Domain-adapted pre-trained language models for implicit information extraction in crash narratives

Transportation Research Part C 192 (2026) 105863

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Domain-adapted pre-trained language models for implicit information extraction in crash narratives

Xixi Wang [a,*], Jordanka Kovaceva [b], Miguel Costa [a], Shuai Wang [c], Francisco Camara Pereira [a], Robert Thomson [b]

[a] Department of Technology, Management and Economics, Technical University of Denmark, Akademivej 2800, Kongens Lyngby, Denmark
[b] Department of Mechanical Engineering, Chalmers University of Technology, Chalmersgatan 4, Gothenburg, 412 96, Sweden
[c] Department of Computer Science and Engineering, Chalmers University of Technology, Chalmersgatan 4, Gothenburg, 412 96, Sweden

ARTICLE INFO

Keywords:
Traffic safety
Crash narratives
Pre-trained language models
Fine-tuning
Information extraction

ABSTRACT

Traffic safety agencies worldwide collect detailed crash narratives to understand the circumstances and contributing factors of road crashes. This information is critical for informing policy decisions and designing safer infrastructure. Unlike structured crash data fields (e.g., injury severity, vehicle type, weather), these free-text narratives capture sequences of events, driver behaviors, and environmental context that are hard to represent as structured codes. However, this also renders them difficult to analyze at scale. Manual coding by trained analysts is time-consuming and has limited throughput, creating a bottleneck in the processing of crash reports. Recent advances in pretrained language models (PLMs) and large language models (LLMs) offer new opportunities for extracting information from such narratives. However, two challenges limit their practical application: 1) limited performance on tasks needing both explicit and implicit reasoning that require domain knowledge; and 2) privacy and data governance concerns when processing sensitive crash reports through closed commercial APIs. To address these challenges, we investigate whether open-source PLMs can achieve expert-level performance in information retrieval from crash narratives through fine-tuning with Low-Rank Adaptation (LoRA). We evaluate our approach on two tasks using the CISS dataset: 1) identifying the *Manner of Collision*, and 2) extracting *Crash Type* for involved vehicles. Our method achieves 96.1% accuracy on *Manner of Collision* (+4.3% over the strongest baseline) and exceeds the strongest baseline by 16.9–28.7% on the more complex *Crash Type* task. Beyond these improvements, we provide a practical, privacy-preserving pipeline deployable locally without external API dependence, and demonstrate that our approach can actively assist human annotators by identifying potential labeling inconsistencies, offering a dual benefit of automation and quality assurance for crash database curation. Our code and data are publicly available[1].

1. Introduction

Traffic crashes remain one of the leading causes of death worldwide, with an estimated 1.19 million fatalities annually (World Health Organization, 2023). Improving road safety is therefore a critical global challenge that requires analyzing high-quality real-world data from crash databases (Imprialou and Quddus, 2019; Xie et al., 2019). These databases typically contain

* Corresponding author.
E-mail address: s232253@dtu.dk (X. Wang).

https://doi.org/10.1016/j.trc.2026.105863
Received 9 October 2025; Received in revised form 22 April 2026; Accepted 5 July 2026
Available online 10 July 2026

information recorded by crash investigators as part of official reports, including crash circumstances, involved parties, severity outcomes (injuries or fatalities), and crash narratives. Among these, crash narratives provide rich, contextual descriptions of the actual crash event, including vehicle travel directions, impact points, road conditions, and other important contextual factors (Jaradat et al., 2024a).

Despite their value, crash narratives remain severely underutilized in large-scale safety analysis. Written in unstructured text with highly diverse styles, inconsistent terminology, and varying levels of detail, these narratives cannot be readily processed using conventional statistical techniques or modeling approaches which rely on structured information. To extract structured information (e.g., collision manner, crash type, or contributing factors) investigators must manually code these narratives, a process that is time-consuming, expensive, and requires expert knowledge. While some databases like the U.S. Crash Investigation Sampling System (CISS) (Administration, 2023) invest heavily in expert manual coding to produce high-quality structured labels, other country or local agencies lack the resources needed for such detailed coding. In addition, when new classification schemes are introduced, retrospective analysis requires costly re-coding and for real-time applications, such as for emergency response, it is unfeasible to manual process crash narratives and descriptions.

To address these challenges, researchers have explored natural language processing (NLP) to automate information extraction from crash narratives. Early approaches relied on topic modeling (e.g., Latent Dirichlet Allocation (Blei et al., 2003)), keyword weighting via Term Frequency-Inverse Document Frequency (TF-IDF) (Christian et al., 2016), or bag-of-words classifiers (Zhang et al., 2010). While useful for coarse crash classification, these methods treat words independently and cannot capture sentence structure or semantic nuances. Word embeddings and recurrent neural networks (RNNs) improved semantic representation and sequence modeling, but they still only provide shallow explicit information extraction capabilities and remain inadequate for complex reasoning tasks. For example, determining a *Crash Type* for each vehicle in a multi-vehicle crash requires in-depth narrative interpretation: narratives often combine descriptions of multiple vehicles with causal relationships, and a model must identify the target vehicle, understand its spatial relation to others, and classify its crash type without being distracted by irrelevant textual information.

Transformer-based pre-trained language models (PLMs), such as BERT (Devlin et al., 2019) and Large Language Models (LLMs) (Vaswani et al., 2017), have been proposed to tackle such challenges in NLP. These deeper, more complex models can enable more complex semantic understanding by learning richer semantic representations over massive corpora and exploiting different techniques like prompt engineering, chain-of-thought (revision) (Wei et al., 2022), and few-shot learning. However, applying PLMs to crash narrative analysis faces several practical challenges. First, using commercial API-based models (e.g., GPT-4) introduces significant data security and privacy risks, as transport agencies and related organizations often strictly prohibit uploading sensitive crash data to external API services. Second, large-scale models require substantial computational resources, making both training and inference costly for road safety agencies with limited budgets. Third, PLMs trained on general corpora lack domain-specific knowledge about traffic safety terminology and crash mechanics (Zheng et al., 2023), limiting their out-of-the-box performance and making it prone to hallucinations in domain-specific settings (Dahl et al., 2024; Dokas, 2026).

In this work, we address these challenges by developing a practical framework for automated crash narrative information extraction using fine-tuned open-source PLMs. Our key idea is to reformulate information extraction as a structured classification task. We achieve this in two-fold: 1) fine-tune PLMs to focus on safety-critical cues in narratives for a relatively simple classification task with only seven classes; and 2) use an instruction-following LLM to inject oracle knowledge directly into the prompt, ensuring predictions are consistent with domain knowledge and information coding rules. To reduce data and computational requirements, we apply Low-Rank Adaptation (LoRA) (Hu et al., 2022) for parameter-efficient fine-tuning of open-source LLMs.

Our work makes the following contributions that address both methodological gaps and practical deployment needs:

- We demonstrate that fine-tuned PLMs can replicate detailed manual coding performed by CISS investigators, achieving high accuracy on both *Manner of Collision* (a per-crash task) and the more challenging per-vehicle *Crash Type* extraction. Critically, while we validate our approach using CISS's existing labels as ground truth, the value lies in being able to automate such process: fine-tuned models can process narratives in seconds rather than hours, enabling agencies without CISS-level resources to perform detailed coding on their own crash databases, and allowing retrospective analysis of historical data or rapid prototyping of new classification schemes without the need for large-scale manual re-annotation;
- We enable constraint-aware, fine-grained classification of complex information extraction by instruction-tuning LLMs to explicitly respect label-space restrictions imposed by upstream category and configuration choices, ensuring that the information extracted maintains semantic coherence while disentangling vital information from irrelevant ones;
- We conduct extensive experiments on the CISS dataset, comparing seven LLM backbones, BERT, GPT models, and traditional NLP baselines across multiple dimensions: accuracy, data efficiency (performance with limited training samples), computational cost, and robustness. Our results and analysis enable agencies to select appropriate models based on their resource constraints and accuracy requirements, making deployment more transparent and accessible.

Overall, we showcase that our approach enables a substantially faster and more scalable alternative to crash narrative information extraction. In practice, this paves a new way for reducing manual workload by augmenting human investigators while also offering resource-constrained local road safety agencies an alternative for performing high-quality crash data coding.

2. Related work

2.1. Traditional text-analysis methods

Research on crash narrative analysis typically follows a two-stage pipeline: first extracting text features (e.g., word frequencies, topics, or embeddings) and then applying classification models. Early work relied on frequency-based representations most notably Bag-of-words (BoW) (Zhang et al., 2010), which treats text as an unordered collection of words and encodes it as simple word counts or Term Frequency-Inverse Document Frequency (TF-IDF) values. These representations are typically paired with traditional classifiers such as Naive Bayes (Aborisade and Anwar, 2018), Logistic Regression (Aborisade and Anwar, 2018; Akuma et al., 2022), and Support Vector Machines (SVMs) (Cichosz, 2018). Such models have been used, for example, to identify agricultural crashes (Kim et al., 2021) and secondary crashes (Zhang et al., 2020). However, BoW and TF-IDF reflect only surface-level word frequency statistics and fail to capture the latent semantic structure of documents. To incorporate semantic themes, researchers adopted topic modeling. Methods such as Latent Dirichlet Allocation (LDA) learn latent topics from text and represent each narrative as a probability distribution over these topics. Topic modeling has been applied to summarize themes in travel surveys (Baburajan et al., 2018, 2020), study motorcycle crash causation (Das et al., 2021), analyze relationships between latent topics and crash severity (Li et al., 2024), and examine safety concerns associated with transitions of control in autonomous vehicle crash narratives (Alambeigi et al., 2020). Yet, topic models remain coarse-grained, providing only distributions over topic words and lacking detailed semantic understanding. With the rise of word embeddings, text representations have advanced beyond sparse counts. Techniques such as Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014) project words into continuous low-dimensional spaces, capturing both semantic and syntactic similarities. When integrated with downstream classifiers, these embeddings have been used for traffic text classification to detect emerging risks (Kim et al., 2020) or for transportation sentiment analysis (Ali et al., 2019). To further improve performance, neural networks such as CNNs (Qiao et al., 2022) and RNNs (Heidarysafa et al., 2018) have been employed as advanced downstream classifiers following word embeddings. For example, these approaches have been applied in railway accident cause analysis (Heidarysafa et al., 2018), road crash causation analysis (Xiong and Zhou, 2024), and investigation of injury outcomes of horse-and-buggy crashes in rural areas (Qawasmeh et al., 2024).

The above methods decouple text representation from classification, performing feature extraction and task learning independently, and preventing gradients from back-propagating through the entire model during training. As a result, the text representations cannot be optimized for the specific task, which limits the model's overall performance. Recent research has shifted toward unified models that integrate text representation and downstream task learning in an end-to-end manner. For instance, some studies learn richer semantic features for identifying accident causes (Zhang, 2022) or performing traffic classification (Ren et al., 2021). However, RNNs still suffer from gradient vanishing and gradient exploding problems, which restrict layer stacking beyond a few layers and limit their ability to capture complex semantics.

2.2. PLM-based text-analysis methods

With the rise of Transformer architectures, model stacking is no longer a limitation, and model parameters can and have scaled to billions. This scaling has significantly enhanced model capacity, enabling large-scale pretraining on massive corpora to capture contextual semantic representations. For example, BERT has been applied to identify actual wrong-way driving (WWD) crashes (Hosseini et al., 2023), classify traffic injury types into five categories (Oliaee et al., 2023), and extract impact points and pre-collision vehicle maneuvers (Seo et al., 2023). However, BERT is pre-trained using masked language modeling and is designed for classification instead of prompt-driven generation, so it cannot directly interpret and respond to prompts, limiting its applicability in crash narrative analysis.

In contrast, prompt-based LLMs can naturally incorporate task instructions into the input, making them more flexible for classification and reasoning over unstructured crash narratives. Mumtarin et al. (2023) evaluated LLMs in answering safety question tasks from accident narratives by embedding questions directly into prompts. Arteaga and Park (2025) employed prompt engineering to identify unreported alcohol involvement in crash reports. Zhen et al. (2024) further leveraged CoT (Wei et al., 2022) reasoning and prompt engineering with LLMs to enhance traffic crash severity analysis and inference. To address the limited adaptability of PLMs caused by insufficient domain-specific knowledge in their training corpora, fine-tuning Transformer-based models provides an effective solution. For example, Jaradat et al. (2024b) proposed a multi-task learning (MTL) LLM framework, and Golshan Khavas and Nosrati (2025) fine-tuned the LLaMA 3.1 model to extract crash locations and casualty counts, with both approaches achieving high accuracy.

Overall, these studies focus on simple tasks with few categories or information explicitly stated in the narratives, leaving their effectiveness on more complex problems unknown.

3. Methodology

PLMs are effective for text classification, converting unstructured crash narratives into structured fields (Harne et al., 2025). While recent traffic-safety studies can extract explicitly stated details (Du et al., 2023), reasoning-intensive variables remain challenging due to limited crash-specific knowledge and the prohibitive cost of full-model training. We address this by fine-tuning BERT and LLMs with LoRA (Hu et al., 2022) using an annotated crash dataset, injecting domain knowledge at low compute and memory cost and enabling safer, more controllable deployment than closed LLMs accesed via public APIs. Our work targets two high-value variables

Table 1
Class definitions for the MANCOLL task.

Label ID	MANCOLL	Percentage(%)
0	Not Collision with Vehicle in Transport	42.7
1	Rear-End	16.5
2	Head-On	3.6
4	Angle	30.0
5	Sideswipe, Same Direction	3.8
6	Sideswipe, Opposite Direction	1.5
9	Unknown	1.6

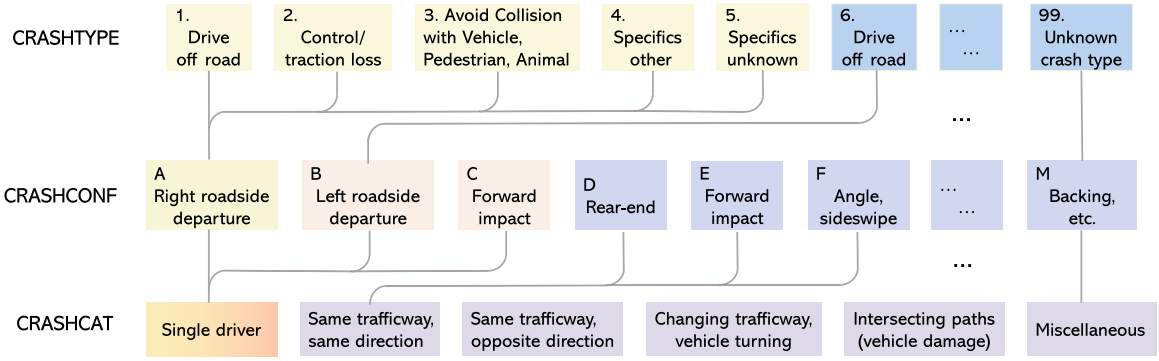


Fig. 1. Hierarchical mapping of CRASHTYPE in a two-step classification procedure.

that support crash risk assessment, and countermeasure design: (i) *Manner of Collision*, a reasoning task over event sequences and interacting agents; and (ii) *Crash Type*, a fine-grained classification with 98 possible classes. We begin by describing the data used in our approach, defining the problem more clearly for both tasks, and presenting our fine-tuning framework for extracting information from crash narratives.

3.1. Dataset: CISS

This paper applies the proposed framework to the Crash Investigation Sampling System (CISS) dataset released by the U.S. National Highway Traffic Safety Administration (NHTSA) (Administration, 2023) as a case study. CISS is a national crash database covering police-reported motor vehicle crashes in the United States, involving passenger cars, light trucks, and vans (Radja et al., 2022). It contains records from 2017 to 2023, with approximately 3700 cases per year. The database contains about 39 relational tables[1], each describing a specific aspect of the crash, such as crash level data (table CRASH), events level record (table EVENT), general vehicle record (table GV). CISS includes not only crash narratives SUMMARY (a basic description of the crash scenario as documented by the crash investigator), but also structured case coding such as MANCOLL (manner of collision) and CRASHTYPE (crash type), which are basically manually determined (Administration et al., 2025).

In the CISS dataset, MANCOLL is not directly annotated but rather derived through rule-based mapping from other manually labeled variables, including OBJECT CONTACTED in table EVENT as well as CRASHTYPE and TRANSPORT in table GV. The resulting MANCOLL consists of seven categories as shown in Table 1, where the proportion of each category is also reported, with one category corresponding to *Unknown*. We observe that the *Unknown* category contains two types of cases: those that could not be properly classified due to technical limitations, and those that do not belong to any of the predefined categories.

The CRASHTYPE is a numeric value derived through a multi-step process: first by selecting the crash category (CRASHCAT) and crash configuration (CRASHCONF) in the GV table (as illustrated in Fig. 1), and then by the crash investigator's assessment based on police reports, scene inspections, vehicle inspections, and interviews. This procedure is preferred because it provides a more structured and interpretable way to visualize crash scenarios.

3.2. Problem definition

This study focuses on the extraction of implicit information from crash narratives. Explicit information refers to details directly stated in the narrative and can be extracted with limited interpretation, such as travel direction, or impact point. By contrast, implicit information is not explicitly expressed and must be inferred by integrating dispersed textual cues. Its identification is more challenging because it requires deeper semantic understanding of entity interactions, temporal structure, and domain-specific crash knowledge,

[1] The number and columns of tables in CISS continuously changes as the data-collection process is updated. For instance, the dataset of year 2017 did not include the VPICDECODE table.

rather than surface-level extraction. Appendix A illustrates this implicit–explicit distinction through a worked example drawn from a real crash narrative.

This paper uses the CISS dataset as a case study to evaluate PLMs on extracting implicit information from unstructured text. We focus on analyzing the crash narratives (the SUMMARY field in the CRASH table) to infer structured collision-type information. Specifically, our goal is (1) to identify the *Manner of Collision* for each crash and (2) the *Crash Type* for each vehicle involved in a crash.

The first task concerns classifying *Manner of Collision* based solely on the free-text crash narrative (i.e., SUMMARY field in the CRASH table), without using any structured metadata. We use the MANCOLL field in the CRASH table as the ground truth labels for accuracy evaluation.

The second task involves extracting and classifying *Crash Type* for each vehicle involved in a crash. Unlike MANCOLL, which involves only seven classes, CRASHTYPE is considerably more challenging, encompassing 97 fine-grained categories. To make this problem more tractable, we exploit the hierarchical mapping illustrated in Fig. 1 and decompose the task into 13 smaller classification subtasks based on CRASHCONF. All subtasks are addressed within a single model. Since CRASHCAT and CRASHCONF classifications are relatively straightforward[2], we treat CRASHCONF as oracle knowledge and concentrate our analysis on the more difficult CRASHTYPE classification.

3.3. LLM fine-tuning workflow

As mentioned before, we formulate the information extraction task as a classification problem by prompting the LLM to output a predefined label rather than free-form text and focus on per-crash *Manner of Collision* and per-vehicle *Crash Type* extraction tasks. We now outline the detailed processes of the two tasks, followed by the supervised fine-tuning process.

The information extraction and reasoning tasks involved in this paper are shown in Fig. 2. In Fig. 2a, MANCOLL directly determines the overall collision mode from the entire crash narrative. The output space is relatively small (7 classes) and can be completed using a fixed implicit CoT prompt. CRASHTYPE extraction for each vehicle, in Fig. 2b, requires first identifying the target vehicle (V1/V2/V3) within a text containing multiple vehicle information intertwined. Based on its crash configuration identifier (CRASHCONF), the corresponding subtask is selected from 13 different dictionaries, and the corresponding 98 *Crash Types* are output. In contrast, CRASHTYPE classification not only involves entity and reference resolution, but also requires discrimination within a heterogeneous and larger label space. The prompts must also be dynamically constructed rather than fixed. Therefore, it is significantly more difficult to understand than Task 1.

To adapt LLMs to traffic-safety tasks, we fine-tune open-source models following the pipeline in Fig. 3. During fine-tuning, we first perform training data preparation. Specifically, we generate prompts using our designed templates in Fig. 2 based on CISS summaries, and combine them with corresponding labels to form prompt–answer pairs. We then conduct parameter-efficient fine-tuning by inserting LoRA adapters into a pre-trained LLM and training only these adapters while freezing the base model. Finally, the trained adapters are merged with the base model to obtain the fine-tuned model for inference.

Prompt engineering

The prompt is crucial for LLM performance, as small changes in wording can greatly affect accuracy. For crash-narrative analysis in traffic domain, prompts must embed traffic safety knowledge to ensure reliable predictions. Accordingly, we design a structured implicit CoT prompt with clearly separated variable slots and a fixed instruction block (Fig. 2): (i) Crash narrative: SUMMARY(V1,V2,V3); (ii) Target vehicle: Vehicle ID (used for CRASHTYPE); (iii) Possible classes: fixed small set for MANCOLL, but dictionary-specific for CRASHTYPE; and (iv) a fixed prompt part, which steers the model to understand the task, reason step-by-step, and produce constrained outputs. The prompt contains mainly the following components:

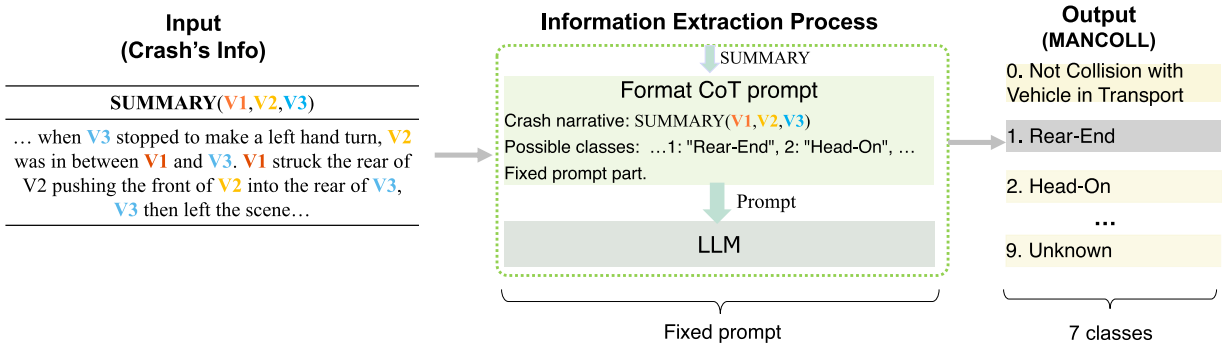
(1) Task Introduction. This part contains introduction of task (fixed prompt part) and expert-verified definitions for each category.

```
You are a helpful assistant that classifies vehicle collisions
into one of the following categories based on...
< Possible classes and their definitions >
```

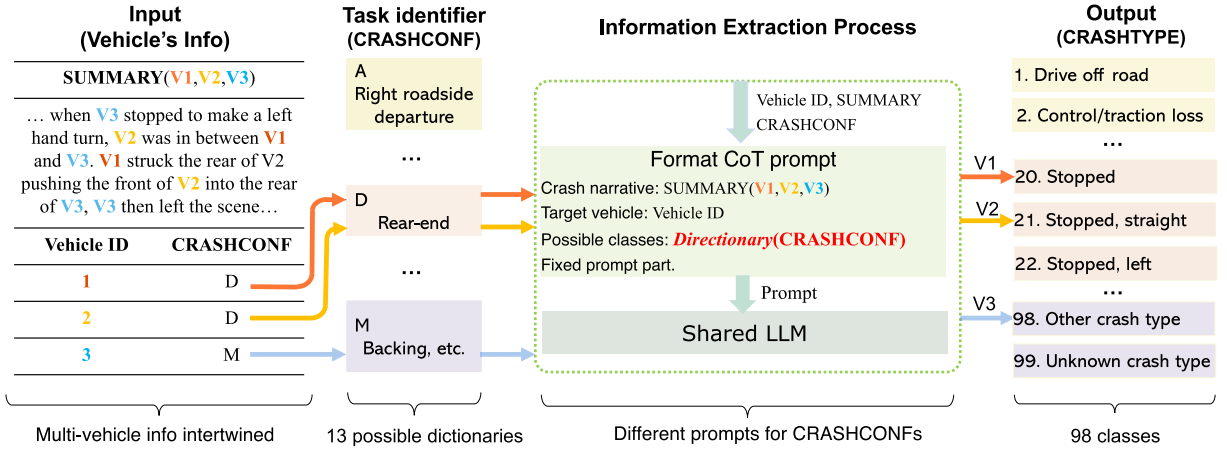
(2) Clarification Rules. These clarification rules are derived from domain-specific heuristics, codified through expert consultation and data observation. We first enhance model performance by applying an implicit CoT prompt strategy. Specifically, the prompts are designed to decompose complex reasoning tasks into step-by-step subproblems, thereby guiding the LLM to follow an explicit reasoning trajectory before arriving at the final categorical prediction. Other rules were distilled from empirical insights obtained via manual inspection and a pilot study over some training examples. For the MANCOLL task, this portion of the prompt is fixed, while for the CRASHTYPE task, it is dictionary-specific and thus varies across subtasks.

(3) Output Instruction. LLMs may produce varied outputs even when the classification result is correct, e.g., returning “4”, “angle”, or “angled side impact” for the same class. Such variations hinder batch processing. To avoid this, we use fixed prompt and explicitly instruct the model to output only a valid class index from the predefined label space.

[2] The CRASHCAT classification is very similar to MANCOLL, consisting of only six categories. After completing the first task, we found the two tasks highly overlapping and therefore did not pursue CRASHCAT further. The CRASHCONF categories are derived from CRASHCAT, with each CRASHCAT corresponding to only 1–3 CRASHCONF classes, making it a relatively simple classification task as well.



(a) Per-crash *Manner of Collision* extraction process.



(b) Per-vehicle *Crash Type* extraction process.

Fig. 2. Overview of the two information extraction tasks: (a) Manner of collision extracted from the crash narrative with a fixed implicit CoT prompt and a small candidate set (7 classes). (b) Per-vehicle *Crash Type* extracted from an intertwined multi-vehicle narrative, using 13 task-specific prompts to predict among 98 fine-grained classes.

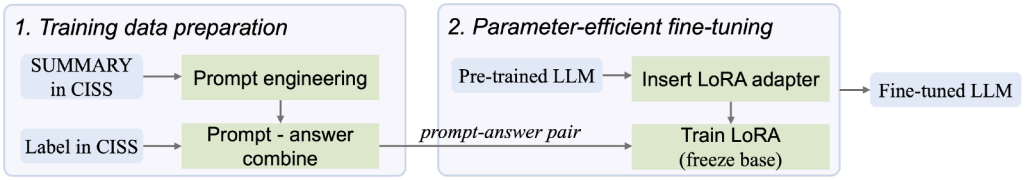


Fig. 3. Overview of the proposed LoRA-based fine-tuning framework.

```
Only respond with a single number from the list above.
Do not add any explanation.
```

All prompts used in this work are included in the Appendix B.

LoRA adapter

The foundation of modern LLMs is the Transformer architecture, whose core component is the self-attention mechanism (Vaswani et al., 2017). By scaling up this architecture to billions of parameters and training on massive text corpora, LLMs acquire broad linguistic and world knowledge. However, traffic-domain narratives (e.g., crash reports, incident logs, work-zone descriptions) are underrepresented in such corpora, so pre-trained LLMs struggle to capture domain-specific entities, relations, and causal patterns critical for crash narrative analysis. To bridge this gap efficiently, we adopt parameter-efficient fine-tuning by keeping the pretrained backbone frozen and learning lightweight adapters specialized for crash narratives analysis.

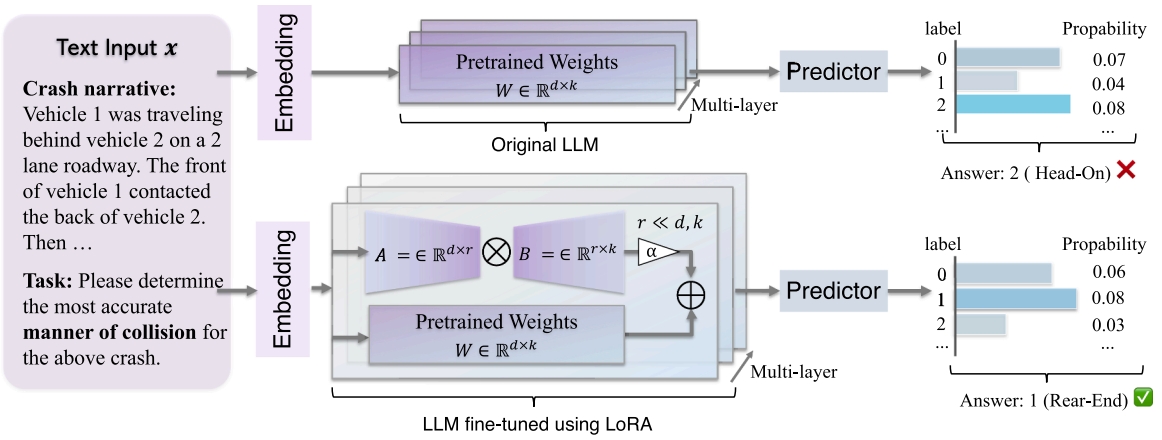


Fig. 4. Prediction processes of the original LLM (top) and the LoRA fine-tuned LLM (bottom).

Self-Attention Primer (Where Adaptation Can Act). Given a token sequence represented by hidden states $X \in \mathbb{R}^{T\times d}$, a single attention head computes

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V,$$

where Q, K, V denote the query, key, and value representations obtained through the projection matrices W_Q, W_K, and W_V, respectively. The attention output is

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{k}}\right)V. \tag{1}$$

This decomposition exposes three linear projections, W_Q, W_K, W_V, as natural adaptation points. The scaling factor $\frac{1}{\sqrt{k}}$, where k is the dimensionality of the key vectors, stabilizes attention scores by preventing overly large dot products. For traffic narratives, adjusting these projections helps the model aligns with crash-specific cues, and emphasize values relevant to causality.

LoRA for Parameter-Efficient Domain Adaptation. Low-Rank Adaptation (LoRA) (Hu et al., 2022) inserts trainable low-rank matrices into selected frozen projections so that only a small number of parameters are updated. A comparison between the original LLM and the LoRA fine-tuned LLM is illustrated in Fig. 4. For the same input, the original model fails to produce the correct answer, whereas the fine-tuned model adapts the LLM's original weights **W** by adding the product of low-rank matrices **A** and **B**, which alters the label prediction probabilities and enables the correct output.

Specifically, for a target weight $W \in \mathbb{R}^{d\times k}$ (e.g., one of W_Q, W_K or W_V), LoRA learns a low-rank update

$$\Delta W = \frac{\alpha}{r} AB, \qquad A \in \mathbb{R}^{d\times r},\ B \in \mathbb{R}^{r\times k},\ r \ll \min(d, k),$$

and uses the adapted weight

$$W' = W + \Delta W \tag{2}$$

During fine-tuning, the large pretrained W is *frozen* and only A, B are trained. This reduces trainable parameters from $d \times k$ to $(d + k)r$ and lowers memory cost, while preserving general linguistic competence. Crucially, because attention is factorized into Q, K, V, placing LoRA on W_Q/W_K directly modulates the *attention weights* in Eq. (1), enabling the model to learn traffic-specific matching patterns; placing LoRA on W_V refines *content aggregation and readout*, improving how incident attributes and causal chains are represented downstream. In our implementation, we apply LoRA to all the query, value projection layers, and key projection layers (W_Q, W_K and W_V) of the transformer blocks, which are empirically found to be effective for language modeling tasks (Mao et al., 2025).

Loss function. In our tasks, each crash or vehicle is assigned a label corresponding to the predefined classes. To simplify the output, we encode all class labels as single-token numeric identifiers using a dictionary mapping, such as in Table 1. Consequently, the model is trained to generate a single token representing the class ID for each input.

We adopt the standard cross-entropy loss for training because it directly measures the divergence between the predicted probability distribution over the vocabulary and the one-hot target distribution (Zhang and Sabuncu, 2018), thereby encouraging the model to assign maximal probability to the correct class token. Formally, let x denote the input sequence (i.e., the crash summary), and let $y \in \{0, 1, \ldots, k\}$ be the target class index, where k is the total number of classes. At the final decoding step, the LLM outputs a vocabulary-sized logit vector $\mathbf{z} \in \mathbb{R}^{|V|}$. The probability of generating the target class token is computed via the softmax function:

$$P(y \mid x) = \frac{\exp(z_y)}{\sum_{i=1}^{|V|} \exp(z_i)} \tag{3}$$

Table 2
Overview of backbone models: parameter size and deployment resources.

Model	Parameters	GPU Memory Required
BERT*	≈ 0.11 B	≈ 0.23GB
Open-source LLMs		
LLaMA3.2-1B*	≈ 1.23 B	≈ 3 GB
LLaMA3.2-3B*	≈ 3.21 B	≈ 6 GB
Qwen2.5-7B-Instruct*	≈ 7 B	≈ 14 GB
Mistral-7B-Instruct-v0.3*	≈ 7.30 B	≈ 14 GB
LLaMA3.1-8B*	≈ 8 B	≈ 15 GB
LLaMA3.3-70B-Instruct	≈ 70 B	≈ 131 GB
Closed LLMs		
GPT-4o	—	—

(*PLMs marked with * are fine-tuned in this work.*)

The training objective is to minimize the negative log-likelihood of the correct class token, which corresponds to the standard cross-entropy loss:

$$\mathcal{L}_{\text{CE}} = -\log P(y \mid x) = -\log\left(\frac{\exp(z_y)}{\sum_{i=1}^{|V|}\exp(z_i)}\right) \tag{4}$$

3.4. BERT fine-tuning

To adapt BERT for classification tasks, a linear layer is added on top of the BERT encoder. Specifically, the hidden representation of the special `[CLS]` token is used as the sentence embedding $h_{[CLS]}$. The prediction is obtained as

$$\hat{y} = \text{softmax}(W \cdot h_{[CLS]} + b), \tag{5}$$

where W and b are trainable parameters of the linear classifier. During fine-tuning, the objective is to minimize the cross-entropy loss between $\hat{y}$ and the ground-truth label, identical to the loss function used as LLM fine-tuning in Section 3.3. Importantly, all parameters of BERT together with the classification layer are updated end-to-end, enabling the model to learn task-specific knowledge while retaining its pretrained linguistic representations.

4. Experiments

4.1. Experimental settings

Backbones and baselines Our pipeline is built upon a range of PLMs, including both open-source and closed-source models. The open-source backbones include BERT (Devlin et al., 2019), the LLaMA3 series (Meta, 2024) (LLaMA3.2-1B, LLaMA3.2-3B, LLaMA3.1-8B, LLaMA3.3-70B), Qwen2.5-7B-Instruct (Yang et al., 2025), and Mistral-7B-Instruct-v0.3 (Jiang et al., 2024). The parameter sizes and deployment resource requirements of PLMs are summarized in Table 2.[3]Some are further adapted to the task via fine-tuning. We also evaluate the closed-source model GPT-4o (OpenAI, 2024). As baselines, we include both neural networks and traditional methods. Neural baselines consist of TextRNN (Liu et al., 2016) and FastText (Joulin et al., 2016). Traditional machine learning baselines include rule-based methods and classifiers such as Random Forest(RF), Conditional Random Fields (CRF), and XGBoost, all implemented with TF-IDF features. To ensure a fair comparison, all experiments were conducted on a single NVIDIA A100 GPU, except for LLaMA3-70B, which required 4 A100 GPUs due to its larger size.

Dataset To assess the generalizability of our approach to future police reports, model training was performed on crash data of year 2020, consisting of 300 annotated examples for prompt design and about 3000 examples for fine-tuning. We evaluated 2000 test cases from 2021 to measure inference performance.

Hyper-Parameters Inference was conducted with a fixed temperature of 0.2, controlling the randomness of output generation. Through multiple pilot experiments, we found that this value offered the best trade-off between accuracy and robustness. All models are trained with AdamW (Loshchilov and Hutter, 2017) optimizer and a learning rate of 2×10^{-5}. For BERT, we adopt full fine-tuning with a weight decay of 0.01 and a batch size of 16, and select the best model based on the macro F1 score. For LLMs, we apply LoRA for parameter-efficient fine-tuning. Following prior work (Raschka, 2024), we set $r = 8$ and $\text{LoRA}_\alpha = 16$, which balance training efficiency and performance for the classification task, and apply LoRA to the query projection (Query-projection) layers. Our fine-tuning and test scripts are publicly available.[4]

[3] Model sizes and deployment requirements are referenced from the official NVIDIA NIM documentation: https://docs.nvidia.com/nim/large-language-models/latest/introduction.html.

[4] https://github.com/hiXixi66/PLMs-Crash-Narrative-Analysis

4.2. Manner of collision

To evaluate both the overall correctness of predictions and the balance of performance across different classes, we assessed model performance using accuracy and Macro F1-score (Sokolova and Lapalme, 2009). In addition, deploying, training, and running inference with LLMs and BERT require substantial computational resources. To better understand the trade-offs, we compare models of different sizes and from different providers, reporting both training and inference times. These results are then analyzed together with accuracy and Macro F1-score to provide a comprehensive evaluation of the whole framework.

During manual inspection, we found that some instances labeled as *Unknown* (label ID: 9) in the original dataset could reasonably be reassigned to other valid categories. Therefore, in addition to reporting results on the full original dataset, we also present evaluations with the *Unknown* category excluded to provide a more precise assessment of model performance.

Results of different baselines and PLMs on Manner of Collision

The overall results of MANCOLL classification are summarized in Table 3. We reported results both including and excluding *Unknown* in the column *ALL* and *–Unknown* respectively.

Among the baselines, traditional machine learning methods combined with TF-IDF features outperform neural network models such as TextRNN and FastText. In particular, XGBoost achieves the best performance among all baselines, reaching 91.8% accuracy and 0.685 macro F1 (All). For open-source LLMs, fine-tuning brings substantial improvements. All models show consistent gains as the number of training steps increases. For example, LLaMA3-3B from an initial accuracy of 50.2% to over 95.1% after fine-tuning. Similar trends are observed across BERT, Qwen, Mistral, and LLaMA3 variants, demonstrating the effectiveness of parameter-efficient fine-tuning. For closed-source models, GPT-4o achieves the best performance with zero-shot guidance. Using few-shot examples alone degrades performance, and even when combined with guidance, the improvement remains limited and does not surpass pure guidance. Therefore, in this task, zero-shot inference with well-designed guidance is both the most effective and the most token-efficient. Overall, fine-tuned open-source LLMs tend to perform competitively with, and in several cases outperform, both traditional baselines and closed-source models on this task. These findings suggest that with a relatively small number of fine-tuning steps, even moderately small-scale PLMs may be effectively adapted for the *Manner of Collision* classification task, though further validation across additional datasets would be needed to confirm the generalizability of this observation.

Effect of training data

Since real-world data often contains label noise and is limited in quantity, we investigate how these factors affect model performance. Specifically, we evaluate the robustness of high-performing models (FastText, BERT, and LLMs) on the MANCOLL classification task under noisy and low-resource conditions, assessing their ability to handle imperfect and scarce data.

Effect of noisy data. In real application scenarios, labeled data are influenced by the knowledge of human investigator, so the data cannot be guaranteed to be fully correct and may inevitably contain noise. However, the downstream application requires reliable predictions on clean inputs. Therefore, to evaluate the robustness of the models, we further examined their generalization by fine-tuning on noisy data and then testing on the clean dataset.

In this experiment, we introduce label noise by randomly assigning a subset of the data samples[5] to one random class, and then train models on this noisy dataset. As shown in Fig. 5, we observe that when the noise ratio is below 30%, the performance of all models does not drop substantially, indicating that these methods are relatively robust to moderate levels of noise. And LLaMA3-3B consistently outperforms BERT, benefiting from background knowledge in its prompts, whereas BERT and FastText lack this advantage. Nevertheless, when the noise ratio increases further (the red box in Fig. 5a), the performance of the LLM drops sharply. This is likely due to knowledge conflicts between the prior knowledge encoded in the prompts and the noisy training signals.

Effect of training samples amount. To enhance the performance of PLMs on traffic data analysis through fine-tuning, a sufficient amount of annotated data is required. However, producing large annotation quantities is labor-intensive and costly. Therefore, we conduct experiments to analyze how the size of the training dataset affects model performance. Out of this concern, we investigate the impact of the number of training samples on model performance.

In this experiment, we fine-tune the models using training sets of varying sizes, ranging from 200 to 2000 cases[6] As shown in Fig. 6, with fewer than 400 examples, LLaMA3-3B requires more data to converge due to its longer prompts and bigger model size, whereas BERT and FastText, with simpler architectures and shorter inputs, achieve better performance on smaller datasets. However, once the training set reaches around 500 samples (the red box in Fig. 6a), LLaMA3-3B experiences an “aha stage” with performance rising sharply and surpassing both BERT and FastText. Although LLM is more data-hungry at the beginning, it can learn from both prompt and training data more effectively once sufficient training data is available, leading to superior accuracy and Macro F1 performance.

[5] Experiments with noise ratios above 40% are not reported, as such levels indicate severely degraded data quality, making fine-tuning on these datasets impractical and meaningless.

[6] BERT and FastText function primarily as feature extractors and require at least some labeled data for task-specific fine-tuning to perform classification effectively. Without labeled data, they can only provide general-purpose text representations and are unable to produce meaningful classification results.

Table 3
Comparison results of various PLMs and baselines on Manner of Collision (MANCOLL) classification task.

Backbones	Training step	Training time (s)	Inference time (ms)	Accuracy(%)		Macro F1	
				All	-Unknown	All	-Unknown
Baselines							
Rule based	–	–	< 1.0	51.1	51.3	0.358	0.336
TextRNN	–	1.9	< 1.0	41.9	42.6	0.093	0.110
FastText	–	1.1	< 1.0	78.8	80.4	0.342	0.407
TF-IDF + RF	–	0.7	< 1.0	87.2	88.6	0.488	0.574
TF-IDF + CRF	–	17.3	< 1.0	88.6	89.3	0.626	0.571
TF-IDF + XGBoost	–	11.8	< 1.0	**91.8**	**93.1**	**0.685**	**0.659**
Open source LLMs							
BERT	169	29.8	3.9	85.3	86.7	0.429	0.507
	507	89.1	4.0	91.2	92.7	0.551	0.650
	845	148.6	4.0	**92.9**	**94.3**	**0.637**	**0.620**
LLaMA3-1B	Original		25.6	1.6	0.0	0.005	0.000
	417	112.9	20.1	45.4	46.2	0.151	0.178
	834	226.7	21.1	81.9	83.3	0.359	0.423
	1251	339.8	20.4	85.8	87.2	0.407	0.479
	1668	452.4	20.0	**87.5**	**88.9**	**0.491**	**0.496**
LLaMA3-3B	Original		33.8	33.7	34.9	0.184	0.143
	417	222.1	33.5	92.8	94.4	0.690	0.815
	834	443.4	35.3	94.0	95.4	0.754	0.745
	1251	668.9	33.7	95.0	96.4	0.762	0.753
	1668	890.5	33.3	**95.1**	**96.4**	**0.779**	**0.753**
Qwen2.5-7B	Original		48.2	76.2	77.0	0.562	0.559
	417	396.2	49.7	92.7	94.1	0.680	0.679
	834	795.3	49.9	94.0	95.5	0.745	0.746
	1251	1192.7	49.5	**94.4**	**95.7**	**0.774**	**0.755**
	1668	1583.9	49.7	94.4	95.7	0.772	0.753
Mistral-7B	Original		49.7	81.7	83.1	0.553	0.561
	417	377.63	48.8	90.1	91.5	0.680	0.679
	834	756.12	49.8	92.4	93.3	0.765	0.729
	1251	1149.84	49.5	91.2	92.2	0.740	0.701
	1668	1512.86	49.7	**94.4**	**95.7**	**0.772**	**0.753**
LLaMA3-8B	Original		54.9	34.3	34.9	0.214	0.251
	417	406.6	56.8	94.4	96.0	0.749	0.886
	834	803.6	56.8	95.2	96.5	0.803	0.773
	1251	1209.8	56.6	95.9	97.1	0.833	0.789
	1668	1615.4	57.1	**96.1**	**97.1**	**0.848**	**0.788**
LLaMA3-70b	Original		508.3	91.4	92.7	0.692	0.704
Closed LLMs							
GPT-4o with zero-shot guidance				**90.9**	**92.3**	**0.674**	**0.805**
GPT-4o with few-shot examples				89.1	90.6	0.656	0.662
GPT-4o with both few-shot example and guidance				90.8	92.3	0.708	0.715

Consistency analysis

To evaluate model stability, we conduct consistency experiments, including self-consistency (the stability of repeated outputs from the same model) and cross-model consistency (the agreement between different models). We report these results only for models with PLM backbones.

Consistency analysis results are summarized in Table 4. Overall, the majority of models show high levels of both self-consistency and cross-model consistency. Besides, the overall consistency results (upper part of table) reveal that excluding ground truth (GT) label slightly increases consistency scores, and removing the *Unknown* label further improves them. This suggests that some inconsistencies may stem from labeling errors: for example, instances marked as *Unknown* in the dataset were consistently assigned to a specific class by the fine-tuned models, and these models strongly agreed on such classifications. This indicates that the models may have identified correct answers that were mislabeled in the ground truth.

In terms of self-consistency, all models achieve scores above 0.96, indicating that their outputs remain stable across repeated runs. Regarding cross-model consistency, the majority of model pairs achieve scores exceeding 0.90, highlighting strong agreement

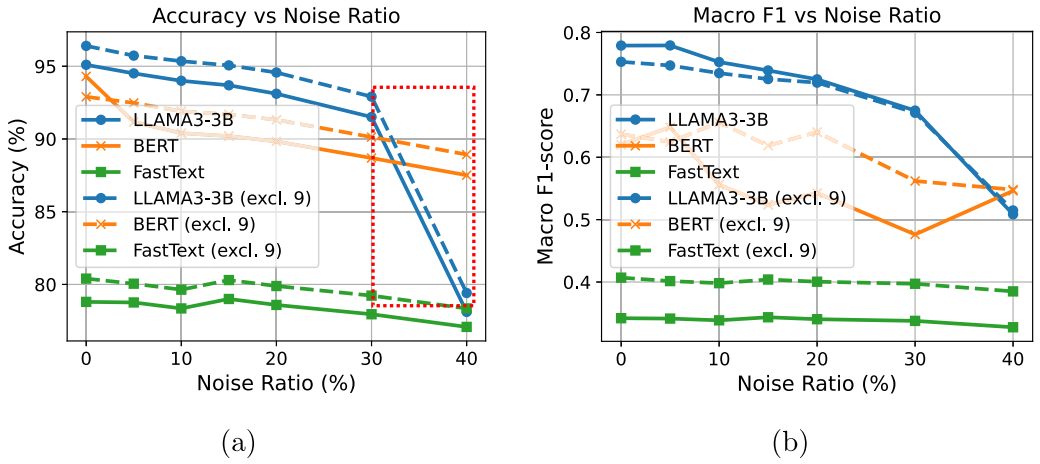


Fig. 5. Accuracy (a) and Macro F1 (b) of LLaMA3-3B, BERT, and FastText under different noise ratios in the training data.

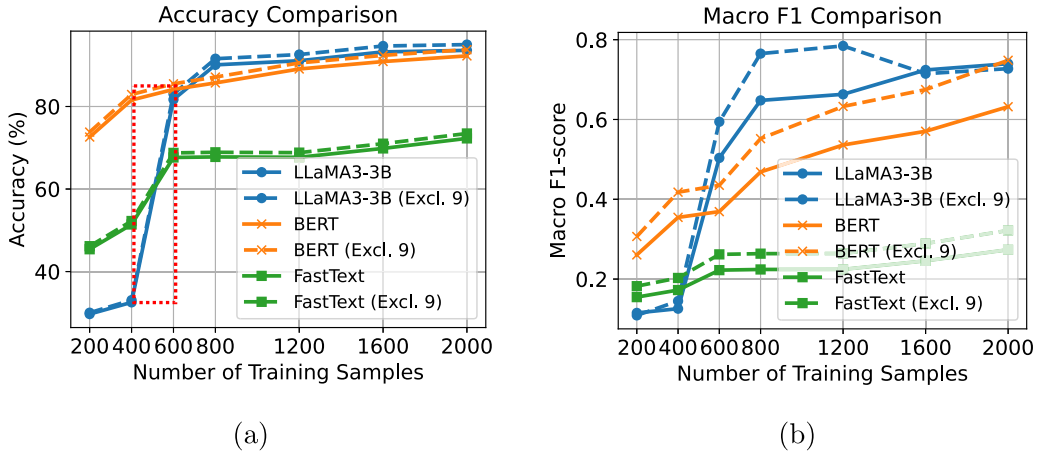


Fig. 6. Accuracy (a) and Macro F1 (b) comparisons of LLaMA3-3B, BERT, and FastText under different numbers of training samples.

Table 4
Self-Model and Cross-Model consistency table.

Overall consistency									
			Models only			Including ground truth (GT)			
Including: *9 - Unknown*			0.9443			0.9438			
Excluding: *9 - Unknown*			0.9502			0.9508			
Self-Model and Cross-Model consistency including: *9 - Unknown*									
	BERT	LLaMA3 1B	LLaMA3 3B	LLaMA3 8B	Qwen2.5 7B	Mistral 7B	LLaMA3 70B	GPT 4o	GT
BERT	1.00	0.87	0.94	0.94	0.93	0.89	0.91	0.90	0.93
LLaMA3-1B	0.87	0.96	0.86	0.86	0.86	0.83	0.84	0.84	0.86
LLaMA3-3B	0.94	0.86	0.98	0.97	0.95	0.91	0.93	0.91	0.95
LLaMA3-8B	0.94	0.86	0.97	1.00	0.96	0.92	0.93	0.92	0.96
Qwen2.5-7B	0.93	0.86	0.95	0.96	0.99	0.91	0.92	0.92	0.94
Mistral 7B	0.89	0.83	0.91	0.92	0.91	0.98	0.89	0.90	0.91
LLaMA3-70B	0.91	0.84	0.93	0.93	0.92	0.89	0.98	0.92	0.91
GPT-4o	0.90	0.84	0.91	0.92	0.92	0.90	0.92	0.97	0.91
GT	0.93	0.86	0.95	0.96	0.94	0.91	0.91	0.91	1.00

between different fine-tuned models. When comparing against the GT, fine-tuned models such as BERT, LLaMA3-3B, LLaMA3-8B, and Qwen2.5-7B achieve significantly higher consistency than non-fine-tuned models like the GPT-4o with best prompt and LLaMA3-70B. This trend aligns with the improvements previously reported in Table 3.

4.3. Crash Type

Compared with *Manner of Collision*, the extraction of *Crash Type* is more challenging. There are 98 categories in total, and due to the hierarchical structure, shown in Fig. 1, the set of candidate `CRASHTYPE` is determined by the associated `CRASHCONF`. This decomposes the task into 13 smaller classification subtasks (one for each `CRASHCONF`), with the largest subtask containing up to 14 classes. Another source of difficulty is that a single crash may involve multiple vehicles. When classifying the `CRASHTYPE` for one vehicle, the narrative often contains descriptions of other vehicles, which can interfere with or complicate the judgment for the target vehicle.

Considering the above, it remains challenging for LLMs to extract information even though all tasks are within the traffic safety domain. To better understand how model performance can be improved when encountering such complex task, we further experiment with different LoRA projection configurations and examine their impact on fine-tuning results. As described earlier in Section 3.1, `CRASHCONF` classification is a relatively easy task to achieve high accuracy. Therefore, in this part we treat `CRASHCONF` as oracle knowledge and focus only on the classification of `CRASHTYPE`.

Results of different PLMs on Crash Type

A single crash may involve multiple vehicles, and descriptions of other vehicles can affect the prediction for the target vehicle. As the number of involved vehicles increases, the narrative becomes more complex, making it harder to accurately identify the target vehicle and associate the correct actions and interactions with it. This multi-vehicle complexity introduces additional ambiguity and increases the difficulty of classification. To account for this factor, we evaluate model performance under different vehicle-count settings, grouping crashes into four categories: 1, 2, 3, and more than 3 vehicles. Given the difficulty of this task, the evaluation is restricted to models that demonstrated strong performance on the *Manner of Collision* classification task, namely RF, CRF, XGBoost, BERT and LLMs.

The results under different vehicle-count settings are shown in Table 5. Overall, this task is challenging, as reflected by the relatively low performance of traditional baselines. Most baselines have an accuracy below 60%, especially in two-vehicle scenarios, indicating the difficulty of accurately identifying the target vehicle and its interactions in complex narratives. In contrast, open-source PLMs show substantial improvements after fine-tuning. Performance consistently increases across all vehicle-count settings, with several models achieving over 80% accuracy in complex scenarios (e.g., LLaMA3-3B, Qwen2.5-7B, and LLaMA3-8B). Notably, large models without fine-tuning, such as GPT-4o and LLaMA3-70B, tend to underperform compared to fine-tuned models on this task, suggesting that these smaller fine-tuned models may offer a potentially favorable balance between efficiency and accuracy given their significantly lower inference requirements. These results indicate that model scale alone may be insufficient for this task, and that task-specific fine-tuning appears to play an important role, though the relative contribution of fine-tuning versus other factors, such as domain specificity of the training data, warrants further investigation.

When analyzing results across different vehicle counts, we observe that collisions involving exactly two vehicles yield the lowest accuracy for all models except for GPT-4o. This is to be expected. Single-vehicle cases are the easiest, as the crash narrative only describes one vehicle and contain no textual interference from other vehicles. Multi-vehicle crashes involving three or more vehicles often correspond to chain collisions (e.g., multi-vehicle rear-end crashes), where multiple vehicles share the same *Crash Type*. This homogeneity reduces ambiguity and makes classification more straightforward, despite the larger number of vehicles. The most challenging setting resides in crashes involving exactly two vehicles. In these cases, the narrative usually mixes descriptions of both vehicles, and their *Crash Types* are often different. Consequently, two-vehicle crashes represent the most difficult scenario, demanding stronger reasoning and disambiguation capabilities from the models.

Consistency analysis

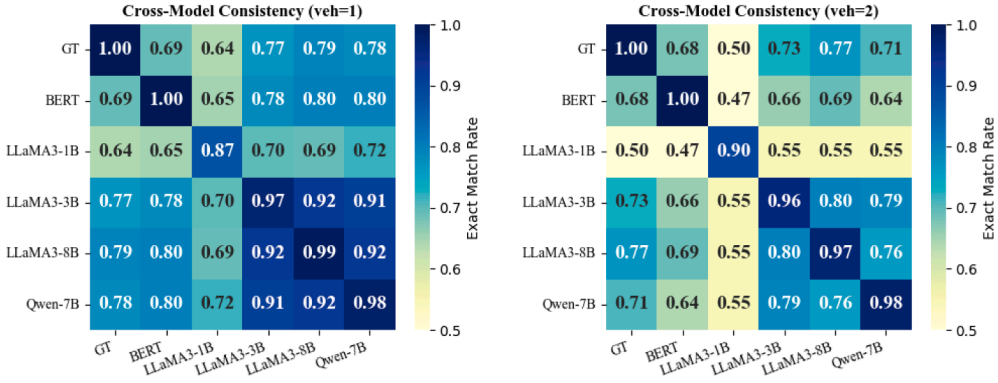
Looking at the available statistics, crashes involving one single vehicle or two vehicles account for nearly the same amount of crashes, and together they represent over 93% of all available crashes. Since such crashes are also the primary interest of researchers, our consistency analysis focuses on these two settings.

For single-vehicle setting, as shown in Fig. 7a, we observe that LLaMA3-3B, LLaMA3-8B and Qwen2.5-7B show near-perfect run-to-run agreement (≈ 0.98) on the diagonal, while smaller models LLaMA3-1B maintain a self-consistency (≈ 0.87). This indicates that fine-tuned models yield stable outputs across runs on `CRASHTYPE` classification. The agreements between LLaMA3-3B, LLaMA3-8B and Qwen2.5-7B are also high (≈ 0.91), suggesting that different LLMs converge to similar predictions despite varying in size and architecture. When compared with CISS ground-truth labels (GT), the average consistency is slightly lower (≈ 0.78), reflecting the possible presence of annotation noise and labeling errors in the dataset, which LLMs may help to mitigate.

For the two-vehicle setting, as shown in Fig. 7b, we observe a similar trend as in the single-vehicle case: larger models such as LLaMA3-3B, LLaMA3-8B, and Qwen2.5-7B maintain high self-consistency, while smaller models like LLaMA3-1B achieve a reasonable level of self-consistency (≈ 0.87). However, cross-model agreements among LLaMA3-3B, LLaMA3-8B, and Qwen2.5-7B drop noticeably (≈ 0.79). Compared with CISS ground-truth labels, the average cross-consistency (≈ 0.78) is also lower than in the single-vehicle setting. All above reflect a reduced accuracy because of the added complexity and ambiguity of multi-vehicle crash narratives.

Table 5
Comparison results of various PLMs and baselines for CRASHTYPE classification under different settings (number of vehicles involved). The best results are highlighted in bold.

Backbones	Training step	Training time (s)	Number of vehicles in a crash			
			=1	=2	=3	>3
Baselines						
TF-IDF + RF	–	2.78	60.2	41.2	51.1	62.2
TF-IDF + CRF	–	869.27	46.6	23.8	38.2	50.7
TF-IDF + XGBoost	–	244.54	58.6	48.6	59.7	67.7
Open source models						
BERT	876	126.4	45.8	42.0	63.8	72.1
	1460	252.9	57.6	53.8	68.6	78.2
	2920	506.6	**68.9**	**68.8**	**77.1**	**78.2**
LLaMA3-1B	Original		7.1	3.6	3.7	1.3
	721	198.4	49.1	26.7	43.3	45.1
	1442	397.2	43.7	42.7	64.1	75.2
	2163	594.7	52.6	49.4	64.5	73.4
	2884	780.3	**62.2**	**51.4**	**64.7**	**75.4**
LLaMA3-3B	Original		50.2	23.3	18.4	12.4
	721	430.1	73.9	53.7	67.1	77.4
	1442	8429	74.0	69.5	78.9	81.9
	2163	1249.8	73.6	71.5	82.6	83.9
	2884	1683.7	**76.0**	**73.7**	**81.8**	**83.6**
Qwen2.5-7B	Original		25.3	21.7	14.8	6.2
	721	659.3	77.2	57.6	68.9	70.3
	1442	1308.5	78.3	68.9	72.8	79.4
	2163	1965.2	**79.1**	**70.3**	**80.6**	**80.9**
	2884	2605.5	77.2	72.9	80.1	81.5
LLaMA3-8B	Original		40.4	18.9	19.0	15.1
	721	694.3	73.9	53.7	67.1	75.2
	1442	1387.5	76.6	77.0	83.3	83.6
	2163	2088.5	**77.1**	**77.3**	**82.0**	**84.8**
	2884	2780.7	77.6	77.3	81.9	81.2
LLaMA3-70B	Original		72.7	41.8	44.3	55.0
Closed models						
GPT-4o			45.3	64.3	58.8	70.3



(a) Single-vehicle setting (b) Two-vehicle setting

Fig. 7. Self-Model and Cross-Model consistency.

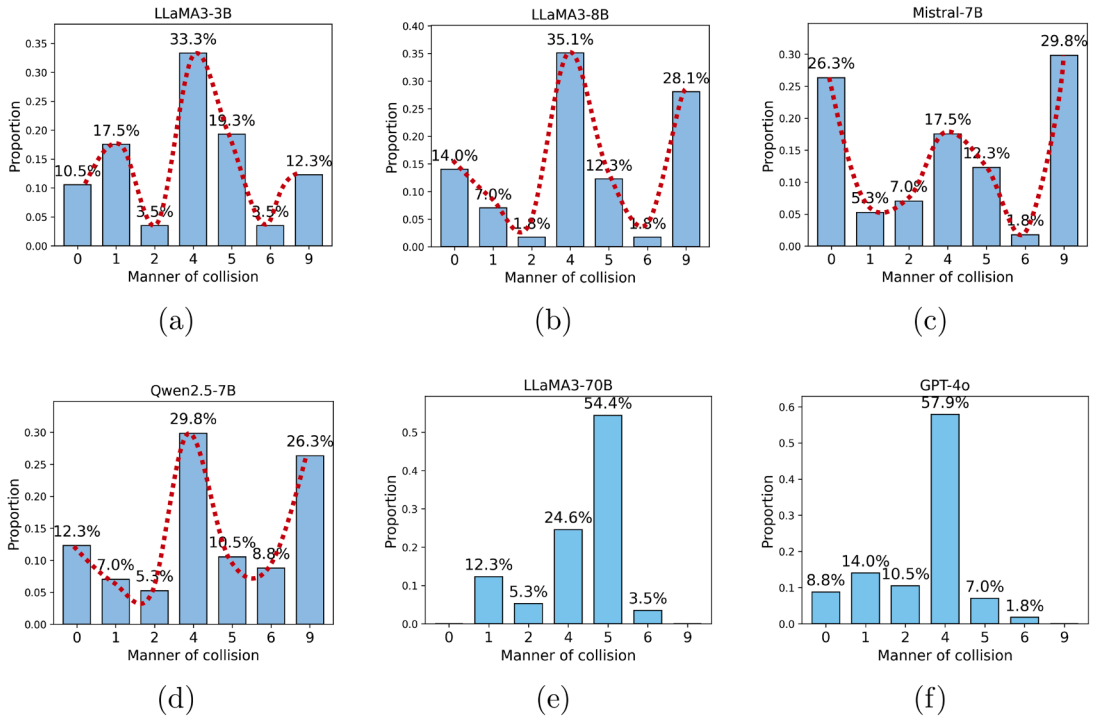


Fig. 8. Comparison of reclassification results for cases originally labeled as *Unknown* in MANCOLL across different LLM backbones. Red dashed lines highlight the overall distribution pattern.

5. Data analysis

5.1. Distribution of reclassified Unknown MANCOLL cases

As shown in Table 1, the proportion of samples labeled as *Unknown* in the original database is about 1.6%. These cases mainly fall into two types. Type I corresponds to mapping-related annotation issues, where errors in underlying indicators lead to incorrect *Unknown* labels. Type II corresponds to out-of-scope cases that do not belong to any predefined categories. Previous experiments demonstrated that our models achieve strong overall performance on the MANCOLL classification task, with some reaching up to 97% accuracy. We also found that our models can reclassify a portion of previously *Unknown* samples into known categories, which is largely attributed to resolving Type I cases. Meanwhile, a portion of samples remains classified as *Unknown*, which mainly corresponds to Type II cases that are inherently out of scope.

To further investigate this behavior, we analyze how PLMs classify the samples originally labeled as *Unknown* in the dataset. As shown in Fig. 8, categories 2 (Head-On) and 6 (Sideswipe, Opposite Direction) consistently appear with very low proportions across all models, which closely aligns with their proportions in the original dataset. In contrast, models (a)–(d) exhibit relatively stable distribution patterns, suggesting that fine-tuned PLMs can effectively reduce ambiguity in *Unknown* cases while maintaining consistent predictions across models. Detailed case studies for both Type I and Type II cases are provided in Appendices C.2 and C.3.

5.2. CRASHTYPE distributions: CISS annotations vs. LLM predictions

To further verify that the labels generated by the LLM do not substantially alter the intrinsic characteristics of the data, we compared some statistics between LLM-generated labels and ground-truth. Here we primarily focus on accidents involving one or two vehicles, which together account for approximately 93% of the dataset. For single-vehicle crashes, we conduct distributional analysis of *Crash Type*, while for two-vehicle crashes, we perform correlation analysis to assess the association between the *Crash Types* of the two vehicles.

Single-vehicle setting

To characterize single-vehicle crashes, we analyze the distribution of their *Crash Types*. Specifically, we compute the frequency distribution of the most common crash categories (1–16), and the overall divergence of the distribution. Since some *Crash Types* in the distribution may have zero or near-zero frequencies, we adopt Jensen–Shannon (JS) divergence (Nielsen, 2019) to measure the difference between the two distributions, as it is symmetric, bounded, and robust to zero-probability events, thereby avoiding the divergence issues inherent in Kullback–Leibler divergence (Kullback, 1951) (Table 6).

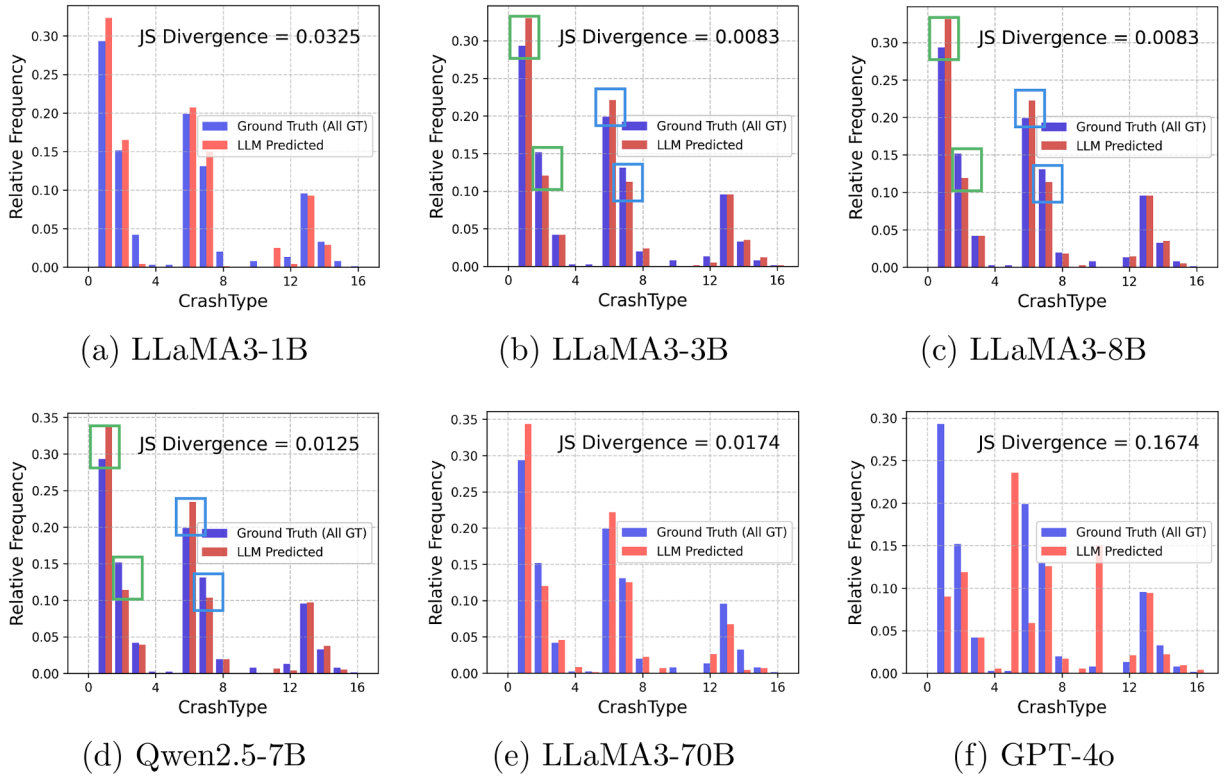


Fig. 9. Comparison of CRASHTYPE distributions between the CISS ground truth (blue) and LLM-predicted results (red) for crashes involving a single vehicle. Each subplot corresponds to a different model, with the JS divergence reported as a measure of distributional similarity.

Table 6
Examples of single-vehicle CRASHTYPE categories related to road departure.

Code	Description	Departure Direction
1	Roadside departure under a controlled situation.	Right
2	Roadside departure because of lost traction or control.	Right
6	Roadside departure under a controlled situation.	Left
7	Roadside departure because of lost traction or control.	Left

Fig. 9 presents the predicted distribution of single-vehicle CRASHTYPE across different models, where only the first 16 crash categories are included since single-vehicle data is restricted to this subset. Overall, the fine-tuned models successfully preserve the distributional characteristics of the ground-truth data: the relative frequency and ranking of each class are largely consistent with the ground truth labels. Moreover, except for the 1B model, all fine-tuned models achieve closer alignment with the ground-truth distribution compared to much larger non-fine-tuned models such as LLaMA3-70B and GPT-4o. Besides, noticeable discrepancies remain for certain categories, particularly CRASHTYPE 1, 2, 6, and 7[7]. The boxed regions in the histogram illustrate a complementary phenomenon between the ground truth distribution and the LLM-predicted results. Specifically, the discrepancies arise mainly from whether the model correctly captures the detail of losing control or not. Because the LLM fails to fully recognize this subtle distinction, its predicted frequencies are shifted compared to the ground truth. Interestingly, similar complementary patterns are consistently observed across fine-tuned models, which indicates that this misalignment is not accidental. Through manual inspection of the CISS dataset, we found that part of this mismatch arises from inconsistencies in the recorded ground truth in CISS, where some cases were not assigned correctly. Illustrative examples of such issues are provided in Appendix C.1.

Two-vehicle setting

Crash types of vehicles in the same accident are naturally related. To quantify this, we compute correlation coefficients using Kendall's Tau (Kumar, 1968). Since the *Crash Type* values are discrete identifiers rather than continuous quantities, *Kendall's Tau* is more appropriate than *Pearson* or *Spearman* correlation (Arndt et al., 1999). It is important to note that in this task, a higher

[7] The detailed definitions of these *Crash Types* are provided in Table 6

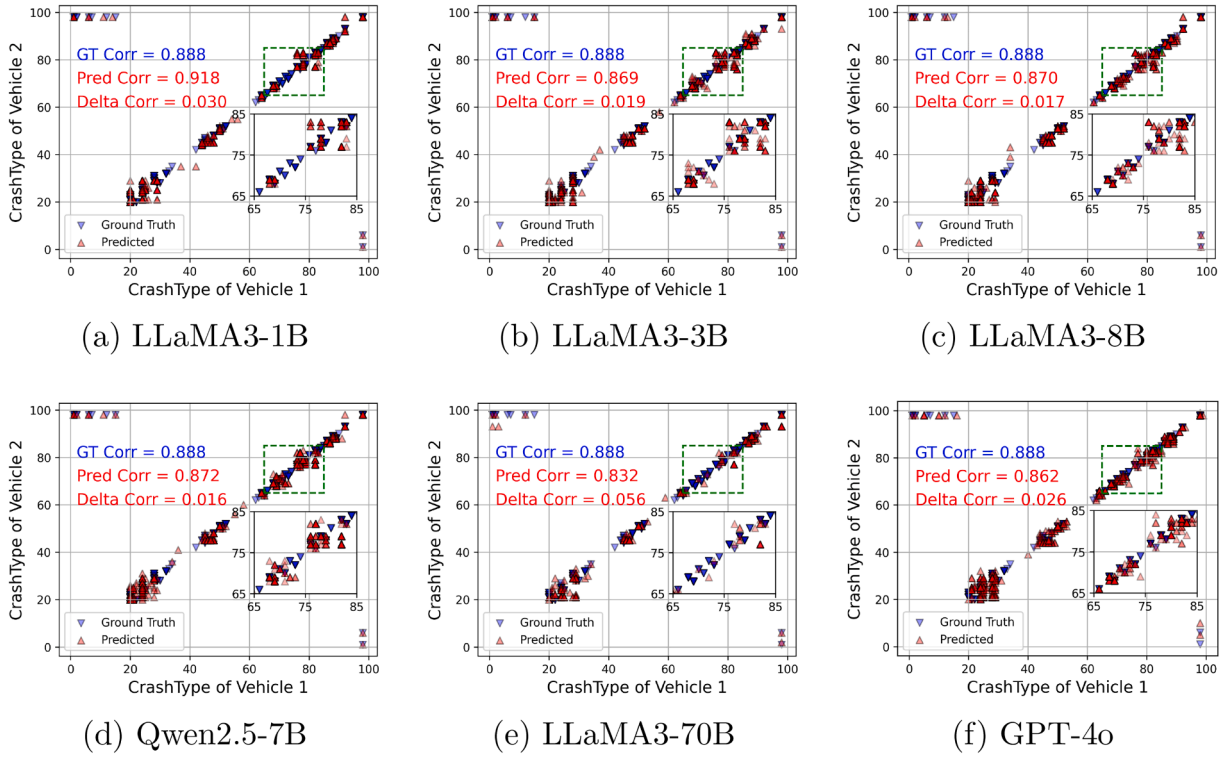


Fig. 10. Correlation analysis of CRASHTYPE combinations between two vehicles in the same crash. Blue markers represent CISS ground truth (GT) and red markers represent LLM predictions, with Spearman correlation coefficients reported for both.

correlation is not necessarily better; instead, what matters is whether the correlation obtained from the predicted labels is closely aligned with the ground-truth correlation.

As shown in Fig. 10, models with higher classification accuracy generally yield correlations closer to ground truth. Across all fine-tuned models, the predicted correlations remain close to the true values, indicating that they not only achieve accurate *Crash Type* predictions but also preserve the inter-vehicle dependency patterns in the CISS dataset.

6. Discussion

Task-specific semantics. We investigate whether fine-tuned LLMs acquire sufficient task-specific semantics, given that such semantics are embedded in the fine-tuning data. During fine-tuning, the model continuously updates its internal representations, making it more sensitive to the semantic cues of the target task and achieving a more fine-grained understanding. Consequently, the fine-tuned smaller LLMs outperform the larger general LLMs. However, these improvements remain limited. For tasks such as *Manner of Collision* extraction, the model only needs to capture the overall semantics of the narrative and candidate categories are semantically well-separated in the embedding space, PLMs perform better without requiring fine-grained discrimination of textual details. In contrast, *Crash Type* requires accurate per-vehicle prediction among interwoven descriptions. To do classification, the model has to carefully identify the target entity, accurately track target vehicles' description, and resolve causal relations between events, while resisting misleading information from other vehicle descriptions. Here, even small errors such as confusing subjects and objects can lead to misclassification as shown in Appendix C.5. As a result, LLMs are less effective for per-vehicle *Crash Type* classification than for per-crash *Manner of Collision* classification.

Model hallucination. Hallucination in LLMs can arise from several factors. One important source is error accumulation in autoregressive generation, where each token prediction depends on previously generated tokens. When generating long sequences, early prediction errors may propagate through subsequent steps, and eventually lead to incorrect or fabricated information. In our task, however, the model is required to predict a single class label rather than generate a multi-step sequence, which largely eliminates this source of error accumulation. Furthermore, through fine-tuning, the model parameters are updated by minimizing a task-specific loss, which reduces the probability of prediction errors at the individual prediction step by encouraging the model toward the desired label format and significantly narrows the reasoning chain. This reduces variability in generation and limits the possibility of irrelevant or fabricated content in this specific task. As a result, the combination of single-step prediction and task-specific fine-tuning substantially reduces the likelihood of hallucination in our framework.

Enhancing crash data annotation. We found that PLMs are highly promising for enhancing crash data annotation under different practical conditions. When labeled data are scarce, powerful LLMs can already support annotation through prompt-based inference by prompt engineering methods, providing a useful starting point for categorizing crash cases. When a small amount of labeled data is available, fine-tuning further improves performance and enables more accurate and task-adapted annotation. We also found that fine-tuned models remain robust under moderate label noise, indicating that imperfect real-world annotations do not substantially undermine their effectiveness. Moreover, the models can reclassify some cases originally labeled as *Unknown* into specific categories and even correct occasional labeling errors, while preserving the original distributional characteristics of the data. Together, these findings provide strong empirical evidence that PLMs can be used to assist crash data annotation, improve label quality, and reduce human workload in realistic settings.

Applications in real-world transportation systems. The extracted variables in this work, such as Manner of Collision and Crash Type, can directly enable multiple downstream applications, including crash prediction and pattern analysis, countermeasure development and safety analysis, and informing infrastructure investment decisions. As these information enables more granular identification of high-risk maneuver patterns and collision configurations and support spatial and temporal analyses of collision patterns. More importantly, beyond supporting downstream applications, the proposed framework establishes a unified pipeline for continuous data enrichment, workflow integration, and model-driven analysis. It is also important to note that many transport agencies restrict uploading crash data to external AI services due to legal and security concerns. Thus, locally deployable open-source LLMs are not only a technical choice but also a governance-aligned solution, enhancing real-world applicability.

7. Conclusion

This study demonstrates that compact open-source PLMs, after taks-specific fine-tuning, can provide state-of-the-art performance for reasoning-intensive extraction from crash narratives, a long-standing challenge in traffic safety research where accurate crash analysis is essential for informing crash prevention strategies and policy development. Our BERT and fine-tuned LLaMA3-3B model outperform GPT-4o and LLaMA3-70B in identifying *Manner of collision* at the crash level and *Crash Type* at the vehicle level in multi-vehicle scenarios, while requiring only a limited number of training steps, modest computational resources, and delivering fast inference. Consistency analyses further demonstrate strong robustness across models and scenarios, enabling reliable per-vehicle *Crash Type* labeling from complex, multi-vehicle narratives.

Analysis of LLM-annotated data reveals that fine-tuned PLMs can correct labeling errors, compensate for deficiencies arising from limited annotator knowledge, and preserve relational characteristics of the data without introducing systematic bias. In addition, the approach mitigates privacy concerns associated with external API usage and reduces deployment cost. Experiments with varying levels of label noise further confirm the robustness of PLMs, while scaling the amount of training data highlights their potential to substantially reduce the human effort required for data annotation.

Taken together, these findings establish automated annotation using fine-tuned PLMs as a viable and scalable alternative to labor-intensive manual coding. While the evaluation is conducted on the U.S. CISS dataset accross two extraction tasks, future work will explore broader safety information extraction (e.g., causal chain identification), uncertainty calibration, and the alignment of different annotation standards for cross-database comparative analysis. Overall, domain-adapted open-source PLMs offer a resource-efficient, privacy-preserving, and robust solution for scalable crash narrative analysis, with broad potential to advance traffic safety research.

8. Future work

Generalization capability. The present study focused on extracting *Manner of Collision*, and *Crash Type* from CISS crash narratives. Other attributes, such as impact point, roadway characteristics, and driver behavior, are also important for a comprehensive understanding of collision dynamics. Future work should extend the proposed framework to these attributes and evaluate its transferability across diverse data sources, including international crash databases and non-traditional report formats such as social media records.

Reasoning-oriented LLMs. The experiments conducted in this study employed BERT and instruction-tuned LLMs configured to produce direct label predictions. Although well-designed zero-shot prompting was applied to the LLMs, the models did not explicitly generate intermediate reasoning steps, nor were reasoning-specific models adopted, primarily to limit computational overhead. Prior studies have shown that encouraging LLMs to articulate their reasoning process can improve prediction accuracy. In addition, during fine-tuning, it is possible to introduce guidance tokens that encourage the model to structure its reasoning prior to producing a final prediction. Nevertheless, such approaches entail increased computational costs and would require careful evaluation of the accuracy-efficiency trade-off in the context of crash narrative analysis.

Glossary

For clarity, all field-specific terms used in this paper are summarized in Tables 7 and 8, and all mathematical symbols are summarized in Table 9.

Table 7
Traffic safety field-specific terms used in this paper.

Term	Description
MANCOLL	Indicates the manner in which vehicles in transport collided.
SUMMARY	A basic text description of the crash scenario, and it may include special circumstances not captured in the normal case coding.
CRASHCAT, CRASHCONF	Variables Crash Category and Crash Configuration are used for categorizing the collisions of drivers involved in crashes. Each Category is further defined by a Crash Configuration.
CRASHTYPE	A numeric value used to classify the first harmful event in a crash and assigned by selecting the Crash Category and the Crash Configuration.

Table 8
Natural language processing field-specific terms used in this paper.

Term	Description
Pre-trained language models	Neural networks trained on vast amounts of text data, enabling them to learn general linguistic knowledge.
Chain-of-thought (Wei et al., 2022)	A prompt engineering technique that guides LLMs to break down complex tasks into intermediate steps.
Low-Rank Adaptation (Hu et al., 2022)	Enable parameter-efficient fine-tuning of open-source LLMs, supporting safe and on-premise deployment.

Table 9
Summary of symbols and their descriptions used in this paper.

Symbol	Description
X	Input hidden states, $X \in \mathbb{R}^{T\times d}$ (T: token length, d: hidden size)
Q	Query matrix in self-attention, $Q = XW_Q$
K	Key matrix in self-attention, $K = XW_K$
V	Value matrix in self-attention, $V = XW_V$
W_Q, W_K, W_V	Projection matrices for query, key, and value
A, B	Low-rank matrices in LoRA update ($A \in \mathbb{R}^{d\times r}$, $B \in \mathbb{R}^{r\times k}$)
ΔW	Low-rank update in LoRA: $\Delta W = \frac{\alpha}{r} AB$
W'	Adapted weight: $W' = W + \Delta W$
h_{CLS}	Hidden representation of the [CLS] token from BERT encoder
$\mathcal{L}_{\text{CE}}$	Cross-entropy loss: $\mathcal{L}_{\text{CE}} = -\log P(y\|x)$
softmax	Normalization function turning logits into probabilities

CRediT authorship contribution statement

Xixi Wang: Writing – original draft, Software, Methodology, Formal analysis, Conceptualization; **Jordanka Kovaceva:** Writing – review & editing, Supervision, Resources, Project administration, Conceptualization; **Miguel Costa:** Writing – review & editing, Supervision, Methodology; **Shuai Wang:** Writing – review & editing, Resources; **Francisco Camara Pereira:** Supervision; **Robert Thomson:** Writing – review & editing, Supervision, Conceptualization.

Funding sources

Part of this research was supported by the Chalmers Transport Area of Advance project TREND.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors would like to thank e-Commons (research infrastructure at Chalmers University of Technology) for their consultations, and extend special thanks to Nora Speicher and Mattia Carlino for valuable discussions related to running the experiments. We acknowledge the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by Swedish Research Council through grant agreement no. 2022-06725, for awarding this project access to the Alvis cluster for computations and data handling.

Data availability

Our data and code are publicly available: https://github.com/hiXixi66/PLMs-Crash-Narrative-Analysis.

Appendix A. Implicit vs explicit

For example, consider the following description:

Vehicle #2 was stopped facing east in lane 1 of a divided trafficway without positive barrier. Vehicle #1 was traveling east in the same lane. The front of V1 contacted the back of V2. Both vehicles came to rest still in the roadway, facing east.

In this case, attributes such as *travel direction* (east) and *lane position* are explicitly stated and can be directly extracted. However, the *manner of collision* or *crash type* is not explicitly mentioned and must instead be inferred from the interaction between vehicles, their relative motion, and the overall scenario. This requires synthesizing multiple pieces of information and applying domain knowledge, making it substantially more complex than extracting explicit facts.

Appendix B. Prompt in use

B.1. Prompt for `MANCOLL` classification

Task Introduction:

You are a helpful assistant that classifies vehicle collisions into one of the following categories based on the description provided. Please choose the most accurate collision type based on the definitions and clarifications below:

Category Definitions (Expert Knowledge):

*0: "Not Collision with Vehicle in Transport - The vehicle did not collide with another vehicle in motion.", 1: "Rear-End - The front of one vehicle strikes the rear of another vehicle traveling in the same direction.", 2: "Head-On - The front ends of two vehicles traveling in opposite directions collide.", 4: "Angle - The front of one vehicle strikes the side of another at an angle (usually near intersections or crossing paths).", 5: "Sideswipe, Same Direction - Both vehicles are moving in the same direction and **their sides make contact**", 6: "Sideswipe, Opposite Direction - Both vehicles are moving in **opposite directions** and their **sides make contact**", 9: "Unknown - The manner of collision cannot be determined."*

Clarification Rules (Special Prompting Instructions)

*If the collision happens at or near an intersection, classify as 4. If it does not occur near an intersection: and both vehicles are traveling in the same direction, classify as 5. and vehicles are traveling in opposite directions, classify as 6. If the collision involves only one vehicle and a non-vehicle object (e.g., animal, fence, tree), classify it as 0. If no collision is described or it is unclear whether any impact occurred, classify as 9. If multiple collisions occur (e.g., chain reaction), classify based on the **first** collision described in the summary.*
Input original Extracted summary:

```
\"\"\"{summary}\"\"\"
```

Output Instruction (Task Constraint)
Only respond with a single number from the list above. Do not add any explanation.

B.2. Prompt for `CRASHTYPE` classification

Task Introduction:

You are a crash analysis assistant. Your task is to assign a Crash Type ID to a specific vehicle involved in a traffic crash, based on the structured context and detailed textual description below.

Category Definitions (Expert Knowledge):

Use the following Crash Type definitions for classification:

```
{crash_type_options} # This is decided by CRASHCONF
```

Clarification Rules (Special Prompting Instructions)

Crash classification must be based on the following inputs:

```
- Vehicle index: {vehicle_index}
- Vehicle Description:
\"\"\"{vehicle_summary}\"\"\"
```

Instructions: - Carefully identify and focus on vehicle index in the text.
- Consider not only this vehicle's motion and behavior but also how it interacted with other vehicles (e.g., which vehicle was backing, struck another, etc.).
- Use both the structured crash context and relevant textual evidence to determine the most appropriate crash type ID.

Output Instruction (Task Constraint)

Respond with only one number or letter corresponding to the correct Crash Type from the options above. Do not include any explanation or extra text.

Appendix C. Case study

C.1. Correct prediction under wrong label

Here is an example where the fine-tuned PLM provides a more accurate classification compared to the ground-truth. In this case, the narrative text shows no clear indication of vehicle loss of control. While the ground-truth labeled it as a loss-of-control departure (7), the fine-tuned model identified it as a controlled left roadside departure (6). According to the coding guidelines in CISS, the first departure event should be considered and "loss of control" is defined as situations where the vehicle spins off due to surface conditions, oversteer phenomena, or mechanical malfunctions. Here, the vehicle first departed to the left roadside without cause explanation. The subsequent loss of control occurred as a consequence of this departure, not as its cause. Furthermore, in cases of doubt, the guideline explicitly advises using code "06" (Left Roadside Departure, Drive Off Road). In this way, to classify this case correctly, the model must capture crash causality, distinguish first events from outcomes, and apply the definition of "loss of control" with coding rules.

***SUMMARY**: V#1 was traveling east, negotiating a curve left, in lane 1 of a 2 lane, two way roadway. **V#1 departed the roadway to the left**, entered a drainage ditch area, and contacted an embankment with its front plane. The impact caused the vehicle to rotate in a counter clockwise direction where its front plane contacted the embankment a second time. The rear of V#1 then elevated vertically into the air and continued to rotate counter clockwise as it re-entered the roadway. The vehicle then traveled backwards, departed the roadway to the left a second time, and began to rotate in a clockwise direction. The vehicle then rolled over one quarter turn onto its right side plane against an embankment, and came to final rest approximately 9 m from the initial contact point, facing west.*
***Ground Truth**: 7 – Left roadside departure because of lost traction or control. (Not True)*
***Mistral-7B fine-tuned with 1251 steps (LLM)**: 6 – Left roadside departure under a controlled situation. (True)*

To illustrate the distinction between controlled departures and loss-of-control scenarios, we present the following example, which clearly represents a loss-of-control crash. In this case, Vehicle #1 departed its lane to the left and began rotating clockwise, providing clear evidence of a loss-of-control condition. Both the ground-truth label and the fine-tuned model correctly classified this as "7 — Left roadside departure because of lost traction or control".

***SUMMARY**: V#1 was traveling in a northern direction, on a two lane, non divided, bidirectional, dirt roadway with a downward grade, and a curve to the right. **V#1 left its lane to the left, where it began to rotate clockwise**. V#1 left the roadway to the left, where it overturned 6 quarter turns coming to final rest on its roof in the south bound lane of travel.*
***Ground Truth**: 7 – Left roadside departure because of lost traction or control. (True)*
***Mistral-7B fine-tuned with 1251 steps (LLM)**: 7 – Left roadside departure because of lost traction or control. (True)*

C.2. From Unknown to specific

In this example, although the dataset assigns the ground truth label as 9 (*Unknown*), the model correctly identifies the crash as an *Angle* collision. This case belongs to the Type I category, where the *Unknown* label arises from mapping or annotation issues in the MANCOLL labeling process. The result demonstrates that fine-tuned LLMs can recover the correct collision type and effectively identify labeling errors in the dataset.

***SUMMARY**: V2 was stopped southbound in a three lane intersection in lane three, awaiting to turn left eastbound. V1 was traveling westbound on a three lane road in lane two approaching the same intersection. V1 entered the south bound travel lanes of the intersecting roadway, where the front of V1 impacted the left side of V2.*
***Ground Truth**: 9 – Unknown*
***Mistral-7B fine-tuned with 1251 steps (LLM)**: 4 – Angle. (True)*

C.3. From Unknown to Unknown

In this example, both the CISS dataset and the LLM classify the crash as category 9 (*Unknown*). In this case, V1 backs up and its rear collides with the front of V2 traveling behind it in the same lane. Such backing-related collisions are not covered by any of the predefined collision categories in the MANCOLL taxonomy. Therefore, this case belongs to the genuinely out-of-scope cases (Type II).

***SUMMARY**: V1 was traveling west on a two way, two lane, undivided roadway approaching an intersection. V2 was traveling west on a two way, two lane, undivided roadway approaching the same intersection and in the same travel lane as V1. V1 began backing and the back plane of V1 contacted the front plane of V2. V1 and V2 both came to final rest on the roadway.*
***Ground Truth**: 9 – Unknown*
***Mistral-7B fine-tuned with 1251 steps (LLM)**: 9 – Unknown (True)*

C.4. Crash type classification in multi-vehicle scenario

Here is an example that illustrates why fine-tuned LLMs perform better in scenarios involving more than two vehicles. The reason is that many vehicles beyond the second one are categorized into the class 98 (Third or subsequent vehicles involved in a crash).

This aggregation reduces the classification granularity for such vehicles, effectively lowering the difficulty of prediction. As a result, when multiple vehicles are present, the model can rely on this broader category assignment, which improves overall performance compared to the more fine-grained distinctions required in two-vehicle crashes.

***SUMMARY**: V1 was traveling north in the #1 lane of a 5 lane divided roadway. V2 was stopped facing west in the #2 lane of a 5 lane undivided roadway. V3 was stopped facing west in the #1 lane of a 5 lane undivided roadway. V5 was stopped facing south in the #5 lane of a 5 lane divided roadway. V1 traveled through the intersection. V2 and V3 started to accelerate and entered the intersection. The front of V1 impacted the left of V2. V2 rotated clockwise and the front of V2 impacted the left side of V3. V1 continued through the intersection where the front of V1 impacted the center median curb. V1 continued forward where the front of V1 impacted the left side of V5.*

***Ground Truth**:*

Vehicle 1: 88 – Straight paths striking from the left
Vehicle 2: 89 – Straight paths struck on the left
Vehicle 3: 98 – Third or subsequent vehicles involved in a crash
Vehicle 4: 98 – Third or subsequent vehicles involved in a crash
Vehicle 5: 98 – Third or subsequent vehicles involved in a crash

***LLaMA3-8B fine-tuned with 2163 steps (LLM)**:*

Vehicle 1: 88 – Straight paths striking from the left (True)
Vehicle 2: 89 – Straight paths struck on the left (True)
Vehicle 3: 98 – Third or subsequent vehicles involved in a crash (True)
Vehicle 4: 98 – Third or subsequent vehicles involved in a crash (True)
Vehicle 5: 98 – Third or subsequent vehicles involved in a crash (True)

C.5. Crash type classification in two-vehicle scenario

In this example, the fine-tuned LLaMA3-8B model fails to correctly distinguish the *Crash Types* of V1 and V2. The primary difficulty arises from the narrative, where the descriptions of V1 and V2 are closely intertwined, making it hard for the model to disentangle their respective roles. Moreover, the fact that both vehicles rotated and came to rest in similar orientations further complicates classification, as these post-crash states provide limited discriminative cues. The only decisive difference is that V1 rear-ended V2, while V2 was rear-ended by V1. This subtle asymmetry requires precise tracking of subject–object relations in the narrative. The model's prediction error suggests that it could not fully capture all semantics encoded in the prompt, highlighting a key limitation of LLMs in fine-grained, semantic-sensitive classification.

***SUMMARY**: V1 was negotiating a right curve traveling east in the first of four lanes on a divided highway. V2 was traveling east in the same lane ahead of V1. The front of V1 contacted the back of V2. V1 rotated counterclockwise and came to rest on the roadway facing northeast. V2 rotated counterclockwise and came to rest on the roadway facing northeast.*

***Ground Truth**:*

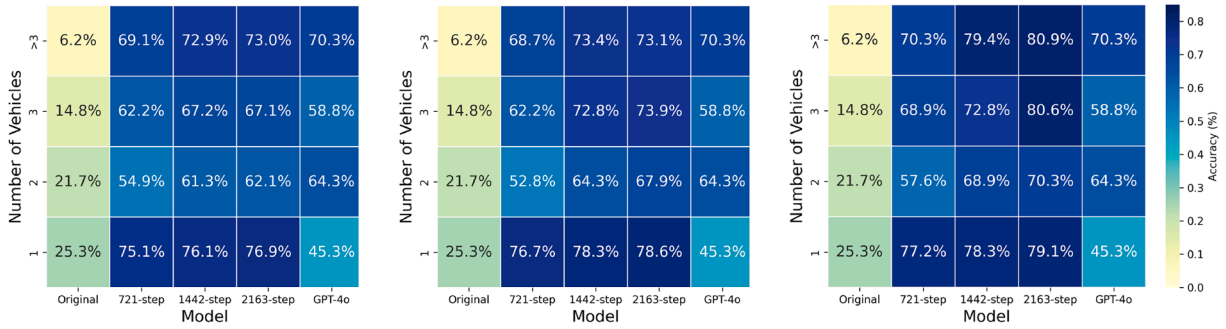
Vehicle 1: 24 – Rear-end: slower (a vehicle that impacts another vehicle from the rear when the struck vehicle was going slower than the striking vehicle.)
Vehicle 2: 25 – Rear-end: slower, going straight (a rear-ended vehicle that was going slower than the other vehicle while proceeding straight ahead.)

***LLaMA3-8B fine-tuned with 2163 steps (LLM)**:*

Vehicle 1: 24 – Rear-end: slower (True)
Vehicle 2: 24 – Rear-end: slower (False)

Appendix D. Effect of LoRA training strategies

Given the complexity of the *Crash Type* classification task, we further explore how LoRA-adapted parameters impacts performance. Specifically, we vary which attention projection matrices to be trained under LoRA, selecting from the query, key, and value projections $W_Q, W_K, W_V \in \mathbb{R}^{d\times k}$. We consider three configurations: (i) LoRA on W_Q only; (ii) LoRA on W_Q and W_V; and (iii) LoRA on all three, W_Q, W_K, and W_V. Results in Fig. D.11 show that the accuracy increases monotonically with the number of adapted attention matrices: Updating W_Q alone yields the weakest performance; adapting both W_Q and W_V performs better; and jointly adapting W_Q, W_K, and W_V achieves the best results. We attribute this to the task's requirement for token interactions: restricting adaptation to queries constrains the model's ability to reshape attention patterns; co-adapting keys improves query–key alignment for retrieval, while adapting values allows the model to better propagate matched evidence through the representation. In short, enabling LoRA on multiple attention projections provides the necessary degrees of freedom to fit the *Crash Type* task more effectively.



(a) Fine-tuned on query projection.

(b) Fine-tuned on query and value projection.

(c) Fine-tuned on query, key and value projection.

Fig. D.11. Accuracy (%) across different numbers of vehicles for various fine-tuning strategies. (a) Fine-tuning only the query projection, (b) fine-tuning both query and value projections, and (c) fine-tuning query, key, and value projections.

References

Aborisade, O., Anwar, M., 2018. Classification for authorship of tweets by comparing logistic regression and Naive Bayes classifiers. In: 2018 IEEE International Conference on Information Reuse and Integration (IRI). IEEE, pp. 269–276.

Administration, N. H. T.S., 2023. Crash investigation sampling system (CISS). Accessed: 2025-07-18. https://www.nhtsa.gov.

Administration, N. H. T.S., et al., 2025. Nhtsa field crash investigation 2023 coding and editing manual. Publ. DOT HS 813, 614.

Akuma, S., Lubem, T., Adom, I.T., 2022. Comparing bag of words and TF-IDF with different models for hate speech detection from live Tweets. Int. J. Inf. Technol. 14 (7), 3629–3635.

Alambeigi, H., McDonald, A.D., Tankasala, S.R., 2020. Crash themes in automated vehicles: a topic modeling analysis of the California Department of Motor Vehicles Automated Vehicle Crash Database. arXiv:2001.11087.

Ali, F., Kwak, D., Khan, P., El-Sappagh, S., Ali, A., Ullah, S., Kim, K.H., Kwak, K.-S., 2019. Transportation sentiment analysis using word embedding and ontology-based topic modeling. Knowl.-Base. Syst. 174, 27–42.

Arndt, S., Turvey, C., Andreasen, N.C., 1999. Correlating and predicting psychiatric symptom ratings: Spearmans r versus kendalls tau correlation. J. Psychiatr. Res. 33 (2), 97–104.

Arteaga, C., Park, J., 2025. A large language model framework to uncover underreporting in traffic crashes. J. Saf. Res. 92, 1–13.

Baburajan, V., e Silva, J.d.A., Pereira, F.C., 2018. Opening up the conversation: topic modeling for automated text analysis in travel surveys. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC). IEEE, pp. 3657–3661.

Baburajan, V., e Silva, J.d.A., Pereira, F.C., 2020. Open-ended versus closed-ended responses: a comparison study using topic modeling and factor analysis. IEEE Trans. Intell. Transp. Syst. 22 (4), 2123–2132.

Blei, D.M., Ng, A.Y., Jordan, M.I., 2003. Latent Dirichlet Allocation. J. Mach. Learn. Res. 3 (January), 993–1022.

Christian, H., Agus, M.P., Suhartono, D., 2016. Single document automatic text summarization using term frequency-inverse document frequency (TF-IDF). ComTech: Comput. Math. Eng. Appl. 7 (4), 285–294.

Cichosz, P., 2018. A case study in text mining of discussion forum posts: classification with bag of words and global vectors. Int. J. Appl. Math. Comput. Sci. 28 (4), 787–801.

Dahl, M., Magesh, V., Suzgun, M., Ho, D.E., 2024. Large legal fictions: profiling legal hallucinations in large language models. J. Leg. Anal. 16 (1), 64–93.

Das, S., Dutta, A., Tsapakis, I., 2021. Topic models from crash narrative reports of motorcycle crash causation study. Transp. Res. Rec. 2675 (9), 449–462.

Devlin, J., Chang, M.-W., Lee, K., Toutanova, K., 2019. Bert: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, volume 1, Long and Short Papers, pp. 4171–4186.

Dokas, I.M., 2026. From hallucinations to hazards: benchmarking LLMs for hazard analysis in safety-critical systems. Saf. Sci. 194, 107056.

Du, W., Dash, A., Li, J., Wei, H., Wang, G., 2023. Safety in traffic management systems: a comprehensive survey. Designs 7 (4), 100.

Golshan Khavas, R., Nosrati, M., 2025. Extracting traffic crash information from social media: an LLM-based approach. Available at SSRN 5379743 .

Harne, S., Agarwal, A., et al., 2025. Llm driven text-to-table generation through sub-tasks guidance and iterative refinement. arXiv:2508.08653.

Heidarysafa, M., Kowsari, K., Barnes, L., Brown, D., 2018. Analysis of railway accidents' narratives using deep learning. In: 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE, pp. 1446–1453.

Hosseini, P., Khoshsirat, S., Jalayer, M., Das, S., Zhou, H., 2023. Application of text mining techniques to identify actual wrong-way driving (WWD) crashes in police reports. Int. J. Transp. Sci. Technol. 12 (4), 1038–1051.

Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al., 2022. Lora: low-rank adaptation of large language models. ICLR 1 (2), 3.

Imprialou, M., Quddus, M., 2019. Crash data quality for road safety research: current state and future directions. Accid. Anal. Prev. 130, 84–90.

Jaradat, S., Alhadidi, T.I., Ashqar, H.I., Hossain, A., Elhenawy, M., 2024a. Exploring traffic crash narratives in Jordan using text mining analytics. In: 2024 IEEE 3rd International Conference on Computing and Machine Intelligence (ICMI), pp. 1–6. https://doi.org/10.1109/ICMI60790.2024.10586010

Jaradat, S., Nayak, R., Paz, A., Ashqar, H.I., Elhenawy, M., 2024b. Multitask learning for crash analysis: a fine-tuned LLM framework using Twitter data. Smart Citi. 7 (5), 2422–2465.

Jiang, A.Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D.S., de las, C.D., Hanna, E.B., Bressand, F., et al., 2024. Mixtral of experts. arXiv:2401.04088.

Joulin, A., Grave, E., Bojanowski, P., Mikolov, T., 2016. Bag of tricks for efficient text classification. arXiv:1607.01759.

Kim, J., Trueblood, A.B., Kum, H.-C., Shipp, E.M., 2021. Crash narrative classification: identifying agricultural crashes using machine learning with curated keywords. Traffic Inj. Prev. 22 (1), 74–78.

Kim, M.-J., Kang, J.-S., Chung, K., 2020. Word-embedding-based traffic document classification model for detecting emerging risks using sentiment similarity weight. IEEE Access 8, 183983–183994.

Kullback, S., 1951. Kullback-Leibler divergence. Tech. Rep.

Kumar, S.P., 1968. Estimates of the regression coefficient based on Kendall's tau. J. Am. Stat. Assoc. 63 (324), 1379–1389.

Li, P., Chen, S., Yue, L., Xu, Y., Noyce, D.A., 2024. Analyzing relationships between latent topics in autonomous vehicle crash narratives and crash severity using natural language processing techniques and explainable XGBoost. Accid. Anal. Prev. 203, 107605.
Liu, P., Qiu, X., Huang, X., 2016. Recurrent neural network for text classification with multi-task learning. arXiv:1605.05101.
Loshchilov, I., Hutter, F., 2017. Decoupled weight decay regularization. arXiv:1711.05101.
Mao, Y., Ge, Y., Fan, Y., Xu, W., Mi, Y., Hu, Z., Gao, Y., 2025. A survey on LoRA of large language models. Front. Comput. Sci. 19 (7), 197605.
Meta, A.I., 2024. Llama 3: open foundation and instruction-tuned models. https://www.llama.com/models/llama-3/.
Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013. Efficient estimation of word representations in vector space. arXiv:1301.3781.
Mumtarin, M., Chowdhury, M.S., Wood, J., 2023. Large language models in analyzing crash narratives–a comparative study of ChatGPT, BARD and GPT-4. arXiv:2308.13563.
Nielsen, F., 2019. On the Jensen–Shannon symmetrization of distances relying on abstract means. Entropy 21 (5), 485.
Oliaee, A.H., Das, S., Liu, J., Rahman, M.A., 2023. Using bidirectional encoder representations from transformers (BERT) to classify traffic crash severity types. Nat. Lang. Process. J. 3, 100007.
OpenAI, 2024. Gpt-4o technical report. https://openai.com/index/gpt-4o.
Pennington, J., Socher, R., Manning, C.D., 2014. Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543.
Qawasmeh, B., Oh, J.-S., Kwigizile, V., 2024. Investigating injury outcomes of horse-and-buggy crashes in rural Michigan by mining crash reports using NLP and CNN algorithms. Safety 11 (1), 1.
Qiao, J., Wang, C., Guan, S., Shuran, L., 2022. Construction-accident narrative classification using shallow and deep learning. J. Constr. Eng. Manag. 148 (9), 04022088.
Radja, G.A., Noh, E.-Y., Zhang, F., 2022. Crash investigation sampling system 2020 analytical user's manual. Technical Report.
Raschka, S., 2024. Practical tips for finetuning llms using LoRA (low-rank adaptation). Ahead AI https://sebastianraschka.substack.com/p/practical-tips-for-finetuning-llms.
Ren, X., Gu, H., Wei, W., 2021. Tree-RNN: tree structural recurrent neural network for network traffic classification. Expert Syst. Appl. 167, 114363.
Seo, Y., Park, J., Oh, G., Kim, H., Hu, J., So, J., 2023. Text classification modeling approach on imbalanced-unstructured traffic accident descriptions data. IEEE Open J. Intell. Transp. Syst. 4, 955–965.
Sokolova, M., Lapalme, G., 2009. A systematic analysis of performance measures for classification tasks. Inf. Process. Manag. 45 (4), 427–437.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. Adv. Neural Inf. Process. Syst. 30.
Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q.V., Zhou, D., et al., 2022. Chain-of-thought prompting elicits reasoning in large language models. Adv. Neural Inf. Process. Syst. 35, 24824–24837.
World Health Organization, 2023. Global status report on road safety 2023. ISBN 978-92-4-008651-7 (electronic version). https://www.who.int/publications/i/item/9789240086517.
Xie, K., Yang, D., Ozbay, K., Yang, H., 2019. Use of real-world connected vehicle data in identifying high-risk locations based on a new surrogate safety measure. Accid. Anal. Prev. 125, 311–319.
Xiong, N., Zhou, H., 2024. Crash causing information extraction via text mining techniques: implementation of the Chinese state-related crash narratives. In: 2024 16th International Conference on Advanced Computational Intelligence (ICACI). IEEE, pp. 220–226.
Yang, A., Yu, B., Li, C., Liu, D., Huang, F., Huang, H., Jiang, J., Tu, J., Zhang, J., Zhou, J., Lin, J., Dang, K., Yang, K., Yu, L., Li, M., Sun, M., Zhu, Q., Men, R., He, T., Xu, W., Yin, W., Yu, W., Qiu, X., Ren, X., Yang, X., Li, Y., Xu, Z., Zhang, Z., 2025. Qwen2.5-1m technical report. arXiv:2501.15383.
Zhang, F., 2022. A hybrid structured deep neural network with word2vec for construction accident causes classification. Int. J. Constr. Manag. 22 (6), 1120–1140.
Zhang, X., Green, E., Chen, M., Souleyrette, R.R., 2020. Identifying secondary crashes using text mining techniques. J. Transp. Saf. Secur. 12 (10), 1338–1358.
Zhang, Y., Jin, R., Zhou, Z.-H., 2010. Understanding bag-of-words model: a statistical framework. Int. J. Mach. Learn. Cybern. 1 (1), 43–52.
Zhang, Z., Sabuncu, M., 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. Adv. Neural Inf. Process. Syst. 31.
Zhen, H., Shi, Y., Huang, Y., Yang, J.J., Liu, N., 2024. Leveraging large language models with chain-of-thought and prompt engineering for traffic crash severity analysis and inference. Computers 13 (9), 232.
Zheng, O., Abdel-Aty, M., Wang, D., Wang, Z., Ding, S., 2023. ChatGPT is on the horizon: could a large language model be suitable for intelligent traffic safety research and applications? arXiv:2303.05382.