Simple Fusion: Return of the Language Model

Felix Stahlberg[†*] and **James Cross**[‡] and **Veselin Stoyanov**[‡]

[†]Department of Engineering, University of Cambridge, UK

[‡]Applied Machine Learning, Facebook, Menlo Park, CA, USA

`fs439@cam.ac.uk, jcross@fb.com, vesko.st@gmail.com`

Abstract

Neural Machine Translation (NMT) typically leverages monolingual data in training through backtranslation. We investigate an alternative simple method to use monolingual data for NMT training: We combine the scores of a pre-trained and fixed language model (LM) with the scores of a translation model (TM) while the TM is trained from scratch. To achieve that, we train the translation model to predict the residual probability of the training data added to the prediction of the LM. This enables the TM to focus its capacity on modeling the source sentence since it can rely on the LM for fluency. We show that our method outperforms previous approaches to integrate LMs into NMT while the architecture is simpler as it does not require gating networks to balance TM and LM. We observe gains of between +0.24 and +2.36 BLEU on all four test sets (English-Turkish, Turkish-English, Estonian-English, Xhosa-English) on top of ensembles without LM. We compare our method with alternative ways to utilize monolingual data such as backtranslation, shallow fusion, and cold fusion.

1 Introduction

Machine translation (MT) relies on parallel training data, which is difficult to acquire. In contrast, monolingual data is abundant for most languages and domains. Traditional statistical machine translation (SMT) effectively leverages monolingual data using language models (LMs) (Brants et al., 2007). The combination of LM and TM in SMT can be traced back to the noisy-channel model which applies the Bayes rule to decompose a translation system (Brown et al., 1993):

$$\begin{aligned}
\hat{\mathbf{y}} &= \underset{\mathbf{y}}{\operatorname{argmax}}\, P(\mathbf{y}|\mathbf{x}) \\
&= \underset{\mathbf{y}}{\operatorname{argmax}}\, P_{TM}(\mathbf{x}|\mathbf{y})P_{LM}(\mathbf{y})
\end{aligned} \tag{1}$$

where $\mathbf{x} = (x_1, \ldots, x_m)$ is the source sentence, $\mathbf{y} = (y_1, \ldots, y_n)$ is the target sentence, and $P_{TM}(\cdot)$ and $P_{LM}(\cdot)$ are translation model and language model probabilities.

In contrast, NMT (Sutskever et al., 2014; Bahdanau et al., 2014) uses a discriminative model and learns the distribution $P(\mathbf{y}|\mathbf{x})$ directly end-to-end. Therefore, the vanilla training regimen for NMT is not amenable to integrating an LM or monoglingual data in a straightforward manner.

An early attempt to use LMs for NMT, also known as *shallow fusion*, combines LM and NMT scores at inference time in a log-linear model (Gulcehre et al., 2015, 2017). In contrast, we integrate the LM scores during NMT training. Our training procedure first trains an LM on a large monolingual corpus. We then hold the LM fixed and train the NMT system to optimize the combined score of LM and NMT on the parallel training set. This allows the NMT model to focus on modeling the source sentence, while the LM handles the generation based on the target-side history. Sriram et al. (2017) explored a similar idea for speech recognition using a gating network for controlling the relative contribution of the LM. We show that our simpler architecture without an explicit control mechanism is effective for machine translation. We observe gains of up to more than 2 BLEU points from adding the LM to TM training. We also show that our method can be combined with backtranslation (Sennrich et al., 2016a), yielding further gains over systems without LM.

---

2 Related Work

2.1 Inference-time Combination

Shallow fusion (Gulcehre et al., 2015) integrates an LM by changing the decoding objective to:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\arg\max} \log P_{\text{TM}}(\mathbf{y}|\mathbf{x}) + \lambda \log P_{\text{LM}}(\mathbf{y}). \quad (2)$$

$P_{\text{LM}}(\cdot)$ is produced by an LSTM-based RNN-LM (Mikolov et al., 2010) which has been trained on monolingual target language data. $P_{\text{TM}}(\cdot)$ can be a typical encoder-decoder Seq2Seq model (Sutskever et al., 2014; Bahdanau et al., 2014; Luong et al., 2015a). λ is a hyper-parameter which is tuned on the development set.

2.2 Cold Fusion

Shallow fusion combines a fixed TM with a fixed LM at inference time. Sriram et al. (2017) proposed to keep the LM fixed, but train a sequence to sequence (Seq2Seq) NMT model from scratch which includes the LM as a fixed part of the network. They argue that this approach allows the Seq2Seq network to use its model capacity for the conditioning on the source sequence since the language modeling aspect is already covered by the LM. Their *cold fusion* architecture includes a gating network which learns to regulate the contributions of the LM at each time step. They demonstrated superior performance of cold fusion on a speech recognition task.

2.3 Other Approaches

Gulcehre et al. (2015, 2017) suggest to combine a pre-trained RNN-LM with a pre-trained NMT system using a controller network that dynamically adjusts the weights between RNN-LM and NMT at each time step (*deep fusion*). Both deep fusion and n-best reranking with count-based LMs have been used in WMT evaluation systems (Jean et al., 2015; Wang et al., 2017). An important limitation of these approaches is that LM and TM are trained independently.

A second line of research augments the parallel training data with additional synthetic data from a monolingual corpus in the target language. The source sentences can be generated with a separate translation system (Schwenk, 2008; Sennrich et al., 2016a) (backtranslation), or simply copied over from the target side (Currey et al., 2017). Since data augmentation methods rely on some balance between real and synthetic data (Sennrich et al., 2016a; Currey et al., 2017; Poncelas et al., 2018), they can often only use a small fraction of the available monolingual data. A third class of approaches change the NMT training loss function to incorporate monolingual data. For example, Cheng et al. (2016); Tu et al. (2017) proposed to add autoencoder terms to the training objective which capture how well a sentence can be reconstructed from its translated representation. However, training with respect to the new loss is often computationally intensive and requires approximations. Alternatively, multi-task learning has been used to incorporate source-side (Zhang and Zong, 2016) and target-side (Domhan and Hieber, 2017) monolingual data. Another way of utilizing monolingual data in both source and target language is to warm start Seq2Seq training from pre-trained encoder and decoder networks (Ramachandran et al., 2017; Skorokhodov et al., 2018). We note that pre-training can be used in combination with our approach.

An extreme form of leveraging monolingual training data is unsupervised NMT (Lample et al., 2017; Artetxe et al., 2017) which removes the need for parallel training data entirely. In this work, we assume to have access to some amount of parallel training data, but aim to improve the translation quality even further by using a language model.

3 Translation Model Training under Language Model Predictions

In spirit of the cold fusion technique of Sriram et al. (2017) we also keep the LM fixed when training the translation network. However, we greatly simplify the architecture by removing the need for a gating network. We follow the usual left-to-right factorization in NMT:

$$P(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{n} P(y_t|y_1^{t-1}, \mathbf{x}). \quad (3)$$

Let $S_{\text{TM}}(y_t|y_1^{t-1}, \mathbf{x})$ be the output of the TM projection layer without softmax, i.e., what we would normally call the logits. We investigate two different ways to parameterize $P(y_t|y_1^{t-1}, \mathbf{x})$ using $S_{\text{TM}}(y_t|y_1^{t-1}, \mathbf{x})$ and a fixed and pre-trained language model $P_{\text{LM}}(\cdot)$: POSTNORM and PRENORM.

POSTNORM This variant is directly inspired by shallow fusion (Eq. 2) as we turn $S_{\text{TM}}(y_t|y_1^{t-1}, \mathbf{x})$

into a probability distribution using a softmax layer, and sum its log-probabilities with the log-probabilities of the LM, i.e. multiply their probabilities:

$$P(y_t|y_1^{t-1}, \mathbf{x}) = \text{softmax}(S_{\text{TM}}(y_t|y_1^{t-1}, \mathbf{x})) \cdot P_{\text{LM}}(y_t|y_1^{t-1}). \quad (4)$$

PRENORM   Another option is to apply normalization after combining the raw $S_{\text{TM}}(y_t|y_1^{t-1}, \mathbf{x})$ scores with the LM log-probability:

$$P(y_t|y_1^{t-1}, \mathbf{x}) = \text{softmax}\Big(S_{\text{TM}}(y_t|y_1^{t-1}, \mathbf{x}) + \log P_{\text{LM}}(y_t|y_1^{t-1})\Big). \quad (5)$$

3.1   Theoretical Discussion of POSTNORM and PRENORM

Note that $P(y_t|y_1^{t-1}, \mathbf{x})$ might not represent a valid probability distribution under the POST-NORM criterion since, as component-wise product of two distributions, it is not guaranteed to sum to 1. A way to fix this issue would be to combine TM and LM probabilities in the probability space rather than in the log space. However, we have found that probability space combination does not work as well as POSTNORM in our experiments. We can describe $S_{\text{TM}}(y_t|y_1^{t-1}, \mathbf{x})$ under POSTNORM informally as the residual probability added to the prediction of the LM.

It is interesting to investigate what signal is actually propagated into $S_{\text{TM}}(y_t|y_1^{t-1}, \mathbf{x})$ when training with the PRENORM strategy. We can rewrite $P(y_t|y_1^{t-1}, \mathbf{x})$ as:

$$\begin{aligned} P(y_t|y_1^{t-1}, \mathbf{x}) &= \frac{P(y_t, y_1^{t-1}|\mathbf{x})}{P(y_1^{t-1}|\mathbf{x})} \\ &= \frac{P(y_t, \mathbf{x}|y_1^{t-1})}{P(\mathbf{x}|y_1^{t-1})} \\ &= \frac{P(\mathbf{x}|y_t, y_1^{t-1})}{P(\mathbf{x}|y_1^{t-1})} P(y_t|y_1^{t-1}). \end{aligned} \quad (6)$$

Alternatively, we can decompose $P(y_t|y_1^{t-1}, \mathbf{x})$ as

Language pair	# Sentences
Turkish-English (WMT)	207.7K
Estonian-English (WMT)	2,178.0K
Xhosa-English (INTERNAL)	739.2K

Table 1: Parallel training data.

Language	# Sentences	LM Perplexity dev	test
English (WMT)	26.9M	91.16	87.77
Turkish (WMT)	3.0M	59.19	70.46
English (INTERNAL)	20.0M	105.28	108.19

Table 2: Monolingual training data.

follows using Eq. 5:

$$\begin{aligned} P(y_t|y_1^{t-1}, \mathbf{x}) &= \text{softmax}\Big(S_{\text{TM}}(y_t|y_1^{t-1}, \mathbf{x}) \\ &\quad + \log P_{\text{LM}}(y_t|y_1^{t-1})\Big) \\ &\propto \exp\Big(S_{\text{TM}}(y_t|y_1^{t-1}, \mathbf{x}) \\ &\quad + \log P_{\text{LM}}(y_t|y_1^{t-1})\Big) \\ &= \exp(S_{\text{TM}}(y_t|y_1^{t-1}, \mathbf{x})) \\ &\quad \cdot P_{\text{LM}}(y_t|y_1^{t-1}). \end{aligned} \quad (7)$$

Combining Eq. 6 and Eq. 7 leads to:

$$\exp(S_{\text{TM}}(y_t|y_1^{t-1}, \mathbf{x})) \propto \frac{P(\mathbf{x}|y_1^t)}{P(\mathbf{x}|y_1^{t-1})} \quad (8)$$

This means that $S_{\text{TM}}(y_t|y_1^{t-1}, \mathbf{x})$ under PRENORM is trained to predict how much more likely the *source* sentence becomes when a particular target token y_t is revealed.

4   Experimental Setup

We evaluate our method on a variety of publicly available and proprietary data sets. For our Turkish-English (tr-en), English-Turkish (en-tr), and Estonian-English (et-en) experiments we use all available parallel data from the WMT18 evaluation campaign to train the translation models. Our language models are trained on *News Crawl 2017*. We use *news-test2017* as development ("dev") set and *news-test2018* as test set.

Additionally, we collected our own proprietary corpus of public posts on Facebook. We refer to it as 'INTERNAL' data set. This corpus consists of monolingual English in-domain sentences and parallel data in Xhosa-English. Training set sizes are summarized in Tables 1 and 2.

Our preprocessing consists of lower-casing, tokenization, and subword-segmentation using joint

Architecture Hyperparameters	
Source vocab size (BPE)	16,000
Target vocab size (BPE)	16,000
Embedding size (all)	256
Encoder LSTM units	512
Encoder layers	2
Decoder LSTM units	512
Decoder layers	2
Attention type	dot product
Training Settings	
Optimization	Vanilla SGD
Learning rate	0.5
Batch size	32
Label smoothing ϵ	0.1
Checkpoint averaging	Last 10

Table 3: Summary of NMT settings for all models.

byte pair encoding (Sennrich et al., 2016b) with 16K merge operations. On Turkish, we additionally remove diacritics from the text.

On WMT we use lower-cased Sacre-BLEU[1] (Post, 2018) to be comparable with the literature.[2] On our internal data we report tokenized BLEU scores.

Our Seq2Seq models are encoder-decoder architectures (Sutskever et al., 2014; Bahdanau et al., 2014) with dot-product attention (Luong et al., 2015b) trained with our PyTorch Translate library.[3] Both decoder and encoder consist of two 512-dimensional LSTM layers and 256-dimensional embeddings. The first encoder layer is bidirectional, the second one runs from right to left. Our training and architecture hyperparameters are summarized in Tab. 3. Our LSTM-based LMs have the same size and architecture as the decoder networks, but do not use attention and do not condition on the source sentence. We run beam search with beam size of 6 in all our experiments.

For each setup we train five models using SGD (batch size of 32 sentences) with learning rate decay and label smoothing, and either select the best one (single system) or ensemble the four best models based on dev set BLEU score.

5 Results

Tab. 4 compares our methods PRENORM and POSTNORM on the tested language pairs. Shallow fusion (Sec. 2.1) often leads to minor improvements over the baseline for both single systems and ensembles. We also reimplemented the

---

English-Turkish (WMT)

Method	Single		4-Ensemble	
	dev	test	dev	test
Baseline (no LM)	12.23	11.56	14.17	13.35
Shallow fusion	12.45	11.61	14.43	13.51
Cold fusion	12.39	11.54	14.20	13.23
This work: PRENORM	12.82	11.93	**14.78**	13.41
This work: POSTNORM	**13.30**	**12.27**	14.77	**13.61**

Turkish-English (WMT)

Method	Single		4-Ensemble	
	dev	test	dev	test
Baseline (no LM)	16.14	16.60	18.01	18.67
Shallow fusion	16.11	16.70	18.01	18.67
Cold fusion	16.25	16.21	17.99	18.40
This work: PRENORM	15.88	16.39	17.95	18.40
This work: POSTNORM	**16.59**	**17.03**	**18.38**	**19.17**

Estonian-English (WMT)

Method	Single		4-Ensemble	
	dev	test	dev	test
Baseline (no LM)	16.02	16.57	16.83	17.91
Shallow fusion	16.02	16.57	16.83	17.91
Cold fusion	15.40	15.99	16.48	17.79
This work: PRENORM	**16.80**	**17.44**	**17.78**	**19.01**
This work: POSTNORM	16.43	17.10	17.62	18.63

Xhosa-English (INTERNAL)

Method	Single		4-Ensemble	
	dev	test	dev	test
Baseline (no LM)	10.39	11.49	13.87	15.43
Shallow fusion	10.69	11.65	14.06	15.54
Cold fusion	10.72	11.29	13.66	15.13
This work: PRENORM	11.06	12.13	14.50	16.07
This work: POSTNORM	**12.34**	**13.27**	**15.45**	**17.79**

Table 4: Comparison of our PRENORM and POSTNORM combination strategies with shallow fusion (Gulcehre et al., 2015) and cold fusion (Sriram et al., 2017) under an RNN-LM.

cold fusion technique (Sec. 2.2) for comparison. For our machine translation experiments we report mixed results with cold fusion, with performance ranging between 0.33 BLEU gain on Xhosa-English and slight BLEU degradation in most of our Turkish-English experiments.

Both of our methods, PRENORM and POSTNORM yield significant improvements in BLEU across the board. We report more consistent gains with POSTNORM than with PRENORM. All our POSTNORM systems outperform both shallow fusion and cold fusion on all language pairs, yielding test set gains of up to +2.36 BLEU (Xhosa-English ensembles).

6 Discussion and Analysis

Backtranslation A very popular technique to use monolingual data for NMT is backtranslation (Sennrich et al., 2016a). Backtranslation
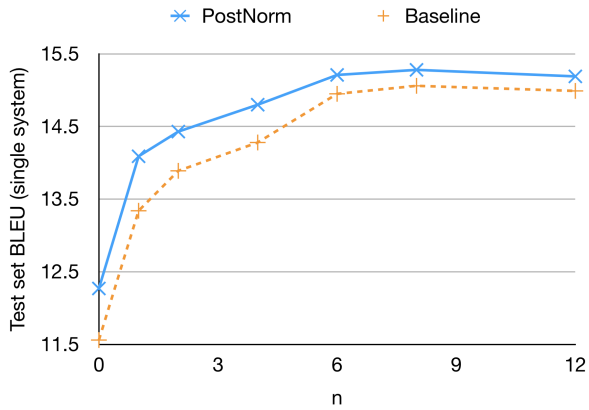
Figure 1: Performance using backtranslation on English-Turkish. Synthetic sentences are mixed at a ratio of 1:n where n is plotted on the x-axis.
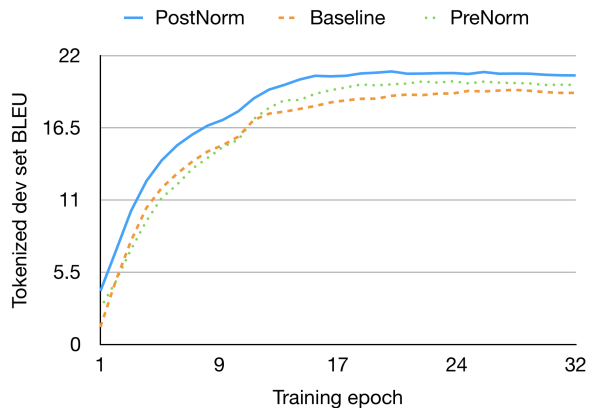

Figure 2: Convergence of NMT training with and without LM on English-Turkish.

uses a reverse NMT system to translate monolingual target language sentences into the source language, and adds the newly generated sentence pairs to the training data. The amount of monolingual data which can be used for backtranslation is usually limited by the size of the parallel corpus as the translation quality suffers when the mixing ratio between synthetic and real source sentences is too large (Poncelas et al., 2018). This is a severe limitation particularly for low-resource MT. Fig. 1 shows that both our baseline system without LM and our POSTNORM system benefit greatly from backtranslation up to a mixing ratio of 1:8, but degrade slightly if this ratio is exceeded. POSTNORM is significantly better than the baseline even when using it in combination with backtranslation.

Training convergence We have found that training converges faster under the POSTNORM loss. Fig. 2 plots the training curves of our sys-

English-Turkish (WMT, single system)

Method	Dev set		Test set	
	FFN	RNN	FFN	RNN
Baseline (no LM)	12.23		11.56	
Shallow fusion	12.25	12.45	11.53	11.61
Cold fusion	12.33	12.39	11.51	11.54
This work: PRENORM	12.76	12.82	11.82	11.93
This work: POSTNORM	12.65	13.30	11.79	12.27

Table 5: Comparison between using a recurrent LM (RNN) and an n-gram based feedforward LM (FFN) on English-Turkish.

English-Turkish (WMT), POSTNORM strategy

LM type		Single		4-Ensemble	
FFN	RNN	dev	test	dev	test
		12.23	11.56	14.17	13.35
✓		12.65	11.79	14.36	13.48
	✓	13.30	12.27	14.77	13.61
✓	✓	12.86	12.02	14.72	13.70

Table 6: Combining an RNN-LM and a feedforward LM with the translation model using the POSTNORM strategy.

tems. The baseline (orange curve) reaches its maximum of 19.39 BLEU after 28 training epochs. POSTNORM surpasses this BLEU score already after 12 epochs.

Language model type So far we have used recurrent neural network language models (Mikolov et al., 2010, RNN-LM) with LSTM cells in all our experiments. We can also parameterize an n-gram language model with a feedforward neural network (Bengio et al., 2003, FFN-LM). In order to compare both language model types we trained a 4-gram feedforward LM with two 512-dimensional hidden layers and 256-dimensional embeddings on Turkish monolingual data. Tab. 5 shows that the PRENORM strategy works particularly well for the n-gram LM. However, using an RNN-LM with the POSTNORM strategy still gives the best overall performance. Using both RNN and n-gram LM at the same time does not improve translation quality any further (Tab. 6).

Impact on the TM distribution With the POSTNORM strategy, the TM still produces a distribution over the target vocabulary as the scores are

Method	Perplexity	Average entropy
Baseline (no LM)	23.46	3.19
RNN-LM	59.19	4.66
TM under POSTNORM	113.69	1.82

Table 7: Perplexity and average entropies of the distributions generated by our systems on the English-Turkish dev set.

208

Method	BLEU	Precisions				BP
		1-gram	2-gram	3-gram	4-gram	
Baseline (no LM)	17.91	53.0	23.7	12.3	6.6	0.996
This work: PRENORM	19.01	54.0	24.9	13.4	7.4	1.000
Relative improvement	+6.14%	**+1.89%**	**+5.06%**	**+8.94%**	**+12.12%**	–

Table 8: BLEU n-gram precisions for Estonian-English.

Source	Eestis ja Hispaanias peeti kinni neli Kemerovo grupeeringu liiget
Reference	Four members of the Kemerovo group arrested in Estonia and Spain
Baseline (no LM)	In Estonia and Spain, four kemerovo groups were held
This work (PRENORM)	Four Kemerovo group members were held in Estonia and Spain
Source	Ta tleb, et elab aastaid hiljem endiselt hirmus.
Reference	He says that years later, he still lives in fear.
Baseline (no LM)	He says that, for years, he still lives in fear.
This work (PRENORM)	He says that many years later he still lives in fear.
Source	"Ma kardan," tleb ta.
Reference	"I'm afraid," he says.
Baseline (no LM)	"I fear," says he.
This work (PRENORM)	"I am afraid," he says.

Table 9: Translation samples from the Estonian-English test set.

normalized before the combination with the LM. This raises a natural question: How different are the distributions generated by a TM trained under POSTNORM loss from the distributions of the baseline system without LM? Tab. 7 gives some insight to that question. As expected, the RNN-LM has higher perplexity than the baseline as it is a weaker model of translation. The RNN-LM also has a higher average entropy which indicates that the LM distributions are smoother than those from the baseline translation model. The TM trained under POSTNORM loss has a much higher perplexity which suggests that it strongly relies on the LM predictions and performs poorly when it is not combined with it. However, the average entropy is much lower (1.82) than both other models, i.e. it produces much sharper distributions.

Language models improve fluency A traditional interpretation of the role of an LM in MT is that it is (also) responsible for the fluency of translations (Koehn, 2009). Thus, we would expect more fluent translations from our method than from a system without LM. Tab. 8 breaks down the BLEU score of the baseline and the PRENORM ensembles on Estonian-English into n-gram precisions. Most of the BLEU gains can be attributed to the increase in precision of higher order n-grams, indicating improvements in fluency. Tab. 9 shows some examples where our PRENORM system produces a more fluent translation than the baseline.

Training set size We artificially reduced the size of the English-Turkish training set even further
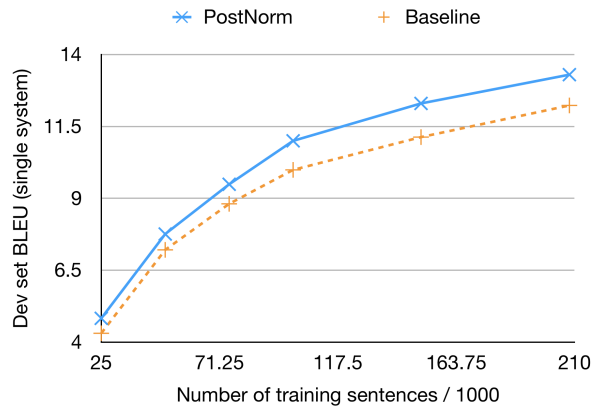

Figure 3: English-Turkish BLEU over training set size.

to investigate how well our method performs in low-resource settings (Fig. 3). Our POSTNORM strategy outperforms the baseline regardless of the number of training sentences, but the gains are smaller on very small training sets.

7 Conclusion

We have presented a simple yet very effective method to use language models in NMT which incorporates the LM already into NMT training. We reported significant and consistent gains from using our method in four language directions over two alternative ways to integrate LMs into NMT (*shallow fusion* and *cold fusion*) and showed that our approach works well even in combination with backtranslation and on top of ensembles. Our method leads to faster training convergence and more fluent translations than a baseline system without LM.

References

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2017. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. 2007. Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 858–867, Prague, Czech Republic. Association for Computational Linguistics.

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

Yong Cheng, Wei Xu, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Semi-supervised learning for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1965–1974. Association for Computational Linguistics.

Anna Currey, Antonio Valerio Miceli Barone, and Kenneth Heafield. 2017. Copied monolingual data improves low-resource neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 148–156. Association for Computational Linguistics.

Tobias Domhan and Felix Hieber. 2017. Using target-side monolingual data for neural machine translation through multi-task learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1500–1505. Association for Computational Linguistics.

Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2015. On using monolingual corpora in neural machine translation. *arXiv preprint arXiv:1503.03535*.

Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, and Yoshua Bengio. 2017. On integrating a language model into neural machine translation. *Computer Speech & Language*, 45:137 – 148.

Sébastien Jean, Orhan Firat, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. Montreal

neural machine translation systems for WMT'15. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 134–140. Association for Computational Linguistics.

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

Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015a. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015b. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421. Association for Computational Linguistics.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*.

Alberto Poncelas, Dimitar Shterionov, Andy Way, Gideon Maillette de Buy Wenniger, and Peyman Passban. 2018. Investigating backtranslation in neural machine translation. *arXiv preprint arXiv:1804.06189*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. *arXiv preprint arXiv:1804.08771*.

Prajit Ramachandran, Peter Liu, and Quoc Le. 2017. Unsupervised pretraining for sequence to sequence learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 383–391. Association for Computational Linguistics.

Holger Schwenk. 2008. Investigations on large-scale lightly-supervised training for statistical machine translation. In *In IWSLT*, pages 182–189.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. 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)*, pages 86–96. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725. Association for Computational Linguistics.

Ivan Skorokhodov, Anton Rykachevskiy, Dmitry Emelyanenko, Sergey Slotin, and Anton Ponkratov. 2018. Semi-supervised neural machine translation with language models. In *Proceedings of the AMTA 2018 Workshop on Technologies for MT of Low Resource Languages (LoResMT 2018)*, pages 37–44. Association for Machine Translation in the Americas.

Anuroop Sriram, Heewoo Jun, Sanjeev Satheesh, and Adam Coates. 2017. Cold fusion: Training seq2seq models together with language models. *arXiv preprint arXiv:1708.06426*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3104–3112, Cambridge, MA, USA. MIT Press.

Zhaopeng Tu, Yang Liu, Lifeng Shang, Xiaohua Liu, and Hang Li. 2017. Neural machine translation with reconstruction.

Yuguang Wang, Shanbo Cheng, Liyang Jiang, Jiajun Yang, Wei Chen, Muze Li, Lin Shi, Yanfeng Wang, and Hongtao Yang. 2017. Sogou neural machine translation systems for WMT17. In *Proceedings of the Second Conference on Machine Translation*, pages 410–415. Association for Computational Linguistics.

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545. Association for Computational Linguistics.