Portuguese-English and English-Portuguese T5 Translation Task

Alexandre Lopes

June 2020

Abstract

This report aims to evaluate the use of T5 [8] for Portuguese-English and English-Portuguese translation. This work proposes a pre-processing and post-processing strategy to deal with special words used in Portuguese language, such as diaeresis, acute and grave accents. This allows an improvement of the results in more than 6 BLEU score points. We compare our results to Google Translate API and MarianMT in ParaCrawl dataset and the winner algorithm in WMT'19 Biomedical Translation tasks in WMT'19 PT/EN and EN/PT datasets. In the WMT'19 Biomedical PT/EN task we achieve a new State-of-Art result without fine-tuning for specific area.

1 Introduction

Machine Translation (MT) systems are one of the oldest Natural Language Processing (NLP) tasks [12]. Recently with the advent of complex deep neural networks, results in Translation improved over classical statistical strategies, specially in parallel corpora [1]. In the Third and Fourth Conference on Machine Translation (WMT18 and WMT19) [2, 7], the best BLEU scores for English-German and German-English competitions were accomplished by systems based on Transformers [13]. Transformer models allow the network to learn more complex relationship between the words in a sentence, letting these models to solve a common problem in MT tasks: lexical choice of words in the case of semantic ambiguity. For instance, the word 'bank' in Portuguese can be translated to 'bench' or 'bank', depending on the usage context.

MT systems can be divided into two main groups: Statistical Machine Translation (SMT) systems and Neural Machine Translation (NMT) systems. The first one rely in statistical techniques like Moses [4] to perform the translation. [**?**] performs such task in Portuguese. They also released a fully automatic compilation of parallel corpora for Brazilian Portuguese, called fapesp-v1 and fapesp-v2 translation datasets.

Recently, the winning paper of WMT'19 biomedical competition for en-pt and pt-en translation tasks [10] uses a NMT system. They used OpenNMT-py to create a Transformer architecture and used seven individual corpora to create a corpus for translation.

The approach used in this work was to add a new task to a **T**ext-to-**T**ext **T**ransfer **T**ransformer (T5) pre-trained model adding a fine-tuning strategy to add Portuguese translation. A representation of T5 training with portuguese data is represented in Figure 1. The T5 implementation used in this work was implemented by HuggingFace [14]. The default tokenizer used in [14] is based on SentencePiece [5]. This tokenizer does not expect characters that do not appear in original translation languages, such as English, German and French. For example, one of the most used words in Portuguese, the word 'não', which means 'no' in English, cannot be formed in SentencePiece, because the tilde over the vowel 'a' is not used in English, German or French. The tilde accent indicates nasalization sound of the base vowel in Portuguese and is very used in common words.

Many commercial systems such as Google Translate (GT) and Amazon Translate (AT) have an excellent performance for MT, but they have a high cost if one intent to translate large amounts of data. For example, we estimate that if one want to translate 20 million sentences of ParaCrawl using GT it would take almost $50,000 to translate it. These systems also lack in providing data for comparing their metric scores with public datasets or providing datasets that allow comparation of performance between public projects and these systems.

This project proposes some additions to the tokenization performed in T5 along with a post processing strategy to deal with Portuguese special characters. This allows us to develop a Portuguese-English (pt-en) and English-Portuguese (en-pt) machine translation system that is available for reproducibility and extension on Github[1]. We compare our results with three other systems in two different datasets. In addition to contributing to the creation of the T5 Portuguese models, this work translated 128,000 sentences of ParaCrawl dataset for both directions (Portuguese-English and English-Portuguese) using Google Translate. All translations are also available in the same repository for other researches to use it as a common database.

2 Methodology

For this work, we proposed an adaptation for the T5 to work with Portuguese. We can divide this into two parts: a pre-processing stage and a post-processing stage. The first one is responsible for allowing the use of Portuguese special

[1] PT-EN-Translator `https://github.com/alelopes/PT-EN-Translator`.

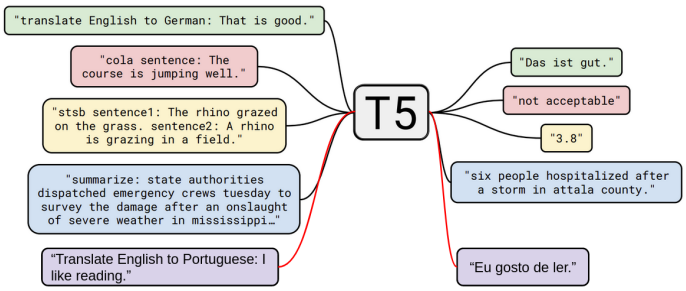

Figure 1: T5 training strategy with adition of red conections. The purple boxes were the fine-tuning part performed by this work. Created using figure from [8]

characters and the second one manages how to get information back to form the words again.

2.1 Pre-processing Tokenizer

The first step of the pre-processing is to identify which accented vowel is not present in tokenizer. We checked all Portuguese accented vowels in the tokenizer and we identified fourteen tokens not included in the tokenizer. We also added the 'não' word that was the most common word in ParaCrawl pt-en dataset to the tokenizer. A list of all added tokens is available in Table 1. Adding these tokens to the system granted it the possibility of learn and predict them in en-pt translation or to learn these tokens relationship to neighbours tokens to predict better English translations in pt-en mode. However, when added directly to the tokenizer, SentencePiece interpretate the result as a new complete token, not part of a word. So, for example, the word 'pão' is broken into three different tokens 'p' 'ã' 'o'. This is fixed in the post-processing step. Table 2 shows some examples of encode and decode after adding tokens to tokenizer.

Table 1: Added Tokens to SentencePiece

ì ò Á Í Ó Ú í ú Â Ê Ã Õ ã õ não

Table 2: Comparing tokenizer results before and after changing it.

Tokenizer without adding tokens	**Tokenizer after adding tokens**
original → after encode/decode	original → after encode/decode
eu gosto de arroz → eu gosto de arroz	eu gosto de arroz → eu gosto de arroz
eu não como → eu n ? o como	eu não como → eu não como
indignação completa → indignaç ? o completa	indignação completa → indignaç ã o completa

2.2 Post-processing Step

The Post-processing step consists in regrouping back the tokens separated erroneously of vowels with accents. Here we find all single special tokens in the text and merge them to their neighbor words. In Figure 2 we can see a representation of our algorithm.

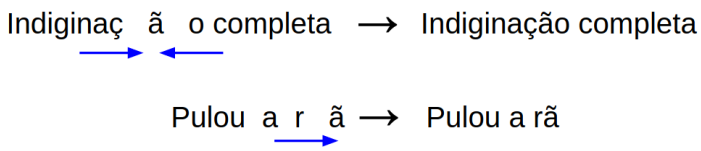

Figure 2: An example of broken tokens reunited back to a single word. The algorithm searches for isolated special token and merge it to its neighbours. It can be merged at the beginning, middle or end of a sentence.

3 Data set

To fine-tune T5 we used ParaCrawl[2] for training and two different datasets for testing. ParaCrawl is a public parallel corpora of many European languages available online. In this dataset, we first randomly selected 250,000 sentences that were going to be separated as test set. The rest of the parallel corpora could be used for training/validation. The first 128,000 sentences of the test set were automatically translated using GT. This allows us to compare the results obtained to this work with a commercial solution. We used 'pt-PT' option in GT since ParaCrawl is in European Portuguese.

The translated sentences from GT are available in our repository, totalizing two datasets from the parallel 128,000 corpora, and making it possible to evaluate performance of translation for both directions.

This work also evaluated its generalizability using WMT'19 Biomedical dataset, which comes with train and test split predefined. The training data provided by WMT'19 challenge is composed of different corpora:

- Medline Dataset[3]: Dataset of titles and abstracts of scientific publications
- Scielo Dataset[4]: Dataset of scientific publications
- ReBEC Dataset[5]: Dataset of clinical trials

We tested our solution using two strategies: without training for biomedical data (only using ParaCrawl as training data); and training with biomedical data.

[2]Paracrawl `https://paracrawl.eu/`
[3]Medline `https://github.com/biomedical-translation-corpora/medline`
[4]Scielo `https://github.com/biomedical-translation-corpora/scielo`
[5]ReBEC `https://github.com/biomedical-translation-corpora/rebec`

4 Experiments

This work conduct several experiments using different configurations of T5. We can divide the experiments into two groups: model hyper-parameter optimization and dataset generalization. All experiments were performed using a Desktop with Nvidia RTX 2070 Super, 32 Gb RAM memory and a 4-core Intel processor running under Ubuntu 18.04 Operational System. We used PyTorch, HuggingFace Transformer and Pytorch-Lightning frameworks for the network implementation.

4.1 Model Hyper-parameter Optimization

To perform such optimization, we first conducted a small training of the data in 100k sentences and evaluated on 50k sentences to determine some basic hyperparameters of T5 archtecture, such as source maximum length of tokens and batch size. We also evaluated the optimizer and the best convergence was obtained with AdamW Optimizer [6] with Pytorch's default parameter, except for the ones in Table 3. All hyperparameters used are available in Table 3. Using the same configuration, we evaluated the performance of adding the pre-processing and post-processing strategy in the models vs leaving the tokens without any pre or post-processing strategy. These results are available in Table 4.

Table 3: Hyperparameters used for training

Batch Size	256
Source Sequence Length	96
Target Sequence Length	160
Learning Rate	$5 \cdot 10^{-3}$
eps	$1 \cdot 10^{-5}$

Table 4: Increasing in performance of preprocessing and postprocessing

Translation Type	**SacreBLEU**
Training Without Pre/Post Processing	31.15
Training With Pre/Post Processing	35.95 (+4.8)

After finding these parameters, we analysed the trade-off between Model Sizes in a subset of ParaCrawl dataset of 1M sentences and evaluated the results in a 150k sentences subset. We did not use any sentence from the 128k pre-selected sentences for testing in this stage. The results of this analysis are reported on Table 5. We trained the T5 small and T5 base models with different epoch sizes, due training time. Training 3 epochs of T5 small takes almost the same time than training one epoch with T5 base. Possibly the performance would increase if using even large models like T5-large, T5-3B or T5-11B, however it would require more than the 8GB needed for the T5-base model. We

used batch accumulation to achieve batch size 256 as the T5 small can only handle batch size 4 in 8GB.

Table 5: Increasing in performance of preprocessing and postprocessing

Translation Type	**SacreBLEU**
Preprocessing adding Top 25 words in Port. + T5 small + 3 Epochs	43.03
Preprocessing adding tokens of Table 1 in Port. + T5 small + 3 Epochs	43.48
Preprocessing adding tokens of Table 1 in Port. + T5 base + 1 Epochs	**44.52**

4.2 Dataset Generalization Experiments

The objective of this experiment is to analyse the performance of our purposed technique with other commercial and non-commercial tools. To compare the results with GT, we first discarded all test data that were similar to any example in training. ParaCrawl is deduplicated, however similar sentences exists in the dataset. To increase the speed in the process we used MinHash and Locality-Sensitive Hashing (LSH) [9] to perform the comparisons between the elements of train and test datasets. We set a jaccard similarity threshold of 0.7, i.e., all elements with similarity greater than 0.7 were discarded. LSH found 28913 sentences with similarity score above 0.7. Finally, the final testing set has 99087 sentences.

We trained in ParaCrawl using approximately 5.4M sentences. It was not possible to add all +20M sentences because we had limited time. It took 7 days to train in en-pt and 7 more days for pt-en. The results for this training are available in Table 6. It is possible to see that our results differ from GT for 0.61 sacreBLEU score in en-pt and 6.78 sacreBLEU score points in pt-en. Although the results differed more in pt-en task, this difference was not perceptive for our human evaluation of some examples.

Table 6: SacreBLEU comparison between GT and our approach in 99k Paracrawl test set.

	pt-en	**en-pt**
GT	51.20	45.17
ours	44.42	44.56

For the WMT19' comparison, first we fine-tuned the previous training result on en-pt to evaluate how the addition of data from similar area would impact in the results. The WMT19' training set has approximately 332k sentences. We also evaluated the results of our model with the winning algorithm of WMT19' Biomedical tasks [11] and MarianMT[3] implementation given by HuggingFace

Transformer Library[6]. The results of this experiment is in Table 7.

Table 7: WMT19' BLEU comparison between BSC [11], MarianMT and two approaches.

	pt-en	**en-pt**
MarianMT	27.91	47.44
BSC [11]	39.90	**48.18**
ours with ParaCrawl only	**40.16**	42.57
ours with WMT Train Data	–	39.31

It is possible to see that we achieved State-of-the-art (SOTA) result in pt-en translation using WMT19' Biomedical task test set. We can see that MarianMT has only 0.74 BLEU difference from the best algorithm for en-pt (BSC algorithm), however it diverges in 12.25 BLEU points in pt-en from our model with ParaCrawl training data only. That happens because the MarianMT only selects the target language in translation and it was trained for multilingual romance-en translation. Therefore, it is ready to translate from many different Latin origin languages, possibly implying in a lower performance as it has a more complex task. It is also interesting to see that our algorithm with WMT Training data performs worst than ours with ParaCrawl dataset only. That possibly happens because the WMT training data is too small compared to ParaCrawl's one. The differences in both datasets (WMT is mostly in Brazilian Portuguese while ParaCrawl is European Portuguese), must be overcoming the advantages of WMT being a dataset in the biomedical area. Besides, WMT has a smaller number of samples.

5 Conclusion

This work aimed to contribute in a model training proposal for T5 dealing with special accented vowels. We also created a dataset for evaluating new purposed MT Systems that is publicly available on our github page. Our algorithm achieved SOTA results in pt-en translation, improving results from WMT'19 winning algorithm.

6 Future Work

One difference of ParaCrawl training and WMT dataset is that ParaCrawl is primarily European Portuguese and WMT test set is primarily Brazilian Portuguese. The results in pt-en and en-pt possibly should improve if trained in Brazilian Portuguese dataset. [10] gathers a parallel corpora of +7M sentences mainly in Brazilian Portuguese. We aim to test this dataset and submit our

[6]MarianMT from HuggingFace `https://huggingface.co/transformers/model_doc/marian.html`

findings in the Fifth Conference on Machine Translation - WMT20' Competition.

References

[1] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Unsupervised statistical machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3632–3642, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[2] Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[3] Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, et al. Marian: Fast neural machine translation in c++. *arXiv preprint arXiv:1804.00344*, 2018.

[4] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[5] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing, 2018.

[6] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[7] Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. Facebook FAIR's WMT19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 314–319, Florence, Italy, August 2019. Association for Computational Linguistics.

[8] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[9] Anand Rajaraman and Jeffrey David Ullman. *Mining of massive datasets*. Cambridge University Press, 2011.

[10] Felipe Soares and Martin Krallinger. BSC participation in the WMT translation of biomedical abstracts. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 175–178, Florence, Italy, August 2019. Association for Computational Linguistics.

[11] Felipe Soares and Martin Krallinger. Bsc participation in the wmt translation of biomedical abstracts. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 175–178, 2019.

[12] Lucia Specia. A hybrid model for word sense disambiguation in english-portuguese machine translation. In *IN PROCEEDINGS OF THE 8TH RESEARCH COLLOQUIUM OF THE UK SPECIAL-INTEREST GROUP IN COMPUTATIONAL LINGUISTICS*, pages 71–78, 2005.

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[14] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.