University *of Ljubljana*
Faculty *of Computer and Information Science*

Automatic generation of Slovenian traffic news for RTV Slovenija

Katarina Gojković, Alen Kurtagić, and Žan Terplan

Abstract

This project explores the automatic generation of traffic news for RTV Slovenija using large language models. We investigate prompt-based and fine-tuned approaches to generate radio-style reports from structured traffic data. After extensive prompt engineering, we fine-tune the GaMS-9B-Instruct model using QLoRA for parameter-efficient training. Our dataset includes structured incident records paired with human-written reports. To enhance output quality, we employ structured JSON inputs and chain-of-thought reasoning. Evaluation reveals that fine-tuned models outperform prompt-based ones in both BLEU scores and factual accuracy, with the standard fine-tuned model yielding the most reliable outputs. Challenges such as inconsistent event hierarchy and ambiguous input data persist, emphasizing the need for improved data quality and representation.

Keywords
Traffic report generation, LLM, Fine-tuning

Advisors: Slavko Žitnik

Introduction

Large Language Models (LLMs) have revolutionized natural language processing (NLP) by achieving state-of-the-art performance across a wide range of tasks, including text generation, translation, and summarization. However, despite their success, LLMs face challenges in less spoken languages with low resources. In this project we utilized this technology to automatically generate traffic reports and tried to overcome its challenges.

Related Work

LLMs have been increasingly used in automated news generation domains, especially in fields like finance, sports, and weather reporting. Despite this, traffic news generation is still a fairly unexplored area. However, some studies have explored this topic, for example, Wan et al. [1] leveraged social media data to complement traditional traffic reporting systems. Ouyang et al. [2] used AI agents to process multi-scale traffic data, conduct analysis and generate suggestions.

Besides traffic news generation, data-to-text generation in general has seen significant advancements with the application of neural network architectures. Several recent studies have explored different methodologies for improving the quality and accuracy of generated text from structured data.

One key advancement in data-to-text generation is the incorporation of entity modeling. Puduppully et al. (2019) [3] proposed an entity-centric neural architecture that dynamically updates entity-specific representations and employs hierarchical attention for generating coherent and factually accurate text. Their approach outperformed existing models on both the benchmark ROTOWIRE dataset and a newly introduced baseball dataset, demonstrating improvements in content selection and text fluency.

Another significant development in the field has been the comparison between pipeline and end-to-end architectures. Castro Ferreira et al. (2019) [4] systematically compared neural pipeline models with end-to-end approaches for generating text from RDF triples. Their findings indicated that pipeline models, which introduce explicit intermediate steps, produce more fluent and accurate text than end-to-end models. Additionally, the pipeline approach was found to generalize better to unseen inputs, reinforcing the notion that structured intermediate representations play a crucial role in high-quality text generation.

Curriculum learning has also been explored to enhance data-to-text generation, particularly in cross-lingual and noisy data settings. Hari et al. (2024) [5] introduced an approach that applies curriculum learning principles by ordering training samples based on an alignment score and using an an-

nealing schedule. Their method resulted in significant improvements in BLEU scores and faithfulness metrics across multiple Indian languages and English.

A recent study by Zhang et al. (2024) [6] proposed optimized LLMs tailored for medical record generation, incorporating data augmentation techniques and domain-specific customization. Their models significantly improved faithfulness scores compared to state-of-the-art general models, demonstrating the importance of high-quality annotated data in generating accurate medical records.

Overall, these studies highlight various approaches to improving data-to-text generation, from entity modeling and pipeline architectures to curriculum learning and domain-specific optimizations.

Data

Dataset

The dataset used for fine-tuning was provided by RTV Slovenija and consists of structured traffic data paired with manually written broadcast reports. It comprises three Excel sheets, each containing approximately 50,000 entries for the years 2022, 2023, and 2024. Each sheet documents real-time traffic incidents using 27 attributes, including affected roads, incident types, severity, and timestamps. Additionally, the dataset contains around 28,000 textual traffic reports composed by human operators for radio broadcasting. These reports aim to summarize the traffic state in a concise and informative manner.

Preprocessing

To prepare the dataset for fine-tuning, we first examined its structure and addressed several inconsistencies. The main finding about the dataset was that while reports are marked with Slovenian local time (UTC +2), eateries in excel are marked with UTC time. After this finding we were able to successfully pair eateries with the reports. After paring data from excel with groundtruth reports we removed embedded markup language. Furthermore, we restructured the data based on semantic categories. This restructuring facilitates clearer representation provides a foundation for generating higher-level abstractions, such as JSON objects describing the state of Slovenian traffic.

Initial Approach and Ideas

Our initial attempt to generate structured traffic reports relied on prompt engineering. We began by testing GaMS-1B, GaMS-9B, Llama 3.1 8B, and mt0-large using a series of prompts. The initial prompts included simple questions about the weather or traffic, as well as similar questions paired with structured data, instructing the models to use the provided data when responding. While all models produced responses, they frequently included fabricated information. These initial tests revealed notable differences in the models' ability to process and apply the input data effectively. **GaMS-1B and GaMS-9B** produced verbose and often irrelevant responses, with GaMS-9B occasionally generating natural-sounding but fabricated content. **Llama 3.1 8B** demonstrated some precision but frequently generated generic or grammatically incorrect Slovenian outputs. Lastly, **mt0-large** struggled with coherence and frequently repeated phrases, particularly in Slovenian.

Following these observations, we experimented with seven distinct prompt designs to improve the structure and coherence of the outputs, utilizing data from the accrual dataset provided by RTV Slovenija. These prompts ranged from basic formats to highly detailed instructions emphasizing event prioritization, road terminology, and concise phrasing. Testing across structured and semi-structured data revealed that while strict formatting improved model performance, significant challenges—such as omissions, fabricated information, and language inconsistencies—remained, highlighting the need for fine-tuning.

In retrospect, a critical oversight during this phase was our use of the standard GaMS models instead of their Instruct variants, which are specifically designed for structured tasks. Upon realizing this, we tested GaMS-9B-Instruct and ultimately selected it for fine-tuning.

Structured Inputs

While the original CSV data was comprehensive, it was also messy, inconsistent, and based on outdated conventions. To enable more robust and modern data handling, we adopted JSON as the standard input format. JSON provides a clean and structured way to represent state, and is widely used in contemporary machine learning pipelines. In addition to using the unstructured text reports, we manually generated structured JSON counterparts for each example. This enables our system to accept both unstructured natural language inputs and structured data formats and transition between the two.

Chain-of-Thought Reasoning

To encourage more accurate and grounded outputs, we incorporated chain-of-thought (CoT) reasoning into the dataset. Prior work has shown that explicitly modeling intermediate reasoning steps can reduce hallucinations and improve factual consistency in large language models. To this end, we generated synthetic `reasoning` fields in the training JSON, describing intermediate inferences about the traffic state.

These fields were then integrated into the training prompts using a dedicated marker (e.g., *"Thinking:"*), inserted between the input and the final output. By exposing the model to this intermediate step, we aimed to help it internalize the structure of logical inference and better justify its output in tasks involving complex traffic states.

Implementation

Setup

To fine-tune our model, we first ensured access to GPUs with sufficient VRAM. This was accomplished using the Arnes

HPC cluster, which provided access to NVIDIA A100 (40GB VRAM) and H100 (80GB VRAM) GPUs. For a consistent and reproducible environment, we utilized a Singularity container. Singularity is particularly suited for HPC environments, offering isolated environments similar to Docker, but with better support for cluster-based workflows.

QLoRA

With the environment prepared, we employed Quantized Low-Rank Adaptation (QLoRA) to fine-tune our large language model, implemented by the PEFT (Parameter-Efficient Fine-Tuning) Python package. QLoRA is a parameter-efficient fine-tuning technique that combines two key strategies:

- **Double Quantization:** Model weights are first quantized to 4-bit representations using the NormalFloat4 format, significantly reducing memory usage. Rather than storing full-precision values, each weight is mapped to a codebook index:

  $$q(w) = \arg\min_i |w - C_i|$$

  where C is a codebook of 4-bit quantized float values. To further reduce memory, the codebook itself is quantized using a second codebook:

  $$C_i = C'_j \text{ where } j = \arg\min_j |C_i - C'_j|$$

  This process allows the storage of codebook entries as low-bit indices into a smaller, full-precision codebook C', thus enabling compression of both weights and codebooks.

- **Low-Rank Adaptation (LoRA):** Instead of updating all parameters in a large neural network, LoRA introduces low-rank matrices $A \in \mathbb{R}^{d \times r}$ and $B \in \mathbb{R}^{d' \times r}$, where $r \ll \min(d, d')$. These matrices adapt the frozen weights:

  $$\Delta W \approx AB^T$$

  and the updated weight is computed as:

  $$W' = W + \Delta W = W + AB^T$$

  This significantly reduces the number of trainable parameters while still allowing effective adaptation.

QLoRA allows us to efficiently adapt a 9-billion parameter model using a single A100 GPU. The 4-bit quantization, coupled with double quantization and other advanced features ensures memory efficiency and training stability.

Hyperparameters

To train the model efficiently using QLoRA, we carefully selected training hyperparameters that balance resource constraints and convergence quality. Our setup utilizes gradient accumulation, enabling fine-tuning even on a single A100 GPU.

Hyperparameter	Value
Base model	GaMS-9B-Instruct
LoRA rank	128
LoRA alpha	256
LoRA dropout	0.1
Quantization	4-bit
Compute dtype	bfloat16
Batch size	1
Gradient accumulation steps	64
Learning rate	5×10^{-5}
Weight decay	0.02
Optimizer	Paged adam
Learning rate scheduler	Linear
Warmup steps	6
Training epochs	10

Table 1. Key hyperparameters used for QLoRA fine-tuning.

Implementation Details

To focus training on the intended output (e.g., a JSON or traffic report), we masked all non-target tokens in the label tensor using `-100`, which excludes them from loss computation in the *transformers* library.

We tokenized both the full input (instruction + input + target) and a truncated version containing only the instruction and input. Tokens belonging to the truncated version were then masked in the labels, ensuring the model learns solely from the output portion without being influenced by the instruction or input.

Evaluation

Empirical Evaluation

To evaluate our two approaches, we compared them with zero-shot and two-shot methods, using the most effective prompt identified during the prompt engineering phase. The evaluation involved manually reviewing the outputs to assess their language quality, adherence to the input data, alignment with the ground truth reports, and compliance with RTV Slovenija's report-writing guidelines.

The zero-shot and two-shot methods were initially evaluated to establish a baseline for comparison. The zero-shot method often failed to maintain proper sentence structure and formulation, instead resorting to listing events with little mention of consequences. Occasionally, it adhered to sentence structures, but only when these were directly present in the input data. A major limitation of the zero-shot approach was its tendency to omit data, fabricate information, or produce incoherent reports. Despite explicit instructions in the prompt on event hierarchy and sentence formulation, these were frequently ignored. Moreover, this method sometimes redundantly repeated events, particularly when multiple consequences were associated with a single event. The two-shot method demonstrated notable improvements in sentence structure and formulation. However, it continued to generate incomplete or nonsensical reports and often inventing events.

Fine-tuning the model resulted in significant improvements. Outputs exhibited better sentence structure and formulation, closely resembling those of the groundtruth reports, but with occasional deviations. For example, while the fine-tuned model sometimes included locations of events, such as specific highway exit, but sometimes highway section, the guidelines stat that road sections should be used. Groundtruth reports themselves were inconsistent in this regard, potentially confusing the model. While the fine-tuned model was successful in determining the hierarchy of the events such as accidents and traffic jams, which are the most often occurred ones, it occasionally misjudged other events hierarchies, favoring the sequence of events in the input data. Also it sometimes omitted consequences if they were not explicitly stated, as well as the detour information. Despite these issues, the main benefit of the fine-tuned model is that it did not fabricate data.

The Chain-of-Thought (CoT) approach captured nuanced patterns (e.g., border directions) but overall performed worse than the standard fine-tuned model. The CoT method often fabricated event details or even entire events, likely due to its step-by-step reasoning, amplifying ambiguity in sparse inputs, causing the model to ”fill gaps” incorrectly.

A persistent challenge across all methods was handling missing or inconsistently presented data in the input. Ground truth reports often included details absent from the input data or rephrased time-bound restrictions (e.g., “today” or “yesterday” instead of specific dates and hours). Addressing this issue might require incorporating explicit report timestamps into the input data. Overall, we believe that the performance of all models, especially the fine tuned ones can be improved a lot by improving input data.

Evaluation using metrics

To quantitatively assess performance, we use a customized BLEU metric with tuned weights and smoothing to reflect the structure and meaning of traffic reports. Table 2 summarizes BLEU scores across different evaluation settings.

Method	BLEU Score
Prompt-based (zero-shot)	0.192
Prompt-based (two-shot)	0.284
Fine-tuned	0.343
Fine-tuned (Chain-of-Thought)	0.310

Table 2. BLEU scores for different evaluation methods. Prompt-based methods rely only on prompting without parameter updates; fine-tuned methods are trained on the dataset.

The results show a clear advantage for fine-tuned models over prompt-based methods. The highest score is achieved by the fine-tuned model without additional reasoning steps, confirming that direct learning from structured examples leads to more reliable, concise, and task-specific outputs. It consistently includes correct phrases (e.g., ”zaradi nesreče je zaprt prehitevalni pas”), proper road names (e.g. ”štajerska avtocesta”) and rarely hallucinates or omits important details. This strong alignment with reference texts boosts both n-gram overlap and BLEU score. The fine-tuned CoT model performs slightly worse, likely due to verbosity and overstructuring. While it improves clarity in complex cases (e.g., multi-incident reports), it sometimes adds redundant phrases or deviates slightly in wording (e.g., ”trčenje” instead of ”prometna nesreča”), which lowers precision. The two-shot method benefits from imitating formatting in the examples but remains inconsistent in terminology and may omit key details like directions (e.g. ”proti Ljubljani”). The zero-shot method performs worst, frequently violating domain conventions, hallucinating content, and introducing irrelevant sections. It struggles with both structure and keyword alignment, resulting in the lowest BLEU score.

We applied weights of (0.3, 0.4, 0.2, 0.1), which assign the highest importance to 2-gram and 1-gram matches respectively, giving more importance to key words and short phrases. This helps the evaluation focus on important content like location names (e.g., ”Ljubljana”, ”Karavanke”), road types (e.g., ”gorenjska avtocesta”, ”hitra cesta”), and traffic-related terms (e.g., ”nesreče”, ”zaprt pas”). The smaller weights on 3-gram and 4-gram matches allow some flexibility in word order and phrasing. This is important for our task, where different sentence structures can still carry the same meaning.

We used *method3* from the NLTK’s SmoothingFunction, which helps prevent a BLEU score of zero when there is no overlap in higher-order n-grams. It does this by adding small values to the n-gram counts. This makes it more suitable for short texts than method4, which adds smoothing uniformly, and method5, which is too forgiving and can inflate scores for incomplete output. This balance is useful for our setting, where correct reports may still phrase things differently from the reference output. Using smoothing helps ensure that these cases are not unfairly penalized.

Conclusion

This project demonstrates that fine-tuning the GaMS-9B-Instruct model on structured traffic data leads to noticeable improvements in the quality of automatically generated traffic news compared to prompt-based methods. The fine-tuned model tends to produce more concise and consistent reports with better use of terminology. However, evaluation results indicate that there is still considerable room for improvement, as issues like occasional inaccuracies and inconsistent phrasing persist. Incorporating CoT reasoning shows potential for handling complex cases but sometimes results in verbosity and minor errors. Overall, these findings highlight the benefits of domain-specific fine-tuning and structured input formats, as well as the efficiency of techniques like QLoRA for adapting large models on limited hardware. Future research could explore further optimization of output clarity and precision, as well as expanding model robustness across various traffic scenarios.

References

[1] Xiangpeng Wan, Michael Lucic, Hakim Ghazzai, and Yehia Massoud. Empowering real-time traffic reporting systems with nlp-processed social media data. *IEEE Open Journal of Intelligent Transportation Systems*, 1:159–175, 01 2020.

[2] Jinhui Ouyang, Yijie Zhu, Xiang Yuan, and Di Wu. Trafficgpt: Towards multi-scale traffic analysis and generation with spatial-temporal agent framework, 2024.

[3] Ratish Puduppully, Li Dong, and Mirella Lapata. Data-to-text generation with entity modeling, 2019.

[4] Thiago Castro Ferreira, Chris van der Lee, Emiel van Miltenburg, and Emiel Krahmer. Neural data-to-text generation: A comparison between pipeline and end-to-end architectures, 2019.

[5] Kancharla Aditya Hari, Manish Gupta, and Vasudeva Varma. Curriculum learning for cross-lingual data-to-text generation with noisy data, 2024.

[6] Xuanyi Zhang, Genghong Zhao, Yi Ren, Weiguang Wang, Wei Cai, Yan Zhao, Xia Zhang, and Jiren Liu. Data augmented large language models for medical record generation. *Applied Intelligence*, 55(2):1–22, 2025.