ZPJa: Neural Machine Translation from Scratch

Bc. Josef Kotoun
xkotou06@fit.vut.cz

Santosh Kesiraju Ph.D.
kesiraju@fit.vut.cz

Abstract

This project focuses on implementation of a Seq2Seq neural machine translation model, utilizing the transformer architecture for English-to-Czech translation, implementation of beam search and greedy search decoding algorithms and evaluating the trained model for both performance, using chrF2 and BLEU metrics, and inference speed. The results reveal that the model works well for short and moderately long sentences, translations capture meaning of source sentences quite well and are syntactically correct in Czech. The longer sentences are more problematic. The greedy decoding shows surprisingly better results than beam decoding algorithm in terms of evaluation metrics. This may be caused by the structure of the dataset, where only 1 reference translation is available for each sentence, as well as the problematic evaluation of correct translation by the metrics themselves, which is further discussed in the Results and Analysis section. The beam search decoding algorithm has slower inference speed compared to greedy decoding.

1 Task Definition

Machine translation is a challenging task in natural language processing that enables automatic conversion of text from one language to another. The key challenge lies in producing sentences in the target language that are grammatically correct and capture the semantic meaning of the source sentence. The complexity arises from linguistic differences between languages. Commonly used architecture for this problem is transformers architecture due to its possibility to selectively focus on different parts of the source sentence when generating each word in the target sentence. This mechanism is called attention.

2 Method

The transformer architecture is used in many different sequence-to-sequence tasks including machine translation. Its innovation lies in relying on multi-head attention mechanism instead of traditional recurrent or convolutional structures. The attention mechanism enables the model to weigh the significance of different input tokens dynamically.

The core building blocks of transformer are encoder and decoder. The encoder processes the input sequence, capturing its contextual information using the self-attention mechanism. The decoder generates output sequence utilizing self-attention to capture contextual information but with an additional cross-attention mechanism, allowing it to consider relevant parts of the input sequence. The decoder generates output sequence by predicting probability distribution over vocabulary for one token at a time. During training process, the most common practice is to use a technique called teacher forcing. Teacher forcing involves providing the true target sequence as input to the decoder, rather than using the previously generated tokens, which helps stabilizing the training process. During evaluation, the model uses an autoregressive approach, generating tokens one at a time and using its own predictions as inputs for subsequent steps.

2.0.1 Decoding algorithms

Decoding in machine translation refers to the process of generating sequence of words in the target language based on the learned representation from the source language. Transformer decoder generates probability distribution over vocabulary for one token at a time. We need to find output sequence of tokens based on tokens distribution such that its joint probability is highest. However, exhaustive search is not efficient for longer sequences, because the complexity of exhaustive search grows exponentially with respect to the length of the output sequence. Therefore, we need a suitable algorithm to find the best possible solution, even though the solution will not be optimal. The two primary decoding algorithms explored in this project are

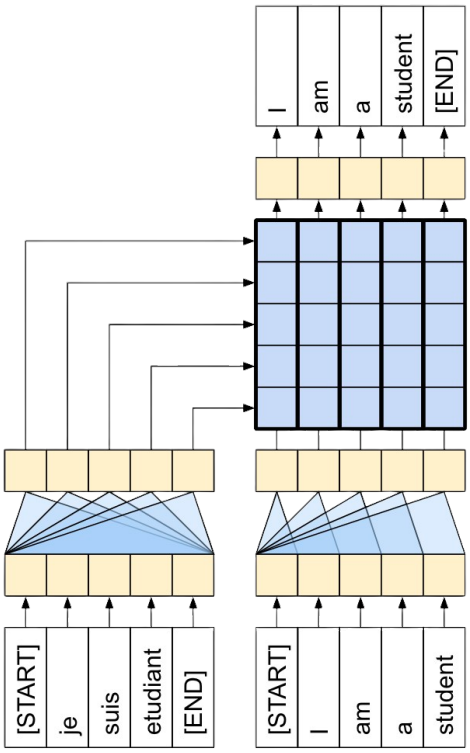


Figure 1: Transformer based translator training process (Tensorflow authors, 2023)

greedy decoding and beam search decoding.

2.0.2 Greedy Decoding

The greedy decoding algorithm is very simple. It selects the token with maximum probability given the current context of already generated tokens. This locally optimal strategy simplifies the decoding process but may overlook global dependencies, potentially resulting in suboptimal translations. However the algorithm is very easy to implement and it is faster than more complex algorithms.

2.0.3 Beam Search Decoding

The beam search maintains a set of top-k possible sequencies at each step, where k determines the beam size. The model generates multiple hypotheses simultaneously and selects the most probable ones. This approach is potentially leading to better results compared to the simpler greedy decoding but at the cost of increased computation and complexity and. It is still not guaranteed that it finds optimal solution either, but it may find better solution than greedy search.

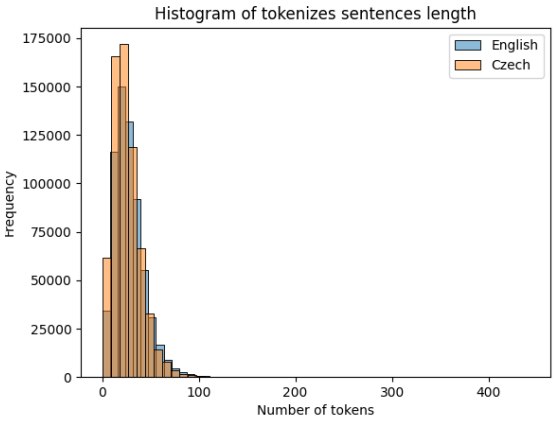
3 Experimental Setup

3.1 Programming Tools

Dataset preprocessing, training and evaluation were implemented using Python. Keras with TensorFlow backend was used to define structure of transformer model, with usage of bultin encoder, decoder and positional embedding from Keras library. Training tokenizers was done with Keras_nlp's wordpiece tokenizer. To check how well the translations worked, Keras_nlp and sacrebleu libraries were used.

3.2 Datasets

The project uses the Europarl dataset(Koehn, 2005), consisting of approximately 650,000 English-Czech sentence pairs. Median length of czech sentences after tokenization is 25 tokens and median length of english sentences is 27 tokens, so the dataset mainly consists of moderately long sentences. Longest sequence of both english and czech sets is over 400 tokens long, however 99.8% of sentences are shorther than 100 tokens, so this constant was chosen as maximum input and output tokens limit. The text, divided into sentences, is extracted from the proceedings of the European Parliament, so the text is mainly formal. This dataset is available at https://www.statmt.org/europarl/ in form of an archive, which consists of two files, each dedicated to one of the languages involved in the chosen language pair. Individual lines of file contain sentences in given language in same order as in other files for other languages. This dataset was divided into training, validation, and test splits, with a 70:15:15 ratio.



]

Figure 2: Dataset sentences length histogram in tokens

3.3 Training Hyperparameters

The transformer model was trained with a batch size of 16, learning rate of 1e-4 for 20 epochs. The Adam optimizer was used with early stopping based on validation loss, with patience of 3. Sequences were limited to 100 tokens. The transformer architecture featured 12 attention heads, an intermediate dimension of 4096 and an embedding dimension of 500.

3.4 Used Hardware

Single NVIDIA A40 48GB GPU was used for both training and evaluation. It was provided by MetaCentrum Virtual Organization.

3.5 Implementation details

Implementation of this project consists of these python scripts: **create_dataset_splits.py**, **preprocess_dataset.py**, **train_tokenizers.py**, **train.py** and **evaluate.py**. The **create_dataset_splits.py** loads both files with english and czech sentences, shuffles it randomly and splits it to train, validation and test splits in 70:15:15 ratio. The **train_tokenizers.py** creates **tf_dataset** from training splits and trains WordPiece tokenizers from **keras_nlp** for Czech and English. The **preprocess_dataset.py** takes care of tokenizing the training and validation splits and other preprocessing of dataset and saving it in tensorflow dataset format for training. The **train.py** defines the model architecture and trains the model using training hyperparameters described in section 3.3. The **eval.py** contains implementation of Greedy and Beam search decoding algorithms. It runs evaluation on specified number of evaluation samples and outputs chrF2 and BLEU metrics. Running the scripts requires downloading and extracting the europarl cs-en dataset archive to **./datasets/europarl/**. All scripts, dataset, pretrained model and pretrained tokenizers are available at https://huggingface.co/jkot/transormer_en_cs_translator/tree/main.

The **preprocess_dataset.py**, definition of model in **train.py** and **decode_sequences** function from **eval.py** were inspired by code examples from Keras documentation[1].

[1] https://keras.io/examples/nlp/

4 Results and Analysis

4.1 Evaluation metrics

4.1.1 chrF

chrF (Character-level F-score) is metric for machine translation evaluation, which calculates similarity between a translation output and reference translation using n-grams of characters instead of n-grams of words. This approach may be better for evaluation of high-morphology languages (community, 2023b). SacreBLEU[2] library was used to compute the chrF2 metric, which is version of the chrF metric, over the test split.

4.1.2 BLEU

BLEU (BiLingual Evaluation Understudy)(community, 2023a) is metric for machine translation evaluation, which calculates the similarity between a translation output and reference translation using word n-gram precision. SacreBLEU[3] library was used to compute the BLEU metric over the test split.

4.2 Experimental results

Evaluation was performed over the test split of the dataset with both greedy decoding and beam search decoding. Both evaluations results were compared to translation references using chrF2 and BLEU metrics. Sacrebleu library outputs the result scores in range from 0 to 100. Evaluation with beam search resulted in chrF2 = 46.5 and BLEU = 58.2. Greedy search results were surprisingly better than beam search with values of chrF2 = 52 and BLEU = 60.2. However, evaluation of translation by metrics such as chrF2 or BLEU may be problematic, because one sentence can be translated in different ways and although all translations are correct, some of the translations are closer to the stylistics of reference translation. For this reason, for a more accurate translation evaluation, the BLEU and chrF2 metrics can be evaluated using multiple possible reference translations, but the Europarl dataset contains only 1 translation for each sentence. This may also be the reason why greedy search comes out as better than beam search, as the reference translations may be stylistically closer to style of the sentences generated by greedy search than by beam search. As an example, the dataset contains this English sentence: *This green paper is important, seeing as it concerns a matter that*

[2] https://github.com/mjpost/sacrebleu
[3] https://github.com/mjpost/sacrebleu

needs to be dealt with. with Czech reference: *Tato zelená kniha je důležitá, protože se týká záležitosti, kterou je třeba řešit.*. The greedy search translated it as: *Tato zelená kniha je důležitá, protože se týká otázky, která se musí řešit.* and the beam search translated it as: *Tato zelená kniha je důležitá, vzhledem k tomu, že souvisí s otázkou, kterou je třeba řešit.*. Both of these translations capture the idea of sentence well, but the beam search translation is written in different style, which results in worse evaluation score. The translator handles short and moderately long sentences pretty well, in most cases it captures semantic meaning of sentences well and the translations are syntactically correct. They are often close to ideal translation. Examples of short sentences translations can be seen in table 1. Long sentences are more problematic. The translation mostly captures the main idea of the source sentence, but sometimes it omits some part of the source sentence and the sentence syntax is not that good. Examples of longer sentences translations can be seen in table 2.

English Sentence	**Czech Translation**
This is unacceptable: I shall vote against.	To je nepřijatelné: hlasuji proti.
I voted in favor of this report with conviction.	Hlasoval jsem ve prospěch této zprávy.
We have already heard about China.	O Číně jsme již slyšeli.
Thanks to the treaty of lisbon, this procedure allows the european parliament to express its view.	Díky lisabonské smlouvě může evropský parlament vyjádřit svůj názor.
One year after the disaster, we would like to pay tribute to those who died, but also to those who have survived.	Rok po katastrofě bychom chtěli vzdát hold těm, kteří zemřeli, ale také těm, kteří přežili.

Table 1: Examples of English to Czech translations of short and moderately long sentences using beam decoding algorithm

English Sentence	**Czech Translation**
But we knew from our experience with the passenger cars regulation that the car industry has made enormous progress in terms of innovation and developing cleaner technology because the legislative framework encouraged them in the right direction.	Ale z našich zkušeností s nařízením o osobních automobilech jsme věděli, že automobilový průmysl v oblasti inovací a rozvoje ekologicky čistšího pokroku, protože právní rámec je podporován v tom, aby se v správným směrem stal.
One of the successes to come out of the european council is that the council has seen that there are risks and dangers - whether it is climate change or the fact that innovation and technology requirements are increased or the regulatory burden upon smes - but there are also opportunities.	Jedním z úspěchů evropské rady je to, že rada viděla, že rada má rizika a rizika - zda jsou změna klimatu nebo že požadavky na inovace a technologie zvyšují regulační zátěž nebo regulační zátěž pro malé a střední podniky, ale existují i příležitosti.
The group of the european people's party, to which i belong, has always defended the principle of the freedom of movement of individuals, following the principle that rules and common procedures regarding visas, residence permits and the control of borders must form part of the full schengen concept.	Skupina evropské lidové strany, ke kterému patřím, vždy vždy hájila zásadu svobody pohybu jednotlivců, po zásadě, že pravidla a společné postupy týkající se víz, povolení k pobytu a kontrole hranic musí tvořit součást plné schengenské koncepce.

Table 2: Examples of English to Czech translations of long sentences using beam decoding algorithm

4.3 Inference speed

The greedy search was approximately 5 times faster than beam search in inference, which is affected by effectivity of my implementation of algorithms, which has definitely room for optimizations.

5 Conclusion

As can be seen in tables 2 and 1, translations of short and moderately long sentences are pretty good, the mostly capture the semantic meaning of source sentence and are syntactically structured well. Longer sentences are more problematic, so there is definitely room for improvement. It would be also better to have more reference translations than only one to each sentence, which could be one way to improve the learning and evaluation of model. The experiments also show, that trained model is better on formal language than informal, probably due to the used dataset, which consists mainly of highly formal sentences. Creating better dataset consisting of texts from multiple sources, both formal and informal, and more than one reference translation to each sentence may lead to better results. There's also room for improvement in optimization of decoding algorithms for better inference speed.

References

Machine Translate community. 2023a. Bleu: Evaluation metric based on n-gram precision. https://machinetranslate.org/bleu.

Machine Translate community. 2023b. chrf: Character-level f-score. https://machinetranslate.org/chrF.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. Phuket, Thailand.

Tensorflow authors. 2023. Tensorflow documentation. online.