Stronger Baselines for Trustable Results in Neural Machine Translation

Michael Denkowski
Amazon.com, Inc.
mdenkows@amazon.com

Graham Neubig
Carnegie Mellon University
gneubig@cs.cmu.edu

Abstract

Interest in neural machine translation has grown rapidly as its effectiveness has been demonstrated across language and data scenarios. New research regularly introduces architectural and algorithmic improvements that lead to significant gains over "vanilla" NMT implementations. However, these new techniques are rarely evaluated in the context of previously published techniques, specifically those that are widely used in state-of-the-art production and shared-task systems. As a result, it is often difficult to determine whether improvements from research will carry over to systems deployed for real-world use. In this work, we recommend three specific methods that are relatively easy to implement and result in much stronger experimental systems. Beyond reporting significantly higher BLEU scores, we conduct an in-depth analysis of where improvements originate and what inherent weaknesses of basic NMT models are being addressed. We then compare the relative gains afforded by several other techniques proposed in the literature when starting with vanilla systems versus our stronger baselines, showing that experimental conclusions may change depending on the baseline chosen. This indicates that choosing a strong baseline is crucial for reporting reliable experimental results.

1 Introduction

In the relatively short time since its introduction, neural machine translation has risen to prominence in both academia and industry. Neural models have consistently shown top performance in shared evaluation tasks (Bojar et al., 2016; Cettolo et al., 2016) and are becoming the technology of choice for commercial MT service providers (Wu et al., 2016; Crego et al., 2016). New work from the research community regularly introduces model extensions and algorithms that show significant gains over baseline NMT. However, the continuous improvement of real-world translation systems has led to a substantial performance gap between the first published neural translation models and the current state of the art. When promising new techniques are only evaluated on very basic NMT systems, it can be difficult to determine how much (if any) improvement will carry over to stronger systems; is new work actually solving new problems or simply re-solving problems that have already been addressed elsewhere?

In this work, we recommend three specific techniques for strengthening NMT systems and empirically demonstrate how their use improves reliability of experimental results. We analyze in depth how these techniques change the behavior of NMT systems by addressing key weaknesses and discuss how these findings can be used to understand the effect of other types of system extensions. Our recommended techniques include: (1) a training approach using Adam with multiple restarts and learning rate annealing, (2) sub-word translation via byte pair encoding, and (3) decoding with ensembles of independently trained models.

We begin the paper content by introducing a typical NMT baseline system as our experimental starting point (§2.1). We then present and examine the effects of each recommended technique: Adam with multiple restarts and step size annealing (§3), byte pair encoding (§4), and independent model ensembling (§5). We show that combining these techniques can lead to a substantial improvement of over 5 BLEU (§6) and that results for several previously published techniques can dramati-

cally differ (up to being reversed) when evaluated on stronger systems (§6.2). We then conclude by summarizing our findings (§7).

2 Experimental Setup

2.1 Translation System

Our starting point for experimentation is a standard baseline neural machine translation system implemented using the Lamtram[1] and DyNet[2] toolkits (Neubig, 2015; Neubig et al., 2017). This system uses the attentional encoder-decoder architecture described by Bahdanau et al. (2015), building on work by Sutskever et al. (2014). The translation model uses a bi-directional encoder with a single LSTM layer of size 1024, multilayer perceptron attention with a layer size of 1024, and word representations of size 512. Translation models are trained until perplexity convergence on held-out data using the Adam algorithm with a maximum step size of 0.0002 (Kingma and Ba, 2015; Wu et al., 2016). Maximum training sentence length is set to 100 words. Model vocabulary is limited to the top 50K source words and 50K target words by frequency, with all others mapped to an `unk` token. A post-processing step replaces any `unk` tokens in system output by attempting a dictionary lookup[3] of the corresponding source word (highest attention score) and backing off to copying the source word directly (Luong et al., 2015). Experiments in each section evaluate this system against incremental extensions such as improved model vocabulary or training algorithm. Evaluation is conducted by average BLEU score over multiple independent training runs (Papineni et al., 2002; Clark et al., 2011).

2.2 Data Sets

We evaluate systems on a selection of public data sets covering a range of data sizes, language directions, and morphological complexities. These sets, described in Table 1, are drawn from shared translation tasks at the 2016 ACL Conference on Machine Translation (WMT16)[4] and the 2016 International Workshop on Spoken Language Translation (IWSLT16)[5].

[1] https://github.com/neubig/lamtram

[2] https://github.com/clab/dynet

[3] Translation dictionaries are learned from the system's training data using `fast_align` (Dyer et al., 2013).

[4] http://statmt.org/wmt16 (Bojar et al., 2016)

[5] https://workshop2016.iwslt.org, https://wit3.fbk.eu (Cettolo et al., 2012)

Scenario	Size (sent)	Sources
WMT German-English	4,562,102	Europarl, Common Crawl, news commentary
WMT English-Finnish	2,079,842	Europarl, Wikipedia titles
WMT Romanian-English	612,422	Europarl, SETimes
IWSLT English-French	220,400	TED talks
IWSLT Czech-English	114,390	TED talks

Scenario	Validation (Dev) Set	Test Set
DE-EN	News test 2015	News test 2016
EN-FI	News test 2015	News test 2016
RO-EN	News dev 2016	News test 2016
EN-FR	TED test 2013+2014	TED test 2015+2016
CS-EN	TED test 2012+2013	TED test 2015+2016

Table 1: Top: parallel training data available for all scenarios. Bottom: validation and test sets.

3 Training Algorithms

3.1 Background

The first neural translation models were optimized with stochastic gradient descent (Sutskever et al., 2014). After training for several epochs with a fixed learning rate, the rate is halved at prespecified intervals. This widely used rate "annealing" technique takes large steps to move parameters from their initial point to a promising part of the search space followed by increasingly smaller steps to explore that part of the space for a good local optimum. While effective, this approach can be time consuming and relies on hand-crafted learning schedules that may not generalize to different models and data sets.

To eliminate the need for schedules, subsequent NMT work trained models using the Adadelta algorithm, which automatically and continuously adapts learning rates for individual parameters during training (Zeiler, 2012). Model performance is reported to be equivalent to SGD with annealing, though training still takes a considerable amount of time (Bahdanau et al., 2015; Sennrich et al., 2016b). More recent work seeks to accelerate training with the Adam algorithm, which applies momentum on a per-parameter basis and automatically adapts step size subject to a user-specified maximum (Kingma and Ba, 2015). While this can lead to much faster convergence, the resulting models are shown to slightly underperform compared to annealing SGD (Wu et al., 2016). However, Adam's speed and reputation

of generally being "good enough" have made it a popular choice for researchers and NMT toolkit authors[6] (Arthur et al., 2016; Lee et al., 2016; Britz et al., 2017; Sennrich et al., 2017).

While differences in automatic metric scores between SGD and Adam-trained systems may be relatively small, they raise the more general question of training effectiveness. In the following section, we explore the relative quality of the optima found by these training algorithms.

3.2 Results and Analysis

To compare the behavior of SGD and Adam, we conduct training experiments with all data sets listed in §2.2. For each set, we train instances of the baseline model described in §2.1 with both optimizers using empirically effective initial settings.[7] In the only departure from the described baseline, we use a byte-pair encoded vocabulary with 32K merge operations in place of a limited full-word vocabulary, leading to faster training and higher metric scores (see experiments in §4).

For SGD, we begin with a learning rate of 0.5 and train the model to convergence as measured by dev set perplexity. We then halve the learning rate and restart training from the best previous point. This continues until training has been run a total of 5 times. The choice of training to convergence is made both to avoid the need for handcrafted learning schedules and to give the optimizers a better chance to find good neighborhoods to explore. For Adam, we use a learning rate (maximum step size) of 0.0002. While Adam's use of momentum can be considered a form of "self-annealing", we also evaluate the novel extension of explicitly annealing the maximum step size by applying the same halving and restarting process used for SGD. It is important to note that while restarting SGD has no effect beyond changing the learning rate, restarting Adam causes the optimizer to "forget" the per-parameter learning rates and start fresh.

For all training, we use a mini-batch size of 512 words.[8] For WMT systems, we evaluate dev set perplexity every 50K training sentences for the first training run and every 25K sentences for subsequent runs. For IWSLT systems, we evaluate every 25K sentences and then every 6,250 sentences. Training stops when no improvement in perplexity has been seen in 20 evaluations. For each experimental condition, we conduct 3 independent optimizer runs and report averaged metric scores. All training results are visualized in Figure 1.

Our first observation is that these experiments are largely in concert with prior work: Adam without annealing (first point) is significantly faster than SGD with annealing (last point) and often comparable or slightly worse in accuracy, with the exception of Czech-English where SGD underperforms. However, Adam with just 2 restarts and SGD-style rate annealing is actually both faster than the fully annealed SGD and obtains significantly better results in both perplexity and BLEU. We conjecture that the reason for this is twofold. First, while Adam has the ability to automatically adjust its learning rate, like SGD it still benefits from an explicit adjustment when it has begun to overfit. Second, Adam's adaptive learning rates tend to reduce to sub-optimally low values as training progresses, leading to getting stuck in a local optimum. Restarting training when reducing the learning rate helps jolt the optimizer out of this local optimum and continue to find parameters that are better globally.

4 Sub-Word Translation

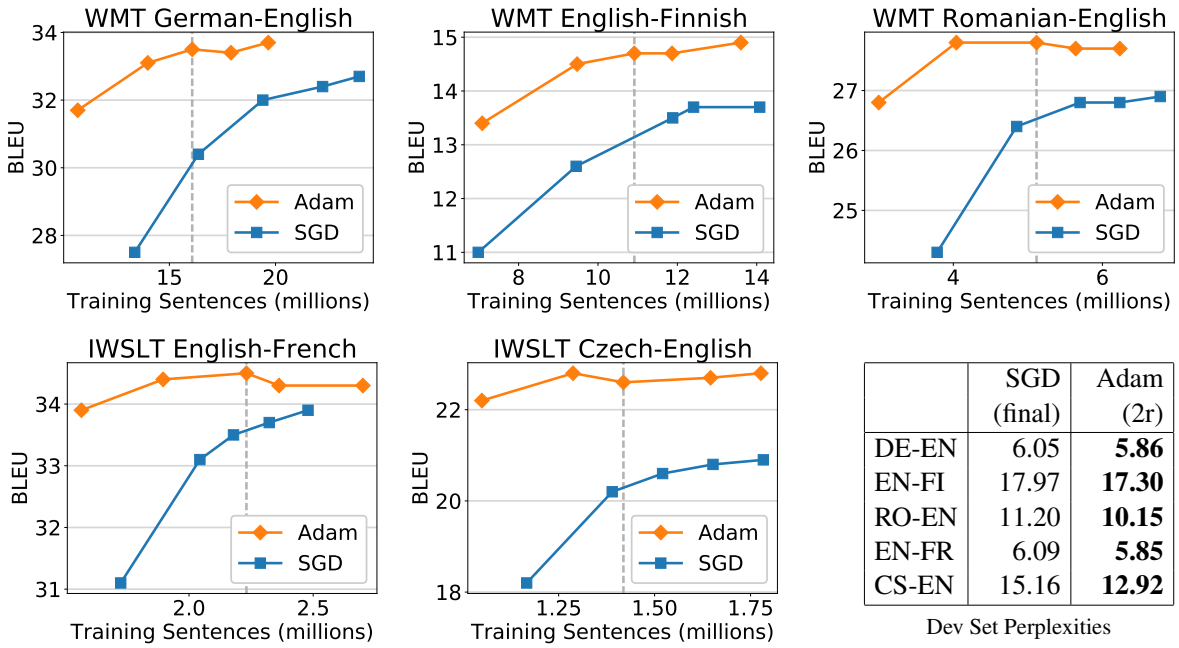
4.1 Background

Unlike phrase-based approaches, neural translation models must limit source and target vocabulary size to keep computational complexity manageable. Basic models typically include the most frequent words (30K-50K) plus a single `unk` token to which all other words are mapped. As described in §2.1, `unk` words generated by the NMT system are translated in post-processing by dictionary lookup or pass-through, often with significantly degraded quality (Luong et al., 2015). Real-world NMT systems frequently sidestep this problem with sub-word translation, where models operate on a fixed number of word pieces that can be chained together to form words in an arbitrar-

[6] Adam is the default optimizer for the Lamtram, Nematus (https://github.com/rsennrich/nematus), and Marian toolkits (https://github.com/amunmt/marian).

[7] Learning rates of 0.5 for SGD and 0.0002 for Adam or very similar are shown to work well in NMT implementations including GNMT (Wu et al., 2016), Nematus, Marian, and OpenNMT (http://opennmt.net).

[8] For each mini-batch, sentences are added until the word count is reached. Counting words versus sentences leads to more uniformly-sized mini-batches. We choose the size of 512 based on contrastive experiments that found it to be the best balance between speed and effectiveness of updates during training.



	SGD (final)	Adam (2r)
DE-EN	6.05	**5.86**
EN-FI	17.97	**17.30**
RO-EN	11.20	**10.15**
EN-FR	6.09	**5.85**
CS-EN	15.16	**12.92**

Dev Set Perplexities

Figure 1: Results of training the NMT models with Adam and SGD using rate annealing. Each point represents training to convergence with a fixed learning rate and translating the test set. The learning rate is then halved and training resumed from the previous best point. Vertical dotted lines indicate 2 Adam restarts. The table lists dev set perplexities for the final SGD model and the 2-restart Adam model. All reported values are averaged over 3 independent training runs.

ily large vocabulary. In this section, we examine the impact of sub-words on NMT, specifically when using the technique of *byte pair encoding* (Sennrich et al., 2016b). Given the full parallel corpus (concatenation of source and target sides), BPE first splits all words into individual characters and then begins merging the most frequently adjacent pairs. Merged pairs become single units that are candidates for further merging and the process continues to build larger word pieces for a fixed number of operations. The final result is an encoded corpus where the most frequent words are single pieces and less frequent words are split into multiple, higher frequency pieces. At test time, words are split using the operations learned during training, allowing the model to translate with a nearly open vocabulary.[9] The model vocabulary size grows with and is limited by the number of merge operations. While prior work has focused on using sub-words as a method for translating unseen words in morphologically rich languages (Sennrich et al., 2016b) or reducing model size (Wu et al., 2016), we examine how using BPE actually leads to broad improvement by addressing inherent weaknesses of word-level NMT.

[9]It is possible that certain intermediate word pieces will not appear in the encoded training data (and thus the model's vocabulary) if all occurrences are merged into larger units. If these pieces appear in test data and are not merged, they will be true OOVs for the model. For this reason, we map singleton word pieces in the training data to `unk` so the model has some ability to handle these cases (dictionary lookup or pass-through).

	WMT			IWSLT	
	DE-EN	EN-FI	RO-EN	EN-FR	CS-EN
Words 50K	31.6	12.6	27.1	33.6	21.0
BPE 32K	**33.5**	**14.7**	**27.8**	34.5	22.6
BPE 16K	33.1	**14.7**	**27.8**	**34.8**	**23.0**

Table 2: BLEU scores for training NMT models with full word and byte pair encoded vocabularies. Full word models limit vocabulary size to 50K. All models are trained with annealing Adam and scores are averaged over 3 optimizer runs.

4.2 Results and Analysis

We measure the effects of byte pair encoding by training full-word and BPE systems for all data sets as described in §2.1 with the incremental improvement of using Adam with rate annealing (§3). As Wu et al. (2016) show different levels of effectiveness for different sub-word vocabulary sizes, we evaluate running BPE with 16K and 32K

merge operations. As shown in Table 2, sub-word systems outperform full-word systems across the board, despite having fewer total parameters. Systems built on larger data generally benefit from larger vocabularies while smaller systems perform better with smaller vocabularies. Based on these results, we recommend 32K as a generally effective vocabulary size and 16K as a contrastive condition when building systems on less than 1 million parallel sentences.

To understand the origin of these improvements, we divide the words in each test set into classes based on how the full-word and BPE models handle them and report the unigram F-1 score for each model on each class. We also plot the full-word and BPE vocabularies for context. As shown in Figure 2, performance is comparable for the most frequent words that both models represent as single units. The identical shapes on the left-most part of each vocabulary plot indicate that the two systems have the same number of training instances from which to learn translations. For words that are split in the BPE model, performance is tied to data sparsity. With larger data, performance is comparable as both models have enough training instances to learn reliable statistics; with smaller data or morphologically rich languages such as Finnish, significant gains can be realized by modeling multiple higher-frequency sub-words in place of a single lower-frequency word. This can be seen as effectively moving to the left in the vocabulary plot where translations are more reliable. In the next category of words beyond the 50K cutoff, the BPE system's ability to actually model rare words leads to consistent improvement over the full-word system's reliance on dictionary substitution.

The final two categories evaluate handling of true out-of-vocabulary items. For OOVs that should be translated, the full-word system will always score zero, lacking any mechanism for producing words not in its vocabulary or dictionary. The more interesting result is in the relatively low scores for OOVs that should simply be copied from source to target. While phrase-based systems can reliably pass OOVs through 1:1, full-word neural systems must generate `unk` tokens and correctly map them to source words using attention scores. Differences in source and target true vocabulary sizes and frequency distributions often lead to different numbers of `unk` tokens in source and target sentences, resulting in models that are prone to over or under-generating `unks` at test time. BPE systems address these weaknesses, although their performance is not always intuitive. While some OOVs are successfully translated using word pieces, overall scores are still quite low, indicating only limited success for the notion of open vocabulary translation often associated with sub-word NMT. However, the ability to learn when to self-translate sub-words[10] leads to significant gains in pass-through accuracy.

In summary, our analysis indicates that while BPE does lead to smaller, faster models, it also significantly improves translation quality. Rather than being limited to only rare and unseen words, modeling higher-frequency sub-words in place of lower-frequency full words can lead to significant improvement across the board. The specific improvement in pass-through OOV handling can be particularly helpful for handling named entities and open-class items such as numbers and URLs without additional dedicated techniques.

5 Ensembles and Model Diversity

The final technique we explore is the combination of multiple translation models into a single, more powerful *ensemble* by averaging their predictions at the word level. The idea of ensemble averaging is well understood and widely used across machine learning fields and work from the earliest encoder-decoder papers to the most recent system descriptions reports dramatic improvements in BLEU scores for model ensembles (Sutskever et al., 2014; Sennrich et al., 2016a). While this technique is conceptually simple, it requires training and decoding with multiple translation models, often at significant resource costs. However, these costs are either mitigated or justified when building real-world systems or evaluating techniques that should be applicable to those systems. Decoding costs can be reduced by using *knowledge distillation* techniques to train a single, compact model to replicate the output of an ensemble (Hinton et al., 2015; Kuncoro et al., 2016; Kim and Rush, 2016). Researchers can skip this time-consuming step, evaluating the ensemble directly, while real-world system engineers can rely on it to make deployment of ensembles practical. To re-

[10] Learning a single set of BPE operations by concatenating the source and target training data ensures that the same word will always be segmented in the same way whether it appears on the source or target side.

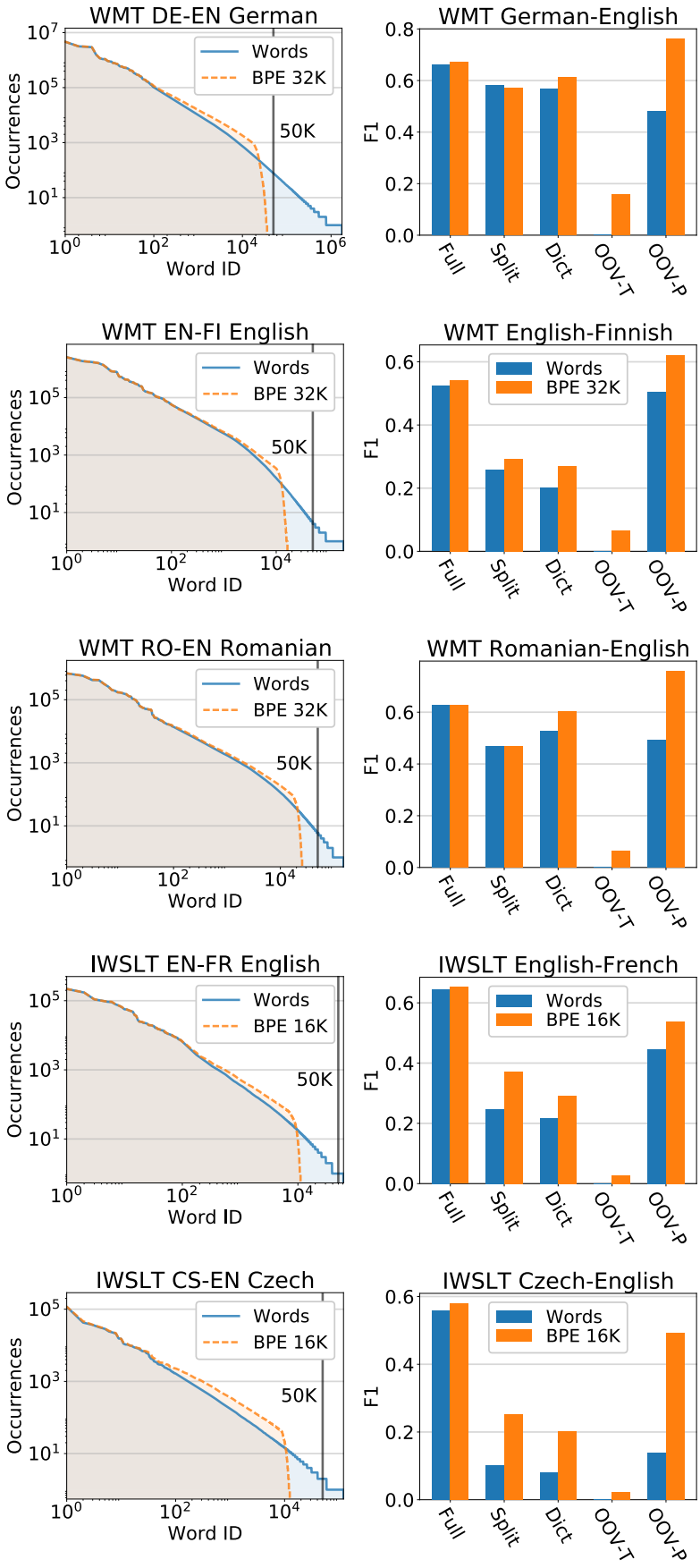


Figure 2: Effects of using subword units on model vocabulary and translation accuracy for specific types of words.
Left figures: Source vocabulary visualizations for NMT training data using full words and byte-pair encoded tokens. The number of merge operations is set to either 32K or 16K, chosen by best BLEU score. BPE reduces vocabulary size by 1-2 orders of magnitude and allows models to cover the entire training corpus. Full-word systems for all scenarios use a much larger vocabulary size of 50K (labeled horizontal line) that leaves much of the total vocabulary uncovered.
Right figures: Class-wise test set unigram F1 scores for NMT systems using full words and byte-pair encoded tokens. Scores are reported separately for the following classes: words in the vocabulary of both the full-word and BPE models (Full), words in the vocabulary of the full-word model that are split in the BPE model (Split), words outside the vocabulary of the full-word model but covered by its dictionary (Dict), words outside the vocabulary of the full-word model and its dictionary that should be translated (OOV-T), and words outside the vocabulary of the full-word model and its dictionary that should be passed through (OOV-P). All reported scores are averaged over 3 independent optimizer runs.

	WMT			IWSLT	
	DE-EN	EN-FI	RO-EN	EN-FR	CS-EN
Vanilla	30.2	11.8	26.4	33.2	20.2
Recommended	33.5	14.7	27.8	34.5	22.6
+Ensemble	**35.8**	**17.3**	**30.3**	**37.3**	**25.5**

Table 3: Test set BLEU scores for "vanilla" NMT (full words and standard Adam), and our recommended systems (byte pair encoding and annealing Adam, with and without ensembling). Scores for single models are averaged over 3 independent optimizer runs while scores for ensembles are the result of combining 3 runs.

duce training time, some work ensembles different training checkpoints of the same model rather than using fully independent models (Jean et al., 2015; Sennrich et al., 2016a). While checkpoint ensembling is shown to be effective for improving BLEU scores under resource constraints, it does so with less diverse models. As discussed in recent work and demonstrated in our experiments in §6, model diversity is a key component in building strong NMT ensembles (Jean et al., 2015; Sennrich et al., 2016a; Farajian et al., 2016). For these reasons, we recommend evaluating new techniques on systems that ensemble multiple independently trained models for the most reliable results. Results showing both the effectiveness of ensembles and the importance of model diversity are included in the larger experiments conducted in the next section.

6 On Trustable Evaluation

6.1 Experimental Setup

In this section, we evaluate and discuss the effects that choice of baseline can have on experimental conclusions regarding neural MT systems. First, we build systems that include Adam with rate annealing, byte pair encoding, and independent model ensembling and compare them to the vanilla baselines described in §2.1. As shown in Table 3, combining these techniques leads to a consistent improvement of 4-5 BLEU points across all scenarios. These improvements are the result of addressing several underlying weaknesses of basic NMT models as described in previous sections, leading to systems that behave much closer to those deployed for real-world tasks.

Next, to empirically demonstrate the importance of evaluating new methods in the context of these stronger systems, we select several techniques shown to improve NMT performance and compare their effects as baseline systems are iteratively strengthened. Focusing on English-French and Czech-English, we evaluate the following techniques with and without the proposed improvements, reporting results in Table 4:

Dropout: Apply the improved dropout technique for sequence models described by Gal and Ghahramani (2016) to LSTM layers with a rate of 0.2. We find this version to significantly outperform standard dropout.

Lexicon bias: Incorporate scores from a pre-trained lexicon (`fast_align` model learned on the same data) directly as additional weights when selecting output words (Arthur et al., 2016). Target word lexicon scores are computed as weighted sums over source words based on attention scores.

Pre-translation: Translate source sentences with a traditional phrase-based system trained on the same data. Input for the neural system is the original source sentence concatenated with the PBMT output (Niehues et al., 2016). Input words are prefixed with either `s_` or `t_` to denote source or target language. We improve performance with a novel extension where word alignments are used to weave together source and PBMT output so that each original word is immediately followed by its

EN-FR	Adam		+Annealing		+Ensemble
	Word	BPE	Word	BPE	BPE
Baseline	33.2	33.7	33.6	34.8	37.3
Dropout	**33.9**	**33.9**	**34.5**	*34.7*	*37.2*
Lexicon Bias	**33.8**	**34.0**	**33.9**	*34.8*	*37.1*
Pre-Translation	–	**34.0**	–	**34.9**	*36.6*
Bootstrapping	**33.7**	**34.1**	**34.4**	**35.2**	**37.4**

CS-EN	Adam		+Annealing		+Ensemble
	Word	BPE	Word	BPE	BPE
Baseline	20.2	22.1	21.0	23.0	25.5
Dropout	**20.7**	**22.7**	**21.4**	**23.6**	**26.1**
Lexicon Bias	**20.7**	**22.5**	*20.6*	*22.7*	*25.2*
Pre-Translation	–	**23.1**	–	**23.8**	**25.8**
Bootstrapping	**20.7**	**23.2**	**21.6**	**23.6**	**26.2**

Table 4: Test set BLEU scores for several published NMT extensions. Entries are evaluated with and without Adam annealing, byte pair encoding, and model ensembling. A bold score indicates improvement over the baseline while an italic score indicates no change or degradation. Scores for non-ensembles are averaged over 3 independent optimizer runs and ensembles are the result of combining 3 runs.

suggested translation from the phrase-based system. As pre-translation doubles source vocabulary size and input length, we only apply it to sub-word systems to keep complexity reasonable.

Data bootstrapping: Expand training data by extracting phrase pairs (sub-sentence translation examples) and including them as additional training instances (Chen et al., 2016). We apply a novel extension where we train a phrase-based system and use it to re-translate the training data, providing a near-optimal phrase segmentation as a byproduct. We use these phrases in place of the heuristically chosen phrases in the original work, improving coverage and leading to more fine-grained translation examples.

6.2 Experimental Results

The immediately noticeable trend from Table 4 is that while all techniques improve basic systems, only a single technique, data bootstrapping, improves the fully strengthened system for both data sets (and barely so). This can be attributed to a mix of redundancy and incompatibility between the improvements we've discussed in previous sections and the techniques evaluated here.

Lexicon bias and pre-translation both incorporate scores from pre-trained models that are shown to improve handling of rare words. When NMT models are sub-optimally trained, they can benefit from the suggestions of a better-trained model. When full-word NMT models struggle to learn translations for infrequent words, they can learn to simply trust the lexical or phrase-based model. However, when annealing Adam and BPE alleviate these underlying problems, the neural model's accuracy can match or exceed that of the pre-trained model, making external scores either completely redundant or (in the worst case) harmful bias that must be overcome to produce correct translations. While pre-translation fares better than lexicon bias, it suffers a reversal in one scenario and a significant degradation in the other when moving from a single model to an ensemble. Even when bias from an external model improves translation, it does so at the cost of diversity by pushing the neural model's preferences toward those of the pre-trained model. These results further validate claims of the importance of diversity in model ensembles.

Applying dropout significantly improves all configurations of the Czech-English system and some configurations of the English-French system, leveling off with the strongest. This trend follows previous work showing that dropout combats overfitting of small data, though the point of inflection is worth noting (Sennrich et al., 2016a; Wu et al., 2016). Even though the English-French data is still relatively small (220K sentences), BPE leads to a smaller vocabulary of more general translation units, effectively reducing sparsity, while annealing Adam can avoid getting stuck in poor local optima. These techniques already lead to better generalization without the need for dropout. Finally, we can observe a few key properties of data bootstrapping, the best performing technique on fully strengthened systems. Unlike lexicon bias and pre-translation, it modifies only the training data, allowing "purely neural" models to be learned from random initialization points. This preserves model diversity, allowing ensembles to benefit as well as single models. Further, data bootstrapping is complementary to annealing Adam and BPE; better optimization and a more general vocabulary can make better use of the new training instances.

While evaluation on simple vanilla NMT systems would indicate that all of the techniques in this section lead to significant improvement for both data sets, only evaluation on systems using annealing Adam, byte pair encoding, and independent model ensembling reveals both the reversals of results on state-of-the-art systems and nuanced interactions between techniques that we have reported. Based on these results, we highly recommend evaluating new techniques on systems that are at least this strong and representative of those deployed for real-world use.

7 Conclusion

In this work, we have empirically demonstrated the effectiveness of Adam training with multiple restarts and step size annealing, byte pair encoding, and independent model ensembling both for improving BLEU scores and increasing the reliability of experimental results. Out of four previously published techniques for improving vanilla NMT, only one, data bootstrapping via phrase extraction, also improves a fully strengthened model across all scenarios. For these reasons, we recommend evaluating new model extensions and algorithms on NMT systems at least as strong as those we have described for maximally trustable results.

References

Philip Arthur, Graham Neubig, and Satoshi Nakamura. 2016. Incorporating discrete translation lexicons into neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, pages 1557–1567.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurelie Neveol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation*. Association for Computational Linguistics, Berlin, Germany, pages 131–198.

Denny Britz, Anna Goldie, Thang Luong, and Quoc Le. 2017. Massive exploration of neural machine translation architectures. *arXiv preprint arXiv:1703.03906* .

M Cettolo, J Niehues, S Stüker, L Bentivogli, R Cattoni, and M Federico. 2016. The iwslt 2016 evaluation campaign .

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit^3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*. Trento, Italy, pages 261–268.

Wenhu Chen, Evgeny Matusov, Shahram Khadivi, and Jan-Thorsten Peter. 2016. Guided alignment training for topic-aware neural machine translation. *AMTA 2016, Vol.* page 121.

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. pages 176–181.

Josep Crego, Jungi Kim, Guillaume Klein, Anabel Rebollo, Kathy Yang, Jean Senellart, Egor Akhanov, Patrice Brunelle, Aurelien Coquard, Yongchao Deng, Satoshi Enoue, Chiyo Geiss, Joshua Johanson, Ardas Khalsa, Raoum Khiari, Byeongil Ko, Catherine Kobus, Jean Lorieux, Leidiana Martins, Dang-Chuan Nguyen, Alexandra Priori, Thomas Riccardi, Natalia Segal, Christophe Servan, Cyril Tiquet, Bo Wang, Jin Yang, Dakun Zhang, Jing Zhou, and Peter Zoldan. 2016. SYSTRAN's pure neural machine translation systems. *CoRR* abs/1610.05540.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. pages 644–648.

M Amin Farajian, Rajen Chatterjee, Costanza Conforti, Shahab Jalalvand, Vevake Balaraman, Mattia A Di Gangi, Duygu Ataman, Marco Turchi, Matteo Negri, and Marcello Federico. 2016. Fbks neural machine translation systems for iwslt 2016. In *Proceedings of the ninth International Workshop on Spoken Language Translation (IWSLT), Seattle, WA*.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, Curran Associates, Inc., pages 1019–1027.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* .

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, pages 1–10.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, pages 1317–1327.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, and Noah A. Smith. 2016. Distilling an ensemble of greedy dependency parsers into one mst parser. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, pages 1744–1753.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2016. Fully character-level neural machine translation without explicit segmentation. *CoRR* abs/1610.03017.

Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals, and Wojciech Zaremba. 2015. Addressing the rare

word problem in neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. pages 11–19.

Graham Neubig. 2015. lamtram: A toolkit for language and translation modeling using neural networks. http://www.github.com/neubig/lamtram.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui, Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. Dynet: The dynamic neural network toolkit. *arXiv preprint arXiv:1701.03980* .

Jan Niehues, Eunah Cho, Thanh-Le Ha, and Alex Waibel. 2016. Pre-translation for neural machine translation. *CoRR* abs/1610.05243.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, pages 311–318.

Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nadejde. 2017. Nematus: a Toolkit for Neural Machine Translation. In *Proceedings of the Demonstrations at the 15th Conference of the European Chapter of the Association for Computational Linguistics*. Valencia, Spain.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Edinburgh neural machine translation systems for wmt 16. In *Proceedings of the First Conference on Machine Translation*. Association for Computational Linguistics, Berlin, Germany, pages 371–376.

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)*. Association for Computational Linguistics, Berlin, Germany, pages 1715–1725.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. pages 3104–3112.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR* abs/1609.08144.

Matthew D. Zeiler. 2012. ADADELTA: an adaptive learning rate method. *CoRR* abs/1212.5701.