EUROPEAN ASSOCIATION FOR MACHINE TRANSLATION

EAMT2018

Proceedings of the

21st Annual Conference of the European Association for Machine Translation

28–30 May 2018

Universitat d'Alacant

Alacant, Spain

Edited by

Juan Antonio Pérez-Ortiz

Felipe Sánchez-Martínez

Miquel Esplà-Gomis

Maja Popović

Celia Rico

André Martins

Joachim Van den Bogaert

Mikel L. Forcada

Organised by

Universitat d'Alacant

Universidad de Alicante

transducens

research group

ISBN: 978-84-09-01901-4

Implementing a neural machine translation engine for mobile devices: the Lingvanex use case

Zuzanna Parcheta[1] **Germán Sanchis-Trilles**[1] **Aliaksei Rudak**[2] **Siarhei Bratchenia**[2]
[1]Sciling S.L., Carrer del Riu 321,Pinedo, 46012, Spain
[2]Nordicwise LLC, 1 Apriliou, 52, Athienou, Larnaca, 7600, Cyprus
{zparcheta, gsanchis}@sciling.com
{alrudak, s.bratchenya}@lingvanex.com

Abstract

In this paper, we present the challenge entailed by implementing a mobile version of a neural machine translation system, where the goal is to maximise translation quality while minimising model size. We explain the whole process of implementing the translation engine on an English–Spanish example and we describe all the difficulties found and the solutions implemented. The main techniques used in this work are data selection by means of Infrequent n-gram Recovery, appending a special word at the end of each sentence, and generating additional samples without the final punctuation marks. The last two techniques were devised with the purpose of achieving a translation model that generates sentences without the final full stop, or other punctuation marks. Also, in this work, the Infrequent n-gram Recovery was used for the first time to create a new corpus, and not enlarge the in-domain dataset. Finally, we get a small size model with quality good enough to serve for daily use.

1 Introduction

Lingvanex[1] is a trademark for linguistic products made by Nordicwise LLC company. The main focus of the company are translator and dictionary applications that work without internet connection on mobile and desktop platforms.

In collaboration with Sciling[2], an agency specialised in providing end-to-end machine learning solutions, a small-sized translation model from English to Spanish was implemented.

When implementing a mobile translator, it is crucial to understand its purpose. In our case, the purpose was to be able to generate translations on a daily usage scenario, without requiring a Internet connection. This is the typical use case in a travel scenario, where travellers often do not have an internet connection, either because they do not want to assume the cost of a roaming connection, because they do not want to purchase a local SIM card, or even because there is no good connection in the places they are travelling to, such as some countries of Africa. In this scenario, the main purpose of the translation engine is to be able to translate correctly short sentences, composed of common words in a traveller domain, but where other words belonging to e.g. a parliamentary or a medical domain are less frequent. In addition, the model requires to be contained in terms of size, since large models would perform poorly in a mobile device.

In this work, we focus on reducing model size mainly through data selection techniques, until a size of 150MB per model. However, there are other techniques which bring promising results as compressing the NMT model via pruning (See et al., 2016).

Along this article we determine what is the main influence to model size. For that, we conducted experiments comparing model size with total vocabulary size and word embedding size. Also, we compare the model size with different layer number on encoder and decoder

[1]https://lingvanex.com/en/

[2]https://sciling.com/

Pérez-Ortiz, Sánchez-Martínez, Esplà-Gomis, Popović, Rico, Martins, Van den Bogaert, Forcada (eds.)
Proceedings of the 21st Annual Conference of the European Association for Machine Translation, p. 297–302
Alacant, Spain, May 2018.

side, and the size of recurrent layer. Next step is to select data for training the engines through sentence length filtering and leveraging a DS technique. During the implementation of our translation engines we found several problems in the translations generated. We describe each of the problems and we propose appropriate solutions. After implementing these solutions, we evaluate the quality of our final model on a test set, and compare the results with Google's and Microsoft's mobile translators.

2 Data description

The data used to train the translation model was obtained from the OPUS[3] corpus. In total, there were 76M parallel sentences. We also leveraged the Tatoeba corpus for DS described in Section 4. Tatoeba is a free collaborative online database of example sentences geared towards foreign language learners. The development set was also built from the Tatoeba corpus, by selecting 2k random sentence pairs. Main figures of Tatoeba corpus are shown in Table 1. As the test set we create small corpus of more useful sentences in English found in different websites. Also we add some sentences of unigrams and bigrams. In total we selected 86 sentences.

Table 1: Tatoeba main figures. k denotes thousands of elements, $|S|$ stands for number of sentences, $|W|$ for number of running words, and $|V|$ for vocabulary size.

language	$\|S\|$	$\|W\|$	$\|V\|$
English	136k	964k	40k
Spanish	136k	931k	61k

3 Model size dependency

When confronting the model size reduction, the first question that arises is what hyper-parameters have the most influence on model size. Before moving forward and implementing a NMT system, we conducted experiments comparing model size with total vocabulary size and word embedding size (Mikolov et al., 2013). We also compared model size with different number of layers and units per recurrent layer, both on encoder and decoder sides.

To determine how the previously enumerated hyper-parameters affect model size, we trained different models varying these hyper-parameters. In the first experiment, we set the number of units in the recurrent layer to 128, with a single layer on both encoder and decoder sides. We analysed the effect of considering a total combined (source and target) vocabulary size $|V|$ was pruned to $|V| = \{5\text{k}, 10\text{k}, 20\text{k}, 50\text{k}, 100\text{k}\}$, selected according to the most frequent words in the Opus corpus, with souce and target vocabulary size set to $|V|/2$. In addition, we also studied different embedding vector sizes $|\omega| = \{64, 128, 256, 512\}$. The results obtained are shown in Figure 1a.

Next, we analysed the effect of considering different number of hidden units and the number of layers. In this case, we fixed to $|\omega| = 128$. We found that the number of layers, using 128 hidden units, has almost no effect on model size. In Figure 1b, we only show 1 and 4 layers for 128 units. Looking at Figure 1, we can conclude that the number of layers has small effect on model size comparing with number of hidden units and embedding size. Figure 1 can be leveraged to decide on adequate values for these hyper-parameters, once model size has been fixed to 150MB.

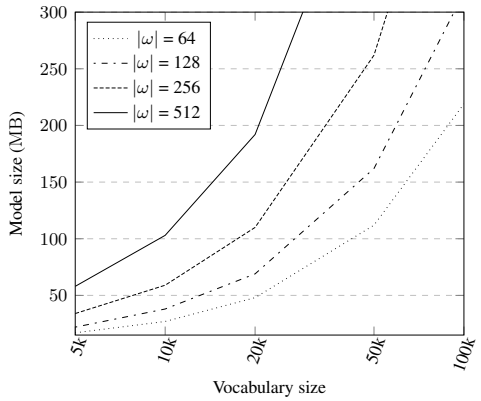
4 Data filtering

Data filtering involved two main steps: first, sentences with length over 20 words were discarded. We did this under the assumption that a mobile translator is mainly designed for translating short sentences. Second, we performed data selection, leveraging Infrequent n-gram Recovery (Gascó et al., 2012). The intuition behind this technique is to select, from the full available bilingual data, those sentences that maximise the coverage of n-grams in a small, domain-specific dataset. The full available bilingual data is sorted by infrequency score of each sentence in order to select first the most informative.

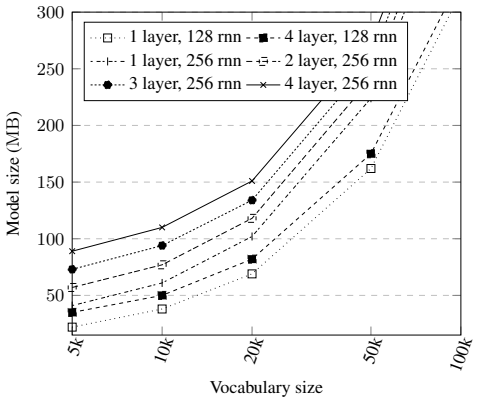
Let be χ the set of *n*-grams that appear in the sentences to be translated and $\mathbf{w}$ one of them; $C(\mathbf{w})$ denotes the counts of $\mathbf{w}$ in the source language training set; t the threshold of counts when an n-gram is considered infrequent, and $N(\mathbf{w})$ the counts of $\mathbf{w}$ in the source sentence $\mathbf{f}$ to be scored. The infrequency score of $\mathbf{f}$ is:

$$i(\mathbf{f}) = \sum_{\mathbf{w} \in \chi} \min(1, N(\mathbf{w})) \max(0, t - C(\mathbf{w})) \quad (1)$$

[3] http://opus.nlpl.eu/



(a) Model size depending of vocabulary size and embedding size. Number of units in the recurrent layer set to 128, and the number of layers is 1.



(b) Model size depending of vocabulary size, number of units in the recurrent layer (*rnn*) and number of layers, with $|\omega| = 128$.

Figure 1: Model size dependency of different parameters. k denotes thousands of elements and MB is an abbreviation for megabyte. The vocabulary size is the sum of source and target vocabulary.

We applied Infrequent n-gram Recovery to the 60M sentences from the Opus corpus as out-of-domain. Intuitively, we selected sentences from the available data until all n-grams, with n up to 5, extracted from the Tatoeba corpus have a maximum of 30 occurrences (if such a thing is possible with the data available). However, applying this technique on the full set of 60M sentences would have led to very long execution time. Hence, we divided the corpus into 6 partitions, and the selection was performed on each one of these partitions. Then, we joined the selections from all 6 partitions and conducted a second selection step on this corpus, since some n-grams could well have $6 \cdot 30$ occurrences. This led to a final selection of 740k sentences. The selected data set presented a vocabulary size of 19.4k words in source and 22.9k on target side. The total (combined) vocabulary was $|V| = 42.4\text{k}$. Note that selection was conducted on the tokenised and lowercased corpus.

5 Experimental setup

The system was trained using the OpenNMT (Klein et al., 2017) deep learning framework based in Torch. OpenNMT is mainly focused at developing sequence-to-sequence models covering a variety of tasks such as machine translation, summarisation, image to text, and speech recognition. Byte-pair encoding (BPE) (Sennrich et al., 2015) was trained on the selected training dataset, and then applied to training, development, and test data. In each experiment we trained a recurrent neural network with long short term memory (LSTM) (Hochreiter and Schmidhuber, 1997). We use global attention layer to improve translation by selectively focusing on parts of the source sentence during translation. Also, we use input feeding to feed attentional vectors as inputs to the next time steps to inform the model about past alignment decisions (Luong et al., 2015). However, this option only had a visible effect with 4 or more layers. Training was performed with 50 epochs using the *adam* (Kingma and Ba, 2014) optimiser, with learning rate of 0.0002. Finally, we selected the best model according to higher BLEU (Papineni et al., 2002) score on the development set, and used that model to translate the test set. Given that the test set is very small, we performed a human evaluation to analyse whether the quality obtained was good enough.

6 Results and analysis

We trained different typologies of neural networks observing the conclusions in Section 3. In each of the experiments we varied the hyper-parameters described in Section 3. Since the total combined vocabulary was fixed to $|V| = 42.4\text{k}$, from Figure 1 we can infer the combination of hyper-parameters with which the allowed model size will not be exceeded.

Table 2 shows the values of the hyper-parameters used in each experiment, together with the BLEU score obtained by each model and its size.

The best model according to the BLEU score on the development set is the model trained with 2

Table 2: Hyper-parameter values for the different experiments (exp) conducted and results obtained. $|\omega|$ is the size of the word embedding vector, expressed in megabytes.

exp	$\|\omega\|$	layers	rnn	size	BLEU	
					dev	test
1	128	2	128	146	39.0	26.6
2	128	3	128	151	36.9	22.8
3	128	4	128	155	37.7	21.8
4	64	4	256	206	38.7	23.8
5	256	4	64	203	32.7	21.1

layers, 128 units on recurrent layer, with $|\omega| = 128$. Also, it is the smallest model among those analysed in Table 2.

6.1 Problems found

Analysing the translations from the test set we found 3 different problems. In the following, we describe each of them and propose the corresponding solutions.

6.1.1 Repeated words problem

Analysing the quality of the best model obtained, we noticed that sentences with more than 7 words were translated correctly. However, translations of very short sentences contained repeated words, e.g. *"perro perro perro perro perro perro"*. The hypothesis for explaining this fact could be because of differences between training and test sentence lengths. To understand the validity of this hypothesis, we analysed the histogram of sentence lengths of training set, shown in Figure 2. As seen, the source side of the training data contains a very few amount of sentences shorter than 8 words, in contrast to the target side, where the distribution of sentence length is more uniform. We believe such difference is caused by the sentence selection algorithm used: selection is conducted in the source language and the selection algorithm tends to assign higher scores to longer sentences, since the more n-grams the source sentence contains, the more likely it includes infrequent n-grams. To cope with this fact, we modified the Infrequent n-gram Recovery strategy as follows:

Re-scoring of sentences: To fix the problem of repeated words we decided to modify the sentence selection procedure modifying the Infrequent n-gram Recovery scoring function by adding a normalisation step. In order to normalise such score, we modified Equation 1 as follows:

$$i(\mathbf{f}) = \sum_{\mathbf{w} \in \mathcal{X}} \frac{\min(1, N(\mathbf{w})) \max(0, t - C(\mathbf{w}))}{|\mathbf{f}| - \mathbf{w} + 1} \tag{2}$$

where the denominator normalises by the number of n-grams of order n in the sentence. With this normalisation, we avoid the side-effect of sentence length on the infrequency score, ultimately leading to selecting shorter sentences and improving the NMT system's translation of such sentences.

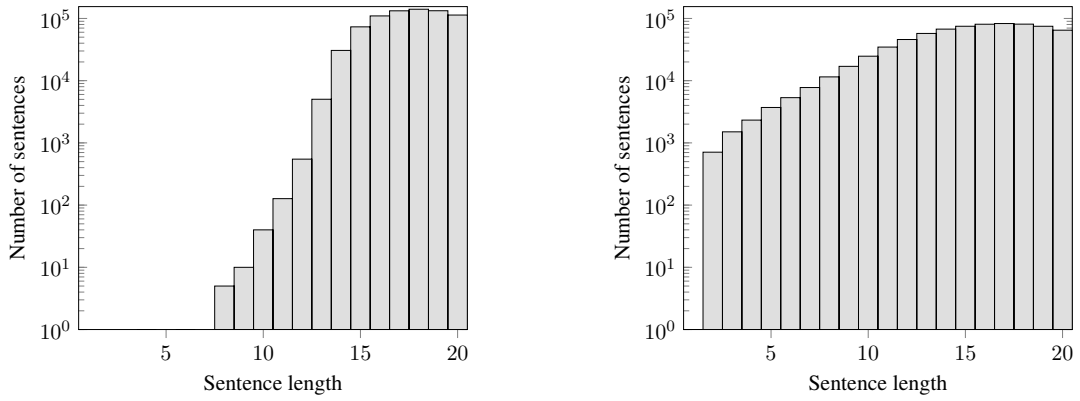
After applying the infrequency score in Equation 2 for selecting the data anew, we obtained 667k sentences. In Table 3 we show the average sentence length in source and target language before and after applying the sentence length normalisation. Average sentence length of Tatoeba is shown for comparison purpose. As shown, we are able to obtain much shorter sentences by including normalisation. The model achieved 36.3 BLEU in development, and 22.8 in test, with a model size of 121MB. Although this score is slightly worse than the one achieved in experiment 1 (Table 2), we believe BLEU is not always the most adequate metric for evaluating translation quality (Shterionov et al., 2017). By manually analysing the hypotheses, we concluded that the repeated words problem had been successfully solved.

Table 3: Average sentence length of Tatoeba and training set before and after applying normalisation in Equation 2.

		source	target
	Tatoeba	7.1	6.8
train	before normalisation	17.4	15.1
	after normalisation	10.4	9.0

6.1.2 Punctuation mark expectation

Analysing the hypotheses generated by our new model, we noticed that the model generated wrong translations with very short sentences, e.g. "dog", or "cat", generating surprising translations such as "amor". However, when adding a punctuation mark to the source sentence, e.g. "dog.", the translation was correctly produced. Our first intuition regarding this was that the model was expecting a punctuation mark at the end of each sentence. This intuition was confirmed by the fact that 94% of the sentences in the source language training set had a dot or other



(a) Histogram of source training set. (English)

(b) Histogram of target training set. (Spanish)

Figure 2: Histogram of training set.

punctuation marks at the end of sentence, and one of the more common final words without a punctuation mark was precisely "amor". Hence, the network was confused (i.e., the model was poorly estimated) when such punctuation mark did not appear. For dealing with this problem, we devised two possible solutions:

Special word ending: We append a special word `@@` at the end of each sentence. Then, the model is forced to learn that a sentence will always finish with `@@`, and the fore-last word might or might not be a punctuation mark. This was applied as a pre- and post-processing steps, and will be referred to as *special word ending*. The model trained using special word ending achieved 36.4 BLEU in development, and 26.3 BLEU in test. This model was reached after 21th epochs and its size was of 121MB.

Double corpus: We enlarged the training corpus by concatenating all existing sentences with punctuation mark at the end, but removing such symbols. By doing so, the model is able to learn that a sentence can finish with or without punctuation marks. This time, the model had a size of 156MB, and reached 37.3 BLEU in development, and 25.1 in test.

Both techniques described previously solved the problem of punctuation mark expectation. However, since the double corpus strategy produced a larger model, with lower BLEU score, we decided to employ the special word ending technique.

6.1.3 Missed segments

Further analysing the translations generated by our model, noticed an additional problem: in case the segment being translated was composed of several short sentences, only the first of them was being actually translated. For instance, "Thank you. That was really helpful." was translated into "Gracias." ("Thank you.").

To solve this problem, we decided to apply a preprocessing step, consisting separating segments composed by several sentences into different segments, according to punctuation marks ".", "?" and "!", in the case of English language, and also in "¿" in case of Spanish language. We split 86 sentences from test set into 118, given that most of them were composed by short sentences. After this preprocessing step was performed, the translations were correctly generated, reaching 36.4 BLEU in development, and 33.7 BLEU, this last one being the highest score so far.

7 Final evaluation

Table 4 summarizes the BLEU scores obtained after applying each one of the solutions described in Section 6. After applying the normalised infrequency score, the special word ending, and preprocessing composed sentences, we improved the quality of test set by about 7 BLEU points.

Table 4: Translation quality, as measured by BLEU, after applying each technique described. Size is given in MB.

technique	size	BLEU	
		dev	test
Base model	146	39	26.6
Re-scoring of sentences	121	36.3	22.8
Special word ending	121	36.4	26.3
Double corpus	156	37.3	25.1
Sentence splitting	121	36.4	**33.7**

As final evaluation of our translation system, we compared its quality with Google's and Microsoft's mobile translators. The BLEU score on the test set obtained by each of the analysed translators, alongside with their corresponding model sizes, are shown in Table 5. In general, all translators generate good quality hypotheses, although some small differences could be observed. We noticed that our model was especially accurate when using punctuation marks and capital letters, whereas Google's translator introduced punctuation marks in wrong places. Also, only in a few cases, Google's translator, uses capital letters. We believe this is the reason why Google's translator achieved such a low BLEU score, as compared to the other two systems. However, Google's translator features a much smaller than the other two others. Also, Google's and Microsoft's models are bidirectional, which means that the size of our model should be doubled ($2 \cdot 121\mathrm{MB}$) to be comparable.

Table 5: Translation quality and model size comparison for Google, Microsoft and our best model.

	Google	Microsoft	our system
BLEU	16.7	28	**33.7**
Model size	29MB	234MB	121MB
Both directions	YES	YES	NO

8 Conclusions

In this work, we have presented our approach to developing a small size mobile neural machine translation engine, in the specific case of English–Spanish. We leveraged a data selection technique to select more suitable data for real use of our translator. We have presented some adjustments to the selection algorithm the translation quality obtained. Also, we proposed a solution to deal with the problem of repeated words, and another one for dealing with missed sentence translations within some segments. Finally, we compared the quality of our model with Google's and Microsoft's mobile translator versions. We overcome significantly the BLEU score of both translators, partially due to being able to translate punctuation marks and capital letters correctly. Our model reached a size of $121\mathrm{MB}$, which is even much smaller than the size we considered initially as acceptable, presenting good translation quality for the specific purpose (travel domain). The translations obtained by our model are perfectly understandable and fluent, and can be used in a scenario where there is no internet connection. In addition, we are still working on improving its quality and on reducing model size even further, using other effective techniques such as weight pruning.

Acknowledgments

Work partially supported by MINECO under grant DI-15-08169 and by Sciling under its R+D programme.

References

Gascó, G. et al. (2012). Does more data always yield better translations? In *Proc. of EACL*, pages 152–161.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, pages 1735–1780.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint*, arXiv:1412.6980.

Klein, G. et al. (2017). OpenNMT: Open-source toolkit for neural machine translation. *arXiv preprints*, arXiv:1701.02810.

Luong, M. et al. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprints*, arXiv:1508.04025.

Mikolov, T. et al. (2013). Distributed representations of words and phrases and their compositionality. *arXiv preprints*, arXiv:1310.4546.

Papineni, K., , et al. (2002). BLEU: a method for automatic evaluation of machine translation. In *Proc. of ACL*, pages 311–318.

See, A. et al. (2016). Compression of neural machine translation models via pruning. *arXiv preprints*, arXiv:1606.09274.

Sennrich, R. et al. (2015). Neural machine translation of rare words with subword units. *arXiv preprint*, arXiv:1508.07909.

Shterionov, D. et al. (2017). Empirical evaluation of NMT and PBSMT quality for large-scale translation production. In *Proc. of EAMT*, pages 75–80.