Machine Translation with Europarl
Machine Learning for Natural Language Processing 2022

Arrazki Hajar
hajar.arrazki@ensae.fr

Blart Louise
louise.blart@ensae.fr

Abstract

Machine Translation or automated translation is a process when a computer software translates text from one language to another without human involvement. It works with large amounts of source and target languages that are compared and matched against each other by a machine translation engine. We differentiate three types of machine translation methods:

The colaboratory notebook of this project is available on github. In this report, we first introduce the dataset with descriptive analyzes and preprocessing, then we focus on the model and its results, and finally we use transfert learning to adapt this model to another dataset.

1 Problem Framing

Our goal is to create and train a machine translation model on the European Parliament Proceedings Parallel Corpus. We wish to evaluate the performance of our model on this dataset. We also want to see how it performs on other datasets that were not trained on. In order to do that, we will analyse the performance of the model on the tatoeba dataset.

We also wish to compare our model with already existing machine translation models that are used on Google or Facebook. We expect them to work better than our own model.

2 Experiments Protocol

2.1 Dataset, first analyzes and preprocessing

For this work we use European Parliament Proceedings Parallel Corpus, available on statmt. It is a corpus of 21 European languages where all the sentences are translated into all the languages. Here, we focus on the English and French languages to be able to check the translations. First we load the corpus, then we clean it using regex expressions for char filtering. We then transform the dataset into a pandas dataframe to remove duplicates and perform exploratory data analysis. We use this dataframe to create a Pytorch Dataset to perform the model.

2.2 Encoder-decoder model

We performed an Encoder-Decoder model for machine translation. These kinds of models were first introduced by Ilya Sutskever, et al. in "Sequence to Sequence Learning with Neural Networks".

The model is a sequence to sequence network with attention. It is composed of two sub-models; an encoder and a decoder. The encoder is responsible for stepping through the input time steps and encoding the entire sequence into a fixed length vector called a context vector. The decoder is responsible for stepping through the output time steps while reading from the context vector.

In this work we use another RNN : Gated Recurrent Unit (GRU), introduced by Kyunghyun Cho, et al. in 2014 in "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". The GRU is like a LSTM with a forget gate, but has fewer parameters than LSTM, as it lacks an output gate. GRUs have been shown to exhibit better performance on certain smaller and less frequent datasets on natural language processing.

2.3 Attention model

Attention is proposed as a solution to the limitation of the Encoder-Decoder model encoding the input sequence to one fixed length vector from which to decode each output time step. This issue is believed to be more of a problem when decoding long sequences. It is proposed as a method to both align and translate. A representation of the model is available in Appendix A.

Alignment is the problem in machine translation that identifies which parts of the input sequence are relevant to each word in the output, whereas translation is the process of using the relevant information to select the appropriate output.

Instead of encoding the input sequence into a single fixed context vector, the attention model develops a context vector that is filtered specifically for each output time step. As with the Encoder-Decoder paper, the technique is applied to a machine translation problem and uses GRU units rather than LSTM memory cells. In this case, a bidirectional input is used where the input sequences are provided both forward and backward, which are then concatenated before being passed on to the decoder.

2.4 Test on another dataset

Our model was performed on the Europarl dataset. We have presented a word cloud in Appendix B Figure 2. The vocabulary of this dataset is specific to the governmental and parliamentary domains. We want to evaluate our model in a more common vocabulary. We choose Tatoeba dataset. The word cloud for Tatoeba is not the same Figure 3. The words are not really significant in tatoeba but we want to check if our model performs well on this dataset.

In order to see if the model really performs well, we will check it with the tatoeba dataset. As our model was performed on a specific dataset that was quite short because of computational reasons, we expect it not to work too well on the tatoeba dataset.

2.5 Transfert learning with pretrained models

When we work on Machine learning problem, adapting an existing solution can help to get a solution much faster. Moreover, it save computational costs as it requires less training. Hugging Face is one of companies that provide open source libraries containing pre-trained models. We use two transformer models : T5 model and MarianMT model. "A transformer is a deep learning model that adopts the mechanism of attention, differentially weighting the significance of each part of the input data. It is used primarily in the field of natural language processing". This kind of models were introduced in the paper *Attention is all you need - 2017* . T5 is based on encoder-decoder architecture and the basic idea is to take text as input and produce a new text as output. T5 model is known to workd well with a wide range of tasks out-of-the-box by prepending a prefix of these tasks to the input sequence. MarianMT is also based on encoder-decoder architecture and was originally trained by Jörg Tiedemann using the Marian library. Marian is written entirely in C++. This library supports faster training and translation. There are around 1300 models which support multiple language pairs. All these models follow the same naming convention – Helsinki-NLP/opus-mt-src-target, where src and target are the two-character language codes. Each model is about 298 MB on disk, which means it is smaller compared to other models and can be useful for experiments, fine-tuning, and integrating tests. This is the reason why we chose this model. In fact, our colaboratory sessions were short, and the RAM did not support hight models.
We fine tune these models to adapt them to our europarl dataset.

3 Results

Our model seems to work well when we write our own sentences but does not perform too well on the test set. This is probably due to the fact that we only trained it on short sentences.

The pre-trained model MarianMT seem to perform better than our sequence to sequence model.

This makes sense because those models were trained on datasets that were bigger and probably covered more words. Our dataset is quite specific, it refers to european institutions. When we want to translate words that are out of the range of this field, it perfoms badly.

4 Discussion/Conclusion

Therefore, our model could be improved by first training it on a bigger dataset. We were limited in terms of computation and ressources because of Google Colab's restrictions. Although we have both subscribed to Google Colab Pro account. In addition to training our model on a bigger dataset, we couls use more epochs. We had to restrict our traning to 5 epochs because it took too much time even with a Google Colab Pro account. Our initial intention was to use early stopping. However, as we did not train our model enough, we could not do that. The next step would therefore be to add early stopping and allow our model to train longer and use early stopping.

References

Ochiai, Tsubasa Watanabe, Shinji Hori, Takaaki Hershey, John. (2017). Multichannel End-to-end Speech Recognition.

A Appendix A Encoder-Decoder Model

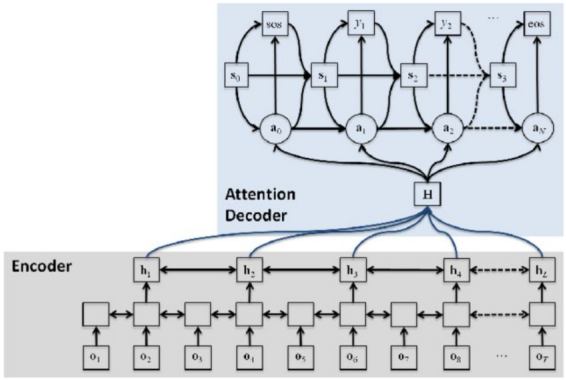

Figure 1: Structure of an attention-based encoder-decoder framework

B Appendix B Dataset Representations

B.1 English Europarl Dataset

Figure 2: Word Cloud for the English Europarl Dataset

B.2 Tatoeba Dataset

Figure 3: Word Cloud for the English Tatoeba Dataset