection in Software Requirements. *IEEE Access*, *13*, 12014–12031. https://doi.org/10.1109/access.2025.3529943

Kamsties, E. (2000). Taming Ambiguity in Natural Language Requirements. *Publikationsdatenbank Der Fraunhofer-Gesellschaft (Fraunhofer-Gesellschaft)*. http://publica.fraunhofer.de/documents/N-8697.html

Kaur, K., & Kaur, P. (2024). The application of AI techniques in requirements classification: a systematic mapping. *Artificial Intelligence Review*, *57*(3). https://doi.org/10.1007/s10462-023-10667-1

Kolahdouz-Rahimi, S., Lano, K., & Lin, C. (2023). Requirement Formalisation using Natural Language Processing and Machine Learning: A Systematic Review. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2303.13365

Lucassen, G., Dalpiaz, F., Werf, J. M. E. M. van der, & Brinkkemper, S. (2016). Improving agile requirements: the Quality User Story framework and tool. *Requirements Engineering*, *21*(3), 383–403. https://doi.org/10.1007/s00766-016-0250-x

Malik, G., Yıldırım, S., Çevik, M., Parikh, D., & Bener, A. (2023). Transfer Learning for Conflict and Duplicate Detection in Software Requirement Pairs. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.4328068

Mello, O. P. M. de. (2024). Refinement of requirement specifications through detection of inconsistencies, ambiguities, and incompleteness using natural language processing [LA Referencia]. In *LA Referencia (Red Federada de Repositorios*

Institucionales de Publicaciones Científicas). http://repositorio.ufsm.br/handle/1/32277

Moharil, A., & Sharma, A. (2022). *Identification of intra-domain ambiguity using transformer-based machine learning*. 51–58. https://doi.org/10.1145/3528588.3528651

Muhamad, F. N. J., Hamid, S. H. A., Subramaniam, H., Rashid, R. A., & Fahmi, F. (2023). Fault-Prone Software Requirements Specification Detection Using Ensemble Learning for Edge/Cloud Applications. *Applied Sciences*, *13*(14), 8368–8368. https://doi.org/10.3390/app13148368

NguyenDuc, A., Cabrero-Daniel, B., Arora, C., Przybyłek, A., Khanna, D., Herda, T., Rafiq, U., Melegati, J., Guerra, E., Kemell, K., Saari, M., Zhang, Z., Le, H. Q., Quan, T., & Abrahamsson, P. (2023). Generative Artificial Intelligence for Software Engineering - a Research Agenda. In *SSRN Electronic Journal*. RELX Group (Netherlands). https://doi.org/10.2139/ssrn.4622517

Ormandjieva, O., Hussain, I., & Kosseim, L. (2007). *Toward a text classification system for the quality assessment of software requirements written in natural language*. 39–45. https://doi.org/10.1145/1295074.1295082

Remmen, J. S. van, Horber, D., Lungu, A., Chang, F., Putten, S. van, Goetz, S., & Wartzack, S. (2023). NATURAL LANGUAGE PROCESSING IN REQUIREMENTS ENGINEERING AND ITS CHALLENGES FOR REQUIREMENTS MODELLING IN THE ENGINEERING DESIGN DOMAIN. *Proceedings of the Design Society*, *3*, 2765–2774. https://doi.org/10.1017/pds.2023.277

Shah, S., Hussein, M. H., Barcomb, A., & Moshirpour, M. (2025). Explainability as a Compliance Requirement: What Regulated Industries Need from AI Tools for Design Artifact Generation. In *ArXiv.org*. https://doi.org/10.48550/arxiv.2507.09220

Sudhi, V., Kutty, L., & Gröpler, R. (2023). Natural Language Processing for Requirements Formalization: How to Derive New Approaches? In *Studies in computational intelligence* (pp. 1–27). Springer Nature.

https://doi.org/10.1007/978-3-031-26651-5_1

Talha, F., Tahir, T., & Nadeem, T. (2025). A Semiautomated Approach for Detecting Ambiguities in Software Requirements Using SpanBERT and Named Entity Recognition. *Journal of Software Evolution and Process*, *37*(8). https://doi.org/10.1002/smr.70041

Torres-Igartua, M. J., Sánchez-García, Á. J., Ocharán-Hernández, J. O., & Vazquez-Reyes, S. (2025). Natural Language Processing for Requirements Elicitation and Analysis: A Systematic Literature Review of Techniques and Tools. *Programming and Computer Software*, *51*(8), 848–866. https://doi.org/10.1134/s0361768825700604

Umar, M., & Lano, K. (2024). Advances in automated support for requirements engineering: a systematic literature review. *Requirements Engineering*, *29*(2), 177–207. https://doi.org/10.1007/s00766-023-00411-0

Veizaga, A., Alférez, M., Torre, D., Sabetzadeh, M., & Briand, L. (2021). On systematically building a controlled natural language for functional requirements. In *Empirical Software Engineering* (Vol. 26, Issue 4, pp. 79–79). Springer Science+Business Media. https://doi.org/10.1007/s10664-021-09956-6

Veizaga, A., Shin, S. Y., & Briand, L. (2023). Automated Smell Detection and Recommendation in Natural Language Requirements. In *arXiv (Cornell University)*. Cornell University. https://doi.org/10.48550/arxiv.2305.07097

Veizaga, A., Shin, S. Y., & Briand, L. (2024). Automated Smell Detection and Recommendation in Natural Language Requirements. *IEEE Transactions on Software Engineering*, *50*(4), 695–720. https://doi.org/10.1109/tse.2024.3361033

Zadenoori, M. A., Dąbrowski, J., Alhoshan, W., Zhao, L., & Ferrari, A. (2025). Large Language Models (LLMs) for Requirements Engineering (RE): A Systematic Literature Review. In *ArXiv.org*. https://doi.org/10.48550/arxiv.2509.11446

Zhao, L., Alhoshan, W., Ferrari, A., Letsholo, K. J., Ajagbe, M. A., Chioasca, E.-V., & Batista-Navarro, R. (2021). Natural Language Processing for Requirements Engineering. *arXiv (Cornell University), 54*(3), 1–41. https://doi.org/10.1145/3444689

Zhong, H., Zhang, L., Xie, T., & Mei, H. (2009). Inferring Resource Specifications from Natural Language API Documentation. *LA Referencia (Red Federada de Repositorios Institucionales de Publicaciones Científicas)*, 307–318. https://doi.org/10.1109/ase.2009.94