Investigating Softmax Tempering for Training Neural Machine Translation Models

Raj Dabre raj.dabre@nict.go.jp
Atsushi Fujita atsushi.fujita@nict.go.jp
National Institute of Information and Communications Technology,
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-0289, Japan

Abstract

Neural machine translation (NMT) models are typically trained using a softmax cross-entropy loss where the softmax distribution is compared against the gold labels. In low-resource scenarios, NMT models tend to perform poorly because the model training quickly converges to a point where the softmax distribution computed using logits approaches the gold label distribution. Although label smoothing is a well-known solution to address this issue, we further propose to divide the logits by a temperature coefficient greater than one, forcing the softmax distribution to be smoother during training. This makes it harder for the model to quickly over-fit. In our experiments on 11 language pairs in the low-resource Asian Language Treebank dataset, we observed significant improvements in translation quality. Our analysis focuses on finding the right balance of label smoothing and softmax tempering which indicates that they are orthogonal methods. Finally, a study of softmax entropies and gradients reveals the impact of our method on the internal behavior of our NMT models.

1 Introduction

Neural machine translation (NMT) (Sutskever et al., 2014; Bahdanau et al., 2015) enables end-to-end training of translation models and is known to give state-of-the-art results for a large variety of language pairs. NMT for high-resource language pairs is straightforward: choose an NMT architecture and implementation, and train a model on all existing data by minimizing the softmax cross-entropy loss, i.e., cross-entropy between the softmax distribution and the label distribution typically represented with a one-hot vector. In contrast, for low-resource language pairs, this does not work well due to the inability of neural networks to generalize from small amounts of data. One reason for this is over-fitting (Zoph et al., 2016; Koehn and Knowles, 2017), where the softmax distribution (sparse vector) ends up resembling the label distribution (one-hot vector).

There are several solutions that address this issue, of which the two most effective ones are transfer learning and model regularization. Transfer learning can sometimes be considered as data regularization and comes in the form of monolingual or cross-lingual (multilingual) fashion (Zoph et al., 2016; Song et al., 2019), pseudo-parallel data generation (back-translation) (Sennrich et al., 2016), or multi-task learning (Eriguchi et al., 2017). On the other hand, model regularization techniques place constraints on the learning of model parameters in order to aid the model to learn robust representations that positively impact model performance. Among existing model regularization methods, dropout (Srivastava et al., 2014) is most commonly used and is known to be effective regardless of the size of data. Label smoothing (Szegedy

et al., 2016) is another effective approach that uses smoothed label vectors as opposed to one-hot label vectors. Previous work on NMT has shown that label smoothing is very effective in low-resource settings (Sennrich and Zhang, 2019) and we believe that this deserves further study. We thus focus on a technique that does not need additional data and can complement dropout and label smoothing in an extremely low-resource situation.

In this paper, we propose to apply *softmax tempering* (Hinton et al., 2015) to the training of NMT models. Softmax tempering is realized by dividing the pre-softmax logits with a positive real number greater than 1.0. This leads to a smoother softmax probability distribution, which is then used to compute the cross-entropy loss. Softmax tempering has been devised and used regularly in knowledge distillation (Hinton et al., 2015; Kim and Rush, 2016) and model calibration (Guo et al., 2017) albeit for different purposes. We regard softmax tempering as a means of deliberately making the softmax distribution noisy during training with the expectation that this will have a positive impact on the final translation quality. It is especially important to note that calibration involves tempering after a model has been trained whereas we perform tempering during training.

We primarily evaluate the utility of softmax tempering on extremely low-resource settings involving English and 11 languages in the Asian Languages Treebank (ALT) (Riza et al., 2016). Our experiments reveal that softmax tempering with a reasonably high temperature improves the translation quality. Furthermore, greedy-search performance of models trained with softmax tempering becomes comparable to or better than the beam-search performance of models that are trained without softmax tempering. Our analysis focuses on the orthogonality of softmax tempering and label smoothing. We additionally compare these methods with the related softmax entropy maximization method (Pereyra et al., 2017). Finally, we analyze the impact of softmax tempering on the softmax distributions and on the gradient flows during training.

2 Related Work

The method presented in this paper is a training technique aimed to improve the quality of NMT models in low-resource scenarios.

Work on knowledge distillation (Hinton et al., 2015) for training compact models is highly related to our application of softmax tempering. However, the purpose of softmax tempering for knowledge distillation is to smooth the student and teacher distributions which is known to have a positive impact on the quality of student models. In our case, we use softmax tempering to make softmax distributions noisy during training a model from scratch to avoid over-fitting. In the context of NMT, Kim and Rush (2016) conducted experiments with softmax tempering. However, their focus was on model compression and they did not experiment with low-resource settings. Softmax tempering is also used in model calibration (Guo et al., 2017; Kumar and Sarawagi, 2019), where the temperature coefficient is optimized on the development set in order to penalize overconfident predictions, which is a common practice in low-resource settings. While model calibration is performed after a model is trained, we use softmax tempering during training.

We regard softmax tempering as a regularization technique, since it adds noise to NMT model training. Thus, it is related to techniques, such as L_N regularization (Ng, 2004), dropout (Srivastava et al., 2014), and tuneout (Miceli Barone et al., 2017). The most important aspect of our method is that it is only applied at the softmax layer whereas other regularization techniques add noise to several parts of the entire model. Label smoothing (Szegedy et al., 2016), which is known to help low-resource NMT (Sennrich and Zhang, 2019), is highly related to our idea, where the key difference is that label smoothing affects the label distributions whereas softmax tempering affects the softmax distributions. On a related note, softmax entropy maximization (Pereyra et al., 2017) seeks to mitigate overconfident predictions but is not known to work well

for NMT. Our method is intended to complement these techniques, i.e., label smoothing and softmax entropy maximization, and not necessarily replace them.

Existing methods effective for low-resource language pairs include data augmentation via back-translating additional monolingual data (Sennrich et al., 2016), exploitation of multilingualism (Firat et al., 2016; Zoph et al., 2016; Dabre et al., 2019), and pre-training on monolingual data (Devlin et al., 2019; Song et al., 2019; Mao et al., 2020). These require more training time and resources, while ours does not.

3 Softmax Tempering

Softmax tempering (Hinton et al., 2015) consists of two tiny changes in the implementation of the training phase of any neural model used for classification.

Assume that $D_i \in \mathbb{R}^V$ is the logit output of the decoder for the i-th word prediction in the target language sentence, Y_i, where V stands for the target vocabulary size, and that $P_i = P(Y_i|Y_{<i}, X) = softmax(D_i)$ represents the softmax function producing the probability distribution, where X and $Y_{<i}$ indicate the given source sentence and the past decoder output, respectively. Let $R_i \in \mathbb{R}^V$ be the label-smoothed reference label for the i-th prediction. Then, the cross-entropy loss for the prediction is computed as $\mathcal{L}_i = -\langle \log(P_i), R_i \rangle$, where $\langle \cdot, \cdot \rangle$ is the inner product of two vectors.

Let $T \in \mathbb{R}_+$ be the temperature hyper-parameter. Then, the prediction with softmax tempering (P_i^{temp}) and the corresponding cross-entropy loss ($\mathcal{L}_i^{temp}$) are formalized as follows.

$$P_i^{temp} = P^{temp}(Y_i|Y_{<i}, X) = softmax(D_i/T), \tag{1}$$

$$\mathcal{L}_i^{temp} = -\langle \log(P_i^{temp}), R_i \rangle \cdot T \tag{2}$$

By referring to Equation (1), when T is greater than 1.0, the logits, D_i, are down-scaled which leads to a smoother probability distribution before loss is computed. The smoother the distribution becomes, the higher its entropy is and hence the more uncertain the prediction is. Because loss is to be minimized, back-propagation will force the model to generate logits to counter the smoothing effect of temperature. During decoding with a model trained in this way, the temperature coefficient is also used which mitigates overconfident predictions[1] stemming from tempering during training.

The gradients are altered by tempering, and we thus re-scale the loss by the temperature as shown in Equation (2). This is inspired by the loss scaling method used in knowledge distillation (Hinton et al., 2015), where both the student and teacher's softmax distributions are tempered and the loss is multiplied by the square of the temperature.

4 Experiments

To evaluate the effectiveness of softmax tempering, we conducted experiments on both low-resource and high-resource settings.

4.1 Datasets

We experimented with the Asian Languages Treebank (ALT),[2] comprising English (En) news articles consisting of 18,088 training, 1,000 development, and 1,018 test sentences manually translated into 11 Asian languages: Bengali (Bn), Filipino (Fil), Indonesian (Id), Japanese (Ja), Khmer (Km), Lao (Lo), Malay (Ms), Burmese (My), Thai (Th), Vietnamese (Vi), and Chinese

[1] This is characterized by sharp probability distributions where the most probable word has an extremely high probability value approaching 1.0.

[2] http://www2.nict.go.jp/astrec-att/member/mutiyama/ALT/ALT-Parallel-Corpus-20190531.zip

(Zh). We focused on translation to and from English to each of these 11 languages. As a high-resource setting, we also experimented with the WMT 2019 English-to-German (En→De) translation task.[3] For training, we used the Europarl and the ParaCrawl corpora containing 1.8M and 37M sentence pairs, respectively. For evaluation, we used the WMT 2019 development and test sets consisting of 2,998 and 1,997 lines, respectively.

4.2 Implementation Details

We evaluated softmax tempering on top of the Transformer model (Vaswani et al., 2017), which gives the state-of-the-art results for NMT. More specifically, we employed the following models.

- En→XX and XX→En "Transformer Base" models where XX is an Asian language.
- En→De "Transformer Base" and "Transformer Big" models.

We modified the code of the Transformer model in the tensor2tensor v1.14.[4] For "Transformer Base" and "Transformer Big" models, we used the hyper-parameter settings in *transformer_base_single_gpu* and *transformer_big_single_gpu*, respectively. Label smoothing of 0.1 was used. We used the internal sub-word tokenization mechanism of tensor2tensor with separate source and target language vocabularies of size 8,192 and 32,768 for low-resource and high-resource settings, respectively.

We trained our models for each of the softmax temperature values, 1.0 (default softmax), 1.2, 1.4, 1.6, 1.8, 2.0, 3.0, 4.0, 5.0, and 10.0. We used early-stopping on the BLEU score (Papineni et al., 2002) for the development set which was evaluated every 1k iterations. Our early-stopping mechanism halts training when the BLEU score does not improve over 10 consecutive evaluation steps. For decoding, we averaged the final 10 checkpoints, and evaluated beam search and greedy search. Note that the training time temperature coefficient was used during decoding as well. If this is not done then the softmax distributions will be extremely sharp and beam search will collapse to greedy search.

4.3 Evaluation Criteria

We evaluated translation quality of each model using BLEU (Papineni et al., 2002) provided by *SacreBLEU* (Post, 2018).[5] The optimal temperature (T_{opt}) for the tempered model was determined based on greedy-search BLEU score on the development set, given that beam- and greedy-search score improvements are almost always correlated. We therefore used these optimal temperature models to perform beam search, where the beam width (among 2, 4, 6, 8, 10, and 12) and length penalty (among 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, and 1.4) were tuned on the development set. We performed statistical significance testing[6] to determine if differences in BLEU are significant.

4.4 Results in Low-Resource Settings

Table 1 shows the greedy- and beam-search BLEU scores along with the optimal temperature (T_{opt}) for translation to and from Asian languages and compare them against those obtained by non-tempered models. In most cases, the greedy-search BLEU scores of the best performing tempered models are higher than the beam-search BLEU scores of non-tempered models.

Figure 1 shows how the greedy- and beam-search results vary with the temperature, taking Ms→En and Id→En translation tasks as examples. As the temperature is raised, both the greedy- and beam-search BLEU scores increase peaking between a temperature of 3.0 and 5.0.

[3] http://www.statmt.org/wmt19/translation-task.html

[4] https://github.com/tensorflow/tensor2tensor

[5] https://github.com/mjpost/sacrebleu, BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.5.0

[6] https://github.com/moses-smt/mosesdecoder/blob/master/scripts/analysis/bootstrap-hypothesis-difference-significance.pl

T	Decoding	En→XX										
		Bn	Fil	Id	Ja	Km	Lo	Ms	My	Th	Vi	Zh
1.0	Greedy	3.5	24.3	27.4	13.4	19.3	11.5	31.5	8.3	13.7	24.0	10.4
1.0	Beam	4.1	25.8	28.7	15.0	21.3	13.0	32.6	9.1	15.9	26.5	12.1
T_{opt}	Greedy	4.5	25.7	29.5†	15.5	20.7	11.8	33.7†	9.3	15.6	25.8	12.9†
T_{opt}	Beam	**4.7**	**27.0**†	**30.2**†	**17.5**†	**22.3**†	**13.3**†	**34.7**†	**10.6**†	**17.4**†	**27.5**†	**15.1**†
Value for T_{opt}		5.0	3.0	4.0	4.0	5.0	5.0	4.0	5.0	5.0	3.0	5.0
T	Decoding	XX→En										
		Bn	Fil	Id	Ja	Km	Lo	Ms	My	Th	Vi	Zh
1.0	Greedy	7.1	22.2	25.1	8.7	14.9	9.8	27.4	7.8	10.5	19.4	9.4
1.0	Beam	8.5	24.0	26.3	9.9	16.4	11.9	28.5	9.3	12.4	20.9	10.8
T_{opt}	Greedy	9.1	24.7	27.5†	11.0†	16.8	11.4	29.7†	11.7†	12.2	21.3	11.5
T_{opt}	Beam	**10.4**†	**26.3**†	**28.2**†	**12.9**†	**18.0**†	**12.9**†	**30.3**†	**13.3**†	**13.7**†	**22.1**†	**12.9**†
Value for T_{opt}		5.0	5.0	3.0	5.0	4.0	5.0	4.0	4.0	4.0	4.0	5.0

Table 1: BLEU scores for the ALT En→XX and XX→En tasks, where XX is one of the Asian languages in the ALT dataset, obtained by non-tempered ($T = 1.0$) and tempered ($T = T_{opt}$) NMT models with greedy and beam search. Best BLEU scores are in bold. "†" marks scores that are significantly ($p < 0.05$) better than non-tempered model's beam-search scores.

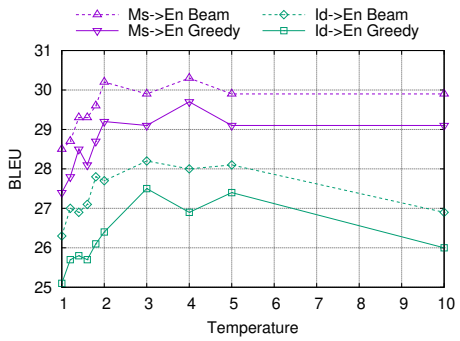

Figure 1: Improvement of greedy- and beam-search BLEU scores with temperature for the Ms→En and Id→En translation tasks.

While the gap between greedy- and beam-search scores was around 1.3 for a temperature of 1.0 (non-tempered), it narrows down to about 0.8 for temperatures which give the best greedy-search score. Furthermore, the best beam-search score almost always corresponds with the best greedy-search score which justifies our choice of the optimal temperature based on greedy-search performance. However, increasing the temperature beyond 10.0 always has a negative effect on the translation quality, because it leads to an excessively smoothed distribution, quantified by high entropy, that does not seem to be useful for NMT training. Consequently, we conclude that training with reasonably high temperature (between 3.0 and 5.0), softmax tempering has a positive impact on translation quality for extremely low-resource settings.

4.5 Results in High-Resource Settings

Table 2 gives the BLEU scores for the high-resource En→De translation task. The results indicate that compared to the low-resource settings, relatively lower temperature values are effective for improving translation quality. Greedy and beam search respectively improve by 0.8 to 2.3 BLEU points for temperature values around 1.2 to 1.4. For the models trained only on the Europarl corpus (EP), the greedy- and beam-search performances of the Transformer Base model starts approaching those of the Transformer Big model. However, when the very large ParaCrawl corpus is used (PC), the gains are around 1.0 BLEU and thus the impact of tempering

Model	Training	T	BLEU	
			Greedy	Beam
Base	EP	1.0	23.6	25.8
		1.4	25.5	**27.6**†
	PC	1.0	28.2	29.2
		1.2	29.1	**30.3**†
Big	EP	1.0	26.8	29.4
		1.2	29.1	**30.2**†
	PC	1.0	32.7	33.7
		1.2	33.6	**34.7**†

Table 2: BLEU scores for the En→De task obtained by non-tempered ($T = 1.0$) and tempered ($T = T_{opt}$) NMT models exclusively trained on Europarl (EP) and ParaCrawl (PC) corpora.

appears to reduce as corpora sizes increase. Using higher temperature values deteriorates translation quality and thus we do not recommend using high temperature values in high-resource settings. This happens presumably because the larger corpora sizes (cf. ALT corpora) enable data regularization and do not need model regularization. Overall, these experiments show that softmax tempering is very important in low-resource settings but not that important in high-resource settings. Note that we did not use any advanced methods, such as back-translation or ensembling, since our focus here was to examine the effectiveness of softmax tempering. In the future, we will explore the impact of softmax tempering on these advanced methods along with a study of how optimal temperatures vary with corpora sizes.

4.6 Impact on Training and Decoding Speed

Although training with softmax tempering makes it difficult for a model to over-fit the label distributions, we did not notice any large impact on the training time. This indicates that the improvements are unrelated to longer training times. With regard to decoding, in low-latency settings, we can safely use greedy search with tempered models given that it is as good as, if not better than, beam search using non-tempered models. Thus, by comparing the greedy- and beam-search decoding speeds, we can determine the benefits that softmax tempering brings in low-latency settings. Greedy-search decoding of the Vi→En[7] test set requires 37.6s on average, whereas beam search with beam sizes of 4 and 10 require 56.4s and 138.2s, respectively. For non-tempered models, where beam-search scores are higher than greedy-search scores by over 2.0 BLEU points, and the best BLEU scores are obtained using beam sizes between 4 and 10. Given the improved performance with greedy search, we can decode anywhere from 1.5 to 3.5 times faster. This also justifies our decision to choose optimal temperature using greedy-search scores. Subjecting softmax tempering to model compression methods, such as weight pruning, might further reduce decoding time.

5 Analysis and Further Exploration

Softmax tempering directly manipulates the softmax distribution making it noisy (smoother). In this section, we explore the relationship between softmax tempering and its closest related methods that directly affect the softmax distribution, i.e., label smoothing and entropy maximization. We also study the internal working of the softmax-tempered models during training. For these analyses, we focus on extremely low-resource settings, since our results in Section 4.5 demonstrate that softmax tempering is less impactful in high-resource settings. We especially take Bn→En, Ja→En, Ms→En, and Vi→En as examples due to lack of space. We focus on the models with optimal hyper-parameters determined via a grid search on greedy-search BLEU scores on the development set, and report on greedy-search BLEU scores on the test set.

[7]For ALT tasks, decoding times are very similar when translating into English due to it being a multi-parallel corpus.

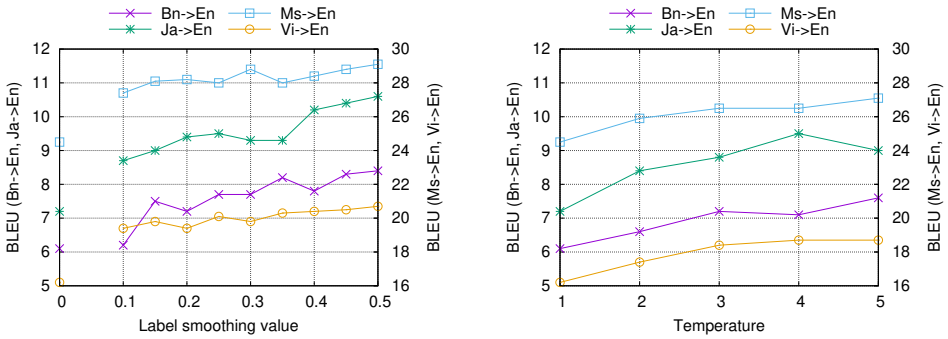

Figure 2: Independent investigation of impact of label smoothing and softmax tempering. On the left, label smoothing value, S, is varied from 0.1 to 0.5 in increments of 0.05 without softmax tempering (i.e., $T = 1.0$). In contrast, on the right, temperature value for softmax tempering, T, is varied from 1 to 5 in increments of 1 without label smoothing (i.e., $S = 0$).

Config.	Bn→En		Ja→En		Ms→En		Vi→En	
	BLEU	(T, S)	BLEU	(T, S)	BLEU	(T, S)	BLEU	(T, S)
Default	7.1	(1.0, 0.1)	8.7	(1.0, 0.1)	27.4	(1.0, 0.1)	19.4	(1.0, 0.1)
LS	8.4	(1.0, 0.5)	10.6	(1.0, 0.5)	29.1	(1.0, 0.5)	20.7	(1.0, 0.5)
Temp.	9.1	(5.0, 0.1)	11.0	(5.0, 0.1)	29.7	(4.0, 0.1)	21.3	(4.0, 0.1)
Temp. & LS	**9.4**	(5.0, 0.5)	**11.8**	(5.0, 0.5)	**30.1**	(5.0, 0.45)	**22.1**	(4.0, 0.45)

Table 3: Results of empirically searching the optimal values of softmax tempering and label smoothing. For each configuration, we present the greedy-search BLEU score and the hyper-parameter set (T for softmax tempering and S for label smoothing) that gives the score.

5.1 Softmax Tempering vs. Label Smoothing

Label smoothing (LS) involves using a smoothed reference label vector instead of a one-hot vector. Let S ($0 \leq S \leq 1$) be the amount of smoothing, where 0 indicates no smoothing. Then the label-smoothed vector contains a value of $(1 - S) + \frac{S}{V}$ in the position corresponding to the correct word and a value of $\frac{S}{V}$ elsewhere, where V is the size of the vocabulary. This prevents the softmax from being sharp which is known to have a strong impact on the final performance (Szegedy et al., 2016; Sennrich and Zhang, 2019). To this end, we first show how the individual impacts of LS and softmax tempering and then we show how they can be effectively combined for the best translation quality. Note that the results in the previous section were obtained using LS of 0.1 which is the default value in tensor2tensor.

The left figure in Figure 2 shows the effect of increasing the LS value (S) from 0.1 to 0.5 in increments of 0.05 for models trained without softmax tempering. BLEU scores for $S = 0.1$ are those in Table 1. We also give scores for $S = 0$ for reference. It is clear that $S = 0$ gives the worst BLEU scores indicating the fundamental importance of LS. Increasing the LS value leads to a general improvement in BLEU which peaks for an LS value of 0.5. Even though we did not test, LS values greater than 0.5 may give better results. On the other hand, the right figure in Figure 2 shows the effect of increasing the temperature with LS value (S) of 0, where translation quality improves with softmax tempering even without any LS. Earlier in Figure 1, we have also shown that increasing temperature while keeping $S = 0.1$ leads to an improvement in BLEU. As this may indicate complementarity between LS and softmax tempering, we examined their combination in further detail.

Table 3 shows that the best BLEU scores are obtained by combining softmax tempering and LS ("Temp. & LS"), along with the temperature and LS values. It also shows the scores of

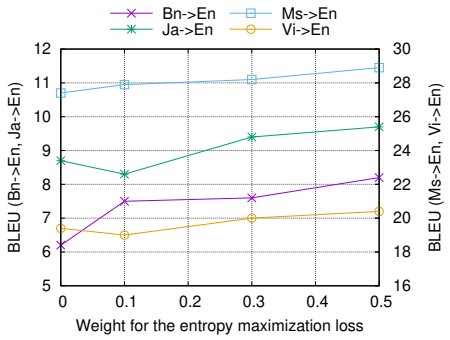

Figure 3: Effect of softmax entropy maximization on models trained without either softmax tempering (default temperature of 1.0) or fixed label smoothing (default value of 0.1).

the models with default LS value ($S = 0.1$) and no tempering ("Default"), best label-smoothed models ("LS") without softmax tempering, and the best softmax tempered models with a fixed LS value of $S = 0.1$ ("Temp.") for reference. Consistently with the results in Figure 2, using high values of temperatures and LS individually lead to better results compared to using no tempering and low values of LS. High values of LS, e.g., $[0.4, 0.5]$, can also lead to translations whose quality approaches those of the best tempered models with a low value of LS ($S = 0.1$). However, softmax tempering is often slightly, if not significantly, better than LS. Ultimately, their combination gives a further improvement in translation quality ranging from 0.3 to 1.0 BLEU points. Even when combining, the best temperature and LS values are between 3.0 to 5.0 and 0.4 to 0.5, respectively. These results show that while increased LS can significantly improve translation quality, softmax tempering is what pushes the translation quality to its limit. Nevertheless, the importance of LS should not be discounted because the combination of softmax tempering and LS is consistenty better than softmax tempering, even if by a small amount, for all translation directions we experimented with.

5.2 Softmax Tempering vs. Softmax Entropy Maximization

Softmax entropy maximization (SEM) is a method to penalize overconfident predictions by producing smoother softmax distributions (Pereyra et al., 2017) which should help in mitigating over-fitting issues prevalent in NMT. As this method directly affects the softmax and has the opposite impact of tempering, we consider its comparison with tempering to be important. SEM can be done by an additional loss which can be combined with the cross-entropy loss. Let P_i be the softmax distribution (regardless of tempering). Then the negative softmax entropy, $\mathcal{L}_{\text{NSE}} = \langle \log(P_i), \log(P_i) \rangle$, is regarded as a loss, which is combined with the cross-entropy loss, $\mathcal{L}_{\mathcal{X}}$. For instance, one can define the final loss to be minimized by linearly interpolating them with a weight w ($0 \leq w \leq 1$) as $\mathcal{L} = w \cdot \mathcal{L}_{\text{NSE}} + (1 - w) \cdot \mathcal{L}_{\mathcal{X}}$.

First, we determined whether SEM really helps or not with SEM loss weights 0.0, 0.1, 0.3, and 0.5.[8] Here, softmax tempering is not applied whereas label smoothing is performed with the default value of 0.1. As shown in Figure 3, in the absence of strong label smoothing and softmax tempering, SEM is important in these low-resource settings: improvements of 1.0 to 2.0 BLEU points can be observed. Previous research on high-resource NMT showed that SEM is not very useful (Pereyra et al., 2017) and, to the best of our knowledge, ours is the first work that confirms its importance in a low-resource setting.

Encouraged by these results, we further experimented with SEM in addition to the com-

[8] In our preliminary experiments, we observed drops in translation quality when $w > 0.5$.

Config.	Bn→En		Ja→En		Ms→En		Vi→En	
	BLEU	(T, S, w)	BLEU	(T, S, w)	BLEU	(T, S, w)	BLEU	(T, S, w)
SEM	8.2	(1.0, 0.1, 0.5)	9.7	(1.0, 0.1, 0.5)	28.9	(1.0, 0.1, 0.5)	20.4	(1.0, 0.1, 0.5)
LS & SEM	8.8	(1.0, 0.5, 0.1)	11.2	(1.0, 0.5, 0.3)	28.9	(1.0, 0.5, 0.3)	20.7	(1.0, 0.3, 0.3)
Temp. & SEM	8.8	(5.0, 0.1, 0.5)	11.5	(5.0, 0.1, 0.5)	**30.1**	(5.0, 0.1, 0.5)	21.6	(5.0, 0.1, 0.1)
Temp. & LS & SEM	**9.4**	(5.0, 0.5, 0.0)	**11.8**	(5.0, 0.5, 0.0)	**30.1**	(5.0, 0.45, 0.0)	**22.1**	(4.0, 0.45, 0.0)

Table 4: Results of empirically searching the optimal values of softmax tempering, label smoothing, and softmax entropy maximization. For each configuration, we present the greedy-search BLEU score and the hyper-parameter set (T for softmax tempering, S for label smoothing, and w for softmax entropy maximization) that gives the score.

bination of softmax tempering with T ranging from 1.0 to 5.0 (increments of 1.0) and label smoothing with S ranging from 0.1 to 0.5 (increments of 0.05). We compared four training configurations: SEM is done for default values of tempering and label smoothing ("SEM"), SEM and label smoothing is done without tempering ("LS & SEM"), tempering and SEM is done for the default label smoothing ("Temp & SEM"), and when tempering, label smoothing, and SEM are performed jointly (last row). Table 4 shows the results. When label smoothing and temperature are kept to their default values, giving a high weight to the SEM loss gives improvements of over 1.0 BLEU points as observed in Figure 3. This shows that in low-resource settings, mitigating overconfident predictions by controlling softmax predictions is crucial even if mild label smoothing is already applied. When label smoothing and SEM are combined, without tempering, the translation quality improves but the SEM loss seems to matter less as lower values for SEM loss are preferred. This indicates that these two methods might cause the model to have similar behavior and thus are not strongly complementary. In contrast, tempering and SEM seem to be complementary as high temperatures and high weights for SEM loss lead to better results. This behavior can be explained by a visualization of softmax entropy in Section 5.3. The final row shows that when tempering and label smoothing are already used, SEM is more often than not useless, i.e., the optimal SEM loss weight is 0.0. This is partially observed in Pereyra et al. (2017) where SEM was not seen to be useful in high-resource settings. Regardless of our observations, we encourage readers to duly experiment with a combination of tempering, label smoothing, and SEM when working in low-resource scenarios.

5.3 Temperature and Model Learning

We expected that tempering leads to a smoother softmax distribution and that loss minimization using such a softmax makes it sharper as training progresses. With softmax tempering, the model will continue to receive strong gradient updates even during later training stages due to the deliberate perturbation of the softmax distribution. We examined whether our model truly behaves this way through visualizing the softmax entropies and gradient values.

Figure 4 visualizes the variation of softmax entropy averaged over all tokens in a batch during training. The left-hand side shows the entropy of tempered softmax distribution in Equation (1), where there is no visible differences between charts with different values for temperature, i.e., T. Considering that the distribution is tempered with T, this indicates that the distribution of logits, D_i, is sharper when tempered with a higher T. The right-hand side plots the entropy of softmax distribution derived from the logits without dividing them by T. The lower entropies confirm that the distribution of logits is indeed sharper with higher T and that division by T as in Equation (1) counters the effect of sharpening. This means that the distribution of logits is forced to become sharper and thus confidently produce exactly one word that the model believes is the best. Pereyra et al. (2017) discouraged overconfident predictions and given that tempering does not reduce translation quality, we suspect that non-tempered models in low-

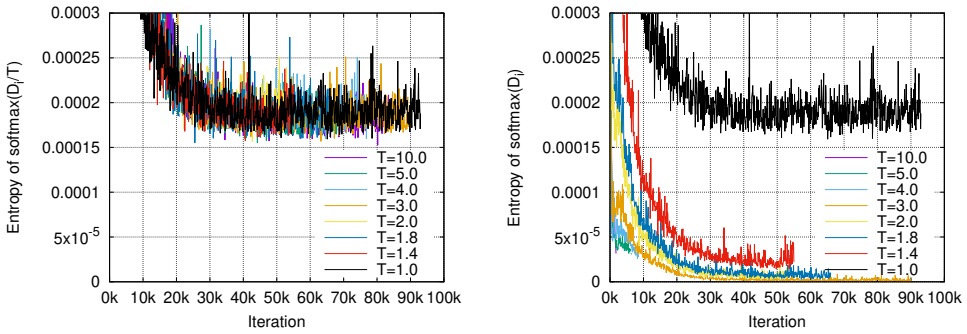

Figure 4: Variation of entropy: the left-hand side shows $softmax(D_i/T)$ in Equation (1) actually used for computing the loss during training, whereas the right-hand side shows $softmax(D_i)$ drawn for this analysis.

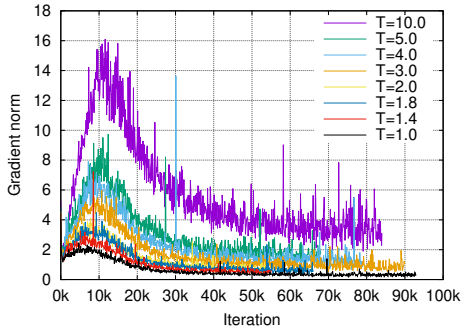

Figure 5: Global gradient norms during training models with softmax tempering.

resource settings are not confident enough. Tempering and softmax entropy maximization have opposite effects on the softmax but they combine together to give better translation quality than when they are used individually as seen in Section 5.2. The same applies for tempering and label smoothing in Section 5.1. Consequently, future efforts should focus on methods that automatically determine the appropriate levels of confidence, especially in low-resource settings.

Figure 5 shows the gradient norms during training with softmax tempering. This revealed that, similarly to ordinary non-tempered training, gradient norms in softmax tempering first increase during the warm-up phase of training and then gradually decrease. However, the major difference is that the norm values significantly decrease for the non-tempered training, whereas they are much higher for training with softmax tempering. Note that we re-scaled the loss for softmax tempering as in Equation (2), which is one reason why the gradient norms are higher. Larger gradient norms indicate that strong learning signals are being back-propagated and this will continue as long as the softmax is forced to make erroneous decisions because of higher temperature values. We can thus conclude that the noise introduced by softmax tempering and subsequent loss re-scaling strongly affect the translation quality of NMT models.

6 Conclusion

In this paper, we explored the utility of softmax tempering for training NMT models. Our experiments in low-resource and high-resource settings revealed that softmax tempering leads to an improvement in the greedy- and beam-search decoding quality. As an indirect consequence,

in latency sensitive scenarios, we can use greedy search while achieving better translation quality than non-tempered models leading to 1.5 to 3.5 times faster decoding. We also explored the compatibility of softmax tempering with label smoothing and softmax entropy maximization where we showed that the combination of tempering and label smoothing is very important. We also identified settings where each method works best. Furthermore, our analysis of the softmax entropies and gradients during training confirms that tempering gives precise softmaxes while enabling the model to learn with strong gradient signals even during late training stages. In the future, we will explore the effectiveness of softmax tempering in other natural language processing tasks.

Acknowledgments

A part of this work was conducted under the commissioned research program "Research and Development of Advanced Multilingual Translation Technology" in the "R&D Project for Information and Communications Technology (JPMI00316)" of the Ministry of Internal Affairs and Communications (MIC), Japan. Atsushi Fujita was partly supported by JSPS KAKENHI Grant Number 19H05660.

References

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, USA. International Conference on Learning Representations.

Dabre, R., Fujita, A., and Chu, C. (2019). Exploiting multilingualism through multistage fine-tuning for low-resource neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1410–1416, Hong Kong, China. Association for Computational Linguistics.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, USA. Association for Computational Linguistics.

Eriguchi, A., Tsuruoka, Y., and Cho, K. (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)*, pages 72–78, Vancouver, Canada. Association for Computational Linguistics.

Firat, O., Cho, K., and Bengio, Y. (2016). Multi-way, multilingual neural machine translation with a shared attention mechanism. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 866–875, San Diego, USA. Association for Computational Linguistics.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1321–1330, Sydney, Australia. JMLR.org.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531.

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

Koehn, P. and Knowles, R. (2017). Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver, Canada. Association for Computational Linguistics.

Kumar, A. and Sarawagi, S. (2019). Calibration of encoder decoder models for neural machine translation. *CoRR*, abs/1903.00802.

Mao, Z., Cromieres, F., Dabre, R., Song, H., and Kurohashi, S. (2020). JASS: Japanese-specific sequence to sequence pre-training for neural machine translation. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3683–3691, Marseille, France. European Language Resources Association.

Miceli Barone, A. V., Haddow, B., Germann, U., and Sennrich, R. (2017). Regularization techniques for fine-tuning in neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1489–1494, Copenhagen, Denmark. Association for Computational Linguistics.

Ng, A. Y. (2004). Feature selection, L1 vs. L2 regularization, and rotational invariance. In *Proceedings of the 21st International Conference on Machine Learning*, Banff, Canada. Association for Computing Machinery.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL)*, pages 311–318, Philadelphia, USA. Association for Computational Linguistics.

Pereyra, G., Tucker, G., Chorowski, J., Kaiser, Ł., and Hinton, G. (2017). Regularizing neural networks by penalizing confident output distributions. *CoRR*, abs/1701.06548.

Post, M. (2018). A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Riza, H., Purwoadi, M., Gunarso, Uliniansyah, T., Ti, A. A., Aljunied, S. M., Mai, L. C., Thang, V. T., Thai, N. P., Chea, V., Sun, R., Sam, S., Seng, S., Soe, K. M., Nwet, K. T., Utiyama, M., and Ding, C. (2016). Introduction of the Asian Language Treebank. In *Proceedings of the 2016 Conference of the Oriental Chapter of International Committee for Coordination and Standardization of Speech Databases and Assessment Technique (O-COCOSDA)*, pages 1–6, Bali, Indonesia.

Sennrich, R., Haddow, B., and Birch, A. (2016). Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), Volume 1: Long Papers*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Sennrich, R. and Zhang, B. (2019). Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221, Florence, Italy. Association for Computational Linguistics.

Song, K., Tan, X., Qin, T., Lu, J., and Liu, T. (2019). MASS: masked sequence to sequence pre-training for language generation. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, pages 5926–5936, Long Beach, USA.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Proceedings of the 27th Neural Information Processing Systems Conference (NIPS)*, pages 3104–3112, Montréal, Canada. Curran Associates, Inc.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, Las Vegas, USA. Institute of Electrical and Electronics Engineers.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Proceedings of the 30th Neural Information Processing Systems Conference (NIPS)*, pages 5998–6008, Long Beach, USA. Curran Associates, Inc.

Zoph, B., Yuret, D., May, J., and Knight, K. (2016). Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1568–1575, Austin, USA. Association for Computational Linguistics.