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

# Machine translation

Januš Likozar, Janja Koželj

**Abstract**

**Keywords**
neural machine translation, Slovenian-English, domain specific translation, literary texts

*Advisors: Slavko Žitnik*

## Introduction

Machine translation is a field that connects computer science with linguistics. The most basic approach is replacing words one by one with its translation, but these translations are rarely good. More complex methods such as statistical machine translation use probabilistic models to generate translations based on the analysis of corpora from both languages.

In this project we use neural machine translation (NMT) methods that use artificial neural network to generate translations. Compared to more basic approaches to translation it does not need any knowledge about the languages, like grammatical structure or similar. NMT methods use one neural network that is trained end to end on source and target language corpora. The corpora used has to be aligned, so that sentences match between both languages.

Earlier NMT methods use encode-decoder recurrent neural networks (RNN), which account for sequential data, so this style of networks can also be used for sequence to sequence prediction tasks in other fields. These methods use multiple Long Short-Term Memory (LSTM) blocks to encode input vectors and decode target output vector, one such method is described in [1]. It maps input sequences to fixed internal representation, but still manages to translate longer sentences as well. To address problems with rare words and lack of robustness, in [2] they add attention layers to similar encode-decoder architecture as described before.

Latest state of the art results use transformer based neural networks, which rely on self attention mechanisms to compute internal representation of inputs and outputs [3].

## Methods

There are many NMT frameworks that make training and inference of translation models easier. Below we list some of the most widely known ones:

- **MarianNMT** [4] is NMT toolkit written in C++. It is made to be fast and self-contained. It offers training and translating using some common models, but has a bit lacking documentation.
- **OpenNMT** [5] is a NMT toolkit that has TensorFlow and PyTorch version. It supports translation, summarization and other tasks, and includes tools for fast inference and tokenization. It is made to be easy to use with many examples and lots of documentation.
- **Trax** [] is a library for deep learning using TensorFlow that supports training of varous common types of networks and reinforcement learning algorithms for translation and other natural language processing tasks. It is a successor to seq2seq toolkit [6], but does not have the same extensive support and documentation yet.
- **Fairseq** [7] is a sequence modeling toolkit written using PyTorch that supports translation, summarization, language modeling and other text generation tasks. It offers lots of pretrained models, models and examples for lots of specific tasks and has extensive documentation and support.
- **Transformers** [8] is a natural language processing library that uses PyTorch, TensorFlow and Jax. It contains a lot of pretrained models, model configurations and other tools for various tasks. It offers extensive documentation and support with many examples, but it is not command line end to end tool.

Considering our use case we chose to use OpenNMT toolkit, the PyTorch version, because it is easy to use, has

many examples and good documentation. We also used Transformers library for tokenization.

OpenNMT toolkit can be used as a python library or from the command line using scripts and configuration files. There are three main features, building vocabulary, training the model and model inference (translation of text). For scripts there has to be a configuration file (or many command line arguments), which defines all the parameters. For use in OpenNMT we have source (in our case English) and target (Slovenian) text files of aligned sentences, with one sentence per line.

When building vocabulary we can set maximum number of tokens in vocabulary. We can have separate vocabulary for each language or one that is shared by both. By default it takes input as words separated by spaces, but we can set it to use a tokenizer. There are two tokenizers available, byte-pair encoding (BPE) and SentencePiece.

For training we can define custom models, there are many available architectures for encoder and decoder parts, like RNN, CNN (convolutional neural networks), Transformers and other variations, we can also define attention and self attention blocks. We can also set many parameters for each building block in architecture. It supports use of pretrained embedding and training on multiple gpus.

For translation we can choose any trained model to generate translations of a given text.

### Data

We decided to make a NMT translator for literary texts. There are multiple approaches how to train a model that is domain specific [9]. One is to train on only domain specific data, which is difficult if there is little appropriate available corpora, so the quality of model depends on chosen domain. Another approach is to train on all available data that is not necessarily domain specific and then fine tune the model on specific corpora. Since there is not much already prepared available corpora of literary texts to train model from scratch, we used more general corpora for inital training.

Translation of literary texts is a difficult task since the source text is lexically rich and literal translation is not necessarily the best. [10, 11, 12]. In case of translation from English to Slovenian there is also a challenge of Slovenian being morphologically complex language and the small number of available aligned corpora of these two languages [13].

We used EUparl (general, a bit political text), Wikipedia (general texts), TRANS3 (articles about various topics) and translated handbooks (technical guides) as general corpora. For domain specific corpora we use SPOOK (translated books) and Orwell's 1984 (from ELAN dataset). Additionally we tried to use TC3 (EUparl, OpenSubtitles 2018, EMEA - European Medicines Agency, DGT - translation memories and ELRC - statistical reports) as a general corpora.

SPOOK contains 67920 sentence pairs, and TC3 contains 24419755 sentence pairs.

### Data preprocessing

First we converted all data to a format that is expected from OpenNMT toolkit training script, two text files, one for each language, with one sentence per line. We checked all matched sentences so that there were no missing ones in any language, and checked that they matched. Then we cleaned the text so that we had uniform symbols across different corpora, for example we switched occurrences of *&quot* with *"* and similar.

### Tokenization and Vocabulary

We have tried different tokenization methods, like WordPiece and SentencePiece [2]. Both are subword tokenization algorithms, WordPiece generates tokens from characters in each word separately and SentencePiece generates tokens from subword units from whole sentence. SentencePiece can be trained end to end on raw sentences, but in our case we used WordPiece since we already had available pretrained version on much more data, than what we had available.

WordPiece is a subword algorithm where we start with individual characters and merge them together based on most frequent combination of characters in text. To calculate WordPiece tokens we used BERT tokenizer from Transformers library with pretrained vocabulary from CroSloEngual BERT model [14].

### Embeddings and models

Inputs to the models are vectors that we get by embedding input tokens. In OpenNMT toolkit the embeddings are learnt during training of the model. We tried different types of models, like RNN and Transformers based ones.

#### Encoder decoder RNN models

#### Transformers self attention model

### Evaluation

For evaluation we will use automatic evaluation methodologies like BLEU (Bilingual Evaluation Understudy), METEOR (Metric for Evaluation of Translation with Explicit ORdering), CHRF (Character n-gram F-score), GLEU (Google BLEU), NIST (n-gram co-occurrence statistics) and RIBES (Rank-based Intuitive Bilingual Evaluation Score). They determine quality of machine translated text compared to the true translation. The implementations for score calculation are available from nltk (natural language toolkit) library.

Besides automatic evaluation we will also use human evaluation.

## Experiments

### Simple LSTM model

For training, we use the command line approach with a configuration file. First we convert the SPOOK dataset from the .xml files into a source and target text files, containing aligned sentences in English and Slovenian. We do the same for the Orwell dataset, which we use for evaluation.

We create a training.yaml file, which we will use to store all our parameters. First we define our data and their paths, and the path where we want to save our vocabulary. We also define SentencePiece tokenization and the path where the SentencePiece model will be saved. When these parameters are defined, we can run `onmt_build_vocab -config training.yaml` to build our vocabulary and tokenization model. This resulted in two separate vocabularies for each language, each containing 16000 tokens.

Before we start training, we define an Adam optimizer with a learning rate of 0.001, and set our batch type to tokens (default is sentences) and batch size to 1024. Since we intend to use the default LSTM network with 2 layers and 500 units on both encoder and decoder, we do not have to define it. We also set number of steps to 100000. We then run `onmt_train -config training.yaml` to start training. We can also use TensorBoard to make it easier to view how the model is training. This model reached an accuracy of 46%.

To look at how well our model is translating, we can use the `onmt_translate` script, but it does not support tokenization, meaning we need a separate script to tokenize our data with the SentencePiece model, use the script to translate it, then use another script to detokenize it before we can evaluate it. It is easier to setup an translation server with `onmt_server`, where we can then send translation requests and receive the translations using http requests with curl.

### Transformer models

For use with transformers, the SPOOK dataset is far too small and quickly causes overfitting. We first train a general NMT model on the provided TC3 dataset and evaluate it on provided test dataset.

Instead of training our own SentencePiece model, we use a pretrained BERT WordPiece model, which was trained on far more data and should be more representative of the language. Since OpenNMT does not support WordPiece tokenization, we use a script to tokenize our data prior to building our vocabulary. We also clean up the data by replacing all symbols with the same meaning with one symbol, such as using " for all types of quotation marks, in order to reduce the number of unknown symbols in our data. We also provide a script to shuffle the training sentences. Path to our cleaned, tokenized and shuffled data is then written to the configuration file.

Running the build vocabulary script will in this case only write our pretrained vocabulary into a format that OpenNMT expects and count how often each token appears in our training data. This results in a shared vocabulary of 35802 tokens.

We also write all the parameters of our transformer model to the file. We then train the model for 100 000 steps, which results in an accuracy of 56%.

### Next steps

- Add other general corpora to our training
- Finetune the transformer model on SPOOK dataset
- Try different transformer models (depth, number of heads)
- Automatic evaluation of translated texts
- Human evaluation

## Results

### Intermediate results

| | |
|---|---|
| English | You stabbed me with the sword three years ago. |
| LSTM | Pred tremi leti si me klical meči z mečom . |
| Transformer | pred tremi leti si me zabodel z mečem . |
| English | I wish I could. |
| LSTM | Rad bi , |
| Transformer | želim si , da bi lahko . |
| English | Now it is time for you to attend to me. |
| LSTM | Zdaj pa sedite k menim , da ga boš ubogal . |
| Transformer | zdaj je čas , da se mi pridružiš . |
| English | Why are you still draining water from the spring? |
| LSTM | Zakaj si še živ ? |
| Transformer | zakaj še vedno puščaš vodo iz spomladi ? |
| English | This whole sneaking around thing... does it ever get any easier? |
| LSTM | To je mogoče ... |
| Transformer | je to vse lažje ? |

**Table 1.** Comparison of model translations.

As we can see in Table 1 the model learn to roughly translate the meaning. Especially Transformers based model is sometimes very close to source text, but it can translate the wrong meaning of words (like spring - potok/pomlad). The models also have problems with translating longer sentences, we think that is due to some training corpora having longer English source sentence and shorter Slovenian translation.

## Discussion

## References

[1] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*, 2014.

[2] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[4] Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom

Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, Melbourne, Australia, 2018.

[5] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[6] D. Britz, A. Goldie, T. Luong, and Q. Le. Massive Exploration of Neural Machine Translation Architectures. *ArXiv e-prints*, March 2017.

[7] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

[8] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[9] Chenhui Chu and Rui Wang. A survey of domain adaptation for neural machine translation. *arXiv preprint arXiv:1806.00258*, 2018.

[10] Margot Fonteyne, Arda Tezcan, and Lieve Macken. Literary machine translation under the magnifying glass: Assessing the quality of an nmt-translated detective novel on document level. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 3790–3798, 2020.

[11] Antonio Toral and Andy Way. What level of quality can neural machine translation attain on literary text? In *Translation Quality Assessment*, pages 263–287. Springer, 2018.

[12] Evgeny Matusov. The challenges of using neural machine translation for literature. In *Proceedings of the Qualities of Literary Machine Translation*, pages 10–19, 2019.

[13] Taja Kuzman, Špela Vintar, and Mihael Arcan. Neural machine translation of literary texts from english to slovene. In *Proceedings of the Qualities of Literary Machine Translation*, pages 1–9, 2019.

[14] M. Ulčar and M. Robnik-Šikonja. FinEst BERT and CroSloEngual BERT: less is more in multilingual models. In P Sojka, I Kopeček, K Pala, and A Horák, editors, *Text, Speech, and Dialogue TSD 2020*, volume 12284 of *Lecture Notes in Computer Science*. Springer, 2020.