# Kyoto University participation to the WMT 2019 news shared task

**Fabien Cromieres**
Graduate School of Informatics
Kyoto University
fabien@nlp.ist.i.kyoto-u.ac.jp

**Sadao Kurohashi**
Graduate School of Informatics
Kyoto University
kuro@i.kyoto-u.ac.jp

## Abstract

We describe here the experiments we performed for the news translation shared task of WMT 2019. We focused on the new German-to-French language direction, and mostly used current standard approaches to develop a Neural Machine Translation system. We make use of the Tensor2Tensor implementation of the Transformer model. After carefully cleaning the data and noting the importance of the good use of recent monolingual data for the task, we obtain our final result by combining the output of a diverse set of trained models through the use of their "checkpoint agreement".

## 1 Introduction

The 2019 edition of WMT's news translation shared tasks was proposing the German-French pair for the first time. The inclusion of two not-so-closely related languages which both have a richer morphology than English is interesting and can in theory provide additional challenges to the more English-X pairs most frequently used for Machine Translation. Due to the rather large computation time investment required by the training of a modern Neural Machine Translation system, we focused on the German-to-French direction.

Overall, our submission mostly relied on carefully following current best practices for Neural MT, while trying to analyze results and find simple ways to improve them. We used a Transformer sequence-to-sequence model (Vaswani et al., 2017) as our base system. After cleaning and selecting data, we ran experiments with different settings, and finally tried to combine the results of all of these models. In these combination, we tried to use what we dubbed "checkpoint agreement" as a proxy to measure the confidence of a system in its translation.

We could obtain a final improvement of more than +3.5 BLEU over the baseline trained only on bilingual data. However, the greater part of this improvement was simply due to the addition of relevant monolingual data.

## 2 Basic setting

All of our experiments are based on the Transformer sequence-to-sequence model (Vaswani et al., 2017). We used the `Tensor2Tensor` implementation[1] (Vaswani et al., 2018). For hyper-parameters, we used the predefined "big" setting of Tensor2Tensor:

- 6 layers for the encoder
- 6 layers for the decoder
- Hidden size of 1024
- Feed-forward hidden size of 4096
- 16 attention heads

A dropout of 0.3 was used during training. Training was done with the Adam (Kingma and Ba, 2014) algorithm.

Like Popel and Bojar (2018), we also observed that parallel training on a large number of GPUs (thus with a larger effective batch size) was leading to a better final results than only using one or two GPUs at once. We therefore always ran training on five to eight GPUs in parallel[2]. Using a per-GPU batch size of 2048 tokens, this means our effective batch-size was in the range of 10 000 to 16 000 tokens.

Except when indicated otherwise, training was run for at least 500 000 iterations on 8 GPUs (with more iterations when using fewer GPUs to keep the number of training epochs roughly equivalent).

[1] https://github.com/tensorflow/tensor2tensor

[2] Since we are using a shared computation environment, it was not practical to always have a batch of 8 GPUs available for training.



*Proceedings of the Fourth Conference on Machine Translation (WMT)*, Volume 2: Shared Task Papers (Day 1) pages 163–167
Florence, Italy, August 1-2, 2019. 

## 3 Data preprocessing

### 3.1 Data used

For bilingual data, we used the provided corpora: europarl (≈ 1.7M sentence pairs), common crawl(≈ 620k sentence pairs) and news-commentary (≈ 255k sentence pairs). We did not use the paracrawl corpus.

In addition, we also used the 2018 set of the news crawl corpus (≈ 8M sentences) as additional monolingual data.

### 3.2 Data cleaning

Inspecting the training data exposed some minor issues, most notably of encoding and mixed languages (eg. Spanish and English sentences in the French part of the corpus).

Encoding issues were mostly due to sentences encoded in the "Latin-1" character set being mixed with "UTF-8" encoded sentences. Encoding was fixed using the convenient Python library `ftfy`[3] (Speer, 2019). In addition, we removed all uncommon[4] special unicode characters: such characters waste embeddings/softmax capacity for no benefits.

In order to remove non-French/German sentences from the corpus, we chose to apply a simple heuristic that was fast enough to be applied to millions of sentences. Comparing corpuses of French, German, English, Spanish and Portuguese, we selected "characteristics" words and characters that were frequent in French or German but rare or inexistent in other languages (eg. character "ç" or words "mais", "donc" for French). We then filtered out any sentence longer than 4 words that did not contain any of these characteristics words/characters. A few dozen thousands sentences were filtered out this way, with a rate of false positive empirically estimated at less than 1%.

### 3.3 Subwords units

As is now common practice, we tokenized all data with subwords units. We relied on the subword tokenization algorithm implemented in Tensor2Tensor. This algorithm is different from the popular BPE tokenization algorithm (Sennrich et al., 2015b), but is expected to be similarly efficient. We targeted a joint subword vocabulary of 32 000 units. In other experiments we had observed that smaller subword vocabulary size can work better for language pairs with many common prefixes (such as Spanish and Portuguese); this did not seem to be the case here.

## 4 The importance of recent news data

### 4.1 Baseline Experiment and Error Analysis

We ran a first baseline experiment using the setting described in section 2 and the cleaned bilingual data of section3. We obtained a cased BLEU score of 33.18.

Manual inspection of the results showed us that the trained model could have serious trouble translating terms or personal names who had only recently appeared in the news. A typical example would be the translation of German "Gelbwesten" ("Yellow vests") into French "Gibiers jaunes" ("Yellow game[5]"), instead of the correct "Gilets jaunes". The "Yellow vests" are a French protest movement that appeared during 2018 fall, and has received much attention in news from that time into 2019. The collocation "Gilets jaunes" is therefore unlikely to appear in the bilingual training data (which is typically older), which explains why the model seems to prefer the similar (in terms of subwords units) "Gibiers jaunes".

Another common problem was the literal translation of German terms that are normally quoted as-is in French News. For example, the German political Party "Die Linke" ("The Left") was translated as "le parti de gauche" ("the left-wing party"), even though French journalists usually refer to it with its German name ("le parti Die Linke").

### 4.2 Backtranslating recent news

The problem above prompted us to make use of the provided monolingual data, which includes more recent pieces of news. We used backtranslation (Sennrich et al., 2015a), which is currently the most popular approach for using monolingual data in NMT. Concretely, we trained a French-to-German model with the sam bilingual data, and backtranslated into German the 2018 section of the news crawl data. We expect that using the data from previous years would have been useful as well, but we focused on the year 2018, first out

[3] https://github.com/LuminosoInsight/python-ftfy

[4] our definition for uncommon was any character whose frequency rank was beyond 500 and that was not appearing in any sentence of the dev set.

[5] with the meaning of "hunted animal", not (board) game.

of concern with time constraints, and second considering the most recent pieces of news should be by far the most relevant to translate the development set and the test set (which are mostly made of recent news).

We added the backtranslated data to the bilingual data and trained a new model. The new model had a cased BLEU score of 35.92, almost a 3 BLEU improvement. Manual inspection showed a large improvement in the translation of recent terms (eg. "Gelbwesten" was now correctly translated as "Gilets jaunes"). However, the problem of litterally translating terms such as "Die Linke" remained.

### 4.3 Checkpoint Averaging

In order to improve results further, we tried checkpoint averaging[6]. Averaging was done over 20 checkpoints, each checkpoint being taken with a one hour interval. This led to a modest improvement of +0.2 BLEU.

## 5 Output combination

An efficient technique for improving the results of a given Neural MT system is to train several models and to compute their ensemble translations. The ensemble translation is obtained by letting each model predict the probability of the next words to be generated, and then combine these probabilities to choose which word is actually generated to create the final translation. The price for the improved translation quality is an increase in training time, decoding time and memory usage proportional to the number of models used.

In the course of this shared task, we trained several different models, but could not use classic ensemble techniques to combine them, due to several factors: absence of a ready-made ensemble implementation in Tensor2Tensor and models being trained with different preprocessing (eg. different subword units). This is why we considered a simple system combination algorithm that proved to be useful.

### 5.1 Checkpoint agreement

While we could have used some more advanced system combination techniques, such as (Freitag et al., 2014), we experimented with the idea that what we call "checkpoint agreement" gives us useful indication about the reliability of a given translation.

The idea is, essentially, to keep many checkpoints for each models (as in section 4.3). Each checkpoint can be used to generate a translation candidate. If all checkpoints generate the same translation candidate, we can have higher confidence in the translation than if they all generate different translation candidates. Further, if twenty checkpoints lead to a set of, say, three different translations, we can have more confidence in the translation that was generated by the most checkpoint. This provides us with a model-independent and implementation-independent way to estimate the confidence we can have in the output of a model. We empirically check to which extent this is true in section 5.2.

Then, in section 5.3, we make use of this checkpoint agreement to simply combine the output of different systems.

### 5.2 Empirical evaluation of checkpoint agreement

We first evaluate this idea with the checkpoints of a single model. The first thing to verify is whether different checkpoints actually produce different translations. Using the same checkpoints as in section 4.3 (ie. 20 one-hour-spaced checkpoints), we compute the translations they generate for the development set. We find that for 9% of the input sentences, the 20 checkpoints generate the same translation. For 2% of the input sentences, they all produce distinct translations. For the remaining 89% of inputs, there therefore exists at least one translation candidate generated by at least two checkpoints.

If, for each input, we select the most often generated translation candidate, we obtain a BLEU score improvement of +0.3 ("selection by checkpoint agreement" in table 1). This is a bit better than simply doing checkpoint averaging, but of course it takes 20 times more decoding time to obtain a translation.

### 5.3 Models output combination through checkpoint agreement

Given that we now have a model-independent way of estimating the reliability of a translation, we can use this to combine the output of different models. This is what we try here.

[6] using the t2t-avg-all script (Popel and Bojar, 2018).

| Model | Dev cased BLEU | Improvement |
|---|---|---|
| Baseline (bilingual data only) | 33.18 | - |
| +2018 news data (monolingual) | 35.92 | +2.74 |
| Checkpoint averaging | 36.12 | +0.2 |
| Selection through checkpoint agreement | 36.23 | +0.31 |
| All Models combined with Checkpoints agreement | **36.73** | +0.81 |

Table 1: Cased-BLEU score on the development set for the different experiments. Improvements of checkpoint averaging and checkpoint agreement combination are computed with respect to the "Baseline+2018 monolingual data" BLEU.

#### 5.3.1 Combined models

The additional models we trained include:

- A model with a subword vocabulary size of 8000
- A model with a subword vocabulary size of 512
- A model trained with a reversed French-side word order

The models with alternative vocabulary size were trained to evaluate the effect of the coarseness of the subword segmentation on the final quality. We had observed this can have an important impact on language pairs with many common substrings (like Spanish and Portuguese), but did not find it to give better results for German-French.

The model trained with a reversed French-side order was to evaluate if the model could produce better results by generating the translation from right-to-left. Again, we did not find this to lead to better results in our case.

Note that we could not combine these models with a "classic" ensemble of models: due to different subwords units or word order generation, these models cannot compute consistent "next-word" probabilities that could be easily combined.

#### 5.3.2 Results

We combine the results of our models through a simple "majority vote" weighted by the confidence deduced from the checkpoint agreement. We could possibly obtain better results by integrating the confidence score given by checkpoint agreement in a more complex system combination algorithm such as Freitag et al. (2014).

We obtain an improvement of +0.8 BLEU ("All Models Checkpoints combination" in table 1).

## 6 Conclusion

We experimented with the translation of German into French in the context of the WMT 2019 shared tasks. Our approach mostly followed the currently known best practices. We detailed how we cleaned an pre-processed the training data, and, in particular, we found it crucial for the task to make good use of recent monolingual data. We also evaluated the idea that a set of checkpoints from a given training run can be used to evaluate the confidence in the quality of the output of a model. We used this to combine simply the output of a set of different models.

## References


Markus Freitag, Matthias Huck, and Hermann Ney. 2014. Jane: Open source machine translation system combination. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 29–32.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Martin Popel and Ondřej Bojar. 2018. Training tips for the transformer model. *The Prague Bulletin of Mathematical Linguistics*, 110(1):43–70.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015a. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015b. Neural machine translation of rare words with subword units. In *WMT2015*.

Robyn Speer. 2019. ftfy. Zenodo. Version 5.5.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan N. Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob

Uszkoreit. 2018. Tensor2tensor for neural machine translation. *CoRR*, abs/1803.07416.

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