Automotive Document Labeling Using Large Language Models

Thin Van Dang[1] **and Cuong Xuan Chu**[2] **and Christian Graf**[2]
Tobias Kaminski[2] **and Trung-Kien Tran**[2]

[1]Bosch Global Software Technologies, Ho Chi Minh City, Vietnam
[2]Bosch Center for Artificial Intelligence, Renningen, Germany
`firstname.lastname@de.bosch.com`

Abstract

Repairing and maintaining car parts are crucial tasks in the automotive industry, requiring a mechanic to have all relevant technical documents available. However, retrieving the right documents from a huge database heavily depends on domain expertise and is time-consuming and error-prone. By labeling available documents according to the components they relate to, concise and accurate information can be retrieved efficiently. However, this is a challenging task as the relevance of a document to a particular component strongly depends on the context and the expertise of the domain specialist. Moreover, component terminology varies widely between different manufacturers. We address these challenges by utilizing Large Language Models (LLMs) to enrich and unify a component database via web mining, extracting relevant keywords, and leveraging hybrid search and LLM-based re-ranking to select the most relevant component for a document. We systematically evaluate our method using various LLMs on an expert-annotated dataset and demonstrate that it outperforms the baselines, which rely solely on LLM prompting.

1 Introduction

For documenting repair and maintenance processes, the automotive industry relies on a vast amount of technical documents reflecting the increased technical depth of modern automotive products. These documents contain detailed information about automotive components, ranging from mechanical parts to complex electronic systems. However, as the volume and complexity of technical documents continue to grow, efficiently retrieving and identifying those relevant to a specific component in need of repair has become a significant challenge for engineers and technical personnel. In addition, each car manufacturer uses different terminology to describe various components (Robert Bosch GmbH, 2022). For example, some manufacturers use the term `powertrain`, to refer to the engine transmission, and driveshafts, while others may use `drivetrain`[1] to indicate everything beyond the engine.

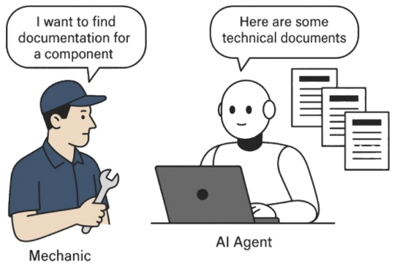

Figure 1: Our method enables technical documents from the automotive domain to be searchable, thus providing the right documents to a car mechanic.

Traditionally, mechanics manually browse lengthy documents to locate necessary component information, a process that is time-consuming and unreliable. This inefficiency not only delays development and maintenance cycles but also increases the risk of overlooking critical details. On the other hand, diverse terminology used by different automakers often leads to confusion and ambiguity (Zellmer et al., 2024). To address these issues, labeling all documents with their most relevant components before providing them to the mechanics becomes a reasonable strategy. This significantly reduces the search space, thereby improving efficiency and enabling the retrieval of more accurate and concise documents. Zhu (2025) also demonstrates that automating the labeling of technical documents is critically important within the automotive industry.

With recent advances of LLMs (Chang et al., 2024; Zhang et al., 2025) in various natural language processing (NLP) tasks–such as text understanding (Sun et al., 2023; Zheng et al., 2025), text generation (Wu et al., 2025), information extrac-

[1]https://en.wikipedia.org/wiki/Powertrain

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 1588–1595
November 4-9, 2025 ©2025 Association for Computational Linguistics

tion (Xie et al., 2023; Kang and Shin, 2025) and text labeling (Alaofi et al., 2024; Thomas et al., 2024)–LLMs have demonstrated specialized linguistic knowledge in narrow domains, thanks to extensive training data. As a result, LLMs are an excellent tool for labeling automotive documents. However, although LLMs are capable of handling significantly larger context windows recently, it is often not feasible to feed all components along with the document to an LLM and prompt it to pick the most relevant components, especially when thousands of components are being used.

This paper presents a solution for a real-world automotive use case: automatically extracting and identifying automotive components from technical documents. The goal is to provide accurate, concise, and contextually relevant component information to assist mechanics during diagnostic or repair activities (as shown in Figure 1). To address existing challenges, we utilize LLMs to enrich the component database via web mining and index them into a search database. Given a technical document, our method relies on LLMs to extract relevant keywords, then performs hybrid search followed by semantic and LLM-based re-ranking to select the single most relevant component. We evaluate the method on an expert-annotated dataset, achieving promising results and outperforming pure LLM-based prompting approaches. Our main contributions are as follows.

- We address a novel and practical problem within the automotive industry. While document labeling is a well-studied task, its application to the automotive domain, particularly in handling thousands of automotive components, remains underexplored.
- We systematically designed our method to scale to large amounts of documents and components, overcoming the limitations of off-the-shelf LLMs by integrating LLMs with hybrid search, advanced prompting, and semantic re-ranking.
- We evaluated our method using various LLMs on an expert-annotated dataset and demonstrated that it outperforms the baselines, which rely solely on LLM prompting.

2 Related work

LLMs on NLP Extraction Tasks. With the development of LLMs (Chang et al., 2024; Zhang et al., 2025), many NLP downstream tasks, including information extraction and document labeling, can now be solved through prompt engineering techniques, eliminating the need for task-specific training or fine-tuning. In particular, Xie et al. (2023) explored the performance of ChatGPT on the zero-shot named entity recognition task. Wei et al. (2024) conducted an empirical study on various complex IE tasks, decomposed them into multiple simpler sub-tasks and used a multi-turn prompting strategy to obtain the final results. Martínez-Cruz et al. (2024) demonstrated the superiority of ChatGPT over state-of-the-art models for the key-phrase generation on various public datasets in diverse domains and for various document lengths. Another work (Kang and Shin, 2025) also investigated four prompting strategies for the keyphrase extraction and experimented on six benchmark datasets, demonstrated that the prompting-based solution was more effective than the traditional approach for this task.

Document Labeling. For text or document labeling tasks, the work of Alaofi et al. (2024) demonstrated that LLMs can assign relevant labels to texts with accuracy comparable to human judgments, especially when using larger and more expensive models. However, these models are prone to false positives due to the presence of query terms in passages, even when the actual content is irrelevant. The research by Thomas et al. (2024) investigated the application of LLMs to directly generate high-quality relevance labels from actual searcher interactions. This approach demonstrated concrete advantages, including notable enhancements in search ranker performance. However, with respect to specific domains such as automotive, to the best of our knowledge, our study is the first to address the real-world application of document labeling, particularly in the context of mechanical engineering tasks involving car components.

3 Methodology

In our setting, we define that a component is considered *relevant* to a document if the document is necessary and useful for mechanics during fixing or maintaining the component. Let $C = \{c_1, c_2, \ldots, c_n\}$ denote a predefined set of automotive components, where $n \geq 7000$. Given an input document d, constituted by an HTML file containing the technical content and metadata m in XML format, the objective is to identify the subset

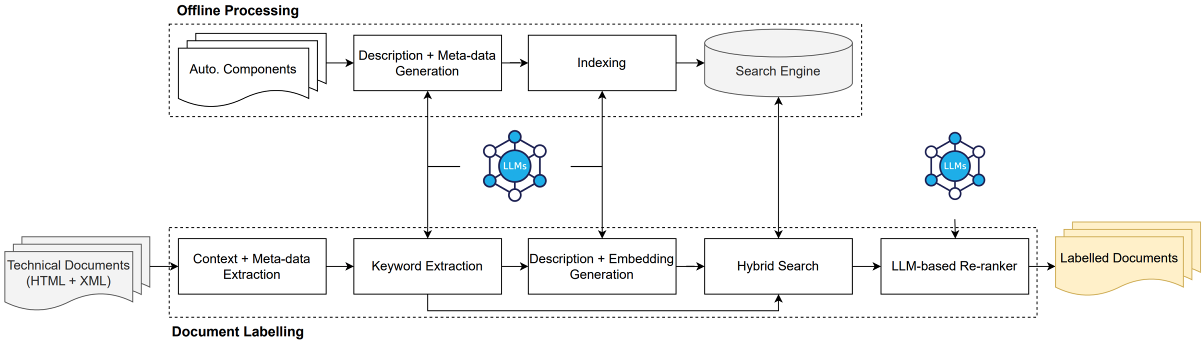

Figure 2: Our pipeline for automotive document labeling.

of components $c^* \subseteq C$ for which the content of d is most relevant.

Due to the large number of target labels and domain-specific data, we propose a multi-step framework consisting of LLM-based keyword extraction and generation, hybrid search over all automotive components and re-ranking to retrieve the relevant components for a document. Figure 2 depicts details of our method.

3.1 Automotive Component Data Enrichment

Labeling automotive components across different document sources is challenging for several reasons: the lack of detailed definitions for the terminology used, the dependence on context and the variations in terminology between different car brands. For instance, `Data Bus Diagnostic Interface` is used in some original equipment manufacturer (OEMs) documents while others use `Gateway`, which is a synonym. Manually defining or collecting this information with human experts is difficult and resource-intensive. To address these challenges, we design a two-step approach to enrich the information of automotive component data in our database, which includes description generation and meta-data collection.

Description Generation. The goal of this step is to generate a unified description for each automotive component. To avoid LLM-hallucination, we utilize Bing-Search[2] to retrieve relevant information from the internet with a specific search query for each component c (e.g. *what is c?*). This enables the system to capture diverse terminology and richer semantic context, improving the accuracy of component retrieval and labeling across different manufacturers. From retrieved web pages/files, an LLM is used to generate the description Desc_{c_i} for the component c_i. For example, a description of `Gateway` is: *"A gateway in automotive systems manages communication between different networks within the vehicle, ensuring data is correctly routed and translated between various electronic control units (ECUs)".*

Meta-data Collection. Along with the descriptions, meta-data is collected via the web search. The meta-data includes system types and a synonym list, instance collections or alternative terminology for components used in the automotive industry. This information could enhance the search of similar/relevant components (e.g. instances of `Gateway` could be `network gateway`, `data bus interface`, `ECU gateway`, etc.). After generating and collecting additional data, we design a component data index that includs all relevant information, such as component names, descriptions, instance collections and system types, along with their textual embeddings (for name and description) generated by an embedding agent. Given a document, the data index is used to search for relevant components.

3.2 Automotive Document Labeling

Given an automotive document, we first extract important keywords, map these keywords to standardized components, and then rank the these components based on the relevance to the input data.

Keyword Extraction. The goal of this step is to extract the most relevant keywords that appear in the technical documents. Our system takes a pair of files as input: the technical document (in HTML format) and its metadata (in an XML file). The metadata includes information such as title, car brand (e.g. *BMW, Audi*), document category (e.g. *Diagram, General information*), and directory path (e.g. *Installation* $\rightarrow$ *Engine* $\rightarrow$ *Injection System* $\rightarrow$

[2] https://www.bing.com

Switch on Clutch Pedal). We extract the content and metadata information to provide as input to a LLM for keyword extraction:

$$T = \text{Extract}(d, m), \\ K = \text{LLM}\big(\text{Prompt}(T)\big), \quad (1)$$

where d and m represent the technical document (.html) and its associated metadata (.xml). K represents the single most relevant keyword to the context of document d. The Prompt is designed in JSON format (Xia et al., 2024) with the role playing template (Wang et al., 2024). Based on the extracted keyword K, we use an LLM to generate a descriptive text Desc_K. This information will be used for searching similar components from the database.

Hybrid Search. Given the most relevant extracted keyword and its description, we apply a hybrid search technique combining keyword search and semantic search over the component embedding and keyword description embedding.

$$\text{Score}(K, c_i) = \gamma \cdot \text{Sim}_{kw}\left(K, c_i\right) \\ + \delta \cdot \text{Sim}_{sem}\left(E\left(\text{Desc}_K\right), E\left(\text{Desc}_{c_i}\right)\right) \quad (2)$$

$$C_{\text{topK}} = \text{Top-k}\left\{c_i \in C \text{ by Score}(K, c_i)\right\}, \quad (3)$$

where Sim_{kw} denotes the keyword-based similarity function, Sim_{sem} denotes the semantic-based similarity function, γ and δ are weighting coefficients for each search mode, and $E(\cdot)$ is a general embedding generator for descriptions. The Top-k function is retrieving the k components with the highest scores.

LLM-based Re-ranker. The hybrid search returns C_{topK} candidate components. However, due to a lack of context or difference in terminology, the correct component might not be at the highest rank. To address this, we employ a verification step, an LLM-based re-ranker, which re-ranks these candidates based on the document content. The LLM-reranker assigns a relevance score to each component candidate by utilizing expert-designed criteria reflected in the prompt. For example, *"is the document really useful to fix the component?"* or *"mentioning the component in the document does not reflect the meaning of usefulness in terms of mechanical engineering tasks"*. The final ranking of the most relevant components c^* is thus defined as:

$$c^* = \text{LLM\_ReRanker}\left(C_{\text{topK}}, \text{Prompt}(T)\right). \quad (4)$$

Table 1: The statistic of evaluation dataset.

Information	Value
Number of documents	110
N.o components appearing in title	35
N.o components appearing in metadata	45
N.o document categories	25
N.o automotive brands	3

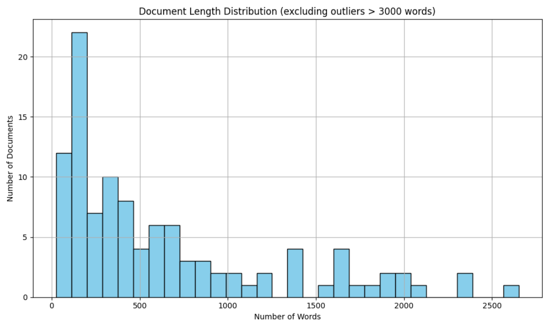

Figure 3: Length distribution of our evaluation dataset.

Large Language Model Backbone. The proposed system leverages both: closed-source LLMs such as GPT-4o, GPT-4.1 (Hurst et al., 2024) and open-source LLMs such as Llama (Grattafiori et al., 2024) or DeepSeek (DeepSeek-AI et al., 2025) as backbone components for various stages. These models provide the semantic understanding necessary for enhancing retrieval accuracy and overall system performance. They are considered based on task complexity, model availability, and computational cost, ensuring that the system remains scalable, adaptable, and efficient across both closed and open deployment environments.

4 Experiments

To evaluate our method, we conducted experiments on a dataset of 110 technical documents, where the most relevant component for each was collaboratively annotated by three mechanical experts. The whole set of available labels comprised ~7K component names. Additional statistics about the dataset can be found in the Table 1 and Figure 3.

In this work, we adopt **Hit@K** as the primary evaluation metric, which quantifies the proportion of documents where the correct component is retrieved within the top K predictions. Let D be the set of evaluation documents. For each document $d \in D$, let $\hat{y}_d^{(K)} = [c_1, c_2, \ldots, c_K]$ denote the top K predicted components, and let y_d denote the ground-truth most relevant component.

$$\text{Hit@K} = \frac{1}{|D|} \sum_{d \in D} \mathbb{1}\left[\exists c_i \in \hat{y}_d^{(K)} : c_i = y_d\right], \quad (5)$$

Table 2: Performance comparison of different approaches and models (best results in bold). Extract-Only retrieves only a single component. For traditional IR-based baseline, we achieved a Hit@1 of 19.09%, Hit@2 of 27.27%, Hit@3 of 31.82%, and Hit@5 of 40.91%.

Approach	Llama-3-70B			DeepSeek-V3			GPT 4.1			GPT 4o		
	Hit@1	Hit@3	Hit@5	Hit@1	Hit@3	Hit@5	Hit@1	Hit@3	Hit@5	Hit@1	Hit@3	Hit@5
Extract-Only	30.30	-	-	37.27	-	-	**47.27**	-	-	22.72	-	-
Extract-Reflect	31.81	37.27	37.27	40.00	45.54	52.72	46.36	**68.18**	**70.81**	25.45	37.27	40.00
Extract-Reflect-Sort	35.27	37.45	37.27	38.18	50.90	54.54	45.45	65.45	66.36	43.63	64.18	65.09
Retrieve-then-Extract	40.32	42.68	47.45	42.86	**52.15**	58.96	**47.27**	64.90	66.36	46.36	**65.45**	66.52
Our approach	**42.14**	**48.36**	**52.96**	**44.25**	51.48	**60.64**	**47.27**	64.55	66.36	**48.36**	63.63	**69.27**

where $\mathbb{1}[\cdot]$ is the indicator function.

4.1 Implementation Details

We access the LLMs via the provided API, using consistent parameters to ensure reproducibility. The temperature was set to 0 to eliminate randomness, and the maximum completion length was limited to 2500 tokens. A fixed seed value of 42 was used to maintain deterministic behavior across runs. Each configuration was executed three times, and the results were averaged to reduce potential variance. The values of γ and δ in the hybrid search component are both set to 0.5. For score aggregation, we employ the Reciprocal Rank Fusion (RRF) algorithm in the Azure Search configuration.

4.2 Baselines

We compare the performance of our method against several baselines:

- **Standard IR-based baseline**: A traditional IR-based method that encodes the document title, content summary, and metadata. We employ a hybrid search strategy that combines BM25 with dense embeddings to retrieve results from the component database.
- **Extract-Only**: a straightforward approach where the complete set of components together with the respective document data is given as input to an LLM and prompt engineering is used to task the LLM to predict the single most relevant component.
- **Extract-Reflect**: this approach supplements the previous approach with a reflection step that provides the initially extracted components together with the document data to an LLM to potentially correct the initial output and to expand it to Top-K components.
- **Extract-Reflect-Sort**: in this approach, in addition to the previous reflection step, an LLM is tasked to re-rank the Top-K candidates based on the content of the document.
- **Retrieve-then-Extract**: before applying the Extract-Reflect-Sort approach, hybrid search is leveraged to reduce the complete list to Top-K components using the document title and content. We report the best performance among $K \in \{30, 50, 100, 200\}$.

4.3 Results and Discussion

Table 2 shows the performance of our approach compared to the baselines using different language models. We observe that our approach achieves the highest Hit@1 values independently of the LLM model, specifically with the highest Hit@1 value of 48.36% using GPT 4o. This demonstrates the robustness and effectiveness of our method across different LLMs. Among four language models used in the experiments, the GPT-family models consistently achieve higher performance than the two open-source models, DeepSeek-V3 and Llama-3-70B. Furthermore, we observe that adding reflection and sorting steps as well as reducing the initial set of components via search beforehand can already improve the performance significantly, compared to prompting an LLM only once with the complete set of components. However, the baseline approaches that include the complete component list in a single prompt depend on this set not being prohibitively large, while our approach does not have this limitation. In this regard, the Retrieve-then-Extract approach can be viewed as an intermediary between the purely prompt-based approaches and our approach based on hybrid search.

4.4 Analysis of Error Types

To better understand the strengths and weaknesses of our approach, we conducted a detailed error analysis on the results based on feedback from domain experts. We then categorized the false top-1 predictions into three different error types (as

Table 3: Experimental results of varying input information in our system.

Input Information	GPT 4.1			GPT 4o		
	Hit@1	**Hit@3**	**Hit@5**	**Hit@1**	**Hit@3**	**Hit@5**
Only Title	37.27	50.00	52.73	34.55	49.09	54.55
Only Content	43.64	55.45	59.08	47.27	58.18	62.73
Content + Metadata	46.36	60.91	65.45	47.27	59.08	65.45
Title + Content + Metadata	47.27	64.55	66.36	48.36	63.63	69.27
Summary + Metadata	47.27	64.55	66.36	50.00	65.45	70.00
Title + Summary + Metadata	**48.73**	**66.36**	**66.91**	**52.82**	**67.27**	**70.91**

Table 4: Comparison inference costs across GPT models, including content summarization.

GPT version	Hit@K			Execution Cost		Total Cost
	Hit@1	**Hit@3**	**Hit@5**	**Input**	**Output**	
GPT 4.1-mini	45.18	62.73	65.17	$\sim$ 0.32$	$\sim$ 0.20$	$\sim$ 0.52$
GPT 4.1	48.73	66.36	66.91	$\sim$ 1.61$	$\sim$ 0.99$	$\sim$ 2.60$
GPT 4o-mini	42.73	63.64	65.45	$\sim$ 0.12$	$\sim$ 0.07$	$\sim$ 0.19$
GPT 4o	52.82	67.27	70.91	$\sim$ 2.01$	$\sim$ 1.25$	$\sim$ 3.26$

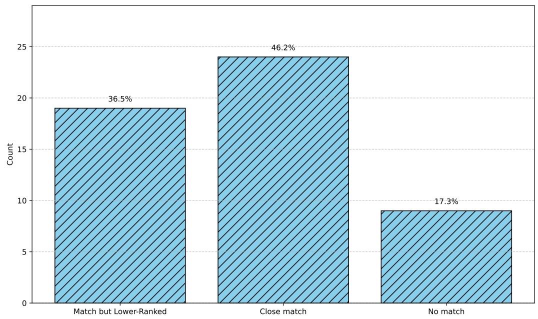

Figure 4: Error Category Distribution.

shown in Figure 4).

Match but Lower-Ranked where the top-1 prediction does not precisely match the ground truth due to a difference in semantic scope, but we predict the correct component within a lower rank. This error type accounts for 36.5 % of our false top-1 predictions. We observed that the top-1 predictions often identified component groups, whereas the ground truth referred to semantically related terms, such as sub-parts or super-types, of those predictions. For instance, if the prediction was `Cooling system` and the ground truth was `Coolant circuit`, the experts commented that these are quite similar, as `Coolant circuit` is a sub-part of the `Cooling system`. Moreover, we found that some prediction components (e.g. `Transmission`) are more generic than the ground truth (e.g. `Manual transmission`).

Close Match which holds the highest proportion among the three identified error types. This error is counted when the predicted component is very similar, or a slightly more generic term that's still considered acceptable by the experts. This error type accounted for 46.2 % of our false Top-1 predictions. This can be explained by the fact that the definition of components embedded within the knowledge base of the language model differs from definitions in knowledge of domain experts, particularly it also depends on the car brands and manufacturers. The availability of expert-defined automotive component definitions could lead to improved performance in this specific situation.

No Match predicted component neither matches nor is similar to the ground truth. We found that most incorrect predictions in this category related to general components like `Assembly frame` or `Connector`, whereas our system tended to predict more specific components. However, the error rate for this category is not high, accounting for only about 17.3% among the three identified error categories.

5 Ablation Study

We conducted a series of ablation studies to evaluate how different input configurations and processing strategies influence our system. First, we analyze the impact of varying the amount of document data provided to our system. Subsequently, we compare the the effect of LLM size on the performance and inference cost of two GPT models.

Effect of Document Input. To understand the impact of context information used for prediction, we conducted an ablation study varying the nature of the input. As a default, we used all available data (document title, content and metadata) for our main experiment in Sec. 4. Here, we present results based on using only subsets of this data as well as only using a summary of the document content. As can be seen in Table 3, using more document data usually improves the overall performance for both GPT-models. However, when the document content is already summarized in a pre-processing step, this can enhance the performance even further, especially in the case of GPT-4o. This is likely due to the fact that the most relevant component is usually the core topic of the document such that summarization can help to reduce the amount of redundant information.

Effect of LLM Size on Inference Costs. To investigate the trade-offs between the performance and computation costs, we evaluated our system on LLMs of different size. Table 4 shows the performance as well as input and output costs[3] and approximate total costs of four LLMs. It can be observed that GPT-4o outperforms other models in Hit@1 score but with 1.5x higher costs than GPT-4.1 and ∼17x higher costs than the mini version. On average, labeling one document costs ∼$0.029 using our approach with the GPT-4o model.

Effect of Expert Assessment on Ontology Ambiguities. Beyond the initial 110 documents in the test set mentioned above, we extended the evaluation in close collaboration with domain experts to a larger set of 348 additional documents from the automotive domain. In this extended evaluation, our labels were manually verified by experts rather than being compared to a fixed pre-annotated ground truth. This setup allowed experts to account for **ontological ambiguity** during the validation process, which likely contributed to improved performance. The evaluation results on the 348 documents are summarized in Table 5.

These results indicate that our approach performs reasonably well when judged by domain experts, and the consistency across different **Hit@K** levels demonstrates its robustness. Overall, the human expert evaluation confirms both the practicality and the promise of our method in real-world applications.

[3] https://azure.microsoft.com/en-us/pricing/details/cognitive-services/openai-service/

Table 5: Evaluation results on labeled documents by domain experts.

Metric	Count	Percentage
Total documents	348	—
Hit@1	206	59.20%
Hit@2	215	61.78%
Hit@3	218	62.64%
Hit@4	222	63.79%
Hit@5	222	63.79%

6 Conclusion

We present a system to address a practical document labeling problem in the automotive domain. Our approach leverages LLMs to enrich the label targets (i.e. components) and ensures unified terminology definitions across car manufacturers. By performing hybrid search with LLM-based reranking, the method significantly improves the accuracy of labeling the most relevant components for automotive documents.

7 Limitations

Although achieving promising results, the limitations of the current method come from the constraints on LLM model availability, data availability and scalability. Due to data security protocols, our investigation is restricted to commercially available GPT-family models and only two open-source models in our cloud service. The evaluation dataset is also relatively small, primarily due to the necessity of specialized automotive domain expertise to construct accurate ground-truth labels. To scale the current pipeline, e.g. on millions of documents, it is essential to reduce the number of requests to LLMs. Prioritizing on the labeling accuracy to generate a high-quality training dataset that can be used to train an open-source LLM is our current strategy.

8 Ethics statement

We acknowledge and respect the importance of intellectual property rights while utilizing search engines such as Bing. We are committed to ensuring that all content accessed and referenced adheres to copyright laws and ethical standards. Furthermore, we want to assure our users that we do not store any personal data during our search activities. Our commitment to data security means that we prioritize the privacy and confidentiality of all users.

References

Marwah Alaofi, Paul Thomas, Falk Scholer, and Mark Sanderson. 2024. Llms can be fooled into labelling a document as relevant. In *Proceedings of the ACM SIGIR Asia-Pacific*.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, and 1 others. 2024. A survey on evaluation of large language models. *ACM TIST*, 15(3):1–45.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, and 1 others. 2025. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Byungha Kang and Youhyun Shin. 2025. Empirical study of zero-shot keyphrase extraction with large language models. In *Proceedings of the COLING*, pages 3670–3686.

Roberto Martínez-Cruz, Alvaro J. López-López, and José Portela. 2024. Chatgpt vs state-of-the-art models: a benchmarking study in keyphrase generation task. *Applied Intelligence*, 55(1):50.

Robert Bosch GmbH. 2022. *Automotive handbook*. John Wiley & Sons.

Xiaofei Sun, Xiaoya Li, Jiwei Li, Fei Wu, Shangwei Guo, Tianwei Zhang, and Guoyin Wang. 2023. Text classification via large language models. In *Findings of the EMNLP*, pages 8990–9005.

Paul Thomas, Seth Spielman, Nick Craswell, and Bhaskar Mitra. 2024. Large language models can accurately predict searcher preferences. In *Proceedings of ACM SIGIR*, pages 1930–1940.

Zhenhailong Wang, Shaoguang Mao, Wenshan Wu, Tao Ge, Furu Wei, and Heng Ji. 2024. Unleashing the emergent cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration. In *Proceedings of the NAACL*, pages 257–279.

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, Yong Jiang, and Wenjuan Han. 2024. Chatie: Zero-shot information extraction via chatting with chatgpt. *Preprint*, arXiv:2302.10205.

Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Lidia Sam Chao, and Derek Fai Wong. 2025. A survey on llm-generated text detection: Necessity, methods, and future directions. *Computational Linguistics*, pages 1–66.

Congying Xia, Chen Xing, Jiangshu Du, Xinyi Yang, Yihao Feng, Ran Xu, Wenpeng Yin, and Caiming Xiong. 2024. FOFO: A benchmark to evaluate LLMs' format-following capability. In *Proceedings of the 62nd ACL*, pages 680–699.

Tingyu Xie, Qi Li, Jian Zhang, Yan Zhang, Zuozhu Liu, and Hongwei Wang. 2023. Empirical study of zero-shot NER with ChatGPT. In *Proceedings of the EMNLP*, pages 7935–7956.

Philipp Zellmer, Lennart Holsten, Jacob Krüger, and Thomas Leich. 2024. The terminology of automotive product-structuring concepts: A systematic mapping study. *IEEE Transactions on Engineering Management*, 71:14974–14990.

Zikang Zhang, Wangjie You, Tianci Wu, Xinrui Wang, Juntao Li, and Min Zhang. 2025. A survey of generative information extraction. In *Proceedings of the COLING*, pages 4840–4870.

Zibin Zheng, Kaiwen Ning, Qingyuan Zhong, Jiachi Chen, Wenqing Chen, Lianghong Guo, Weicheng Wang, and Yanlin Wang. 2025. Towards an understanding of large language models in software engineering tasks. *Empirical Software Engineering*, 30(2):50.

Shengliang Zhu. 2025. Enhancing competitive advantage through ai and digital technology: A case study of jiangling motors group. *Asia Pacific Economic and Management Review*, 2(1).