Analyzing Knowledge Distillation in Neural Machine Translation

Dakun Zhang, Josep Crego, Jean Senellart

SYSTRAN / 5 rue Feydeau, 75002 Paris, France

`firstname.lastname@systrangroup.com`

Abstract

Knowledge distillation has recently been successfully applied to neural machine translation. It allows for building shrunk networks while the resulting systems retain most of the quality of the original model. Despite the fact that many authors report on the benefits of knowledge distillation, few have discussed the actual reasons why it works, especially in the context of neural MT. In this paper, we conduct several experiments aimed at understanding why and how distillation impacts accuracy on an English-German translation task. We show that translation complexity is actually reduced when building a distilled/synthesised bi-text when compared to the reference bi-text. We further remove noisy data from synthesised translations and merge filtered synthesised data together with original reference, thus achieving additional gains in terms of accuracy.

1. Introduction

Neural machine translation (NMT) achieves state-of-the-art results in several translation tasks and for multiple language pairs [1, 2]. Equivalent to its phrase-based predecessor, neural networks directly learn from parallel bi-texts, consisting of large amounts of human created sentences with their corresponding translations. Therefore, the quality of an MT engine is heavily dependent on the amount and quality of parallel sentences.

Several techniques aimed at boosting the quality [3, 4] and quantity [5] of training data are successfully applied to neural MT. Parallel to these techniques, knowledge distillation [6] has attracted the focus of many researchers given its simplicity and the quality of the results. However, despite the fact that a growing number of private entities have begun to include distillation into their NMT systems [7, 8, 9] and that knowledge distillation has demonstrated its performance for multiple tasks [10, 11, 12], none of them give a detailed analysis of the reasons why it works.

In most cases, the availability of parallel corpora is a prerequisite to build Neural MT systems. The process of compiling parallel bi-texts is usually composed of several steps: crawling, filtering, cleaning, etc. As a result, parallel corpora usually contain parallel sentences that are often not as parallel as one might assume. And even for parallel sentences that truly convey the same meaning, in some cases translations follow a more or less word-for-word pattern (more literal translations). While in many other cases, translations show greater latitude of expression (more flexible translations) with higher degrees of variability, which humans often judge as good. However, machine translations are usually "closer" in terms of syntactic structure and present lower levels of variability when considering word choice. It is rather an intuitive idea that feeding more "literal" translations to a neural MT network should facilitate the training process compared to training with less literal translations (original bi-text).

In this paper, we report on the results of experiments where we automatically distill a human translation bi-text which is then used to train neural translation engines. Thus, aiming at boosting the learning ability of neural translation models. We show that the resulting models perform even better than a neural translation engine trained on the original reference dataset.

Our contribution is as follows:

- We analyse the reason why and how distillation works for neural machine translation.
- We analyse in detail the difference between original reference translations and synthesised translations.
- We further filter out noise from synthetic data and measure the impact on using both synthetic and reference translations.

The remainder of this paper is structured as follows. Section 2 briefly surveys previous work. Section 3 outlines our neural MT engine and details the distillation approach presented in this paper. Sections 4, 5 and 6 report training configurations and experimental results with detailed analysis. Section 7 draws conclusions and outlines future work.

2. Related Work

Sequential knowledge distillation for neural machine translation was first detailed by [6]. The authors trained a smaller student network to perform better by learning from a larger teacher network allowing more compact neural MT models to be built. [7] followed this idea and proposed a similar language simplification method based on distillation. They reported improvements on English to German and English to French translations. [8] further demonstrated distillation experiments from both an ensemble teacher model and a single model. After that, they improved training efficiency and performance by removing noisy sentences from the training corpus. As a comparison, we performed detailed experiments and analysed the reasons why and how distillation works for neural machine translation.

Other than neural MT, [10] and [11] show that distillation also works well when transferring knowledge from a network ensemble or from a large highly regularised model into a smaller, distilled network for image classification and speech recognition. [12] applied distillation based on a search based structured prediction on dependency parsing and machine translation. These works demonstrate that knowledge distillation is adapted to many different tasks. In this work, we focus on the influence of knowledge distillation to neural machine translation and suggest directions for further improvements.

3. Neural Machine Translation

We train two types of NMT systems in this work, an RNN-based model and a Transformer-based model. The RNN model follows the architecture presented in [13]. It is implemented as an encoder-decoder network with multiple layers of an RNN with Long Short-Term Memory hidden units [14]. The Transformer model follows the work in [15]. It encodes the representation of sentences in a way a self attention only and is reported as the current state-of-the-art in many machine translation tasks [1, 9].

For the RNN model, the encoder is a bidirectional neural network that reads an input sequence $s = (s_1, ..., s_J)$ and calculates a forward sequence of hidden states $(\overrightarrow{h_1}, ..., \overrightarrow{h_J})$, and a backward sequence $(\overleftarrow{h_1}, ..., \overleftarrow{h_J})$. The decoder is an RNN that predicts a target sequence $t = (t_1, ..., t_I)$, being J and I respectively the source and target sentence lengths. Each word t_i is predicted based on a recurrent hidden state h_i, the previously predicted word t_{i-1}, and a context vector c_i. We employ the attentional architecture from [16] and use the implementation of *OpenNMT*[1]. Additional details are given in [17].

Unlike the RNN model, the Transformer model directly models the representations of each sentence with a self-attention mechanism. Hence, it reduces the number of operations related between tokens in different positions, especially for distant positions, in input and output sequence. The Transformer model stacks a so-called multi-head self attention layer and a position-wise, fully connected layer for non-linear conversions on both the encoder and decoder side. On the decoder side, it uses masked self-attention to prevent positions to attend to unseen positions.

The notion of time step is encoded automatically in the sequence in the RNN model. Whereas the Transformer model uses positional embedding to record the position information of each word in the sequence. In addition, the Transformer model is easy to be parallelized for the MLE training process across multiple GPUs. This allows the benefit of accelerating the training speed compared with the RNN model. In this work, we use the implementation of *OpenNMT-tf*[2] to train our Transformer based systems.

3.1. Knowledge Distillation

Knowledge distillation is a method to train different deep neural networks on the same data. Information learned from a large teacher model with the original reference data can be learned quite well with a smaller student model with the synthesised data [10]. Thus, a compact smaller model is generated and used to replace the larger model, especially in some resource limited devices.

For machine translation, we follow the approach described by [6]. The machine translation model is trained to minimise the Kullback-Leibler divergence, either between the model distribution and ground-truth distribution $\mathcal{L}_{NLL}$, or between the model distribution and synthesised data distribution $\mathcal{L}_{KD}$, which is from the teacher system, or an interpolation of both:

$$\mathcal{L} = (1 - \alpha) \cdot \mathcal{L}_{NLL} + \alpha \cdot \mathcal{L}_{KD}$$

In [6], three distillation methods are proposed by tuning the weight (α) in sequence-level loss. We simplify this process and perform experiments with $\alpha = 1$ in this work. First, we train a teacher system with the original source/target data. Second, we train another student system with the source/synthesised target data. The synthesised target language data is generated by running beam search (with beam size 5) over the training set with the teacher system (forward translation),

The objective during the training of the student system is the same as the teacher system. The only difference is the student system's objective is not to maximise log likelihood toward the ground-truth reference, but toward a generalised "soft" target, which is from the teacher system. From this point of view, the student system is directed by how the teacher system acts. Hence, in general a stronger teacher system is preferred. E.g. an ensemble teacher system is used in [8].

4. Experimental Conditions

4.1. Data

Experiments are performed using a preprocessed and tokenised version of WMT English-German translations[3]. The training set contains $4.5M$ sentence pairs. We use *newstest2013* as validation set and both *newstest2014* and *newstest2015* as test sets. We applied joint byte-pair encoding (BPE) [18] with $32K$ merge operations. The actual training vocabulary size is of $34K$ tokens after BPE tokenization. We

[1]`https://github.com/OpenNMT/OpenNMT`

[2]`https://github.com/OpenNMT/OpenNMT-tf`

[3]The corpus is already tokenised and can be downloaded from `https://nlp.stanford.edu/projects/nmt`

tfm	*L*arge	N=6, d=512, d_{ff}=4096, h=8
	*M*iddle	N=6, d=512, d_{ff}=2048, h=8
	*S*mall	N=4, d=256, d_{ff}=2048, h=8
rnn	*L*arge	Bi-LSTM, 4x1024, emb=512
	*M*iddle	Bi-LSTM, 2x1024, emb=512
	*S*mall	Bi-LSTM, 2x512, emb=512

Table 1: Configurations of the networks used in this paper. In the remainder of this paper we use respectively *tfm.L*, *tfm.M*, *tfm.S*, *rnn.L*, *rnn.M* and *rnn.S* to represent systems trained with different configurations.

limit sentence length to 100 in both source and target sides (excluding 0.31% of the training corpus). After decoding, we remove BPE joiners and evaluate the tokenised output with *multi-bleu.perl* [19].

4.2. Network Configuration

In this paper, we employs several neural MT models based on the Transformer [15] and RNN [13] models. Three different systems are used for each architecture, which differ in network size. Details of the system configurations are given in Table 1.

For RNN based systems, we use stochastic gradient descent, a mini-batch size of 64 in segments with dropout probability set to 0.3. We train our models during 18 epochs and evaluate the performance of the last epoch. Initial learning rate is set to 1.0 and we start decaying after epoch 10 by a fixed decay rate of 0.7. In decoding, we use a beam size of 5.

In the case of Transformer systems, we use Lazy Adam optimiser, which starts the learning rate at 1.0. We train the systems with a batch size of 8, 192 in tokens and save checkpoints in every 5, 000 steps. We terminate training after $400K$ iterations and average the last 8 checkpoints to get the final evaluation.

5. Analysis of Synthetic Translations

Aiming for a better understanding of the translated languages, we first conduct an elementary human analysis of the German hypotheses (synthesised translations) produced by the best performing network. We observe that in many cases, automatic translations produced by our neural MT systems consist of paraphrases of the reference translations. While both, reference and automatic translations, preserve the same meaning and are grammatically correct, automatic translations are closer in terms of syntactic structure to the source sentences than reference translations, which seams a key factor to train machine translation systems.

Examples in Table 2 illustrate this fact. In the first example, the English and German synthetic sentences follow a very similar structure. While in the German reference translation, the sentence: *you are sure to find the nightclub you like* is expressed by *clubbers (Disco-Gänger) are guaranteed*

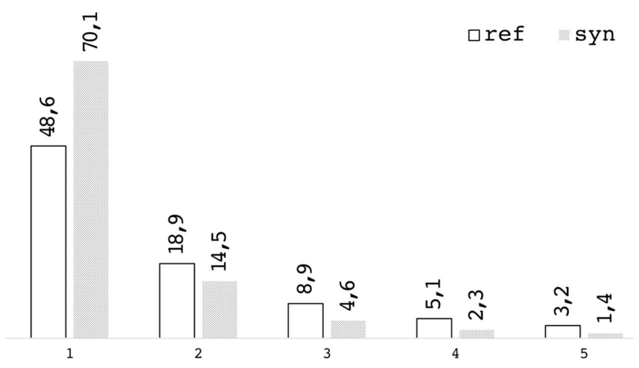

Figure 1: Histogram indicating the percentage (%) of source words aligned to n (x-axis) distinct target words in the training set.

to get their money (common garantiert auf ihre Kosten). In the second example, we can see a very similar situation. The verb *verwendet* is shifted to the end of the sentence when comparing the structure of the English and the German synthetic sentences. In contrast, the reference German translation employs a greater latitude of expression.

Next, we conduct several experiments in order to confirm the hypothesis that automatic translations are closer to the input sentence than reference translations. We compare reference German translations (*ref*) to automatic translations (*syn*) produced by our neural MT network over the entire training set. In our experiments, we employ word alignments computed using `fast_align`[4].

5.1. Translation Fertility

First, we measure translation fertility. We identify the number of different target words aligned to each source word in the training corpus. We regard this number as the "fertility" between parallel sentences. Figure 1 shows a histogram indicating the percentage of source words aligned to n distinct target words.

As it can be seen, English words are in average related to less German words in the case of the automatic translations (*syn*) than for reference translations (*ref*). 70.1% of English tokens are aligned to a single German token in the case of automatic translations while this number is reduced to 48.6% in the case of reference translations. As expected, the opposite situation is also observed when considering target words aligned to multiple source words. In this case, reference translations show always a higher percentage of tokens.

5.2. Translation Distortion

In addition, we compare the translation distortion in order to validate the closeness (similarity) of syntactic structures. The translation distortion is calculated by the number of crossed alignments on automatic (*syn*) and reference (*ref*) translations. Given a sentence pair with its set of alignments, we

[4] `https://github.com/clab/fast_align`

Src:	[In Cala Ratjada]$_1$ [you are sure to find]$_2$ [a nightclub]$_3$ [you like]$_4$.
Ref:	**Disco-Gänger kommen** [in Cala Ratjada]$_1$ **garantiert auf ihre Kosten**.
Syn:	[In Cala Ratjada]$_1$ [finden Sie sicher]$_2$ [einen Nachtclub]$_3$, [den Sie mögen]$_4$.
Src:	[Your personal information]$_1$ [will only be]$_2$ [used]$_3$ [to process your booking]$_4$.
Ref:	**Sie** [werden nur]$_2$ **in dem Umfang weitergegeben , wie es** [für eine Buchung]$_4$ **notwendig ist**.
Syn:	[Ihre persönlichen Daten]$_1$ [werden nur zur]$_2$ [Bearbeitung Ihrer Buchung]$_4$ [verwendet]$_3$.

Table 2: Examples of English-to-German translation. Subscripts in these examples indicate the alignment between multi-words expressions.

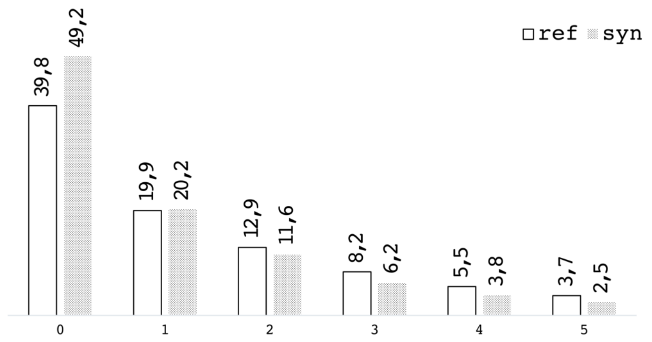

Figure 2: Difference in number of crossed alignments (in percentage (%)) between reference and synthesised translations. 0 means no crossed alignment, i.e. monotonic, between source and target sentences.

compute for each source word s_i the number of alignment crossings between the given source word and the rest of the source words. We consider that two alignments (i, j) and (i', j') are crossed if $(i-i')*(j-j') < 0$. Figure 2 illustrates the difference in number of crossed alignments between reference and synthetic translations.

As it can be seen, automatic (*syn*) translations show a higher number of words with no crossed alignments (49.2%) than reference (*ref*) translations (39.8%). In contrast, when considering larger numbers of crossings, the reference data set shows higher ratios than the automatic data set. This shows the synthesised target is much "closer" to the source in grammatical order compared with original reference.

Note that automatic translations carry important levels of noise (translation errors) that cannot be neglected from the view of a human. However, since it's the generalised output from the teacher system, it is indeed compatible to the machines. The next section evaluates the suitability of automatic translations as a training set for our neural MT systems compared to reference translations.

6. Results

6.1. Basic systems

We first summarise translation accuracy results (BLEU scores) of our 6 basic systems learned over the reference training set. As shown in Table 3, systems implementing the

Config	*newstest2014*	*newstest2015*
*tfm.L**	**27.87**	**30.04**
*tfm.M**	27.59	29.73
tfm.S	24.60	27.58
rnn.L	24.11	26.62
rnn.M	24.11	26.74
rnn.S	22.94	25.85

Table 3: BLEU scores on systems trained over the original dataset. Systems with * are candidates for teacher systems.

Transformer architecture outperform RNN networks. The best score is achieved by *tfm.L*, which is 27.87 for *newstest2014* and 30.04 for *newstest2015*. The smallest transformer network (*tfm.S*) clearly outperforms the largest RNN network (*rnn.L*) in about 0.5 BLEU points for *newstest2014*. We choose the best two systems *tfm.L* and *tfm.M* as the candidates for teacher systems.

According to BLEU scores, similar performance is obtained by large and middle size versions of the Transformer and RNN models. However, the smallest systems show a clear drop in performance for both architectures. We argue that, when the network is big enough, the performance relies more on the amount of training data. That is, if the training data is fixed, there is a proper size of the neural network to learn the information embedded inside this training data. An even larger network could achieve a better performance, but not significantly.

6.2. Comparison between different teachers

For the distillation based method, the teacher system is important because its output will be used as the student's training reference. As shown in [8], a student system trained with an ensemble teacher usually performs better than that trained with a single teacher system. In our experiments, we train student systems based on a single teacher system. Table 3 shows that system *tfm.L* achieves similar accuracy compared with system *tfm.M* with original dataset. This means that there is no major difference between *tfm.L* and *tfm.M*, and these two systems can both be used as teacher systems. Therefore, in this section, we compare the performance between student systems trained on different teachers. We show that a strong teacher will lead to better students. Results are

Student System	Teacher System	*newstest 2014*	*newstest 2015*
tfm.S	*tfm.L*	26.07	28.96
	tfm.M	25.33	27.79
rnn.S	*tfm.L*	24.84	27.99
	tfm.M	24.22	27.10

Table 4: Results of comparison between different teacher systems.

	synthesised data	*newstest 2014*	*newstest 2015*
tfm.S	100%	26.07	28.96
	95%	**26.24**	**29.42**
	90%	26.20	29.21
rnn.M	100%	26.22	28.87
	95%	26.11	28.92
	90%	25.98	28.80

Table 5: Impact on different amounts of synthesised data for student systems by removing noisy data from the output of the teacher system *tfm.L*.

shown in Table 4.

Both student systems outperform basic systems trained with original data (in Table 3). For *newstest2014*, systems trained with original reference data achieve 24.60 for model *tfm.S* and 22.94 for model *rnn.S*. When systems are trained with synthesised data from teacher system *tfm.L*, the performance improves to 26.07 (+1.47) and 24.84 (+1.90) respectively. This proves the distillation method works for neural machine translation.

We also notice that the difference between the two teacher systems *tfm.L* and *tfm.M* trained with original data is 0.28 for *newstest2014*. However, the difference between same student model trained with these two teachers increases to 0.74 for *tfm.S* and 0.62 for *rnn.S*. We argue that this is because of noisy data present in the synthesised target side. We found from the training synthesised data that some target sentences are exactly the same regardless of the source sentences. We further checked the original reference target sentence and confirmed that it is because the original bi-text is not parallel (noise in the original training data). During the training of the teacher system, neural models tend to normalise all these "bad" instances into a uniformed one by minimising the log likelihood. However, during the training of the student system, such noise is amplified and leads to the larger gap between the different systems. We therefore analyse in detail the influence of noise in the next section.

6.3. Influence of synthesised data noise

Based on Table 4, we can conclude that a stronger teacher is usually beneficial to train student systems. Similarly, we compared two similar student systems *rnn.M* and *rnn.S* based on teacher *tfm.L*. We found *rnn.M* can achieve 26.22 in BLEU score in *newstest2014*, which is $+1.38$ BLEU points higher than *rnn.S*. We therefore choose *tfm.L* as our teacher system and *tfm.S* and *rnn.M* as our default student systems in the following experiments. In this section, we train *tfm.S* and *rnn.M* with a different proportion of the synthesised data to see the influence of data noise for student systems.

As we showed in section 5, the synthesised data is the translation of the whole training set by a teacher model. Sequences in these generated hypotheses contain noise as they are from machine translated results. Noise includes ungrammatical sentences, wrong words selection, word ordering problems, etc. Also there are inconsistent target sentences with the source in semantic and under/over translation[5] problems in the synthesised data. We regard all these problems as noise because they are not correct translations.

We use an embedding based method proposed by [20] to calculate the similarity between source and target sentences. In [20], a sentence embedding was first built based on word similarity, relying on a neural architecture, which is able to identify several types of cross-lingual divergences. The resulting embeddings are then used to measure semantic equivalence between sentences[6]. In our case, the target sentences are synthesised data from a teacher system. We filter out sentence pairs which are not similar based on the similarity score and train the student system with the remaining data. Table 5 shows the results from distilled *tfm.S* and *rnn.M* systems.

We compare two different student systems, Transformer based *tfm.S* and RNN based *rnn.M*. For *rnn.M* system, we can not see any gains by removing different proportions of noisy data. While for *tfm.S* system, when we remove 5% noisy data, we found an increase from 26.07 to 26.24 (+0.17) in BLEU score for *newstest2014*, which is also the best performance we have achieved until now.

We argue that the Transformer based system is more sensitive to the noisy data compared with the RNN based system. When a little amount noisy data (e.g. 5%) is removed, the performance improves because the remaining 95% is enough to train a good system. Along with the further reduction to 90%, both the Transformer based model and the RNN based model starts to decrease because there are fewer instances used for training. This also shows that the size of training data is another crucial factor to the final accuracy.

Another interesting phenomenon is that the differences between student systems with different architecture are not so big compared with systems trained with an original reference. In this experiment, *rnn.M* performs well compared with *tfm.S* given synthesised training data, while it is not the case for them to be trained with original data. We speculate that this is because the diversity in distilled bi-text is much

[5]During translating, under translation is when the words/phrases in the source sentence are missing (not translated) in the target side. Over translation is when there are duplicated translations for the same source words/phrases present on the target side.

[6]`https://github.com/jmcrego/similarity`

	merged data (hyp+ref)	*newstest 2014*	*newstest 2015*
tfm.S	95%+5%	**26.27**	29.09
	90%+10%	26.20	**29.11**
	80%+20%	25.76	28.88
	50%+50%	25.58	28.54
rnn.M	95%+5%	25.94	28.83
	90%+10%	25.52	28.76
	80%+20%	25.66	28.44
	50%+50%	25.04	27.60

Table 6: Results of merged corpus with synthesised data and original data. 95%+5% means the training data is composed of 95% data from the synthesised target translations (hypothesis) according to the similarity and additional 5% data from original target data (reference).

less than in the original reference. Systems with different architecture show different sensitivity of data diversity. That is also to say, the distilled bi-text is consistent and compact, which is much suitable for training machine translation systems.

6.4. Replacing noisy data with the original reference

Previous experiments show that systems trained with synthesised data usually perform better than systems trained with original reference data. At the same time, when we remove some noisy data from the synthesised training set, there is further improvement for the student system. In this section, we test experiments with merged synthesised data and the original reference as the training corpora to see which part is more crucial for the final performance.

First, we use a similarity score between source and target sentences calculated beforehand to rank the synthesised data. We select top $X\%$ "similar" data and for the remaining $(1 - X\%)$ data, we replace the target side with the original references to merge into a new data set. Table 6 shows the evaluation results on two student systems *tfm.S* and *rnn.M*.

Results show that synthesised data greatly contribute to the final accuracy. Along with the increase of data from the synthesised target side, the performance increases as well for both *tfm.S* and *rnn.M*. However, when comparing with systems trained with 100% synthesised data or systems trained with 95% synthesised data, the performance is different between the Transformer and RNN based models.

For *tfm.S*, when training with merged 95% data, the system reaches its highest performance in *newstest2014*. As for *rnn.M*, on the contrary, the performance starts to drop a little. It is even worse than training with the filtered 95% synthesised data. We argue this inconsistency stems from the architecture differences. The performance of the RNN based model is difficult to be improved. However, the Transformer-based model, due to its sensitivity to data diversity, can perform quite well as long as the training data is well controlled.

	2X data (hyp+ref)	*newstest 2014*	*newstest 2015*
tfm.S	100%+100%	25.95	28.74
rnn.M	100%+100%	25.78	28.85

Table 7: Results of the concatenated synthesised data and original reference. Twice the training cost is needed as the corpus is doubled.

6.5. Doubled training data

Lastly, we combine the synthesised data with the original reference. This will double the training data size and lead to twice the training cost. However, based on the results shown in Table 7, we found that even though the data size was doubled, we could not achieve further improvement.

We analyse that this is because the synthesised data is generated from the teacher system. All the information embedded in the synthesised data is already in the original data. In other words, adding such synthesised data is somewhat equivalent to adding the same original data. It is similar to training the system with the same data but with a 2X data size. Furthermore, considering noise existed in the synthesised data, systems trained with this 2X data is even worse than the original doubled data.

7. Conclusions

We have presented distillation experiments for neural machine translation. Results indicate the suitability of using synthetic translations to train neural MT systems. Higher accuracy results are obtained by the systems when trained using synthetic data. We show data noise present in both the original translation references and synthesised translations is a key factor that influence the final performance.

Meanwhile, the Transformer-based and RNN-based systems perform differently given different amounts of synthesised and/or merged data. We further prove that much "closer" translations contribute the most to the system's accuracy and that is also the reason why distillation works for neural machine translation.

In conclusion, we summarise that for neural machine translation:

- Having a stronger teacher system usually helps the student systems.
- Removing noise from the synthesised data of teacher systems also helps.
- Replacing noisy data with the original reference data can get further improvements.

A clear drawback of distillation-based methods is the efficiency of the training process. Student systems must be trained after the teacher system. In addition, we must also consider the cost of translating the entire training set. As such, one solution is to integrate this procedure during the training process. Since data noise is one of the key factors during training, we believe that identifying noisy instances during training may alleviate the time problem. We leave that for future work.

8. Acknowledgements

We would like to thank all the anonymous reviewers for their insightful comments.

9. References

[1] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-attention with relative position representations," *CoRR*, vol. abs/1803.02155, 2018. [Online]. Available: http://arxiv.org/abs/1803.02155

[2] O. Bojar, Y. Graham, A. Kamran, and M. Stanojević, "Results of the wmt16 metrics shared task," in *Proceedings of the First Conference on Machine Translation*, Berlin, Germany, August 2016.

[3] Y. Vyas, X. Niu, and M. Carpuat, "Identifying semantic divergences in parallel text without annotations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, 2018, pp. 1503–1515. [Online]. Available: http://aclweb.org/anthology/N18-1136

[4] H. Schwenk, "Filtering and mining parallel data in a joint multilingual space," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, 2018, pp. 228–234. [Online]. Available: http://aclweb.org/anthology/P18-2037

[5] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2016, pp. 86–96. [Online]. Available: http://www.aclweb.org/anthology/P16-1009

[6] Y. Kim and A. M. Rush, "Sequence-level knowledge distillation," *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1317–1327, November 2016.

[7] J. M. Crego and J. Senellart, "Neural machine translation from simplified translations," *CoRR*, vol. abs/1612.06139, 2016. [Online]. Available: http://arxiv.org/abs/1612.06139

[8] M. Freitag, Y. Al-Onaizan, and B. Sankaran, "Ensemble distillation for neural machine translation," *CoRR*, vol. abs/1702.01802, 2017. [Online]. Available: http://arxiv.org/abs/1702.01802

[9] A. Birch, A. Finch, M.-T. Luong, G. Neubig, and Y. Oda, "Findings of the second workshop on neural machine translation and generation," *arXiv preprint arXiv:1806.02940*, 2018.

[10] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *CoRR*, vol. abs/1503.02531, 2015. [Online]. Available: http://arxiv.org/abs/1503.02531

[11] J. H. Wong and M. J. Gales, "Sequence student-teacher training of deep neural networks," 2016.

[12] Y. Liu, W. Che, H. Zhao, B. Qin, and T. Liu, "Distilling knowledge for search-based structured prediction," *CoRR*, vol. abs/1805.11224, 2018. [Online]. Available: http://arxiv.org/abs/1805.11224

[13] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *CoRR*, vol. abs/1409.0473, 2014, demoed at NIPS 2014: http://lisa.iro.umontreal.ca/mt-demo/. [Online]. Available: http://arxiv.org/abs/1409.0473

[14] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," *CoRR*, vol. abs/1409.2329, 2014. [Online]. Available: http://arxiv.org/abs/1409.2329

[15] 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. 6000–6010.

[16] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, September 2015, pp. 1412–1421. [Online]. Available: http://aclweb.org/anthology/D15-1166

[17] J. Crego, J. Kim, G. Klein, A. Rebollo, K. Yang, J. Senellart, E. Akhanov, P. Brunelle, A. Coquard, Y. Deng, S. Enoue, C. Geiss, J. Johanson, A. Khalsa, R. Khiari, B. Ko, C. Kobus, J. Lorieux, L. Martins, D. Nguyen, A. Priori, T. Riccardi, N. Segal, C. Servan, C. Tiquet, B. Wang, J. Yang, D. Zhang, J. Zhou, and P. Zoldan, "Systran's pure neural machine translation systems," *CoRR*, vol. abs/1610.05540, 2016. [Online]. Available: http://arxiv.org/abs/1610.05540

[18] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," *arXiv preprint arXiv:1508.07909*, 2015.

[19] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, *et al.*, “Moses: Open source toolkit for statistical machine translation,” in *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*. Association for Computational Linguistics, 2007, pp. 177–180.

[20] M. Q. Pham, J. Crego, J. Senellart, and F. Yvon, “Fixing translation divergences in parallel corpora for neural mt,” *Conference on Empirical Methods in Natural Language Processing*, 2018.