# Special Topics in Computational Linguistics
# Romanian-to-Romani Translation

**1st Semester of 2024-2025**

**Zibileanu Sabin** **Grigore Dragoș - Gabriel** **Vasilescu Costin - Tiberiu** **Handac Alexandru - Radu**

## Abstract

We present the steps and the tools necessary to build a machine translation deep learning model for Romanian to Romani. Even though it is a low - resource language, the Romani language is a very popular one around the world and this project represents a starting point for future research and improvements on building machine translation models with this language.

## How to use the project

1. in the jupyter notebooks available in the repository, all the important parts of the project can be found: data acquisition, tokenization, training, loading the data, etc.

2. knowing the fact that we are working with a low - resource language (Romani), we decided that it was best to upload the whole project for future research and why not, future usage.

3. the dataset is structured in two txt files: romani.txt and romanian.txt containing the acquired samples and also 2 tokenizer directories generated from creating a custom Romanian-to-Romani tokenizer with a base MarianMT Model and another directory containing an extended pre-trained tokenizer.

4. there are also 2 directories in the repository: "data" and "tokenizer", where you can find the code for obtaining the dataset and the tokenizer

5. if you want to reproduce the results, you can run the notebook in the main branch (MTProject.ipynb).

1. **Data Acquisition**

   working with a low-resource language, we took into account the fact that we had to build a corpora from scratch almost.

   however, we found a corpus online with around 7900 samples from Romanian to Romani **from HelsinkiNLP** (Christodoulopoulos and Steedman, 2014), which to our knowledge is in the Carpathian Romani dialect.

   the available corpora was a great starting point, as it provided a considerable amount, but we wanted to add more data.

   to add more data, we scraped samples from **Glosbe** as it provided Carpathian Romani samples translated into Romanian.

   we chose the dialect Carpathian Romani as it is one that provides the most examples.

   because Glosbe is a dictionary website, we had to search for keywords first in order to scrape the resulted data. Those keywords can be: conjuctions, verbs, adverbs, prepositions etc.

   it is important to mention that some samples were added manually, because of the fact that certain keywords had multiple available phrases split into different pages and we could not load more pages with our scraper.

   next on, we manually gathered a set of samples from different dictionaries, conversational guides, and a Romani course support. We added those samples manually, due to the fact that we mostly chose phrases and also the dialectal mix between the already existing samples in Carpathian Romani and these samples in Kalderash Romani.

   a high mix of dialects between the samples could negatively affect the performance of the model.

2. **Tokenization**

   the Romanian-to-Romani tokenizer was created using SentencePiece, with a vocabulary size with respect to the size of the unique tokens in the dataset, which was around 10000.

after it was created, the tokenizer was loaded using MarianTokenizer (Junczys-Dowmunt et al., 2018)

the second tokenizer was an extended version of a pre-trained tokenizer from HelsinkiNLP: roa - en.

3. **Evaluation**

   for a low resource language like Romani, human evaluation would be the best.

   in our project, however, we present the results obtained with the BLEU metric (Papineni et al., 2002).

# 1 Introduction

The Romani language, although a low - resource one is, according to some estimates in the top 3% of the world's most spoken languages, containing multiple dialects: Carpathian, Kalderash, Balkan etc. Considering the fact that there are no available Romanian - Romani machine translation models (to our knowledge), we:

- tried building a machine translation model for Romanian to Romani
- our approach involved using the MarianMT model (Junczys-Dowmunt et al., 2018) and using 2 different variatons when it comes to the initialization of the model. Precisely, we used a MarianMT model with our custom tokenizer and a pre-trained MarianMT model with the already existing tokenizer that we combined with the custom one. We extended the already existing pre-trained tokenizer with our custom made Romanian to Romani tokenizer.
- we chose this approach because the MarianMT model supports multiple romance languages and it is also a very well structured Sequence2Sequence model for machine translation.
- to our knowledge there are no papers covering this task.
- During this project we learned:
  - more about augmenting techniques for datasets
  - how a tokenizer can be built
  - how a Sequence2Sequence model works
  - more about the metrics used to evaluate a machine translation system
  - how to build a corpora almost from scratch
  - how to use large language models for machine translation tasks
  - how can preprocessing tehcniques can affect the performance of the model
  - more about tokenization strategies
  - different types of machine translation

# 2 Approach

This section covers our approach on this project divided into detailed steps

1. Firstly we gathered the data: merged 2 datasets : one built from scratch using Glosbe, dictionaries, conversational guides and a romani course and the other one **from HelsinkiNLP**. Afterwards, the dataset was cleaned for duplicate samples.

2. Moving on, we felt that it was necessary to try and use some data augmentation techniques, especially working with a low resource language. The techniques that we used were: **back translation**, where the Romanian sentence was translated into English and then back to Romanian ; **sentence insertion** where we inserted a Romanian token and a Romani token at the begining of a sentence, "Po del chavo - " for Romani and "Începutul propoziției -" for Romanian. Finally, the last technique was: **random char insertion** where a random character was introduced in a word from the sentence, simulating a typo. The augmentation was made on 15% of the training set.

3. The training proces took around 1 to 2 hours depending on the model and was done in Google Colab with a T4 GPU.

4. The deep learning tool used, was: MarianMT Transformer Model. For augmentation (random char) we used nlpaug, and Google - Translator for back translation

5. The results can be seen in Table 1

6. Afterwards, we decided to use a large language model to see what performance it can give.

| Results | | | | |
|---|---|---|---|---|
| **Data** | **Model** | **BLEU** | **Epochs** | **Tokenizer** |
| Augmented | Base | 1.969900 | 10 | Custom |
| Augmented | Pre-trained | **6.726000** | 3 | Extended |
| Non-augmented | Base | 0.823531 | 10 | Custom |
| Non-augmented | Pre-trained | 2.415000 | 3 | Extended |

Table 1: BLEU scores for augmented and non-augmented data using different models

It can be seen that the best performance is obtained with the pretrained model and the augmented data. With that in mind, we decided to furtherly enhance our experiments and see if the results can be improved in different environments, those being: using the custom created tokenizer with the base model and with the removal of the diacritics for the Romani data and using the already existent tokenizer for the pre-trained model with no additional changes and the removal of the diacritics for the Romani data.

The new results can be seen in Table 2

| Results | | | | |
|---|---|---|---|---|
| **Data** | **Model** | **BLEU** | **Epochs** | **Tokenizer** |
| Augmented and preprocessed | Base | 2.2694901 | 10 | Custom |
| Augmented and preprocessed | Pre-trained | **16.8106000** | 3 | Non-modified |

Table 2: BLEU scores for augmented and preprocessed data using different models

## 3 Limitations

The most notable limitation that can be seen is the quality of the results. This is caused by the low amount of data that was gathered. In order to build a high - standard machine translation model for Romanian to Romani, a large corpus is needed first (millions of sentences) that can be verified by a Romani speaker. It can be built from books, dictionaries, conversational guides and even songs why not. While working on this project, we found that there are not that many Romani books, that are translated in Romanian (excluding dictionaries).

## 4 Conclusions and Future Work

This work presented the steps and the tools necessary to build a machine translation model for a low resource language. We liked this project because it allowed us to build something from the ground up and learn more about machine translation in the process. We believe that this project will surely leave room for improvement for the future and is a starting point for this task.

### 4.1 References

## References

Christos Christodoulopoulos and Mark Steedman. 2014. A massively parallel corpus: the bible in 100 languages. *Language Resources and Evaluation*, 49:1–21.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Aji, Nikolay Bogoychev, André Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in c++. pages 116–121.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL ’02, page 311–318, USA. Association for Computational Linguistics.