Improving Low-Resource NMT through Relevance Based Linguistic Features Incorporation

Abhisek Chakrabarty Raj Dabre Chenchen Ding Masao Utiyama Eiichiro Sumita
National Institute of Information and Communications Technology
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-0289, Japan
{abhisek.chakra,raj.dabre,chenchen.ding,
mutiyama,eiichiro.sumita}@nict.go.jp

Abstract

In this study, linguistic knowledge at different levels are incorporated into the neural machine translation (NMT) framework to improve translation quality for language pairs with extremely limited data. Integrating manually designed or automatically extracted features into the NMT framework is known to be beneficial. However, this study emphasizes that the relevance of the features is crucial to the performance. Specifically, we propose two methods, 1) self relevance and 2) word-based relevance, to improve the representation of features for NMT. Experiments are conducted on translation tasks from English to eight Asian languages, with no more than twenty thousand sentences for training. The proposed methods improve translation quality for all tasks by up to 3.09 BLEU points. Discussions with visualization provide the explainability of the proposed methods where we show that the relevance methods provide weights to features thereby enhancing their impact on low-resource machine translation.

1 Introduction

Neural machine translation (NMT) (Bahdanau et al., 2015; Luong et al., 2015; Vaswani et al., 2017; Kitaev et al., 2020) is known to give state-of-the-art translation quality for language pairs having abundance of parallel corpora. In case of resource poor scenarios, additional translation knowledge is acquired either through transfer learning in the form of pre-trained model parameters or by supplying external monolingual corpora. However, exploiting linguistic information effectively in low-resource conditions is still an under-researched field. Annotating the source side with various syntactic features e.g. part-of-speech (POS), lemma, dependency labels etc. can help in accurate translation when either a surface word is polysemous i.e. having multiple senses on varying context or the same word form is shared by different root words due to the inflectional nature of the language. Hence, for morphologically rich languages, ideally, the use of language specific knowledge should improve the translation quality.

To the best of our knowledge, there are not an adequate amount of research works on effectively incorporating arbitrary syntactic information into NMT. One possible reason could be that in high resource scenario the network learns from the large amount of training data to handle the problem caused by polysemy and morphological variants. In this direction, the notable works are done by (Sennrich and Haddow, 2016; Hoang et al., 2016; Li et al., 2018). Sennrich and Haddow (2016) incorporated several features at the source side by employing a separate embedding matrix for each component of a source token including the word and its associated features. Finally, all embeddings are concatenated to enrich the representation. Inspired by this work, Hoang et al. (2016) developed a method to process feature sequences of the source sentence by separate recurrent neural networks (RNNs) and combined the output of all RNNs using a hybrid global-local attention strategy. Li et al. (2018) proposed a complex RNN architecture to model source-side linguistic knowledge. Their approach, at the first level, passes the morphological properties of each word sequentially through an RNN to build a composite representation of the features. These representations are further encoded by an RNN at the sentence level. Very recently, Pan et al. (2020) came up with a dual-source Transformer model to process words and features in

Proceedings of the 28th International Conference on Computational Linguistics, pages 4263–4274
Barcelona, Spain (Online), December 8-13, 2020

isolation. The output of the word Transformer and the feature Transformer are fed to the decoder by two encoder-decoder attention sub-layers put in a series. All the models discussed so far ignore to consider whether a particular feature of a given word is really relevant for the translation task.

The above point motivates us to conduct the present research. We hypothesize that only including the features alongside the word by using a generalized embedding layer or forming a composite representation by taking all features together does not exploit the features completely. There should be some mechanism which can justify the relationship between a word and its supporting features as well. Driven by this idea, we come up with two simple strategies which measure the relevance of the word and the feature embeddings obtained from the output of the embedding layers. The first one is self relevance which considers the relevance of a feature with respect to itself. We apply an attention function to the feature embedding, which in turn generates a mask determining the importance of that feature. Finally, the mask is applied on the feature to effectuate the attention. Our second approach considers the feature relevance with respect to the corresponding word which is the most vital component of a source token. In this case, the attention function operates on a word-feature pair and returns the mask determining the word-based relevance of the input feature. For experimentation, we choose the Transformer network of Vaswani et al. (2017) and assess our proposed techniques on eight low-resource language pairs having diverse morphological variations taken from the Asian Language Treebank (Riza et al., 2016). Our hypothesis is empirically validated showing the fruitfulness of the relevance checking mechanisms in low-resource scenario. We achieve up to 3.09 BLEU points gain over the standard baseline models of Sennrich and Haddow (2016) and Vaswani et al. (2017). In the next section, the related works are briefly described.

2 Related Works

Incorporating morphological information for NMT is a challenging area of research. A significant number of works involve dependency structure at the source side (Eriguchi et al., 2016; Shi et al., 2016; Bastings et al., 2017; Chen et al., 2017; Hashimoto and Tsuruoka, 2017; Li et al., 2017; Wu et al., 2018; Zhang et al., 2019). Eriguchi et al. (2016) proposed a syntax-aware encoding mechanism that encodes the source sentence maintaining the hierarchy of its dependency tree. A Long-Short-Term Memory (LSTM) network (Hochreiter and Schmidhuber, 1997) is used to encode the constituent phrases recursively in bottom-up direction. On the contrary, Shi et al. (2016) claimed that an RNN encoder can capture the inherent syntactic properties automatically from a source sentence as a by-product of training. They used a multi-layer LSTM and found that its different layers represent different types of syntax. This work gives the intuition that using linguistic prior at the source side may be redundant for translation. Following Eriguchi et al. (2016), Chen et al. (2017) presented a bidirectional tree encoder which builds the sentence representation considering both top-down and bottom-up directions of the dependency tree. A different approach was taken by Bastings et al. (2017) to propose a syntactic graph-convolutional network for encoding source sentences. Hashimoto and Tsuruoka (2017) came up with a model that learns the latent graph structure of a source sentence optimized by the translation objective. In the study by Li et al. (2017), the parse tree of the source sentence are linearized to label sequence. Next, three different encoding strategies using RNN namely parallel, hierarchical and mixed, are tried to integrate the dependency information. Wu et al. (2018) used dependency information of both the source and the target languages. As their model needs multiple encoders and decoders, so it is not worthy for use under low-resource condition. In the work of Zhang et al. (2019), the authors employed a supervised encoder-decoder dependency parser and used the outputs from the encoder as a syntax-aware representations of words, which in turn, are concatenated to the input embeddings of the translation model. Most recently, Bugliarello and Okazaki (2020) proposed dependency-aware self-attention in the Transformer that needs no extra parameter. For a pivot word, its self-attention scores with other words are weighted considering their distances from the dependency parent of the pivot word.

Apart from the above works, there are some studies which use the factors in the target side (Burlot et al., 2017; García-Martínez et al., 2016a; García-Martínez et al., 2016b). In general, their approach is to predict the roots and other morphological tags of the target words instead of producing the surface forms. Additionally, a morphological analyzer is applied for the reinflection task.

3 NMT Architecture

We employ the Transformer architecture for execution of our experiments. It is a specific type of encoder-decoder neural model that can be applied for sequence modelling tasks. Unlike RNN, the working policy of the Transformer does not rely on recurrence and hence, is much more parallelizable. As in general encoding-decoding framework, the learned embeddings of the source sentence $s = (s_1, ..., s_m)$ are mapped by the encoder to continuous representations $z = (z_1, ..., z_m)$. Each representation z_i contains the contextual information about its surrounding tokens. To keep track of the order of the sequence, positional encodings are added to the input embeddings.

The encoder comprises a stack of identical layers each having two sub-layers - multi-head self-attention and position-wise fully connected feed-forward network. For each position in the source sentence, the self-attention block calculates a probability distribution over all positions. This distribution is then used to make a new representation of the reference position. Multi-head self-attention repeats the process a number of times resulting multiple representations in different subspaces. Finally, all of them are concatenated and the output is passed to the fully connected feed-forward network which contains two linear layers with ReLU activation function in between. As no recurrence is involved, so the encoding process can be parallelized during both training and inference phases.

The decoder is also made of a stack of identical layers having multi-head self-attention and feed-forward networks with the addition of an extra computation called multi-head encoder-decoder attention over the output of the encoder stack. Here, the self-attention sub-layer is masked so that at a particular position it would not be able to consider the subsequent positions. This must be enforced for preserving the auto-regressive property. All sub-layers in the encoder and the decoder stacks have residual connections followed by layer normalization.

3.1 Generalized Source Embedding for Input Features

Sennrich and Haddow (2016) incorporated arbitrary morphological features through a generalized source embedding. Formally, let each token in the source language sentences be annotated with K number of features . For the k^{th} feature, V_k and E_k denote the vocabulary and the feature embedding matrix respectively. $E_k \in \mathbb{R}^{d_k \times |V_k|}$ where d_k is the dimension of the feature embedding. Finally, the embeddings of all features are concatenated to form the generalized embedding of a source token. For the source token s_i, its embedding e_i is formulated as follows.

$$e_{ik} = E_k s_{ik}$$
$$e_i = \|_{k=1}^{K} e_{ik}$$

Where s_{ik} denotes the k^{th} feature of s_i and e_{ik} denotes the vector embedding of the feature s_{ik}. $\|$ is the vector concatenation operation. Hence, the dimension of the resultant vector e_i is $\sum_{k=1}^{K} d_k$. Finally, e_i is given as input to the encoder.

3.2 The Proposed Relevance Checking Methods

Our hypothesis is that only including the features through separate embedding matrices and then, combining all together by concatenation does not exploit the features completely. It would be beneficial to weight the feature vectors according to their relevance in translation. Now, the challenge is how to estimate the relevance of each feature component in order to improve the translation quality. To address this issue, we propose two empirical methods - self relevance and word-based relevance, which are described below.

3.2.1 Self Relevance

Let among K feature components of the source token $s_i = (s_{i1}, \ldots, s_{iK})$, the first component s_{i1} denotes the corresponding word[1] and the rest of them from s_{i2} to s_{iK} denote various morphological properties of s_{i1}. The corresponding vector embeddings are $e_{i1}, \ldots, e_{iK}$. The self relevance of each

[1] In case of subword-NMT, the corresponding subword is the main component of a source token.

Figure 1: Self relevance of a feature.

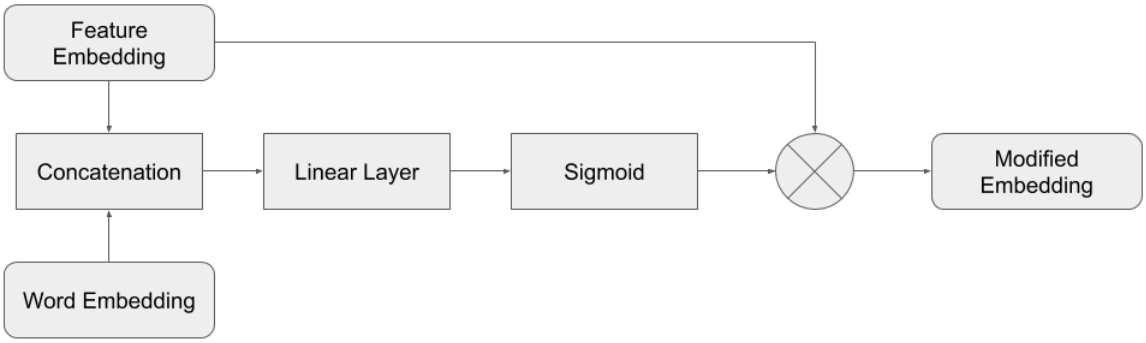

Figure 2: Word-based relevance of a feature.

component (including the word and its all morphological features) of the source token s_i is evaluated as follows. For $k \in \{1, \ldots, K\}$,

$$mask_{ik} = sigmoid(W_k e_{ik})$$
$$e'_{ik} = mask_{ik} \odot e_{ik}$$

Where $W_k \in \mathbb{R}^{d_k \times d_k}$ is the weight matrix for the k^{th} feature, and $\odot$ is the element-wise multiplication operation between two vectors. The feature embedding e_{ik} is given input to a linear layer with output dimension same as that of e_{ik} followed by sigmoid activation function. The output is a mask vector with values in the range between 0 to 1, which signifies the self relevance of e_{ik}. Next, the element-wise multiplication operation between the input embedding and the output mask produces the modified feature embedding e'_{ik}. Finally, all modified feature embeddings $e'_{i1}, \ldots, e'_{iK}$ are concatenated to make the resultant embedding e'_i which is given as input to the Transformer encoder. In this way the features can determine their own impact on the final word representation. The process is depicted in Figure 1.

$$e'_i = \|_{k=1}^{K} e'_{ik}$$

3.2.2 Word-based Relevance

This strategy evaluates the relevance of a morphological feature with respect to its corresponding word. For $k \in \{2, 3, \ldots, K\}$, the word-based relevance of the embedding e_{ik} is measured with respect to e_{i1}. Formally,

$$mask_{ik} = sigmoid(W_k(e_{i1} \| e_{ik}))$$
$$e'_{ik} = mask_{ik} \odot e_{ik}$$

Where $W_k \in \mathbb{R}^{d_k \times (d_1 + d_k)}$ is the weight matrix. While the self relevance measures the importance of a feature with respect to itself, in contrast the word-based relevance gives priority to the word component. The final embedding e'_i of the source token s_i is obtained by concatenating $e'_{i2}, \ldots, e'_{iK}$ with e_{i1}. We present the word-based relevance checking mechanism in Figure 2.

$$e'_i = e_{i1} \| e'_{i2} \| \ldots \| e'_{iK}$$

Original sentence:		there is no show without winners !						
Subwords:	there	is	no	show	without	win@@	ners	!
Lemmas:	there	be	no	show	without	winner	winner	!
Subword tags:	S	S	S	S	S	B	E	S
POS:	EX	VBZ	DT	NN	IN	NNS	NNS	.
Dep:	expl	root	det	nsubj	prep	pobj	pobj	punct

Table 1: Annotation of a sample source sentence.

4 Experiments

Datasets and Preprocessing: As mentioned in section 1, we carry out our experimentation on eight language pairs under low-resource scenario. For all of them, the source side is fixed to English (en) and the target sides are Bengali (bg), Filipino (fi), Hindi (hi), Indonesian (id), Khmer (khm), Malay (ms), Myanmar (my), and Vietnamese (vi). The datasets are taken from the multi-lingual, multi-parallel Asian Language Treebank (Riza et al., 2016)[2]. We use the official train/dev/test split of the datasets having $18,088/1,000/1,018$ parallel sentences respectively. The following preprocessing operations are done on the data. Out of the eight reference Asian languages, the Khmer data is unsegmented. So we use the nova annotation system for Khmer segmentation (Ding et al., 2018) and tokenization. The remaining languages are tokenized to separate the delimiters and the punctuation symbols.

Linguistic Features Used: We use three linguistic features of the source language in our experiments. They are - (i) lemma, (ii) POS tag and (iii) dependency label. For morphologically rich languages where roots have multiple variants, there tagging the raw text with these three features helps to disambiguate homonymy and polysemy. In particular, if experiments are done at subword-level, then annotating each subword with the word-level features is expected to feed into the model's performance. As the source side is fixed to English, we annotate the English data using Stanford CoreNLP (Manning et al., 2014) toolkit. The vocabulary sizes of the three features are $26,414;43$ and 45 respectively.

Subword Tags: All experiments are done at subword-level to reduce the out-of-vocabulary cases during inference. We segment the datasets into subword units using byte-pair encoding (BPE) (Sennrich et al., 2016) technique keeping the number of merge operations to be $10,000$. Note that in BPE segmentation there is no explicit word boundary and a symbol may form either of the beginning/inside/end/whole of a word. Hence, following Sennrich and Haddow (2016), we add an extra feature to each subword in the source side in addition to the three linguistic features stated above. Every subword is annotated with one of the four markers - B (beginning), I (inside), E (end), S (single). The annotation is done with the help of the script provided in the corresponding url[3]. Table 1 depicts the structure of a sample source sentence.

Baselines: We compare our proposed self relevance and word-based relevance methods with the following baselines.

- Transformer-base: It is the base configuration as proposed in (Vaswani et al., 2017). The experiments are done at subword-level without using external linguistic knowledge.
- Concat: This is the technique proposed by Sennrich and Haddow (2016). The embeddings of a subword and its supporting features are concatenated. Sennrich and Haddow (2016) reported the results on the RNN model. Whereas, we apply the same strategy on the Transformer.
- Add: Subword and feature embeddings are added to form the resultant embedding i.e. $e_i = \sum_{k=1}^{K} e_{ik}$.
- Linear: The embeddings are passed through a linear transformation followed by a ReLU activation. Here, e_i =ReLU$(W(\|_{k=1}^{K} e_{ik}))$ where W is the weight matrix.

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

[3] https://github.com/rsennrich/wmt16-scripts/blob/master/preprocess/conll_to_factors.py

	Subword	Lemma	Subword tag	POS	Dep
Base	512	-	-	-	-
Concat	250	250	6	15	15
Add	512	512	512	512	512
Linear	250	250	6	15	15
Self-rel	250	250	6	15	15
Word-rel	250	250	6	15	15

Table 2: Embedding dimensions of the components.

		en-bg	en-fi	en-hi	en-id	en-khm	en-ms	en-my	en-vi
Baseline models	Base	4.97	25.59	18.54	27.93	22.88	32.40	13.93	24.99
	Concat	5.56	23.75	20.69	27.99	23.53	32.92	14.92	26.50
	Add	4.66	22.02	15.45	24.78	21.65	30.45	11.86	22.78
	Linear	4.89	24.26	20.65	27.17	23.42	32.64	13.79	25.36
Proposed models	Self-rel	6.10	**26.26**	21.27	**30.41**	24.76	**34.71**	**16.53**	**27.74**
	Word-rel	**6.25**	26.01	**21.63**	26.53	**25.13**	33.20	15.62	27.66

Table 3: BLEU scores of the models for all reference language pairs.

Hyperparameters: We use the OpenNMT PyTorch implementation (Klein et al., 2017) to build our models and mostly follow the Transformer-base hyperparameter setting mentioned there[4]. There are 6 layers in each of the encoder and the decoder stacks. The number of multi-heads used is 8. The dimension of the fully-connected-feed-forward network is $2,048$. Total number of training steps is set to $200,000$ and after each $10,000$ steps validation checking is performed. We use the early-stopping strategy in training. If the validation accuracy does not improve for 5 consecutive validation checking steps, then training stops. Following (Sennrich and Haddow, 2016), we keep the dimension of the final embedding which is fed to the Transformer, comparable across the models without and with using features so that the number of model parameters does not influence the performance. In Table 2 we list the embedding dimensions of the subword and its features for all experimental settings. Inference is done keeping beam size equal to 5. We carry out our experiments using single GPU with the specification of 32 GB Tesla V100-SXM2.

4.1 Results

BLEU Scores: We present the BLEU scores[5] (Papineni et al., 2002) of our proposed methods and the baselines in Table 3. The scores are computed after undoing the BPE segmentation of the translations. For en-fi, en-id, en-ms, en-my and en-vi, the self relevance checking strategy yields the best results (26.26, 30.41, 34.71, 16.53 and 27.74 respectively). For en-bg, en-hi and en-khm, the word-based relevance method outperforms others (6.25, 21.63 and 25.13 respectively). Compared to the base configuration, maximum improvement is obtained for en-hi ($18.54 \rightarrow 21.63$) and minimum for en-fi ($25.59 \rightarrow 26.26$). Compared to (Sennrich and Haddow, 2016) i.e. the concat combination, we get maximum BLEU points gain for en-fi ($23.75 \rightarrow 26.26$) and minimum for en-bg ($5.56 \rightarrow 6.25$). We also check the significance test[6] and found that these improvements are statistically significant with p-value < 0.05. Overall, the two methods proposed in this work come out to be the top two performers for seven out of the eight language pairs except en-id. These results unquestionably prove the effectiveness of the self relevance and the word-based relevance checking strategies.

Comparison of Model Parameters: Table 4 provides the number of model parameters in each configuration. The base model and the addition combination require the lowest and the highest number of parameters respectively. The remaining configurations (linear, concat, self relevance and word-based relevance) use comparable number of parameters. Larger models in low-resource settings like ours often overfit leading to low translation quality. Thus, trivially increasing the number of parameters should not

[4] `https://opennmt.net/OpenNMT-py/FAQ.html`

[5] We calculate BLEU scores using the script *multi-bleu.perl* obtained from *Moses*.

[6] Significance test is done using the script *bootstrap-hypothesis-difference-significance.pl* obtained from *Moses*.

	en-bg	en-fi	en-hi	en-id	en-khm	en-ms	en-my	en-vi
Base	59.45	59.42	59.34	59.25	58.98	59.20	59.12	59.02
Concat	66.94	66.91	66.83	66.73	66.40	66.67	66.59	66.49
Add	73.02	72.99	72.92	72.82	72.51	72.77	72.69	72.59
Linear	67.23	67.20	67.11	67.02	66.69	66.95	66.88	66.78
Self-rel	67.07	67.04	67.04	66.86	66.53	66.79	66.72	66.62
Word-rel	67.08	67.05	67.05	66.87	66.54	66.80	66.73	66.63

Table 4: Number of model parameters (in million) for all configurations.

	en-bg	en-fi	en-hi	en-id	en-khm	en-ms	en-my	en-vi
Base	410.33	34.62	74.44	30.13	46.21	23.18	132.40	36.23
Concat	287.32	36.44	64.56	31.10	45.62	21.20	103.8	36.71
Add	367.07	45.63	92.06	37.90	55.91	25.27	157.25	41.97
Linear	349.91	43.06	79.22	36.01	52.36	23.24	131.80	39.97
Self-rel	274.51	**34.60**	**60.09**	29.80	**43.95**	**19.15**	97.91	35.12
Word-rel	**268.32**	35.92	60.91	**28.52**	44.18	20.13	**97.63**	**33.84**

Table 5: Best validation set perplexity for all reference language pairs.

be the reason behind any improvements in translation quality. Despite the increase in the number of parameters, our methods show strong improvements in BLEU scores. Therefore, we can say that the performance gain is due to our proposed methods and not because of the excess parameters.

Best Validation Set Perplexity: We report the best validation set perplexity of the models in addition to BLEU scores. Table 5 shows the results. Perplexity of a translation model indicates that given a source sentence, how good the model is at predicting the reference translation. So, we assess all models in terms of perplexity to test whether our proposed methods are really effective or not. As the lower perplexity suggests the better model, hence Table 5 shows that the self relevance gives the best scores for en-fi, en-hi, en-khm and en-ms. For the rest of the language pairs, the word-based relevance outperforms others. Overall, these two methods are the top two performers for all language-pairs except en-fi.

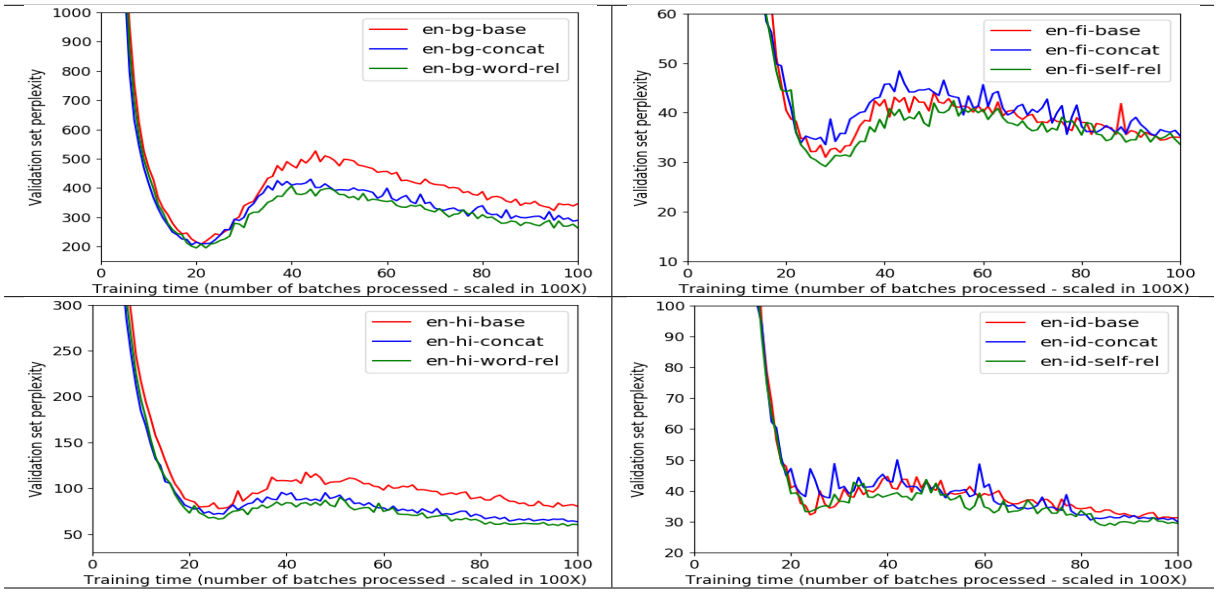

Table 6: Training time vs. validation set perplexity plot.

To ensure that the best validation set perplexity obtained is not just due to randomness, we plot the training time (number of batches processed) vs. perplexity on the validation set for initial $10,000$ batches

of training with the interval of 100 batches. The experiments are done taking the best models in terms of BLEU score obtained for every individual language pair (self relevance model for en-fi, en-id, en-ms, en-my, en-vi and word-based relevance model en-bg, en-hi, en-khm as shown in Table 3) and we compare their performances with the base and the concat configurations. The plots are given in Tables 6 and 7. During initial $2,000 - 2,500$ batches all models exhibit nearly the equal perplexity for each language pair. After that the differences are prominent showing that the relevance-based models yield the lowest perplexity. This is clearly seen in the plots of en-bg, en-hi, en-ms, en-my and en-vi. For en-fi, en-id and en-khm, though the baseline systems are sometimes better but overall, our proposed methods are superior.

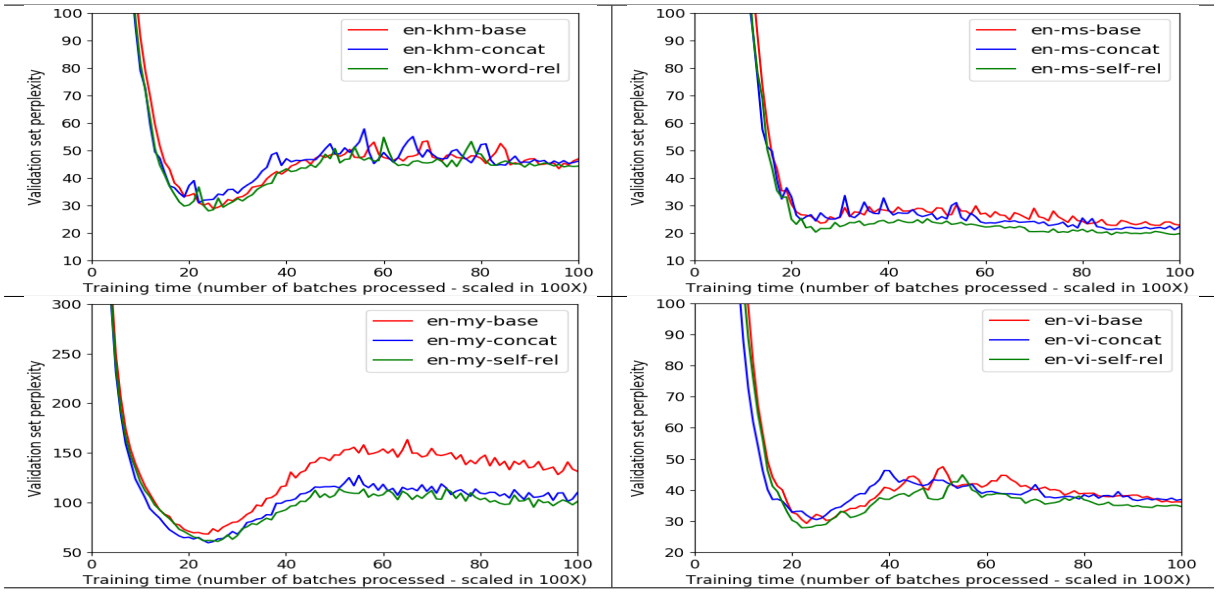

Table 7: Training time vs. validation set perplexity plot.

Impact of Individual Linguistic Features: Further we conduct the experiments taking linguistic features (lemma, POS and dependency labels) in isolation i.e. only one feature is used at a time. The goal is to check their potential exclusively. For each feature we execute our proposed methods as well as the base and the concat baselines. The results are reported in Table 8. In comparison with Table 3, combination of the three features proves to yield the best results. Table 8 shows that use of any of the features in isolation cannot beat the base configuration for en-fi (BLEU 25.59). For the remaining language pairs the best results are produced by either of the two proposed methods. Out of the three features, lemma and dependency labels prove to be most effective. Because, using the dependency labels gives the maximum

		en-bg	en-fi	en-hi	en-id	en-khm	en-ms	en-my	en-vi
	Base	4.97	**25.59**	18.54	27.93	22.28	32.40	13.93	24.99
Lemma	Concat	5.32	24.49	18.34	26.72	23.08	31.77	13.51	26.49
	Self-rel	5.31	25.44	19.81	**28.89**	23.15	32.20	14.81	24.88
	Word-rel	5.26	25.32	19.64	27.69	23.14	**32.62**	14.71	**26.83**
POS	Concat	5.11	23.73	18.80	25.23	21.99	29.83	13.69	22.63
	Self-rel	5.66	24.24	19.61	28.80	22.84	31.87	14.94	24.86
	Word-rel	5.74	24.61	19.73	26.81	23.26	29.81	13.94	23.06
Dep	Concat	5.24	24.51	20.14	25.50	22.94	31.17	13.95	23.26
	Self-rel	**5.78**	23.78	**20.40**	26.27	23.40	31.19	**14.95**	24.13
	Word-rel	5.30	22.20	19.78	26.37	**23.48**	32.42	14.36	24.17

Table 8: BLEU scores of the models using features separately.

BLEU scores for en-bg, en-hi, en-khm and en-my. Whereas, the use of lemma gives the best results for en-id, en-ms and en-vi.

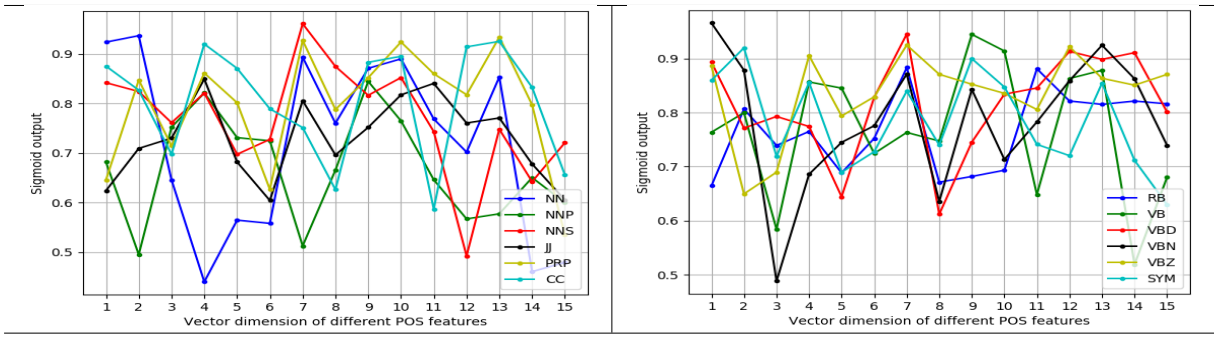

Table 9: Visualization of the sigmoid activation of feature embeddings in self relevance method.

Visualization of Sigmoid Outputs: To investigate the effectiveness of checking feature relevance, we plot the sigmoid outputs of some features from the test data and analyze them. In case of the self relevance, the features get fixed sigmoid outputs. For example, we plot the values for 12 primary POS categories - NN, NNP, NNS, JJ, PRP, CC, RB, VB, VBD, VBN, VBZ, SYM (noun, proper noun, plural noun, adjective, pronoun, conjunction, adverb, base verb, verb past form, verb past participle, verb third person singular and symbol). The plots are shown in Table 9. The x-axis denotes the dimensions of the vector (from 1^{st} to 15^{th} as the POS embedding size is 15 given in table 2) and the y-axis denotes the sigmoid output. In contrast to the model of Sennrich and Haddow (2016), where feature vectors are just appended to the word vector, the proposed relevance methods customize the features according to their importance in the translation task, which is empirically proved to be beneficial for translation.

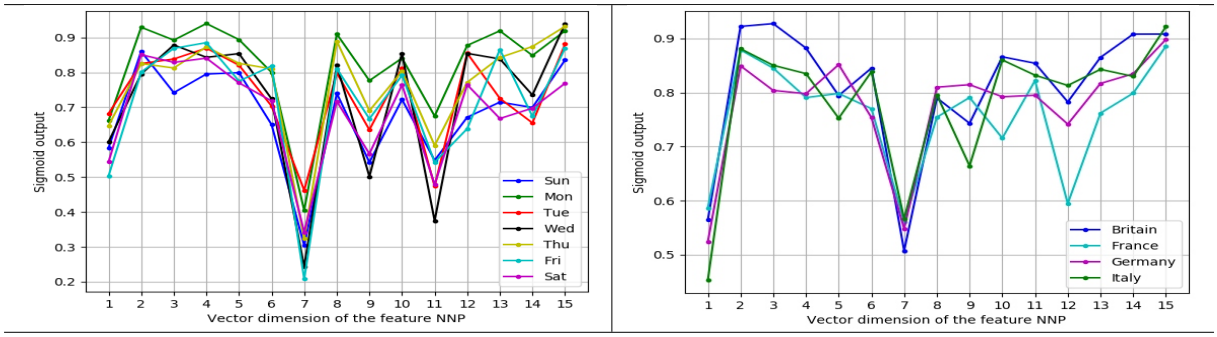

Table 10: Visualization of the sigmoid activation of feature embeddings in word-based relevance method.

In case of the word-based relevance method, the sigmoid output of a particular feature follows specific pattern depending on the semantics of the associated word. For example, we take the part-of-speech NNP (proper noun) and show its sigmoid activation for two different categories of named entities. The first category is the days in a week i.e. from Sunday to Saturday. The other category is the names of different countries. These proper nouns are chosen as they have not been divided into subwords by BPE in our experiments. The plots are shown in Table 10. In the left graph, the similarities among the plots are clearly seen as all curves have the global minima at the 7^{th} dimension and the local minimas at the $1^{st}, 9^{th}$ and 11^{th} dimensions. The maximas are found at the dimensions 4^{th} and 15^{th}. In the right graph, there are local minimas at the dimensions $1^{st}, 7^{th}$ and 12^{th} and the maximas are at the $2^{nd}, 3^{rd}$ and 15^{th} dimensions. It gives the justification of why the word-based relevance checking is effective as the features are weighted similarly for semantically close words leading to more compact embedding representation of a source token.

Results Using RNN Models: Further we explore the effect of feature relevance on attention-based RNN model. To do the experiments, we choose a LSTM network with the encoder composed of a stack of

	en-bg	en-fi	en-hi	en-id	en-khm	en-ms	en-my	en-vi
Base	5.52	24.10	18.37	25.73	22.60	30.29	13.42	24.64
Concat	6.85	25.43	20.72	27.43	22.94	32.25	15.69	26.12
Self-rel	5.69	24.57	19.61	26.62	23.18	32.18	5.16	26.08
Word-rel	6.26	25.23	20.83	27.01	23.96	32.62	13.94	25.38

Table 11: BLEU scores obtained for different feature configurations on RNN.

2 bidirectional layers and a single layer unidirectional decoder. We apply Luong-attention (Luong et al., 2015) in our model. Four different configurations have been tested. They are - (i) no feature used (ii) concat (iii) self relevance (iv) word-based relevance. We use the same feature set as used for the Transformer. The BLEU scores obtained are presented in Table 11. Contrast to the results in Table 3, the relevance checking mechanisms do not perform significantly well when applied for RNN models. Out of the 8 language pairs, the word-based relevance method produces the best results for en-hi, en-khm and en-ms. For the remaining language pairs, the concat configuration outperforms others. Compared to the Transformer model, we get the highest BLEU score in en-bg (6.25 in Table 3 vs 6.85 in Table 11) using RNN. For the rest of the experiments, the Transformer produces the best results. These results disclose an important finding that although our objective is to enrich the source word representation by masking feature embeddings with attention, the proposed relevance methods are not model-agnostic. Rather, their efficacy is influenced by the network architecture where they favor the Transformer architecture.

5 Conclusion

In this article we revisit the ways to incorporate linguistic features in NMT. We argue that it is important to check the relevance of the features instead of just plugging them into the model. To establish our claim two novel methods are proposed and evaluated under extremely low-resource condition on eight language pairs. We design word-dependent as well as word-agnostic relevance checking mechanisms and show that by controlling the effects of features we can obtain substantial improvement in translation quality. Our models yield significantly higher BLEU scores compared to the baselines with a modest increase in the number of model parameters. Additionally we observe lower validation perplexity that shows applying feature relevance helps to reduce prediction uncertainty. The methods are further analyzed by visualization of the relevance weights. In case of the word-based relevance, the feature embeddings are tuned similarly based on the semantics of the corresponding word. It indicates that the proposed models actually pay attention to morphological features leading to enriched word representation. Moreover, we assess the proposed relevance methods on RNN model with attention. The results are not as satisfactory as obtained for the Transformer model, which requires further investigation. One notable issue in the present work is that the source language has been annotated by a robust and highly accurate parser which is unlikely to exist for a low-resource language. Checking the effectiveness of relevance methods is necessary when the annotation is noisy. The future extension of the present work will also focus to exploit the features under high resource scenario. In a resource rich setting usually features are redundant as the model learns from a large variety of context. Hence, using them effectively in that case will be beneficial for NMT research.

References

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations*, January. 3rd International Conference on Learning Representations, ICLR 2015 ; Conference date: 07-05-2015 Through 09-05-2015.

Jasmijn Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical*

Methods in Natural Language Processing, pages 1957–1967, Copenhagen, Denmark, September. Association for Computational Linguistics.

Emanuele Bugliarello and Naoaki Okazaki. 2020. Enhancing machine translation with dependency-aware self-attention. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1618–1627, Online, July. Association for Computational Linguistics.

Franck Burlot, Mercedes García-Martínez, Loïc Barrault, Fethi Bougares, and François Yvon. 2017. Word representations in factored neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 20–31, Copenhagen, Denmark, September. Association for Computational Linguistics.

Huadong Chen, Shujian Huang, David Chiang, and Jiajun Chen. 2017. Improved neural machine translation with a syntax-aware encoder and decoder. *CoRR*, abs/1707.05436.

Chenchen Ding, Masao Utiyama, and Eiichiro Sumita. 2018. Nova: A feasible and flexible annotation system for joint tokenization and part-of-speech tagging. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 18(2), December.

Akiko Eriguchi, Kazuma Hashimoto, and Yoshimasa Tsuruoka. 2016. Tree-to-sequence attentional neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 823–833, Berlin, Germany, August. Association for Computational Linguistics.

Mercedes García-Martínez, Loïc Barrault, and Fethi Bougares. 2016a. Factored neural machine translation. *ArXiv*, abs/1609.04621.

Mercedes García-Martínez, Loïc Barrault, and Fethi Bougares. 2016b. Factored neural machine translation architectures.

Kazuma Hashimoto and Yoshimasa Tsuruoka. 2017. Neural machine translation with source-side latent graph parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 125–135, Copenhagen, Denmark, September. Association for Computational Linguistics.

Cong Duy Vu Hoang, Reza Haffari, and Trevor Cohn. 2016. Improving neural translation models with linguistic factors. In *Proceedings of the Australasian Language Technology Association Workshop 2016*, pages 7–14, Melbourne, Australia, December.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *International Conference on Learning Representations*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proc. ACL*.

Junhui Li, Deyi Xiong, Zhaopeng Tu, Muhua Zhu, Min Zhang, and Guodong Zhou. 2017. Modeling source syntax for neural machine translation. *CoRR*, abs/1705.01020.

Qiang Li, Derek F. Wong, Lidia S. Chao, Muhua Zhu, Tong Xiao, Jingbo Zhu, and Min Zhang. 2018. Linguistic knowledge-aware neural machine translation. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 26(12):2341–2354, December.

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

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Yirong Pan, Xiao Li, Yating Yang, and Rui Dong. 2020. Dual-source transformer model for neural machine translation with linguistic knowledge. 02.

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

Hammam Riza, Michael Purwoadi, Gunarso, Teduh Uliniansyah, Aw Ai Ti, Sharifah Mahani Aljunied, Luong Chi Mai, Vu Tat Thang, Nguyen Phuong Thai, Rapid Sun, Vichet Chea, Khin Mar Soe, Khin Thandar Nwet, Masao Utiyama, and Chenchen Ding. 2016. https://ieeexplore.ieee.org/document/7918974/ Introduction of the Asian language treebank. In *Proc. of O-COCOSDA*, pages 1–6.

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, pages 83–91, Berlin, Germany, August. Association for Computational Linguistics.

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

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural MT learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1526–1534, Austin, Texas, November. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

S. Wu, D. Zhang, Z. Zhang, N. Yang, M. Li, and M. Zhou. 2018. Dependency-to-dependency neural machine translation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(11):2132–2141.

Meishan Zhang, Zhenghua Li, Guohong Fu, and Min Zhang. 2019. Syntax-enhanced neural machine translation with syntax-aware word representations. 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 1151–1161, Minneapolis, Minnesota, June. Association for Computational Linguistics.