KIT's Submission to the IWSLT 2019 Shared Task on Text Translation

Felix Schneider, Alex Waibel

Karlsruhe Institute of Technology

felix.schneider@kit.edu, waibel@kit.edu

Abstract

In this paper, we describe KIT's submission for the IWSLT 2019 shared task on text translation. Our system is based on the transformer model [1] using our in-house implementation. We augment the available training data using backtranslation and employ fine-tuning for the final model. For our best results, we used a 12-layer *transformer-big* configuration, achieving state-of-the-art results on the WMT2018 test set. We also experiment with student-teacher models to improve performance of smaller models.

1. Introduction

The performance of state-of-the-art NMT systems can often be difficult to reproduce. It is highly dependent not only on the amount and kind of training data as well as on a number of subtle hyperparameter choices and implementation details.

The Karlsruhe Institute of Technology perticipated in the IWSLT 2019 shared task on English to Czech text translation. In this paper we describe our data preprocessing, the model architectures we have chosen, as well as attempt to give an exhaustive list of the implementation and training tricks that we used to achieve our final performance.

We describe the data preprocessing in section 2, backtranslation in section 3, training speedups in section 4.1 and finetuning in section 4.2.

2. Data Preprocessing

The allowed training data for the task was the special MUST-C release containing TED talks as well as all WMT 2019 data. We made use of all allowed data, which is broken down in table 1. The allowed parallel data from WMT consists of Commoncrawl, CzEng (which makes up the vast majority of the parallel training data), Europarl, news commentrary and paracrawl.

WMT also provides a large amount of monolingual data for Czech. We used backtranslation (section 3) to synthesize this data into parallel corpora, more than doubling the total amount of available training data.

We performed very little data preprocessing: We first remove noise, by fixing typical encoding errors (such as replacing `u+0080`, the € character in cp1252, but a control character in utf-8, with `u+20ac`, the unicode representation of the € character), and removing html tags which may be left in the data.

Dataset	sentence pairs (K)	words (K) EN	CS
Commoncrawl	162	3 348	2 927
CzEng	57 065	618 423	543 184
Europarl	641	15 623	12 994
MUST-C	128	2 413	2 000
News-Commentary	240	5 166	4 610
Paracrawl	2 982	48 918	44 100
Newscrawl CS 2007-18	72 155	—	1 019 492
Total Parallel	61 218	693 891	609 815
Total	133 373	—	1 692 308

Table 1: Data Overview

Then we filtered the data, with the same method used in [2]: Filter any sentence pair where the Czech side does not contain any accented characters. We also filtered any sentence pair where source and target are the same or where one side is empty. This filtering removes about 8% of the training data.

Then we trained and applied a shared SentencePiece model [3] with a vocabulary size of 32 000. Notably, we did not perform regular tokenization. Finally, we filtered out any sentence pairs where either side is longer than 150 tokens, bringing the total amount of training data to 53 million parallel sentences, 888 million English and 902 million Czech tokens.

3. Backtranslation

We trained a backtranslation model (i. e. Czech to English) with the *transformer-big* configuration [1]. While it would be possible to iteratively train better backtranslation models, the long time it takes to train each model and to apply it to the large amount of training data meant that we could not take this approach, using only our initial backtranslation model. Our model reached a BLEU score of 31.6 on the WMT 2018 test set. Including the backtranslation data, the total amount of training data is 123 million parallel sentences, 2.5 billion English and 2.6 billion Czech tokens.

Model		BLEU		Parameters	training	training
	WMT18	MUST-C	IWSLT19		steps	time
Transformer Big (TF6)	22.94	28.68	26.82	209M	384k	4d23h
TF6 + avg	23.10	28.64	27.08	209M	—	—
TF6 + bt + avg	25.56	27,81	26.86	209M	546k	5d3h
TF8 + bt + avg	26.06	27.88	—	267M	550k	9d10h
TF12 + bt + avg	**26.37**	28.34	27.49	385M	467k	11d7h
TF12 + bt + fine	24.55	29.10	27.96	385M	50	—
TF12 + bt + fine + avg	25.56	**29.51**	28.62	385M	—	—
Transformer Base (tf6) + bt + avg	24.26	25.86	25.16	60M	476k	5d
tf6 + teacher + bt + avg	24.99	26.11	25.73	60M	470k	5d
TF6 + teacher + bt + avg	25.96	27.37	26.85	209M	386k	5d
CUNI WMT 2018 [2]	26.01				788k	8d
LIG			22.72			
CUNI			**29.03**			
Sharp			26.67			
CMU			16.93			
UEDIN			28.07			

Table 2: Main training results

4. Main Training

We trained a number of different configurations of transformer models, evaluating on the WMT 2018 test set, as well as on the MUST-C `tst-COMMON` set. Except for the number of layers, all models are similar to the *transformer-big* configuration: Layer size is 1024, the feed forward hidden dimension is 4096. We did however have better results with a dropout of only 0.1 (rather than 0.3) and only 8 attention heads (rather than 16), differing from [1].

We evaluated models with 6, 8 and 12 layers (each in encoder and decoder), yielding progressively better results, at the cost of evaluation speed and memory usage.

4.1. Training Procedure

We grouped training examples into minibatches of 6000, 5000 or 4000 tokens each for 6, 8 and 12 layers, respectively. In each case, this is the maximum that fits into GPU memory. In order to have more consistent training, we simulated a batch size of 30 000 tokens by accumulating gradients for several steps before updating.

We used Adafactor [4] to train all models, using the parameters from the original paper. We did however increase the number of warmup steps in the learning rate scheduler to 10 000. We saved checkpoints every 2000 training steps.

Each training was run on a single GPU, either an NVIDIA GeForce RTX 2080 with 11GB or an NVIDIA Titan V with 12GB of VRAM. Both of these support two important optimizations, which greatly accelerate training: Mixed-precision training and tensor cores. Using NVIDIA Apex[1], the majority of calculations are performed in half precision, saving a significant amount of memory as well as running about twice as fast. This allowed us to even run the 12 layer model without gradient checkpointing [5].

Another significant (about 20%) speed increase is gained by utilizing the GPU's tensor cores to speed up vector operations. This required padding tensors in the model to factors of 8.

With all optimizations, during training the model processes approximately 23 000, 19 000 or 12 000 tokens per second for 6, 8 and 12 layers, respectively. For the final evaluations, we average the 10 most recent checkpoints for each model.

We use our own implementation of the transformer model to train[2].

4.2. Finetuning

For the final submission, we also finetuned our best model (TF12) by training only on MUST-C. We found that finetuning requires very careful tuning of hyperparameters and close supervision to yield any benefits. For finetuning, we decreased the learning rate to 0.2, the warmup steps to 100 and got the best results after only 70 training steps.

4.3. Student-Teacher

Because we achieved our best results on a very large model, we decided to experiment whether the performance of the very large model could be transferred to a smaller model using sequence-level knowledge distillation [6]. We therefore translated the entire training data with our best performing model before finetuning, the 12 layer *transformer-big* configuration and trained several smaller models with this output as labels.

[1] `https://github.com/NVIDIA/apex`

[2] `https://github.com/felix-schneider/xnmtorch`

It should be noted that for the backtranslation data, after applying the teacher model to it, *both* sides of the parallel data are now synthetic. It may be an interesting future experiment to compare whether this data still gives any benefit.

5. Results

We evaluate our model on the newstest2018 test set, the MUST-C `tst-COMMON` dev set as well as the IWSLT2019 test set. We use sacreBLEU [7] to score our results. For wmt18, we apply postprocessing as in [2], fixing quotation marks. However, as we observed that the quotation marks are not normalized in the MUST-C test set, we do not apply this postprocessing for the other two test sets. The sacreBLEU fingerprint for the reported BLEU score is `BLEU+case.mixed+lang.en-cs+numrefs.1+smooth.exp+test.wmt18+tok.13a+version.1.3.5`.

The in-domain data, i. e. the MUST-C corpus of TED talks makes up only a small part of the training data. This imbalance is increased further by adding the backtranslation data. Because of this, we observed that at first our best performing in-domain model was the very first model trained without any backtranslation data. However, due to their greater representational capacity, the larger models are once again able to generalize to this small portion of the training data and, after finetuning, were able to outperform the simpler model.

Our best performing model on the in-domain data is the finetuned 12-layer model, so this is our primary submission, all other models are contrastive.

5.1. Student-Teacher

For our student-teacher experiments, neither of the models reached the performance of the teacher, the best being a 6-layer *transformer-big*, which came to within 0.4 BLEU of the teacher model. However, both models outperformed models of the same size which were trained on the original data, by 0.7 BLEU (transformer base) or 0.4 BLEU (transformer big). We therefore conclude that in order to achieve the best possible performance with a given number of parameters, student-teacher networks are a viable option regardless of the size of the training data.

6. Conclusions

In this evaluation we trained and compared a number of different transformer configurations for the English to Czech text translation task. Using our largest model, we achieve state-of-the-art performance on the wmt18 test set (evaluation on IWSLT 2019 is still not available).

7. Acknowledgments

The work leading to these results has received funding from the European Union under grant agreement No. 825460 and the Federal Ministry of Education and Research (Germany)/DLR Projektträger Bereich Gesundheit under grant agreement No. 01EF1803B.

8. References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[2] M. Popel, "Cuni transformer neural mt system for wmt18," in *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, 2018, pp. 482–487.

[3] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," *arXiv preprint arXiv:1808.06226*, 2018.

[4] N. Shazeer and M. Stern, "Adafactor: Adaptive learning rates with sublinear memory cost," *arXiv preprint arXiv:1804.04235*, 2018.

[5] T. Chen, B. Xu, C. Zhang, and C. Guestrin, "Training deep nets with sublinear memory cost," *arXiv preprint arXiv:1604.06174*, 2016.

[6] Y. Kim and A. M. Rush, "Sequence-level knowledge distillation," *arXiv preprint arXiv:1606.07947*, 2016.

[7] M. Post, "A call for clarity in reporting BLEU scores," in *Proceedings of the Third Conference on Machine Translation: Research Papers*. Belgium, Brussels: Association for Computational Linguistics, Oct. 2018, pp. 186–191. [Online]. Available: https://www.aclweb.org/anthology/W18-6319