Learning to Generate Word- and Phrase-Embeddings for Efficient Phrase-Based Neural Machine Translation

Chan Young Park **Yulia Tsvetkov**
Language Technologies Institute
Carnegie Mellon University
{chanyoun,ytsvetko}@cs.cmu.edu

Abstract

Neural machine translation (NMT) often fails in one-to-many translation, e.g., in the translation of multi-word expressions, compounds, and collocations. To improve the translation of phrases, phrase-based NMT systems have been proposed; these typically combine word-based NMT with external phrase dictionaries or with phrase tables from phrase-based statistical MT systems. These solutions introduce a significant overhead of additional resources and computational costs. In this paper, we introduce a phrase-based NMT model built upon continuous-output NMT, in which the decoder generates embeddings of words or phrases. The model uses a fertility module, which guides the decoder to generate embeddings of sequences of varying lengths. We show that our model learns to translate phrases better, performing on par with state of the art phrase-based NMT. Since our model does not resort to softmax computation over a huge vocabulary of phrases, its training time is about 112x faster than the baseline.

1 Introduction

Despite the successes of neural machine translation (Wu et al., 2016; Vaswani et al., 2017; Ahmed et al., 2018), state of the art NMT systems are still challenged by translation of typologically divergent language pairs, especially when languages are morphologically rich (Burlot and Yvon, 2017). One of the reasons lies in increased sparsity of word types, which leads to the demand for (often unavailable) significantly larger training corpora (Koehn and Knowles, 2017). Another reason is an implicit assumption of sequence to sequence (seq2seq) models that input sequences are translated into a target language word-by-word or subword-by-subword (Sennrich et al., 2016).

This is not the case for typologically divergent language pairs, for example when translating into English from agglutinative languages with high rates of morphemes per word (e.g., Turkish and Quechua) or languages with productive compounding processes like German or Finnish (Matthews et al., 2016). Another ubiquitous source of one-to-many correspondences is a translation of idiomatic phrases and multi-word expressions (Rikters and Bojar, 2017).

While outperformed by NMT overall, translation models in traditional statistical phrase-based approaches (Koehn, 2009, SMT) provide an inventory of phrase translations, which can be used to address the above challenges. To combine the benefits of NMT and phrase-based SMT, phrase-based NMT systems have been proposed (Huang et al., 2017; Lample et al., 2018) which combine word-based NMT with external phrase memories (Tang et al., 2016; Dahlmann et al., 2017). However, prior approaches to phrase-based NMT introduced a significant overhead of additional resources and computation.

We introduce a phrase-based continuous-output NMT (PCoNMT) model built upon continuous-output NMT (Kumar and Tsvetkov, 2019), in which the decoder generates embeddings of words or phrases (§2). The model extracts phrases in the target language from one-to-many word alignments and pre-computes word and phrase embeddings which constitute the output space of our model (§2.2). A fertility module guides the decoder, providing the probability of generating a word or a phrase at each time step (§2.3). Experimental results show that the proposed model outperforms the conventional attention-based NMT systems (Bahdanau et al., 2014) by up to 4.8 BLEU, and the baseline continuous-output models by up to 1.6 BLEU, and beat the state-of-the-art phrase-based NMT system in translation from German and Turkish into English.

Since our model does not resort to softmax

Proceedings of the 3rd Workshop on Neural Generation and Translation (WNGT 2019), pages 241–248
Hong Kong, China, November 4, 2019.
www.aclweb.org/anthology/D19-56%2d

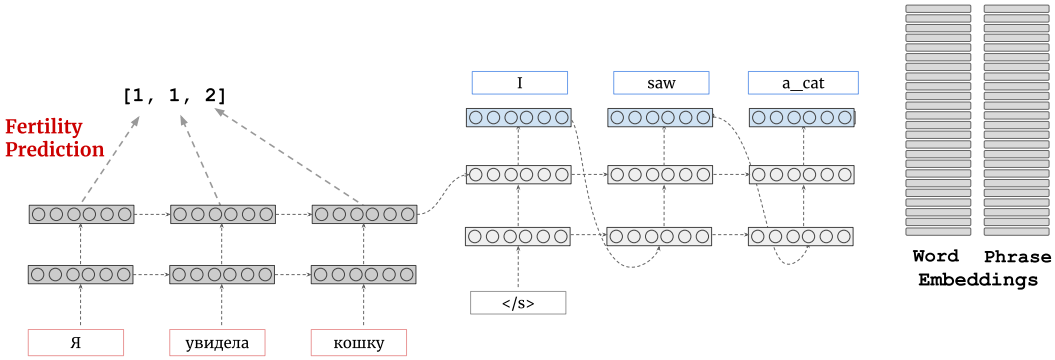

Figure 1: Phrase-based neural machine translation architectures generate word- and phrase embeddings at each step of decoding. The PCoNMT models are guided by on the fertility prediction and the attention.

computation over a huge vocabulary, it also maintains the computational efficiency of continuous-output NMT, even with additional ngram embedding tables, and is faster than the state-of-the-art baseline by 112x (§3), making our models energy-efficient (Strubell et al., 2019).

The key contributions of our work are twofold: (1) we develop a phrase-based NMT model that outperforms existing baselines and better translates phrases, while (2) maintaining the computational efficiency of NMT end-to-end approaches.[1]

2 Phrase-based Continuous-output NMT

2.1 Embedding output layer

Kumar and Tsvetkov (2019) introduced continuous-output machine translation (CoNMT) which replaces the softmax layer in the conventional seq2seq models with a continuous embeddings layer. The model predicts the embedding of the target word instead of its probability. It is trained to maximize the von Mises-Fisher (vMF) probability density of the pretrained target-language embeddings given the embeddings predicted by the model at every step; at inference time, predicted embedding is compared to the embeddings in the pre-trained embedding table, and the closest embedding is selected as an output word. While maintaining the translation quality of traditional seq2seq approaches, CoNMT approach alleviates the computational bottleneck of the softmax layer: it is substantially more efficient to train and the models are more compact, without limiting the output vocabulary size.

[1]Our code and data are available at https://github.com/chan0park/PCoNMT

Extending the CoNMT approach, we propose phrase-based continuous-output NMT (PCoNMT). As depicted in Figure 1, we augment the original model with (1) additional embedding tables for phrases, and (2) a fertility module that guides the choice of embedding table to look-up in (described in §2.3). Having additional large embedding tables, which significantly increase the vocabulary size, could be a considerable overhead to a word-based model with the softmax layer. However, since we generate embeddings in the final layer and do not resort to the softmax computation, our models maintain the computational efficiency of continuous-output models (§3) during the training time. At inference time, the only overhead incurred by our model is getting another set of vMF scores for the phrase embedding table for each output step. This is almost negligible compared to the computation of the entire network.

In addition to efficiency benefits, since the PCoNMT approach enables us to pre-compute embeddings of less frequent phrases and phrases that do not have a literal translation, e.g., multi-word expressions, it facilitates better translations specifically where the translation is notoriously challenging for NMT.

2.2 Output embedding tables

To construct embedding tables for target-language phrases, we first extract the list of output phrases from parallel corpora. Following Tang et al. (2016), in this work, we focus on one-to-many word alignments in the training corpus. Consider as an example translation of German com-

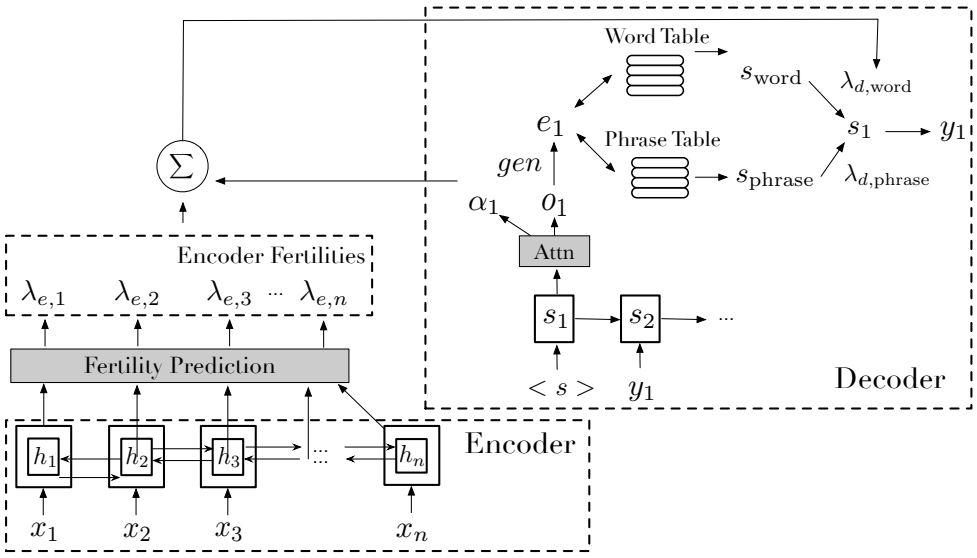

Figure 2: The detailed architecture of our model which consist of three components (encoder, fertility module, and decoder), described in §2. Given an input sentence $\{x_1, x_2 \dots x_n\}$, our model generates the output sentence $\{y_1, y_2 \dots y_m\}$, where y_i corresponds to words or phrases, e.g. *quality_of_life*. At each step, the decoder generates an embedding e_i, then the fertility module guides it to generate a word or a phrase, via the word- or phrase-embedding table, respectively.

pounds to English, e.g., *Lebensqualität* in German is translated as *quality of life*. We extract all such one-to-many word alignments from the parallel corpora using Fastalign (Dyer et al., 2013). There are several standard approaches to extract meaningful phrases from a monolingual corpus, such as using scores like pointwise mutual information (PMI) (Mikolov et al., 2013). However, for our model, we utilize word-alignment results to construct a phrase list since we are particularly interested in multi-word translation cases. Note that with this approach, phrases in target-side embedding tables can be different depending on which language pair and which corpus are being used.

After extracting all noisy one-to-many alignments from the parallel corpus, we filter our phrase list in order to keep only the useful phrases and to remove potential erroneous phrases coming from alignment errors. We filter according to the following heuristics: (1) a phrase should appear at least twice in the parallel corpus; (2) it should not contain any punctuation; (3) PMI of the phrase should be positive; (4) a bigram phrase should not repeat the same word; and (5) the phrase should not contain only stopwords.

We train embeddings for the resulting list of words and phrases as follows. First, we preprocess the target language's large monolingual corpus to concatenate words to match the longest phrase in the extracted phrase list. For example, the sentence 'I went to a graduate school' will be converted into 'I went_to a graduate_school' if we have *went_to* and *graduate_school* in our phrase list. This concatenated corpus is then used to train fastText (Peters et al., 2018) embeddings for both phrases and words simultaneously. We use fastText because it encodes subword-level information which may provide a signal about each word in a phrase. From this training, we obtain both the word- and phrase-tables, which are of the same dimension.

2.3 Fertility module

We introduce a fertility module, similar to the fertility concept in SMT (Brown et al., 1993). The fertility indicates how many target words each source word should produce in the output. The SMT models keep the fertility probabilities over fertility count, typically from zero to four, and use it to produce probability over words. We integrate this fertility concept into our PCoNMT model.

Our fertility module predicts the fertility probability $\phi_e = [\phi_{e0}, ..., \phi_{eN}]$, where ϕ_{ei} indicates the scalar probability of the source word at position e being translated into i words. This is predicted based on the word embedding and encoder's output of the word: $\phi_e = \text{FFNN}(x_e; h_e)$. FFNN is the feed-forward neural network, and $(x_e; h_e)$ denotes the concatenation of x_e and h_e, which are embedding and encoder's hidden state of eth

source word, respectively. The dimension of fertility vector ϕ, N, can be arbitrarily large, but in this paper we explore two different variants; the first one is Fertility_4 where each dimension corresponds to zero to three words to produce respectively ($N \in \{0, 1, 2, 3\}$), and the second one is Fertility_2 which simplifies the fertility prediction into binary classification by setting N=1 as a cut-off point, i.e., whether the model should generate a word ($N \leq 1$) or a phrase ($N > 1$). Therefore, ϕ_e becomes a four-dimensional vector of $[\phi_{e0}, \phi_{e1}, \phi_{e2}, \phi_{e3}]$ for Fertility_4, and two-dimensional vector of $[\sum_{n=0}^{1} \phi_{en}, \sum_{n=2}^{\infty} \phi_{en}]$ for Fertility_2.

At decoding time, we combine this fertility probability of each source word and the attention to guide the decoder to generate a phrase or a word. To get the probability of producing a word $\lambda_{d,\text{word}}$ for timestep d, we use attention given to each source word as a weight to its fertility probability and sum over the entire source sentence:

$$\lambda_{d,\text{word}} = \begin{cases} \sum_e \mathbf{a}_{d,e} \left(\phi_{e0} + \phi_{e1}\right) & (\dim = 4) \\ \sum_e \mathbf{a}_{d,e} \left[\phi_e\right]_0 & (\dim = 2) \end{cases}$$

$$\lambda_{d,\text{phrase}} = 1 - \lambda_{d,\text{word}},$$

where $\mathbf{a}_{d,e}$ is a scalar value of attention assigned for source word e at timestep d and $[\phi_e]_0$ is the 0th element of ϕ_e, which basically is the same as $(\phi_{e0} + \phi_{e1})$ in Fertility_4. We use this $\lambda_{d,\text{word}}$ and $\lambda_{d,\text{phrase}}$ to weight the scores in word table and in phrase table, respectively:

$$s_{\text{word}} = \lambda_{d,\text{word}} \cdot \text{Score}(e_d, T_{\text{word}})$$
$$s_{\text{phrase}} = \lambda_{d,\text{phrase}} \cdot \text{Score}(e_d, T_{\text{phrase}})$$
$$y_d = \arg\max(s_{\text{word}}; s_{\text{phrase}}),$$

where s_{word} is a vector of scores for word in the word embedding table T_{word}, and Score is a score function to measure how similar the predicted embedding e_d and the embeddings in T. For the Score function, we use vMF as proposed in Kumar and Tsvetkov (2019). Finally, we get an output, y_d for the timestep d by doing $\arg\max$ over weighted scores from both word and phrase tables.

2.4 Model Training

The training of PCoNMT model is achieved by two separate steps. First, we only train the seq2seq modules as CoNMT does. We use vMF loss to optimize the embedding prediction. Once we find the optimal parameters for the CoNMT components, we freeze those parameters, and separately train parameters of the fertility module.[2] During the preprocessing, we extract the actual fertility value for each source word using the word-alignment model and the filtered phrase list, then set it as a gold label for the fertility prediction training.

	IWSLT	$\text{IWSLT}_{\text{MWT}}$
Attn	23.83	-
NPMT	27.27	-
CoNMT	27.07	24.98
PCoNMT	**28.69**	**28.89**
$+\text{Fertility}_4$	28.04	24.93
$+\text{Fertility}_2$	28.29	25.12

Table 1: Evaluation results (BLEU) on IWSLT 2014 De–En task.

	WMT	WMT_{MWT}
NPMT	3.58	-
CoNMT	7.44	7.67
PCoNMT	**8.87**	7.70
$+\text{Fertility}_4$	8.12	8.53
$+\text{Fertility}_2$	8.39	**8.61**

Table 2: Evaluation results (BLEU) on WMT 2017 Tr–En task.

3 Experiments

In this section, we evaluate our model in terms of translation quality and training efficiency. We used IWSLT 2014 dataset for De–En machine translation task, following the same preprocessing and splits as in Ranzato et al. (2016). For the Tr–En task, we used WMT 17 train and test dataset (Bojar et al., 2018). The training corpora size for IWSLT 2014 and WMT 17 is about 153K and 200K sentences, respectively. All results are reported with case-sensitive BLEU-4 (Papineni et al., 2002). In addition to the two official datasets, we subset the given test sets to sentences that actually contain multi-word translation (MWT) cases by running the word-alignment model. The size of extracted MWT subsets for IWSLT 2014 and WMT 17 are 335 (5%) and 116

[2] Although we have omitted the results due to space, we also have tried jointly training the fertility prediction and translation in a multi-task learning setting. However, the joint-learning has consistently hurt the translation quality.

	speed ↓ (samples/sec)	convergence ↑ (epochs)	total time ↑ (hours)
NPMT	15.4	40	110
CoNMT	256.0	6	1.00
PCoNMT	261.0	6	0.98

Table 3: Training efficiency results on IWSLT 2014 De–En dataset.

(4%), respectively. Also note that following Kumar and Tsvetkov (2019), in this paper, we only used greedy decoding.

We compared our proposed model with three baselines: (1) Attn: Standard attention-based NMT model as in Wiseman and Rush (2016); (2) CoNMT: RNN-based Continuous-output NMT systems (Kumar and Tsvetkov, 2019); (3) NPMT: The state of the art phrase-based NMT model proposed by Huang et al. (2017). For NPMT, we ran its released code with the same preprocessed data we are using without changing any hyperparameters they set.[3] For both De–En and Tr–En CoNMT models, we used the best hyperparameter settings reported by Kumar and Tsvetkov (2019) for De–En. For our model, PCoNMT, we only changed the batch size from the original setting in CoNMT and chose other additional parameters based on the performance on the validation set.

Although we use recurrent architectures in this paper to make our findings comparable to prior work that uses the same setting, we believe using multi-layer self-attention mechanism (Vaswani et al., 2017) as a base of our model has further potential to improve the performance. Even with Transformers, the conventional token-by-token generation scheme will be still prone to mistakes in multi-word generations. Therefore, explicitly handling the phrase generation as we propose is likely to be helpful, which we leave it as future work.

Translation quality De–En and Tr–En translation results are summarized in Tables 1 and 2. PCoNMT significantly outperforms both the conventional attention-based model (by >4 BLEU) and its base CoNMT model (by 1.6 BLEU), and also performs better than NPMT (by 1.4 BLEU). The fertility module is shown to be relatively more helpful in Tr–En task, while showing less impact in De–En task. We also observed that $\mathrm{Fertility}_2$ consistently generates better translations than $\mathrm{Fertility}_4$. On the more difficult MWT subset containing multi-word phrases, PCoNMT obtains large absolute gains in BLEU, confirming their effectiveness in phrase translations. Examples of translations are shown in Table 7.

[3]The number we got from the experiment is different from the one reported in the original paper, which possibly is rooting from slightly different preprocessing steps.

Class	De–En				Tr–En			
	Tot.	P	R	F-1	Tot.	P	R	F-1
$N \leq 1$	97%	0.97	0.96	0.97	97%	0.97	0.95	0.96
$N > 1$	3%	0.33	0.28	0.31	3%	0.17	0.1	0.13

Table 4: The Precision, Recall, and F1 evaluation results on the fertility prediction of $\mathrm{Fertility}_2$. "Tot." is the percentage for the number of occurrences of each label in the gold label.

Class	De–En				Tr–En			
	Total	P	R	F-1	Total	P	R	F-1
$N = 0$	10%	0.59	0.09	0.15	14%	0.56	0.30	0.39
$N = 1$	86%	0.88	0.95	0.91	83%	0.86	0.91	0.89
$N = 2$	4%	0.27	0.35	0.31	3%	0.12	0.19	0.14
$N = 3$	0%	0.16	0.14	0.15	0%	0	0	0

Table 5: The Precision, Recall, and F1 evaluation results on the fertility prediction of $\mathrm{Fertility}_4$. "Tot." is the percentage for the number of occurrences of each label in the gold label.

Computational efficiency We report the training efficiency of models in three metrics: speed, number of training epochs till convergence, and total training time. All results were measured on the same machine with the same batch size. The machine was a single-node local machine with NVIDIA GTX 1080 Ti. During the training, no other process was executed except for the training for the fair comparison.

Table 3 shows that CoNMT and PCoNMT can process 28 times faster than NPMT, and converge six times faster, i.e., reducing the entire training time by 112x. Somewhat surprisingly, PCoNMT further accelerates the CoNMT as it can reduce the timestep needed for a sample by generating phrases. This result proves that additional phrase embeddings of PCoNMT has little impact on computational efficiency while training.

Fertility Prediction Evaluation The fertility prediction can have a significant impact on the translation as it guides the decoder to decide when to generate phrases and when to generate words. We evaluate the prediction results on the test set

with the gold label obtained from the word alignment model in Table 5 and Table 4.

In both datasets, we observe that the data is highly skewed toward word-level classes as most translations are word-to-word generation. This results in Fertility$_4$ not to predict $N = 3$ classes at all in the Tr–En dataset. The comparison between Table 5 and Table 4 shows that the Fertility$_2$ has slighly higher F-1 score than Fertility$_4$ in both datasets. It implies that aggregating the classes into two made the prediction task easier for the model, which thus led to the improved translation quality shown in the previous results.

Analysis on Generated Phrases Table 6 presents further analysis of the generated phrases. We first see in which category of phrases our model performs well compared to the baseline, CoNMT, to know from where the improvement of our model is coming. As for the phrase categories, we consider three categories, compound nouns (CNs, e.g., *thought_experiment*), verb phrases (VPs, e.g., *grow_apart*), and collocations (COs, e.g., *at_risk*). We randomly sampled a hundred generated phrases from the De–En test set, and manually annotate the category of phrases and whether it is the correct translation. We also look at the output of CoNMT baseline for the same test samples, and also annotate if the sampled phrases are well translated in the CoNMT output.

The results in Table 6 show that the most frequently generated phrases are collocations (56%) followed by verb phrases (28%) and compound nouns (16%). Among the entire sampled phrases, 64 percent of phrases were correct in PCoNMT output while CoNMT had 50 percent of them correct. Specifically, our model significantly outperformed the baseline in compound word generation cases while performs worse in verb phrases generation. By looking into the instances of wrong verb phrase generation, we found that a significant amount of those errors are related to the tense of the verb.

Category	Total	PCoNMT	CoNMT
CNs	16%	0.63	0.25
VPs	28%	0.5	0.57
COs	56%	0.71	0.54
Sum	100%	0.64	0.50

Table 6: Percentages of categories of randomly sampled 100 phrases generated by PCoNMT on IWSLT 2014 De–En test set and the accuracy of PCoNMT and CoNMT phrase translations, respectively.

4 Related Work

Multi-word Expressions for NMT There have been several studies that incorporate multi-word phrases into supervised NMT (Tang et al., 2016; Wang et al., 2017; Dahlmann et al., 2017). Most approaches rely on pre-defined phrase dictionaries obtained from methods such as phrase-based Stastitical MT (Koehn et al., 2003) or word-alignment. Tang et al. (2016) use a method that combines phrase probability and word probability obtained from a softmax layer enabling the decoder to decide to switch between phrase generation and word generation based on context. Dahlmann et al. (2017) use a separate SMT model to generate phrases along with an NMT model. Wang et al. (2017) proposed a similar approach to have an SMT model run in parallel, where an additional module decide whether to use a phrase generator from the SMT model or the neural decoder.

Recent works have also explored using an additional RNN to compute phrase generation probabilities. Huang et al. (2017) proposed Neural Phrase MT (NPMT) that is built upon Sleep-WAke Network (SWAN), a segmentation-based sequence modeling technique, which automatically discovers phrases given the data and appends the special symbol $ to the source and target data. The model gets these segmented word/phrase sequences as input and keeps two levels of RNNs to encode and decode phrases. NPMT established state of the art results for phrase-based NMT, but at a price of significant computational overhead.

The main differences between previous studies and our work are: (1) we do not rely on SMT model and adapt in an end-to-end manner only requiring some preprocessing using word-alignment models; and (2) we use phrase embedding tables to represent phrases instead of keeping external phrase memory and its generation probability. By using the phrase embeddings along with the continuous-output layer, we significantly reduce the computational complexity and propose an approach to overcome the phrase generation bottleneck.

Fertility in MT Fertility (Brown et al., 1993) has been a core component in phrase-based SMT

German src	und Sie sollten auch an Dinge wie **Lebensqualität** denken
English ref	and you also want to think about things like **quality of life**
Baseline CoNMT	and you should think of things like **life**
PCoNMT	and you should think of things like **quality_of_life** .
German src	wer ein Gehirn hat , ist **gefährdet** .
English ref	everyone with a brain is **at risk** .
Baseline CoNMT	who has a brain is **risk** .
PCoNMT	who has a brain is **at_risk** .
German src	ich stecke **voller** Widersprüche .
English ref	I am **full of** contradictions .
Baseline CoNMT	I 'm put .
PCoNMT	I 'm **full_of** contradictions

Table 7: Translation output examples from CoNMT and PCoNMT systems.

models (Koehn et al., 2003). Fertility gives the likelihood of each source word of being translated into n words. Fertility helps in deciding which phrases should be stored in the phrase tables. Tu et al. (2016) revisited fertility to model coverage in NMT to address the issue of under-translation. They used a fertility vector to express how many words should be generated per source word and a coverage vector to keep track of words translated so far. We use a very similar concept in this work but the fertility module is introduced with a purpose to guide the decoder to switch over generating phrases and words.

5 Conclusion

We proposed PCoNMT, a phrase-based NMT system built upon continuous-output NMT models. We also introduced a fertility module that guides the decoder by providing the probabilities of generating a phrase and a word by leveraging the attention mechanism. Our experimental results showed that our model outperforms the state of the art phrase NMT systems, and also speeds up the computation by 112x.

Acknowledgments

We gratefully acknowledge Sachin Kumar and our anonymous reviewers for the helpful feedback. This material is based upon work supported by NSF grant IIS1812327 and an Amazon MLRA award.

References

Karim Ahmed, Nitish Shirish Keskar, and Richard Socher. 2018. Weighted transformer network for machine translation. In *Proc. ICLR*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *Proc. of ICLR*.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Philipp Koehn, and Christof Monz. 2018. Findings of the 2018 conference on machine translation (wmt18). In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 272–303.

Peter F Brown, Vincent J Della Pietra, Stephen A Della Pietra, and Robert L Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311.

Franck Burlot and François Yvon. 2017. Evaluating the morphological competence of machine translation systems. In *Proc. of WMT*, pages 43–55.

Leonard Dahlmann, Evgeny Matusov, Pavel Petrushkov, and Shahram Khadivi. 2017. Neural machine translation leveraging phrase-based models in a hybrid search. In *Proc. of EMNLP*.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM Model 2. In *Proc. of NAACL*.

Po-Sen Huang, Chong Wang, Sitao Huang, Dengyong Zhou, and Li Deng. 2017. Towards neural phrase-based machine translation. In *Proc. of ICLR*.

Philipp Koehn. 2009. *Statistical machine translation*. Cambridge University Press.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proc. of WNGT*.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sachin Kumar and Yulia Tsvetkov. 2019. Von mises-fisher loss for training sequence to sequence models with continuous outputs. In *Proc. of ICLR*.

Guillaume Lample, Myle Ott, Alexis Conneau, and Ludovic Denoyer. 2018. Phrase-based & neural unsupervised machine translation. In *Proc. of EMNLP*, pages 5039–5049.

Austin Matthews, Eva Schlinger, Alon Lavie, and Chris Dyer. 2016. Synthesizing compound words for machine translation. In *Proc. of ACL*, pages 1085–1094.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. of ACL*, pages 311–318.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks.

Matss Rikters and Ondej Bojar. 2017. Paying attention to multi-word expressions in neural machine translation. In *Proc. of Machine Translation Summit*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proc. of ACL*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. In *Proc. of ACL*.

Yaohua Tang, Fandong Meng, Zhengdong Lu, Hang Li, and Philip LH Yu. 2016. Neural machine translation with external phrase memory. *arXiv preprint arXiv:1606.01792*.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 76–85.

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.

Xing Wang, Zhaopeng Tu, Deyi Xiong, and Min Zhang. 2017. Translating phrases in neural machine translation. In *Proc. of EMNLP*, pages 1421–1431.

Sam Wiseman and Alexander M Rush. 2016. Sequence-to-sequence learning as beam-search optimization. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1296–1306.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.