Multi-representation Ensembles and Delayed SGD Updates Improve Syntax-based NMT

Danielle Saunders[†] and **Felix Stahlberg**[†] and **Adrià de Gispert**[‡†] and **Bill Byrne**[‡†]

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

[‡]SDL Research, Cambridge, UK

Abstract

We explore strategies for incorporating target syntax into Neural Machine Translation. We specifically focus on syntax in ensembles containing multiple sentence representations. We formulate beam search over such ensembles using WFSTs, and describe a delayed SGD update training procedure that is especially effective for long representations like linearized syntax. Our approach gives state-of-the-art performance on a difficult Japanese-English task.

1 Introduction

Ensembles of multiple NMT models consistently and significantly improve over single models (Garmash and Monz, 2016). Previous work has observed that NMT models trained to generate target syntax can exhibit improved sentence structure (Aharoni and Goldberg, 2017; Eriguchi et al., 2017) relative to those trained on plain-text, while plain-text models produce shorter sequences and so may encode lexical information more easily (Nadejde et al., 2017). We hypothesize that an NMT ensemble would be strengthened if its component models were complementary in this way. However, ensembling often requires component models to make predictions relating to the same output sequence position at each time step. Models producing different sentence representations are necessarily synchronized to enable this. We propose an approach to decoding ensembles of models generating different representations, focusing on models generating syntax.

As part of our investigation we suggest strategies for practical NMT with very long target sequences. These long sequences may arise through the use of linearized constituency trees and can be much longer than their plain byte pair encoded (BPE) equivalent representations (Table 1). Long sequences make training more difficult (Bahdanau et al., 2015), which we address with an adjusted training procedure for the Transformer architecture (Vaswani et al., 2017), using delayed SGD updates which accumulate gradients over multiple batches. We also suggest a syntax representation which results in much shorter sequences.

1.1 Related Work

Nadejde et al. (2017) perform NMT with syntax annotation in the form of Combinatory Categorial Grammar (CCG) supertags. Aharoni and Goldberg (2017) translate from source BPE into target linearized parse trees,

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 319–325
Melbourne, Australia, July 15 - 20, 2018. ©2018 Association for Computational Linguistics

but omit POS tags to reduce sequence length. They demonstrate improved target language reordering when producing syntax. Eriguchi et al. (2017) combine recurrent neural network grammar (RNNG) models (Dyer et al., 2016) with attention-based models to produce well-formed dependency trees. Wu et al. (2017) similarly produce both words and arc-standard algorithm actions (Nivre, 2004).

Previous approaches to ensembling diverse models focus on model inputs. Hokamp (2017) shows improvements in the quality estimation task using ensembles of NMT models with multiple input representations which share an output representation. Garmash and Monz (2016) show translation improvements with multi-source-language NMT ensembles.

2 Ensembles of Syntax Models

We wish to ensemble using models which generate linearized constituency trees but these representations can be very long and difficult to model. We therefore propose a derivation-based representation which is much more compact than a linearized parse tree (examples in Table 1). Our linearized derivation representation ((4) in Table 1) consists of the derivation's right-hand side tokens with an end-of-rule marker, </R> , marking the last non-terminal in each rule. The original tree can be directly reproduced from the sequence, so that structure information is maintained. We map words to subwords as described in Section 3.

2.1 Delayed SGD Update Training for Long Sequences

We suggest a training strategy for the Transformer model (Vaswani et al., 2017) which gives improved performance for long sequences, like syntax representations, without requiring additional GPU memory. The Tensor2Tensor framework (Vaswani et al., 2018) defines batch size as the number of tokens per batch, so batches will contain fewer sequences if their average length increases. During NMT training, by default, the gradients used to update model parameters are calculated over individual batches. A possible consequence is that batches containing fewer sequences per update may have 'noisier' estimated gradients than batches with more sequences.

Previous research has used very large batches to improve training convergence while requiring fewer model updates (Smith et al., 2017; Neishi et al., 2017). However, with such large batches the model size may exceed available GPU memory. Training on multiple GPUs is one way to increase the amount of data used to estimate gradients, but it requires significant resources. Our strategy avoids this problem by using delayed SGD updates. We accumulate gradients over a fixed number of batches before using the accumulated gradients to update the model[1]. This lets us effectively use very large batch sizes without requiring multiple GPUs.

2.2 Ensembling Representations

Table 1 shows several different representations of the same hypothesis. To formulate an ensembling decoder over pairs of these representations, we assume we have a transducer T that maps from one representation to the other representation. The complexity of the transduction depends on the representations. Mapping from word to BPE representations is straightforward, and mapping from (linearized) syntax to plain-text simply deletes non-terminals. Let $\mathcal{P}$ be the paths in T leading from the start state to any final state. A path

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

Representation	Sample	Mean length
(1) Plain-text	No complications occurred	27.5
(2) Linearized tree	(ROOT (S (NP (DT No) (NNS complications)) (VP (VBD occurred))))	120.0
(3) Derivation	ROOT→S ; S→NP VP ; NP→DT NNS ; DT→No ; NNS→complications ; VP→VBD ; VBD→occurred	-
(4) Linearized derivation	S</R> NP VP</R> DT NNS</R> No complications VBD</R> occurred	73.8
(5) POS/plain-text	DT No NNS complications VBD occurred	53.3

Table 1: Examples for proposed representations. Lengths are for the first 1M WAT English training sentences with BPE subwords (Sennrich et al., 2016).

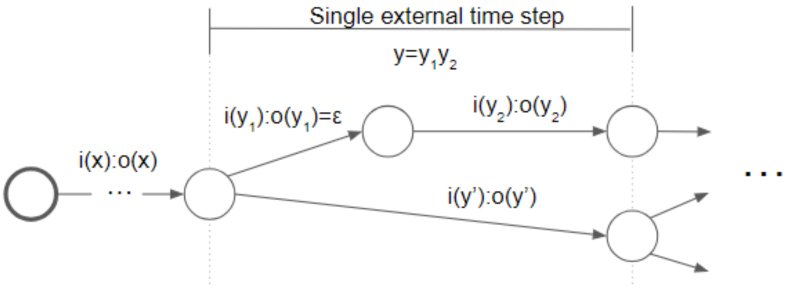

Figure 1: Transducer mapping internal to external representations. A partial hypothesis might be $\mathrm{o}(\mathrm{xy}_2)$ in the external representation and $\mathrm{i}(\mathrm{xy}_1\mathrm{y}_2)$ in the internal representation.

$p \in \mathcal{P}$ maps an *internal representation* $i(p)$ to an *external representation* $o(p)$.

The ensembling decoder produces external representations. Two NMT systems are trained, one for each representation, giving models P_i and P_o. An ideal equal-weight ensembling of P_i and P_o yields

$$p^* = \underset{p \in \mathcal{P}}{\text{argmax}}\ P_i(i(p))\, P_o(o(p)) \tag{1}$$

with $o(p^*)$ as the external representation of the translation.

In practice, beam decoding is performed in the external representation, i.e. over projections of paths in $\mathcal{P}$ [2]. Let $h = h_1 \ldots h_j$ be a partial hypothesis in the output representation. The set of partial paths yielding h are:

$$M(h) = \tag{2}$$
$$\{(x, y) | xyz \in \mathcal{P}, o(x) = h_{<j}, o(xy) = h\}$$

Here z is the path suffix. The ensembled score of h is then:

$$\begin{aligned} P(h_j|h_{<j}) = & P_o(h_j|h_{<j}) \times \\ & \max_{(x,y) \in M(h)} P_i(i(y)|i(x)) \end{aligned} \tag{3}$$

The max performed for each partial hypothesis h is itself approximated by a beam search. This leads to an outer beam search over external representations with inner beam searches for the best matching internal representations. As search proceeds, each model score is updated separately with its appropriate representation. Symbols in the internal representation are consumed as needed to stay synchronized with the external representation, as illustrated in Figure 1; epsilons are consumed with a probability of 1.

[2] See the `tokenization` wrappers in https://github.com/ucam-smt/sgnmt

Reference	low - energy electron microscope (LEEM) and photoelectron microscope (PEEM) were attracted attention as new surface electron microscope .
Plain BPE	low energy electron microscope (LEEM) and photoelectron microscope (PEEM) are noticed as new surface electron microscope .
Linearized derivation	low-energy electron microscopy (LEEM) and photoelectron microscopy (PEEM) are attracting attention as new surface electron microscopes .

Table 2: Sample generated translations from individual models

3 Experiments

We first explore the effect of our delayed SGD update training scheme on single models, contrasting updates every batch with accumulated updates every 8 batches. To compare target representations we train Transformer models with target representations (1), (2), (4) and (5) shown in Table 1, using delayed SGD updates every 8 batches. We decode with individual models and two-model ensembles, comparing results for single-representation and multi-representation ensembles. Each multi-representation ensemble consists of the plain BPE model and one other individual model.

All Transformer architectures are Tensor2Tensor's base Transformer model (Vaswani et al., 2018) with a batch size of 4096. In all cases we decode using SGNMT (Stahlberg et al., 2017) with beam size 4, using the average of the final 20 checkpoints. For comparison with earlier target syntax work, we also train two RNN attention-based seq2seq models (Bahdanau et al., 2015) with normal SGD to produce plain BPE sequences and linearized derivations. For these models we use embedding size 400, a single BiLSTM layer of size 750, and batch size 80.

We report all experiments for Japanese-English, using the first 1M training sentences of the Japanese-English ASPEC data (Nakazawa et al., 2016). All models use plain BPE Japanese source sentences. English constituency trees are obtained using CKYlark (Oda et al., 2015), with words replaced by BPE subwords. We train separate Japanese (lowercased) and English (cased) BPE vocabularies on the plain-text, with 30K merges each. Non-terminals are included as separate tokens. The linearized derivation uses additional tokens for non-terminals with </R> .

3.1 Results and Discussion

Our first results in Table 3 show that large batch training can significantly improve the performance of single Transformers, particularly when trained to produce longer sequences. Accumulating the gradient over 8 batches of size 4096 gives a 3 BLEU improvement for the linear derivation model. It has been suggested that decaying the learning rate can have a similar effect to large batch training (Smith et al., 2017), but reducing the initial learning rate by a factor of 8 alone did not give the same improvements.

Representation	Batches / update	Learning rate	Test BLEU
Plain BPE	1	0.025	27.5
	1	0.2	27.2
	8	0.2	28.9
Linearized derivation	1	0.025	25.6
	1	0.2	25.6
	8	0.2	28.7

Table 3: Single Transformers trained to convergence on 1M WAT Ja-En, batch size 4096

Our plain BPE baseline (Table 4) outperforms the current best system on WAT Ja-En, an 8-model ensemble (Morishita et al., 2017). Our syntax models achieve similar results despite producing much longer sequences. Table

Architecture	Representation	Dev BLEU	Test BLEU
Seq2seq (8-model ensemble)	Best WAT17 result (Morishita et al., 2017)	-	28.4
Seq2seq	Plain BPE	21.6	21.2
	Linearized derivation	21.9	21.2
Transformer	Plain BPE	28.0	28.9
	Linearized tree	28.2	28.4
	Linearized derivation	28.5	28.7
	POS/BPE	28.5	29.1

Table 4: Single models on Ja-En. Previous evaluation result included for comparison.

External representation	Internal representation	Test BLEU
Plain BPE	Plain BPE	29.2
Linearized derivation	Linearized derivation	28.8
Linearized tree	Plain BPE	28.9
Plain BPE	Linearized derivation	28.8
Linearized derivation	Plain BPE	$29.4^{\dagger}$
POS/BPE	Plain BPE	$29.3^{\dagger}$
Plain BPE	POS/BPE	$29.4^{\dagger}$

Table 5: Ja-En Transformer ensembles: † marks significant improvement on plain BPE baseline shown in Table 4 ($p < 0.05$ using bootstrap resampling (Koehn et al., 2007)).

3 indicates that large batch training is instrumental in this.

We find that RNN-based syntax models can equal plain BPE models as in Aharoni and Goldberg (2017); Eriguchi et al. (2017) use syntax for a 1 BLEU improvement on this dataset, but over a much lower baseline. Our plain BPE Transformer outperforms all syntax models except POS/BPE. More compact syntax representations perform better, with POS/BPE outperforming linearized derivations, which outperform linearized trees.

Ensembles of two identical models trained with different seeds only slightly improve over the single model (Table 5). However, an ensemble of models producing plain BPE and linearized derivations improves by 0.5 BLEU over the plain BPE baseline.

By ensembling syntax and plain-text we hope to benefit from their complementary strengths. To highlight these, we examine hypotheses generated by the plain BPE and linearized derivation models. We find that the syntax model is often more grammatical, even when the plain BPE model may share more vocabulary with the reference (Table 2).

In ensembling plain-text with a syntax external representation we observed that in a small proportion of cases non-terminals were over-generated, due to the mismatch in target sequence lengths. Our solution was to penalise scores of non-terminals under the syntax model by a constant factor.

It is also possible to constrain decoding of linearized trees and derivations to well-formed outputs. However, we found that this gives little improvement in BLEU over unconstrained decoding although it remains an interesting line of research.

4 Conclusions

We report strong performance with individual models that meets or improves over the recent best WAT Ja-En ensemble results. We train these models using a delayed SGD update training procedure that is especially effective for the long representations that arise from including target language syntactic information in the output. We further improve on the individual results via a decoding strategy allowing ensembling of models producing different output representations, such as subword units and syntax. We propose these techniques as practical approaches to including target syntax in NMT.

Acknowledgments

This work was supported by EPSRC grant EP/L027623/1.

References

Roee Aharoni and Yoav Goldberg. 2017. Towards string-to-tree neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 132–140.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A Smith. 2016. Recurrent neural network grammars. In *Proceedings of NAACL-HLT*, pages 199–209.

Akiko Eriguchi, Yoshimasa Tsuruoka, and Kyunghyun Cho. 2017. Learning to parse and translate improves neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 72–78.

Ekaterina Garmash and Christof Monz. 2016. Ensemble learning for multi-source neural machine translation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1409–1418.

Chris Hokamp. 2017. Ensembling factored neural machine translation models for automatic post-editing and quality estimation. In *Proceedings of the Second Conference on Machine Translation*, pages 647–654.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics.

Makoto Morishita, Jun Suzuki, and Masaaki Nagata. 2017. NTT neural machine translation systems at WAT 2017. In *Proceedings of the 4th Workshop on Asian Translation (WAT2017)*, pages 89–94.

Maria Nadejde, Siva Reddy, Rico Sennrich, Tomasz Dwojak, Marcin Junczys-Dowmunt, Philipp Koehn, and Alexandra Birch. 2017. Predicting target language CCG supertags improves neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 68–79.

Toshiaki Nakazawa, Manabu Yaguchi, Kiyotaka Uchimoto, Masao Utiyama, Eiichiro Sumita, Sadao Kurohashi, and Hitoshi Isahara. 2016. ASPEC: Asian scientific paper excerpt corpus. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC)*, pages 2204–2208. European Language Resources Association (ELRA).

Masato Neishi, Jin Sakuma, Satoshi Tohda, Shonosuke Ishiwatari, Naoki Yoshinaga, and Masashi Toyoda. 2017. A bag of useful tricks for practical neural machine translation: Embedding layer initialization and large batch size. In *Proceedings of the 4th Workshop on Asian Translation (WAT2017)*, pages 99–109.

Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57. Association for Computational Linguistics.

Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2015. Ckylark: A more robust PCFG-LA parser. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 41–45.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. 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)*, volume 1, pages 1715–1725.

Samuel L Smith, Pieter-Jan Kindermans, and Quoc V Le. 2017. Don't decay the learning rate, increase the batch size. *arXiv preprint arXiv:1711.00489*.

Felix Stahlberg, Eva Hasler, Danielle Saunders, and Bill Byrne. 2017. SGNMT–a flexible NMT decoding platform for quick prototyping of new models and search strategies. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 25–30.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan N. Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob Uszkoreit. 2018. Tensor2tensor for neural machine translation. *CoRR*, abs/1803.07416.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010.

Shuangzhi Wu, Dongdong Zhang, Nan Yang, Mu Li, and Ming Zhou. 2017. Sequence-to-dependency neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 698–707.