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

Automatic generation of Slovenian traffic news for RTV Slovenija

Janez Tomšič, Žan Pušenjak, Matic Zadobovšek

Abstract

In this report we present our initial analysis performed on the dataset of RTV Slovenija traffic news. We describe related work, especially that on prompt engineering and fine-tuning of LLMs, which will be especially important for our project. Lastly, we outline the initial idea of our approach and evaluation methods - including automated (BERTScore) and manual human evaluation (factual accuracy) methods.

Keywords

Traffic news generation, Prompt engineering, Supervised fine-tuning, Natural language generation

Advisor: Slavko Žitnik

Introduction

Generating traffic news using the modern Natural Language Processing (NLP) techniques involves transforming raw traffic data into structured reports, that are in a form that is suitable to be broadcast on radio. Our main goal is to reduce manual workload for editors, and to improve the quality and consistency in prepared reports.

There are a few challenges that arise, mainly in following the formatting rules and correctly reflecting traffic conditions (e.g. road names, directions) - this is what the model has to learn to replicate. To address these conditions, we will use pre-trained language models, structured prompting, and fine-tuning. We combine all these techniques together in order to be able to automatically generate high-quality Slovenian traffic reports.

Slovenian traffic news dataset

In this section we present the provided dataset which consists of two parts. The first part provides tabular data that was retrieved from the Slovenian website **promet.si** [1], which provides real-time traffic updates and road conditions. The second part is a corpus of traffic reports that were manually written to be read on the radio broadcast. Both parts span data from 2022 to 2024. We also worked with a set of provided instructions which outline the expected format of generated traffic news reports and help with consistency.

Tabular traffic updates data structure

Traffic updates are stored in the form of an Excel file, where each row describes a traffic event. Each of the entries has multiple fields that give a structured overview of what has happened at certain time. The main fields we are focusing on are:

- **ID**: A unique identifier.
- **Date and time**: The UTC timestamp indicating when data was obtained.
- **Category-specific**: Various columns related to specific traffic-related topics, which include:
  - **Accidents**: Information about road accidents.
  - **Traffic jams**: Reports on traffic congestion.
  - **Weather**: Weather conditions that affect road traffic.
  - **Road work**: Information about road construction and closures.
  - **Obstacles**: Reports of road blockages.
  - **International information**: Updates on border controls and other restrictions.
  - **General notices**: General traffic-related information.

Corpus with traffic reports

The corpus contains 28.037 traffic reports which are stored in rich text format (*rtf*) files. In our initial analysis we observed the following specific structure of these reports.

- **Title**: Either *Nujna prometna informacija* (Urgent traffic information) or *Prometne informacije* (Traffic information).
- **Date and time**: The local date (DD.MM.YYYY) and time (HH:MM) of the report.
- **Program**: Program in which the report is broadcast.
- **Traffic data**: A structured list of traffic incidents and disruptions.

This structure was not always strictly followed, which made automatic parsing of records more difficult - we did not manage to parse approximately 1.3% of documents which we excluded from our initial analysis. Automatically determining the date and time of report from the documents is crucial to be able to match written reports with events in the tabular data file.

We also observed some anomalies in the corpus. Some documents were empty and some only had a title and a date, but no content. We found some documents that are not even related to traffic reports - one was a weather report, and one was a general news report. Additionally, we noticed that some reports were not always fully up-to-date. This was most common when a traffic update occurred less than 5 minutes before being read and is most likely a consequence of manual preparation of reports.

Traffic report generation guidelines

Every traffic incident that we report on has to follow a structured formulation:

- **Option 1**: Road name and direction + reason + consequence.
- **Option 2**: Reason + road name and direction + consequence.

We have two categories of the traffic reports based on the information they contain: **regular** and **urgent** traffic information. Regular traffic information provide updates on general road conditions (congestion, roadworks) and are broadcast every 30 minutes. Urgent traffic reports are read more frequently and include information about motorway closures and severe congestion, which critically impact the traffic flow.

Traffic events are represented in a form of an event hierarchy, which tells us which ones are to be prioritized:

1. **Wrong-way driver alert**.
2. **Full motorway closure**.
3. **Accident with congestion on a motorway**.
4. **Congestion due to roadworks on a motorway**.
5. **Closure of a regional road due to an accident**.
6. **Accidents on motorways and other roads**.
7. **Broken-down vehicles**.
8. **Animal on the road**.
9. **Objects on a motorway**.
10. **Roadworks in high-risk areas**.
11. **Severe congestion at border crossings**.

There are a few other rules and naming conventions that are more specific and are dependent on the type of event we are reporting on (high-priority events must be repeated every 15-20 minutes, congestion is reported only if it exceeds 1 km in length, indicate road sections with expected closures or delays). Note that some of the naming conventions for reports differ from naming conventions on *promet.si*, where the tabular data was retrieved from. This will need to be taken into account in our analysis.

Related work

We looked at the popular approaches for text style generation adaptations for language models. The two most prominent techniques are *prompting* and *fine-tuning*. Both try to achieve better desired results either by guiding the model to generate data according to determined rules or to change the models structure in accordance with the desired outcome.

The available research suggests that *prompting* is overall less efficient than *fine-tuning* [2] but should not be discarded. Using the right prompt can substantially increase the models desired performance [3]. Additionally smaller and medium-sized models benefit more from prompting than from simply increasing the model size [2]. The prompt should be crafted according to the desired result [4], so it is important to not rush the process of finding the best performing one. Taking that into account we conducted that crafting a good prompt for the style of the output is essential for a good generation of the output data.

If *prompting* is enhanced by using a *fine-tuned* model, we can achieve the best possible results. For our use case we would use *fine-tuning* to make the model adapt the generated content style to the form that is used on RTV. The problem with *fine-tuning* is lack of tuning data and computing power limitations [5]. But apart from those problems the approach can heavily change and customize the generation style with noticeably better results [6].

Attempts at creating an automated traffic incident reporting were already made [7] so looking at the findings and shortcomings of existing systems will allow us to even further improve the results. Since the traffic data and the desired report structure vary greatly even between news channels, let alone countries, it is impossible to create an out-of-the box solution for news reporting.

Methodology

In this section, we describe our initial approach on how we could potentially tackle automatic generation of Slovenian traffic news. Our goal is to develop a system that would be able to convert data from initial Excel files into structured reports that follow predefined formatting rules, and are ready for the radio presenters.

Pre-trained language models, prompt engineering, fine-tuning

As a first step in our development, we will test out *pre-trained language models* that we will run on the Arnes HPC cluster [8]. Since one of the requirements is to do text generation in Slovene, we will also try to explore some of the Slovene models. Because of the computational constraints, we will first try to experiment with smaller models, as they are more efficient to run and fine-tune as well. Depending on the performance of these smaller models, we will consider using larger models to get improved quality of generated reports. Models that we plan to test include:

- **SloT5 (T5-sl-small, T5-sl-large)** [9]: Sequence-to-sequence model for text generation.
- **GaMS (GaMS-1B, Generative Model for Slovene** [10]): A larger model fine-tuned on Slovene.
- **Multilingual models (LLaMA-2 7B [11], Mistral 7B [12])**: Possibly limited in performance in Slovene, but can serve as a good baseline for comparison.

Prompt engineering is the process of preparing input instructions that help navigating model's response to what we want it to be like [13]. In our case, we can provide a formatted input template, where we can explictly define all needed report sections according to the provided instructions and maintain consistenty in reporting. Another advantage is that it does not require additional model training, but might still have limitations in accuracy, which is what motivates us to use fine-tuning approach.

In order to get better results, we will use *supervised fine-tuning*, which involves taking a pre-trained model and then further training it on a prepared dataset. In our case we will be using already prepared traffic reports paired with structured input (raw traffic events). With this approach we aim to improve performance over using simple prompting, as the model will be able to learn from a larger dataset of prepared examples, instead of just relying on the general training data.

Evaluation

In this section we outline our initial ideas for evaluating the generated traffic reports. We plan to use automated evaluation methods as well as manual human evaluation, especially for evaluating structure of the generated reports, their relevance, and factual accuracy.

For automatic evaluation, we are considering the following methods:

- **ROUGE** [14] is commonly used for evaluating summarization methods. It estimates how much of the reference text is captured by the generated text using either N-grams, skip-grams, or longest common subsequences, depending on the variant used.
- **BERTScore** [15] compares semantic similarity of texts rather than just analysing word overlap. It uses contextual embeddings and cosine similarity. We are considering using *SloBERTa* [16], a RoBERTa-based contextual embedding tuned for the Slovenian language.

The biggest advantage of these methods is that they do not require manual human review. However, they are not able to directly check if the structure of a report is suitable, or whether the reported events in fact occurred. They also can not explicitly check if reporting conventions were taken into account. Additionally, as observed in the data exploration phase, some ground-truth reports do not contain the latest traffic information, which might cause our models to be underestimated.

Because of this, we are also considering manual review of generated reports, at least on a subset of the data. Another idea we have is to attempt to define an automated evaluation method that would be better suited for our task.

When defining the evaluation procedure, additional criteria should also be taken into account. The detection of urgent traffic events should also be evaluated, perhaps some events need to have a higher weight assigned to them - for example, a wrong-way driver on the highway is much more crucial to be reported than a congestion at a border crossing.

References

[1] DARS d.d. Prometno informacijski center za državne ceste. https://www.promet.si. Accessed: 20. 03. 2025.

[2] Martina Toshevska and Sonja Gievska. Llm-based text style transfer: Have we taken a step forward? *IEEE Access*, 13:44707–44721, 2025.

[3] Jia He, Mukund Rungta, David Koleczek, Arshdeep Sekhon, Franklin X Wang, and Sadid Hasan. Does prompt formatting have any impact on llm performance?, 2024.

[4] J.D. Zamfirescu-Pereira, Richmond Y. Wong, Bjoern Hartmann, and Qian Yang. Why johnny can't prompt: How non-ai experts try (and fail) to design llm prompts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, New York, NY, USA, 2023. Association for Computing Machinery.

[5] Yuchen Xia, Jiho Kim, Yuhan Chen, Haojie Ye, Souvik Kundu, Cong Callie Hao, and Nishil Talati. Understanding the performance and estimating the cost of llm fine-tuning. In *2024 IEEE International Symposium on Workload Characterization (IISWC)*, pages 210–223, 2024.

[6] Xinyue Liu, Harshita Diddee, and Daphne Ippolito. Customizing large language model generation style using parameter-efficient finetuning, 2024.

[7] Anna Jansdatter Bjørgo and Mads Lundegaard. Automating the process of traffic incident reporting. Master's thesis, NTNU, 2024.

[8] Slovenian National Supercomputing Network. Arnes hpc cluster. https://www.sling.si/en/arnes-hpc-cluster/. Accessed: 21. 03. 2025.

[9] Matej Ulčar and Marko Robnik-Šikonja. Sequence-to-sequence pretraining for a less-resourced slovenian language. *Frontiers in Artificial Intelligence*, 6:932519, 2023.

[10] Domen Vreš, Martin Božič, Aljaž Potočnik, Tomaž Martinčič, and Marko Robnik-Šikonja. Generative model for less-resourced language with 1 billion parameters. *arXiv preprint arXiv:2410.06898*, 2024.

[11] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[12] Gueyoung Jung, Matti A Hiltunen, Kaustubh R Joshi, Richard D Schlichting, and Calton Pu. Mistral: Dynamically managing power, performance, and adaptation cost in cloud infrastructures. In *2010 IEEE 30th International Conference on Distributed Computing Systems*, pages 62–73. IEEE, 2010.

[13] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM computing surveys*, 55(9):1–35, 2023.

[14] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[15] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

[16] Matej Ulčar and Marko Robnik-Šikonja. Slovenian roberta contextual embeddings model: Sloberta 2.0. *clarin.si*, 2021.