Proc. of the European Safety and Reliability Conference (ESREL 2026)
Edited by José C. Matos, Paulo B. Lourenço, Michael Beer, Daniel Oliveira,
Edoardo Patelli, Hélder S. Sousa, Rui A. Silva and Monica Santamaria
 Published by Research Publishing, Singapore.
doi: 10.3850/ESREL2026061419_esrel26-p27901-cd

Annotating Maintenance Reports with a Domain-Specific Ontology and a Large Language Models to Extract Reliability Data from Industrial Systems

Dario Valcamonico

Energy Department, Politecnico di Milano, Italy. E-mail: dario.valcamonico@polimi.it

Piero Baraldi

Energy Department, Politecnico di Milano, Italy. E-mail: piero.baraldi@polimi.it

Enrico Zio

MINES Paris-PSL, Centre de Recherche sur les Risques et les Crises (CRC), France
Energy Department, Politecnico di Milano, Italy. E-mail: enrico.zio@polimi.it

While maintenance reports contain valuable data for supporting reliability analysis, extracting them is hindered by technical jargon, extreme conciseness and lack of standardization. To address these challenges, we propose a method that combines a Large Language Model (LLM) with a domain-specific ontology to automatically annotate maintenance reports. Specifically, the annotation task involves assigning semantic labels to specific text spans and identifying relations between them. The identified entities and relations subsequently serve for deriving quantitative reliability metrics, such as component failure rates. The proposed method is validated using a repository of maintenance reports concerning malfunctions in traction systems of a fleet of freight transport trains. The obtained results show that the method can effectively annotate maintenance reports, achieving an average F1-scores of 0.903 and 0.733 for entities and relations, respectively.

Keywords: Natural Language Processing, Large Language Model, Annotation, Ontology, Maintenance Reports.

1. Introduction

Maintenance reports contain unstructured text written by system operators following their interventions (Bikaun et al. 2024). This rich source of information regarding observed failures, malfunctions and maintenance actions is valuable for conducting reliability analysis (Stenström et al. 2015). However, a major challenge in the automatic extraction of reliability data from these reports lies in the inherent ambiguity and context-dependency of language. This difficulty is further exacerbated by:

- the conciseness of the unstructured text, which may hinder accurate annotation, as the limited information may lead to ambiguous interpretations (Conte et al. 2021);
- the complexity of the language, which uses technical words, domain-related acronyms, abbreviations and codes (Brundage et al. 2021).

To address these challenges, we propose a method to automatically annotate maintenance reports. Annotation involves classifying the words of the unstructured texts in predefined classes (entities) and identifying the relations between them (Bikaun et al. 2024). One of the main difficulties is that the same word or phrase can have different semantic meaning depending on its context, leading to inconsistencies in annotation. For example, the word "*valve"* can be associated to different entities based on the contextual information: in the text "*emergency safety valve*" it is best classified as a protective object, whereas in the text "*gate valve*" it is best classified as a controlling object.

Text annotation in industrial applications relies on the use of ontologies, i.e. formal representations

of knowledge relevant to a particular domain or task. Ontologies are structured using hierarchical levels of entities, which represent classes of semantically homogeneous concepts of the domain knowledge (e.g. components, accidents), and relations (e.g. logic, causal) among them (Batres et al. 2014). By grounding unstructured text in formalized domain knowledge, ontologies allow engineers to systematically extract information from reports.

Since human annotation of a large number of reports can be time-consuming and error-prone, several methods have been developed to facilitate this process. For example, (Bikaun et al. 2022) developed a graphical tool for annotating reports and propagating these annotations throughout the repository. Also, the application of Natural language Processing (NLP) methods to automatically annotate reports has been proposed. For example, (Wang et al. 2026) utilized an annotation method based on Bidirectional Encoder Representation from Transformers (BERT) to identify risk factors from offshore platform accident reports. These factors were, then, used to construct a BN, where the prior and conditional probabilities were derived from keywords counts, thereby enabling quantitative risk assessment of offshore platform operations.

Concerning NLP, traditional methods such as Term Frequency-Inverse Document Frequency (TFIDF) (Amati and Van Rijsbergen 2002) are useful for identifying relevant terms, but they are limited by their dependence on word occurrence or fixed representations, which can neglect word order, contextual variability and long-range semantic relationships (Mikolov et al. 2013). Transformer-based methods such as Large Language Models (LLMs) have been recently introduced to address these limitations through self-attention and context-dependent representations, which have been shown to significantly improve the identification and extraction of information from textual data (Q. Liu et al. 2025).

In this work, the annotation is performed by querying an LLM. A limitation of LLMs is that they can produce hallucinations, i.e., syntactically correct answers that are either factually incorrect or inconsistent with the input prompt (Huang et al. 2025). As a result, hallucinations in annotation tasks can directly compromise the accuracy and consistency of the resulting annotations (Y. Liu et al. 2024). To address this limitation, we consider an LLM equipped with Chain-of-Thought (CoT) to perform step-by-step reasoning before answering (DeepSeek-AI 2025), and, according to (Salewski et al. 2023), we have structured the input prompt in sections. These choices ensure that the LLM adequately understands the context and the task of annotating maintenance reports.

The method is validated considering the maintenance of traction systems of a fleet of freight transport trains and a historical repository of maintenance reports.

The remainder of this work is organized as follows: Section 2 states and formulates the problem; Section 3 illustrates the developed ontology; Section 4 describes the annotation method; Section 5 presents the case study. Section 6 reports the results obtained; Section 7 discusses the work conclusions.

2. Problem statement and formulation

We consider a repository of D maintenance intervention reports $\{d_i, i = 1, \dots, D\}$. A generic report d_i is an unstructured textual description of the accident or malfunction that occurred and the maintenance intervention performed, and it is composed of P_i words $\{w_i^1, \dots, w_i^p, \dots, w_i^{P_i}\}$.

The problem of annotating the reports is addressed by defining an ontology composed of N entities $\{E_1, \dots, E_n, \dots, E_N\}$ and M relations $\{R_1, \dots, R_m, \dots, R_M\}$. Then, an LLM is used to annotate each report d_i by: 1) assigning each word w_i^p or span (e.g. $[w_i^{p-1}, w_i^p]$) to an entity E_n, and 2) assigning a relation R_m between two identified entities of the report. Figure 1 shows an example of annotation of a report.

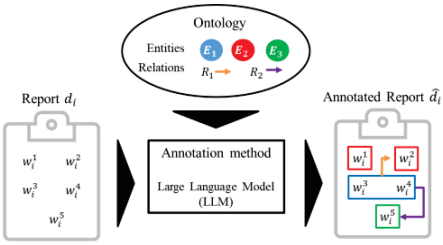

Figure 1: Annotation of a report

3. Ontology

We consider the ontology developed in (Bikaun et al. 2024). It features 224 distinct entities hierarchically organized in three levels. The first level contains five primary entities:

- *Activity*, indicating activities related to maintenance and support actions performed on physical objects (e.g., *refill*, *replace*, *clean*);
- *PhysicalObject*, indicating an object in the system (e.g., *pump*, *engine*, *valve*). The second and third levels classify the objects based on their function (e.g., *HoldingObject*, *ConnectingObject*);
- *Process*, indicating accidental events occurring to physical objects (e.g., *leakage*, *stuck open*);
- *Property*, indicating the attributes of a physical object (e.g., *crack*, *isolated*);
- *State*, indicating the conditions of physical objects (e.g., *broken, degraded*).

and 6 relations:

- *contains*, used to denote the containment of physical objects (e.g., engine *contains* oil);
- *isA*, used to denote a subtype relationship between entities (e.g., diesel engine *isA* engine);
- *hasPart*, used to denote part-whole relationships (e.g., engine *has part* radiator);
- *has Agent*, used to denote entities actively involved or initiating an action or event (e.g., repair *has agent* operator);
- *hasPatient*, used to denote entities or undergoing an action or event (e.g., leakage *has patient* pipe);
- *hasProperty*, used to denote the possession of a particular characteristic by an entity (e.g., pipe *hasProperty* degraded).

In the present work, we consider a reduced version of the ontology proposed in (Bikaun et al. 2024) characterized by a limited number of levels and entities. The objective is to streamline the annotation process by reducing expert effort and the possibility of inconsistent assignments of entities and relations. Specifically, the following modifications are made:

- The structure of the ontology is reduced from three to two levels, considering only the entities in level 2 of the original ontology;
- Some level 2 entities are not considered since they contain entities specific for the industrial domain of application in (Bikaun et al. 2024) (e.g. *PresentingObject*);
- The relation *contains* is merged with the relation *hasPart* due to their semantic similarity.

Table 1 reports the entities and relations of the ontology considered in this work.

Table 1: Ontology used in this work.

	Level 1	Level 2	Examples
Entity	MaintenanceActivity	Adjust	*refill, clean*
		Replace	*replace, change*
		Repair	*refit, tighten*
		Reset	*restart*
	SupportingActivity	Isolate	*isolate*
		Observe	*notice, observe*
		Communicate	*say, announce*
	PhysicalObject	Controlling	*contactor*
		CoveringObject	*panel*
		DrivingObject	*engine*
		EmittingObject	*alarm*
		GeneratingObject	*battery, fan*
		GuidingObject	*cable*
		HoldingObject	*bolt, screw*
		HumanInteractionObject	*screen*
		InformationProcessingObject	*relay*
		InterfacingObject	*terminal box*
		MatterProcessingObject	*filter*
		ProtectingObject	*diode,*
		RestrictingObject	*resistor, brake*
		SensingObject	*sensor*
		StoringObject	*tank*
		TransformingObject	*inverter, pump*
	Substance	Gaseous	*air*
		Liquid	*oil*
		Solid	*dust*
	Personnel	Personnel	*operator*
	Process	DesirableProcess	*no operational impact*
		UndesirableProcess	*leakage*
	Property	DesirableProperty	*active*
		UndesirableProperty	*crack, stuck open*
	State	NormalState	*normal*
		DegradedState	*degraded*
		FailedState	*broken*
Relation	isA	isA	diesel engine *is a* engine
	hasPart	hasPart	engine *hasPart* oil
	hasAgent	hasAgent	operator *has agent* repair
	hasPatient	hasPatient	leakage *has patient* pipe
	hasProperty	hasProperty	pipe *hasProperty* broken

4. Annotation method

The annotation of the reports is performed using an LLM equipped with Chain-of-Thought (CoT) to perform step-by-step reasoning before answering (DeepSeek-AI 2025). Specifically, the LLM receives the report d as input and generates the annotated report $\hat{d}$ as output. The LLM is queried using the prompt reported in Figure 2, which contains a description of the task of identifying entities and relations of the ontologies. The annotation of the reports is performed considering the $N = 34$ entities and $M = 5$ relations in the second level of the ontology of Table 1.

5. Case study

We consider a repository comprising hundreds of maintenance reports of traction systems of freight transport trains. The exact number of reports is not disclosed for confidentiality reasons. Two representative examples of these reports are provided in Table 2.

Table 2: Two examples of reports.

Maintenance report
lp reported ios loss of ed braking. tcu was showing isolated

aux converter isolated normalized vcb open contactor faulty

6. Results

The performance of the proposed annotation method is evaluated by comparison with the manual annotations made by an expert.

Tables 3 and 4 report the performance metrics of precision (fraction of spans labelled by the LLM as a given class that are correctly classified), recall (fraction of spans of a given class that are correctly identified) and F1-score (harmonic mean of precision and recall) of the proposed method evaluated on 50 reports. These metrics have been computed individually for the specific classes (entity and relation) and globally, as average across all classes.

Concerning the classification of the entities, most of the entities have been correctly identified. The lowest performance is observed for the entity *UndesirableProcess*, due to its semantic similarity with the entities *UndesirableProperty* and *State.*

For example, the description of a *pump leaking* due to a *degraded seal* or a valve *occluding* as a consequence of material degradation, can make it difficult for the LLM to discriminate among the undesirable process itself (*UndesirableProcess*), the degradation affecting the component (*UndesirableProperty*), and its resulting state (*State*).

Concerning the classification of relations, the overall performances are less satisfactory than those observed for the classification of the entities. This is in agreement with the findings of (Bikaun et al. 2024), where a global f1-score of 0.655 is obtained for the classification of the relations.

Role
You are an expert annotator working for a railway safety team. Your task is to extract ontology-based entities and relations from maintenance reports of electric freight trains.

Input
Each maintenance report is a sentence or paragraph describing an accident, failure, inspection, or repair activity. The reports are provided in the Python list `reports_prompt`, where each entry is of the form [accident description, severity label, maintenance action label].
Here is the maintenance report list: {reports_prompt}

An ontology is available, defined as:
- A list `ontology_entities` of entities with structure: ['entity category', 'entity ID', 'entity type', 'entity description', 'example spans']
  Here is the ontology entity list: {ontology_entities}
- A list `ontology_relations` with structure: ['relation ID', 'relation type', 'relation description', 'example usage']
  Here is the ontology relation list: {ontology_relations}

Task
For each accident report:
1. Tokenize the accident description into individual words.
2. Identify all entity spans that match ontology-based entity types.
   - For each span, return:
     - `type`: the entity class (e.g., 'SensingObject')
     - `start`: start index of the span (inclusive)
     - `end`: end index of the span (exclusive)
3. Identify all relations between the detected entities.
   - For each relation, return:
     - `type`: the relation type (e.g., 'hasPatient')
     - `head`: index of the head entity (based on order in `entities` list)
     - `tail`: index of the tail entity

Response Format
Return a list of dictionaries, one per report, each with:
- `tokens`: list of tokenized words
- `entities`: list of dictionaries with `type`, `start`, `end`
- `relations`: list of dictionaries with `type`, `head`, `tail`

Examples
Example 1 ...
Example 2 ...
Example 3 ...

Figure 2: Prompt of the LLM

Table 3: Accuracy in the classification of the entities (sorted by number of occurrences in the test dataset).

Entity	Precision	Recall	F1-score	Occurrences in the test dataset
UndesirableProcess	0.714	0.714	0.714	7
ControllingObject	1.000	1.000	1.000	6
Adjust	1.000	1.000	1.000	4
Isolate	0.800	1.000	0.889	4
StoringObject	1.000	0.750	0.857	4
UndesirableProperty	1.000	0.750	0.857	4
Liquid	1.000	1.000	1.000	3
GuidingObject	0.667	1.000	0.800	2
NormalState	1.000	1.000	1.000	2
Observe	1.000	1.000	1.000	2
FailedState	1.000	1.000	1.000	1
Personnel	1.000	1.000	1.000	1
Replace	1.000	1.000	1.000	1
Average	0.903	0.903	0.903	41

Table 4: Accuracy in the classification of the relations (sorted by number of occurrences in the test dataset).

Relation	Precision	Recall	F1-score	Occurrences in the test dataset
hasPatient	0.750	0.808	0.778	26
hasPart	0.500	0.500	0.500	2
hasAgent	0.000	0.000	0.000	1
isA	0.000	0.000	0.000	1
Average	0.733	0.733	0.733	30

The classification of relations is more difficult than that of entities because it requires to correctly identify not only the type of relation but also the two spans involved in the relation. Notice that the occurrence in the reports of the relations *hasPart*, *hasAgent* and *isA* is relatively smaller than that of the relation *hasPatient*. This is due to the conciseness of reports, which tend to contain only information related to the affected objects, their issues and maintenance actions, often omitting information (entities) of the subsystems to which they belong and the technician performing the action.

The obtained annotations can be used to estimate reliability metrics, such as component failure rates. For example, considering a system pump, we can estimate its total number of failures by counting the reports where the text span describing the pump is annotated as a "*TransformingObject"*, the malfunction is annotated as a "*FailedState"* and the relation "*hasPatient*" is identified between the two entities.

7. Conclusion

This paper presented a method that integrates an LLM with a domain-specific ontology to automatically annotate maintenance reports. The classification accuracy obtained for both entities and relations is satisfactory, demonstrating the capability of the method to correctly identify physical objects, malfunctions and maintenance actions from the unstructured text of the reports. Ultimately, these annotated reports form the basis for estimating reliability metrics by counting records that fulfil specific criteria for the targeted components and malfunctions. Future work will

be devoted to the use of the annotated reports for defining probabilistic methods, such as Bayesian Networks, for reliability analysis.

Acknowledgement

The participation of Dario Valcamonico, Piero Baraldi and Enrico Zio to this work is supported by the MOSAIC project under the HORIZON-JU-Chips-2024-2-RIA-T1 call of the European Union, Grant Agreement 101194414. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or Chips Joint Undertaking. Neither the European Union nor the granting authority can be held responsible for them.

References

Amati, Gianni, and Cornelis Joost Van Rijsbergen. 2002. "Probabilistic Models of Information Retrieval Based on Measuring the Divergence from Randomness." *ACM Transactions on Information Systems* 20 (4). https://doi.org/10.1145/582415.582416.

Batres, Rafael, Shinya Fujihara, Yukiyasu Shimada, and Testuo Fuchino. 2014. "The Use of Ontologies for Enhancing the Use of Accident Information." *Process Safety and Environmental Protection* 92 (2). https://doi.org/10.1016/j.psep.2012.11.002.

Bikaun, Tyler K., Tim French, Michael Stewart, Wei Liu, and Melinda Hodkiewicz. 2024. "MaintIE: A Fine-Grained Annotation Schema and Benchmark for Information Extraction from Maintenance Short Texts." *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, 10939–10951. https://aclanthology.org/2024.lrec-main.954.

Bikaun, Tyler K., Michael Stewart, and Wei Liu. 2022. "QuickGraph: A Rapid Annotation Tool for Knowledge Graph Extraction from Technical Text." *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 270–278. https://doi.org/10.18653/v1/2022.acl-demo.27.

Brundage, Michael, Thurston Sexton, Melinda Hodkiewicz, Alen Dima, and Sarah Lukens. 2021. "Technical Language Processing: Unlocking Maintenance Knowledge." *Manufacturing Letters* 27. https://doi.org/10.1016/j.mfglet.2020.11.001.

Conte, Anna, Coline Bolland, Lynn Phan, Michael P. Brundage, and Thurston Sexton. 2021. "The Impact of Data Quality on Maintenance Work Order Analysis: A Case Study in Historical HVAC Maintenance Work Orders." *Proceedings of the European Conference of the PHM Society*. https://doi.org/10.36001/phme.2021.v6i1.2814.

DeepSeek-AI. 2025. *DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning*.

Huang, Lei, Weijiang Yu, Weitao Ma, et al. 2025. "A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions." *ACM Transactions on Information Systems* 43 (2). https://doi.org/10.1145/3703155.

Liu, Qiong, Yuexiong Ding, and Xiaowei Luo. 2025. "Automated Knowledge Graph-Based Risk Assessment for Fall-from-Height Accidents in Construction." *Automation in Construction* 179. https://doi.org/10.1016/j.autcon.2025.106482.

Liu, Yang, Yuanshun Yao, Jean-Francois Ton, et al. 2024. "Trustworthy LLMs: A Survey and Guideline for Evaluating Large Language Models' Alignment."

ArXiv E-Prints, ahead of print. https://doi.org/10.48550/arXiv.2308.05374.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. "Efficient Estimation of Word Representations in Vector Space." *International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.1301.3781.

Salewski, Leonard, Stephan Alaniz, Isabel Rio-Torto, Eric Schulz, and Zeynep Akata. 2023. "In-Context Impersonation Reveals Large Language Models Strengths and Biases." *Proceedings of the 37th International Conference on Neural Information Processing Systems*. https://doi.org/10.5555/3666122.3669274.

Stenström, Christer, Mustafa Aljumaili, and Aditya Parida. 2015. *Natural Language Processing of Maintenance Records Data*.

Wang, Dan, Kai Yin, and Hailong Wang. 2026. "Risk Identification in Prefabricated Building Construction Safety Systems Based on STPA-TM." *Reliability Engineering and System Safety* 268. https://doi.org/10.1016/j.ress.2025.112004.