Syntax-Aware Data Augmentation for Neural Machine Translation

Sufeng Duan, Hai Zhao, and Dongdong Zhang

Abstract—Data augmentation is an effective method for the performance enhancement of neural machine translation (NMT) by generating additional bilingual data. In this article, we propose a novel data augmentation strategy for neural machine translation. Unlike existing data augmentation methods that simply modify words with the same probability across different sentences, we introduce a sentence-specific probability approach for word selection based on the syntactic roles of words in the sentence. Our motivation is to consider a linguistics-motivated method to obtain more ingenious language generation rather than relying on computation-motivated approaches only. We argue that high-quality aligned bilingual data is crucial for NMT, and only computation-motivated data augmentation is insufficient to provide enough extra enhancement data. Our approach leverages dependency parse trees of input sentences to determine the selection probability of each word in the sentence using three different functions to calculate probabilities for words with different depths. Besides, our method also revises the probability for words considering the sentence length. We evaluate our methods on multiple translation tasks. The experimental results demonstrate that our proposed data augmentation method does effectively boost existing sentence-independent methods for significant improvement of performance on translation tasks. Furthermore, an ablation study shows that our method does select fewer essential words and preserves the syntactic structure.

Index Terms—Natural language processing, neural machine translation, data augmentation, dependency parsing.

I. INTRODUCTION

Data augmentation (DA) is a helpful strategy in deep learning to enhance accuracy and has been widely used in computer vision (CV) [1], natural language processing (NLP) [2], [3], [4], [5], [6], [7], [8], [9] and other areas. In CV, data augmentation generates additional data by producing variations of existing data through transformations such as mirroring and random. In NLP tasks like neural machine translation (NMT), data augmentation can improve performance by generating additional training samples [6], [7], [8] or enhance model robustness by adding explicit noise [2], [4].

To perform unsupervised or semi-supervised NMT training using only monolingual data, back-translation [3], [5], an important sentence-level data augmentation method, has been widely used to generate bilingual data. However, back-translation is not a simple method which requires a full NMT system to translate the target sentence into the source. Collecting and cleaning a huge amount of monolingual data also requires substantial efforts. Moreover, some sentences generated by back-translation are not native and are harmful for training, especially for low-resource languages [10].

In contrast to back-translation, existing word-level data augmentation methods offer a simpler and more efficient approach. These methods, such as randomly swapping words [6], dropping words [2], and replacing one word with another [4], primarily focus on changing words within a sentence instead of generating an entirely new sentence, though these approaches can still create diverse variants of the original sentence.

Existing data augmentation methods for NMT have been shown generally effective. However, they also suffer from specific limitations. Current methods are computation-motivated only and randomly select words with the same probability which considers no linguistic clues delivered into the words. This approach can easily result in the generation of flawed sentences by replacing or dropping crucial words from the original sentence. For example, if an essential verb in one sentence is selected and replaced by other words, the meaning of the entire original sentence will be fundamentally modified. Furthermore, replacing or dropping words improperly can disrupt the syntactic structure of the original sentence. With such data augmentation, the original alignment between the source and target sentence may be improperly influenced, consequently limiting performance improvement.

Note that NMT is a challenging language generation task by all means, and high-quality aligned bilingual data is indispensable for NMT. However, most existing data augmentation methods are computation-motivated and insufficient to provide good enough extra enhancement data due to their obvious drawbacks. To alleviate these limitations, this article introduces a linguistics motivated data augmentation strategy for NMT. Different from the existing methods that generate monolingual data by randomly replacing words, we heuristically select words...
for revision based on their roles within one sentence. In detail, we adopt the syntactic dependency parse tree of a sentence as heuristic clues. In one dependency parse tree, words have different positions and depths, which can be considered as essential to the sentence. Generally, the closer a word is to the root, the more important the word is. In practice, we reduce the probability of selecting a word closer to the root and prefer words farther from the root which is supposed to be less important. We propose three different functions to calculate probabilities for words at different depths. Our syntax-aware enhancement method can conveniently combine with standard word-level data augmentation operations such as dropping and replacing.

We evaluate our method on WMT14 English-to-German, IWSLT14 German-to-English and English-French, IWSLT15 English-Vietnamese datasets using the Transformer. The experimental results demonstrate that our proposed strategy can effectively enhance baseline data augmentation methods for significant performance improvement, especially on low-resource datasets.

II. RELATED WORKS

A. Neural Machine Translation

NMT models are based on a sequence-to-sequence (seq2seq) architecture, which uses an encoder to create a vector of the source sentence and a decoder to generate a sequence of target words, along with an attention [11], [12], [13]. Several seq2seq NMT models, such as recurrence neural network based model [12], [13], convolutional neural network model [14], and the Transformer [15], have achieved improvement of performance on MT tasks.

The Transformer is a fully attention-based NMT model empowered by self-attention networks [16]. The encoder of the Transformer consists of one self-attention layer and a position-wise feed-forward layer. The decoder of the Transformer contains one self-attention layer, one encoder-decoder attention layer and one position-wise feed-forward layer. The Transformer uses residual connections around the sublayers, followed by a normalization layer. Vaswani et al. proposed scaled dot-product attention, which is the critical component in the Transformer. Besides, Vaswani et al. also proposed multi-head attention which is used in the Transformer to generate the representation of the sentence by dividing queries, keys and values into different heads and getting information from different subspaces.

Several variants have been proposed to improve the performance of the Transformer. Relative position representations [17] in the self-attention mechanism are proposed to replace the absolute position encoding, and it enhances the ability to capture the local information of the input sentence. He et al. [18] shared the parameters of each layer between the encoder and decoder to coordinate the learning between the encoder and decoder.

Besides, the architecture of the Transformer is also used in other NLP tasks. BERT [19] is a language model to pre-train deep bidirectional representations from unlabeled text by joint conditioning on both left and right context in all layers. Transformer-XL [20] enables the Transformer to learn dependency beyond a fixed length without disrupting temporal coherence. Eriguchi et al. [21] proposed a bilingual pre-trained language model, BIBERT, which can outperform both monolingual and multi-lingual language models for machine translation.

B. Dependency Parsing

As a fundamental NLP task, syntactic dependency parsing aims to predict the existence and type of linguistic dependency relations between words in a sentence [22], [23], [24].

Dependency parser may be roughly put into two categories in terms of searching strategies over parsing trees, graph-based and transition-based [25]. With the development of neural network applied to dependency parsing, there comes continuous progress for better parsing performance [22], [26]. Zhang et al. [27] proposed a neural probabilistic parsing model which explores up to third-order graph-based parsing with maximum likelihood training criteria. Li et al. [28] proposed a full character-level neural dependency parser together with a released character-level dependency treebank for Chinese. Dependency parsing is shown to be more effective than non-neural parser. Wu et al. [29] proposed a system for multilingual universal dependency parsing from raw text. Li et al. [30] proposed a tree encoder and integrate pre-trained language model features for a better representation of partially built dependency subtrees to enhance the model.

Fig. 1 illustrates the dependency parse tree for the sentence *It is a good thing for people*. The tree has only one root and every word in the sentence has only one parent. The label between one word and its parent reflects the relationship between them.

The dependency parse tree may be viewed as one pre-trained information for NMT, which has been incorporated into NMT for better translation. Eriguchi et al. [31] proposed a tree-to-sequence model with a tree-based encoder that encodes the phrase structure of a sentence as vectors. Aharony and Goldberg [32] designed a sequence-to-tree model which translates source sentences to a linearized constituency tree. The method proposed in this article can also be viewed as another way to incorporate dependency parsing information into NMT model.

C. Data Augmentation

Data augmentation is a training enhancement paradigm that has been broadly used in CV [1], [33], [34], [35] and NLP [2], [3], [4], [5], [6], [7], [8], [9], [36], [37].
In NMT, data augmentation is used for either better model robustness by generating more noisy data or better performance by generating more helpful training samples. As a kind of data augmentation approach at sentence level, back translation has been effectively adopted by unsupervised NMT [2], [3], [4], [5], [6], [38], in which data augmentation operation is essentially used to facilitate unsupervised NMT by generating data from monolingual corpora. In addition, back translation may also improve the performance of supervised NMT.

There are multiple data augmentation approaches for NMT. Artetxe et al. [6] proposed a method to swap words randomly with nearby words within a window size. Iyyer et al. [2] proposed a method to randomly drop some words of one sentence. Xie et al. [4] proposed two methods to add noise to sentences, randomly replacing words with a placeholder word and replacing words with other words having similar frequency distribution over the vocabulary. Fadaee et al. [39] proposed a data augmentation approach which targets low-frequency words. Wang et al. [40] proposed SwitchOut to randomly replace words in both the source and the target. Chen et al. [41] proposed a method for lexically constrained NMT by packing constraints into the source sentence with a separation symbol. Gao et al. [7] presented a method which replaces the one-hot representation of a word with a soft word which is a probabilistic distribution over the vocabulary. Guo et al. [42] proposed SeqMix to create new synthetic examples by softly combining sentences from the training set.

For the Transformer, the method in [6] can also be viewed as a special noisy-adding method for position encoding which does not apply impact over the word directly. In the Transformer, the positions of words are considered as features for input sequence, and relative positions of word-pairs are learned by scaled dot-product attention. It makes this method different from replacing words for the position to add noise.

Data augmentations such as Swap [6], Blanking [4], Replacement [4], SwitchOut [40] and WordDropout [43] are based on selecting words randomly and adding noises to sentence. Some methods [7], [39], [44] use language model to create additional training data by calculating the similarities between words or generating the new representation of words. These methods do not use linguistics knowledge directly and are based on different strategies to calculating probabilities, representation and other statistics, which can be regarded as computation-motivated method.

There are also some works motivated by and benefited from linguistic knowledge, which can be regarded as linguistics motivated methods. Iyyer et al. [45] used a syntactically controlled paraphrase networks to generate adversarial examples. Zhou et al. [46] proposed a method to create pseudo-parallel sentences with divergent syntactic structures by re-ordering. Sahin and Steedman [47] proposed a dependency tree based method to generate training data by removing dependency links and moving the tree fragments around the root.

Different from previous methods on data augmentation for NMT which may be regarded as computationally motivated, we propose a simpler and more effective linguistics motivated method to provide more accurate language generation for the high quality requirement in NMT training.

### III. Method

In NMT, data augmentation is used for either enhancing model robustness by generating additional noisy data or improving performance by generating more helpful training samples. In this work, we focus on enhancing word-level data augmentation approaches for improving model robustness. Typically, data augmentation methods for NMT select words from source sentences with a fixed sampling probability, followed by an alteration operation over the selected word to generate a variant of the original sentence. The new generated sentences maintain alignment with the same target sentences as the original source sentence and will be used as new training samples.

As described in Section II-C, we will work on three representative data augmentation operations as follows,

- **Blanking [4]:** Words in the sentence will be randomly replaced with a special placeholder token `<BLANK>`.
- **Dropout [2]:** Words in the sentence will be randomly dropped by simply setting the respective word embedding as a zero vector.
- **Replacement [4]:** Words in the sentence will be randomly selected and replaced with one word which has a similar unigram word frequency over the dataset.

To illustrate the data augmentation, we provide an example in Table I. Given one sentence We shall fight on the beaches., we would like to use this sentence to generate extra sentences using word-level data augmentation methods. Table I displays sentences generated by different methods.

### A. Basic Idea

Our proposed syntax-aware method is an enhancement over standard word-level data augmentation, which applies word-level operations, such as swapping, dropping and replacing to selected words. However, existing methods typically takes a completely sentence-independent and context-free strategy for word selection, which does not demonstrate sufficient effectiveness. In contrast, we propose using syntactic clues to guide the word selection, which results in a sentence-specific (context-dependent) strategy.

It is well known that a small number of important words can determine the meaning of a sentence. Consequently, modifying these syntactically and semantically important words can change the sentence radically. Being a robustness-oriented data augmentation, it is necessary to improve the translation robustness by intentionally introducing moderate noisy data. However,
too many radically altered sentences may become a genuinely harmful noise which eventually hurts the NMT training.

To improve the model robustness, effective data augmentation methods should follow these rules:

- The data augmentation method should select a large enough number of words for word-level operations.
- The data augmentation method should avoid selecting important words that could hurt the NMT training.

To meet such requirements, it is necessary to identify a heuristic clue to measure the importance of words and subsequently determine the probability of word selection to alter the corresponding word for data augmentation.

### B. Dependency Tree Depth as Clue

Given a dependency parsing tree, the root can figure out the most salient word in the sentence which is usually a key verb in linguistics. Additionally, in dependency grammar, the parent in this tree is always more essential than its children. Based on these characteristics of dependency parsing trees, we use the distance between words and the root, or the depths of words in the parsing tree, as initial clues to measure the importance of words in our data augmentation method. Consequently, the leaf node of the dependency parsing tree are more likely to be selected for alteration during data augmentation. Fig. 1 shows an example of word depth in the dependency tree.

In this work, we only use depths of words in the dependency parse tree while exclude other information delivered by the dependency tree such as dependency relationship or label among any word pair, whose reason is twofold:

- As we adopt the Transformer as our NMT model baseline, it has demonstrated the ability to learn the relationship between any word pair through its powerful self-attention mechanism.
- We have to identify a clear enough heuristic clue indicating the importance of a word within its sentence, which needs to measure the relationship between the word and the entire sentence rather than the relationship of a word pair offered by the dependency tree.

Different from existing data augmentation methods which commonly rely on word frequency as a straightforward clue for words selection, our syntax-aware clue has an obvious advantage in which the word selection is context-aware and specific to its sentence, rather than selecting the same words across different sentences based on their frequency in the entire dataset only.

### C. Probability of Word Selection

Taking the tree depth as an initial heuristic clue, we argue that a well-designed word selection probability should satisfy the following conditions:

- The probabilities for words with different tree depths should have diversity for sufficient distinguishability.
- In sentences of different lengths, especially long sentences, it is important to select words with an appropriate chance.

Given a sentence \( s = w_1, w_2, \ldots, w_n \) of length \( n \) to calculate the final probability \( P = \{p_1, \ldots, p_n\} \), we first use a function to calculate a original score \( q_i \) based on \( d_i \), where \( d_i \) is the depth of word \( w_i \). To avoid selecting important words, we propose three functions which are strictly monotonically increasing with increasing depth of \( w_i \) for \( q_i \).

- **Linear function**: This function makes \( q_i \) increase linearly with the increase of word depths as follow,
  \[
  q_i = d_i - 1. 
  \]

- **Logarithmic function**: The growth speed of \( q_i \) gradually decreases with the increase of word depths as follow,
  \[
  q_i = \log(d_i). 
  \]

- **Power function**: Similar to the logarithmic function, the growth speed of \( q_i \) gradually decreases with the increase of word depths in the power function while scores for words are all smaller than 1 which can be used as probability directly. \( d_i \) is calculated as follow
  \[
  q_i = 1 - \frac{1}{2d_i - 1}. 
  \]

Note that \( q_i \) is equal to 0 when \( w_i \) is exactly the root node in which \( d_i = 1 \).

With \( Q = \{q_1, \ldots, q_n\} \) for \( w_i \), we use **Softmax** function to normalize \( Q \) and get an adjusted probability distribution \( P = \{p_1, \ldots, p_n\} \) for \( s \) as follow,

\[
P(s) = \text{Softmax}(Q(s)) \\
= \{p_1, p_2, \ldots, p_n\},
\]

At last, to select words in the longer sentence with proportionable possibility, we introduce a sentence length compensation for the final word selection probability as

\[
p_i^f = \alpha p_i n,
\]

where \( p_i^f \) is the final possibility to select word \( w_i \) and \( \alpha \) is a hyper-parameter to control the magnitude of possibility changing.

Using the dependency tree example in Fig. 2, we demonstrate a procedure to compute the probabilities for word selection as shown in Table II. According to (1), (3), (2) and (5), we get the initial and final probabilities of words. The larger the depth of a word is, the more likely the word be selected. The fifth line in Table II shows that with the smallest depth, the possibility of *thing* is lower than other nodes such as leaf node *for*.  

---

**Fig. 2.** Depths of words in dependency tree of Fig. 1. Fig. 3 and Fig. 4 show dependency trees after removing words *good* and *is* respectively.
TABLE II
COMPUTING THE PROBABILITY OF WORD SELECTION

| It | is | a | good | thing | for | people |
|----|----|----|------|-------|-----|--------|
| 2  | 1  | 3  | 3    | 2     | 3   | 2      |
| $q_i$ | 0.5 | 0.75 | 0.75 | 0.5   | 0.875 | 0.75 | 0.5 |
| $p_i$ | 0.112 | 0.068 | 0.144 | 0.144 | 0.112 | 0.163 | 0.144 | 0.112 |
| $p'_i$ | 0.089 | 0.054 | 0.115 | 0.115 | 0.089 | 0.131 | 0.089 |

$Blanking$ <BLANK> is a good thing for <BLANK>.
$Dropout$ is a good thing for $p_i$.
$Replacement$ That is a good thing for person.

$Syntax-aware Blanking$ It is a <BLANK> thing <BLANK> people.
$Syntax-aware Dropout$ It is a thing <BLANK> people.
$Syntax-aware Replacement$ It is a great thing to people.

The last two lines are the result of the original data augmentation and ours. Note that blanking is to replace word with a special placeholder<BLANK>.

IV. EXPERIMENTS

In this article, data augmentation will only process source data from the training data. The translation quality is evaluated by multiBLEU and sacreBLEU score. Statistical significance ($p < 0.05$) on the difference of BLEU scores is tested by script bootstrap-hypothesis-difference-significance.pl.

A. Datasets

We use three low-resource translation tasks, IWSLT14 German-to-English (De-En), IWSLT14 English-to-French (En-Fr), IWSLT15 English-to-Vietnamese (En-Vi) and one high-resource translation task, WMT14 English-to-German (En-De) for our evaluation. In these four tasks, the word order of Vietnamese is different from English while German and French are same to English.

**IWSLT14 German-English** IWSLT14 De-En dataset contains 153 K training sentence pairs. We use 7 K data from the training set as validation set and use the combination of dev2010, dev2012, tst2010, tst2011 and tst2012 as test set with 7 K sentences same to [48], [49], [50]. Our dataset is preprocessed by script. BPE algorithm is used to process words into subwords, and the number of subword tokens in the shared vocabulary is 10 K.

**IWSLT14 English-French** IWSLT14 En-Fr dataset contains 168 K training sentence pairs. Similar to IWSLT14 De-En, we use 7 K data from the training set as validation set and use the combination of dev2010, dev2012, tst2010, tst2011 and tst2012 as test set with 7 K sentences same to [48], [49], [50]. We use a modified script from De-En task to preprocess the dataset. The number of subword tokens in the shared vocabulary is 10 K which is same to De-En.

**IWSLT15 English-Vietnamese** IWSLT15 En-Vi dataset contains 112 K training sentence pairs. We use 5 K data from the training set as validation set. We use three different test sets to evaluate our model. The first test set has 5.1 K sentences which is the combination of dev2010, tst2010, tst2011 and tst2012. The second test set is tst2013 with 1.3 K sentences which is same to other work [40]. The third test set with 6.3 K sentences is the combination of the first and the second test sets. The number of subword tokens in the shared vocabulary is 8 K.

**WMT14 English-German** WMT14 En-De dataset contains 4.5 M sentence pairs for training. We use the script to preprocess data. We use the combination of newstest2012 and newstest2013 as validation set and newstest2014 as test set. The sentences longer than 80 are removed from the training dataset. All data are segmented by BPE so that number of subwords in the shared vocabulary is 40 K.

Stanford Parser is used to process German corpus to get dependency tree and POS tags. For English corpus, we use Stanford Dependency Parser to generate dependency tree.

Table III shows the number of augmented data for different NMT tasks using our methods.

![Dependency tree and depths of words change removing word good. It shows that the depths of other words are not changed after removing good.](image)

**TABLE III**
THE AMOUNT OF AUGMENTED DATA ON DIFFERENT NMT TASKS WITH OUR METHODS

| Training Data | De-En | En-Fr | En-Vi | En-De |
|---------------|-------|-------|-------|-------|
| Power Function | 150K | 168K | 112K | 4.5M |
| Linear Function | 150K | 168K | 112K | 4.5M |
| Logarithmic Function | 152K | 168K | 112K | 4.5M |

IWSLT14 English-French IWSLT14 En-Fr dataset contains 168 K training sentence pairs. Similar to IWSLT14 De-En, we use 7 K data from the training set as validation set and use the combination of dev2010, dev2012, tst2010, tst2011 and tst2012 as test set with 7 K sentences same to [48], [49], [50]. We use a modified script from De-En task to preprocess the dataset. The number of subword tokens in the shared vocabulary is 10 K which is same to De-En.

**IWSLT15 English-Vietnamese** IWSLT15 En-Vi dataset contains 112 K training sentence pairs. We use 5 K data from the training set as validation set. We use three different test sets to evaluate our model. The first test set has 5.1 K sentences which is the combination of dev2010, tst2010, tst2011 and tst2012. The second test set is tst2013 with 1.3 K sentences which is same to other work [40]. The third test set with 6.3 K sentences is the combination of the first and the second test sets. The number of subword tokens in the shared vocabulary is 8 K.

**WMT14 English-German** WMT14 En-De dataset contains 4.5 M sentence pairs for training. We use the script to preprocess data. We use the combination of newstest2012 and newstest2013 as validation set and newstest2014 as test set. The sentences longer than 80 are removed from the training dataset. All data are segmented by BPE so that number of subwords in the shared vocabulary is 40 K.

Stanford Parser is used to process German corpus to get dependency tree and POS tags. For English corpus, we use Stanford Dependency Parser to generate dependency tree.

Table III shows the number of augmented data for different NMT tasks using our methods.

1[Online]. Available: https://github.com/pytorch/fairseq/blob/main/examples/translation/prepare-iwslt14.sh

2[Online]. Available: https://github.com/pytorch/fairseq/blob/main/examples/translation/prepare-wmt14en2de.sh

3[Online]. Available: https://nlp.stanford.edu/software/lex-parser.html

4[Online]. Available: https://nlp.stanford.edu/software/nndep.html

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
TABLE IV
HYPER-PARAMETERS FOR OUR EXPERIMENTS

| Parameter | De-En&En-Fr&En-Vi | En-De |
|-----------|-------------------|-------|
| Layers    | 6                 | 6     |
| Dimension | 512               | 512   |
| Head      | 4                 | 8     |
| FF        | 1024              | 2048  |
| Dropout   | 0.3               | 0.1   |

FF is short for Feed-forward layer. The number of heads is based on the dimension for word and feature.

B. Hyper-Parameters

Table IV shows the hyper-parameters for our methods. For En-De, we follow the setting of Transformer-base. For De-En, En-Fr and En-Vi, we follow the setting of Transformer-small. The hyper-parameter \( \alpha \) in (5) is set to 0.1.

C. Training

IWSLT14 De-En, En-Fr and IWSLT15 En-Vi tasks are trained on one CPU (Intel i7-5960X) and one nVidia RTX TITAN X GPU, and WMT14 En-De task is trained on one CPU (Intel i7-5960X) and four nVidia RTX TITAN X GPU. The implementation of the model is based on fairseq-0.9.0. We choose Adam optimizer [51] with \( \beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 10^{-9} \) and the learning rate setting strategy, which are all the same as [15],

\[
lr = d^{-0.5} \cdot \min(step^{-0.5}, step \cdot \text{warmup}^{1.5})
\]

where \( d \) is the dimension of embeddings, \( step \) is the step number of training and \( \text{warmup} \) is the step number of warmup. When the number of steps is smaller than the step of warmup, the learning rate increases linearly and then decreases. The warmup steps for WMT and IWSLT tasks are 8000 and 4000 respectively.

We use beam search decoder for De-En, En-Fr and En-Vi tasks with beam width 6. For En-De, following [15], the width for beam search is 6 and the length penalty \( \alpha \) is 0.2. The batch size is 1024 for De-En and 4096 for En-De. To evaluate the translation quality, we use multiBLEU score, which is the aggregated mean of BLEUs computed with 1-gram, 2-gram, 3-gram and 4-gram, for all four tasks, and the multiBLEU score is calculated by script \texttt{multi-bleu.perl}. Besides, we also calculate detokenized sacreBLEU [52] for En-De task. In this article, sacreBLEU is calculated by toolkit \texttt{SacreBLEU}.

In this article, statistical significance \((p < 0.05)\) on the difference of BLEU scores is tested by script \texttt{bootstrap-hypothesis-difference-significance.pl}, which is based on the bootstrap resampling methods designed for NMT results evaluation [53].

V. RESULTS

A. Main Results

Our baselines for WMT and IWSLT tasks are Transformer-base and Transformer-small respectively. We also compare our methods with SwitchOut [40], WordDropout [43] and three original data augmentation methods Blanking, Dropout and Replacement [2], [4]. To reproduce the baseline and data augmentation methods, we use the optimized hyper-parameters and training settings from the corresponding literature for different baseline methods.

Table V compares our data augmentation methods with the baseline method and other data augmentation methods. Note that the results in Table V outperforming the baselines significantly with \( p < 0.05 \) and \( p < 0.01 \) are marked by \( \dagger \) and \( \ddagger \) respectively, and the results outperforming other data augmentation methods are marked by \( \Delta \).

Table V shows that our method enhances all the original data augmentation on all tasks. Our methods not only outperforms the performance of the Transformer but also outperforms other DA methods.

WMT14 En-De: On WMT14 En-De task, our Blanking method with Power function performs the best with 28.3 multi- BLEU scores and 28.1 sacreBLEU scores, which outperforms the Transformer-base and the best DA methods significantly by more than 1.0 and 0.7 sacreBLEU scores respectively. Besides, all our methods improve the performance except the Replacement method with Linear function, which achieves the lowest performance with only 26.9 sacreBLEU scores and hurts the performance.

IWSLT14 De-En: On IWSLT14 De-En task, our Blanking method with Logarithmic function achieves the best performance with 37.3 BLEU scores and outperforms the original Blanking method by 0.7 BLEU scores. Our other methods also outperform the baseline, and all our methods outperform the original data augmentation methods.

IWSLT14 En-Fr: On IWSLT14 En-Fr task, our Dropout method with Power function achieves the best performance with 45.6 BLEU scores and outperforms the baseline by 0.5 BLEU scores. Some of our methods such as Replacement with Power function hurt the performance.

IWSLT15 En-Vi: On IWSLT15 En-Vi task, all our methods outperform the baseline and the other data augmentation methods in three different test sets. In the first test set, our Dropout method with Power function achieves the highest 29.3 BLEU scores and outperforms the baseline by 1.3 BLEU scores. In the second test set, our Blanking method with Power function achieves the highest 31.2 BLEU scores and outperforms the baseline by 0.9 BLEU scores. In the third test set, Dropout with Power function achieves the highest 28.9 BLEU scores and outperforms the baseline by 0.6 BLEU scores.

B. Analysis

Table V illustrates that our method achieves comparable results to other methods without requiring any changes to the training process or the model architecture. It is important to

---

5[Online]. Available: https://github.com/pytorch/fairseq/archive/refs/tags/v0.9.0.zip
6[Online]. Available: https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl
7[Online]. Available: https://github.com/moses-smt/mosesdecoder/blob/master/scripts/analysis/bootstrap-hypothesis-difference-significance.pl
note that our method focuses on straightforward or naive data augmentation approaches instead of methods that leverage additional data in a latent way such as pre-trained language models.

Some methods, such as [39] and [7], rely on representations generated by language models and require complicated pre-training or computationally intensive representation learning from a large scale of additional data. In contrast, our method utilizes syntactic knowledge to enhance performance and is simple to implement. Furthermore, unlike other works using generated representation [7], [39], embedding [57] or loss [55] to generate new sentences during model training, our approach is an offline method that generates new sentences to enlarge the dataset before model training, which makes our method generally applicable and independent of the specific model or training setup.

Table V illustrates that our method can enhance the performance on different tasks with varying scales of training data. Besides, Table V indicates that the performance impact of our method varies based on the language pair. For example, our method utilizes the syntactic knowledge of English for En-Fr, En-De and En-Vi tasks, while the improvements on these tasks are distinct. Notably, the En-Fr and the En-Vi tasks are both low-resource tasks based on the dependency parsing tree of English, while our method improves the En-Vi task more effectively than En-Fr, which indicates that language pairs can impact performance. Additionally, our method uses syntactic knowledge of English and German for En-De and De-En tasks respectively, and the improvement on En-De is better compared with De-En, indicating that the language used for syntactic knowledge can also influence performance.

On low-resource translation tasks, our method with Linear function yields most of the lowest BLEU scores and fails to outperform three original data augmentation methods, especially the method with Linear function hurts the performance of the En-Fr task. Table VI indicates that Linear function results in the largest depths, surpassing the average values of the depth of dependency parsing tree on low-resource tasks. It shows that our method with Linear function selects more words near the leaves, which makes the Transformer learn less syntactic knowledge and limits the flexibility of word selection which may hurt the performance. Therefore, it is necessary to select some words near roots to maintain the flexibility of word selection for optimal performance.

Table V

| Model                          | multBLEU | sacreBLEU |
|-------------------------------|----------|-----------|
|                               | De-En    | En-Fr     | En-Vi     | En-De    | En-De    |
|                               | 1st (5.1K)| 2nd (1.3K)| 3rd (6.5K)|         |          |
| Transformer                    | -        | -         | -         |         | 31.8     | 27.3     |
| [54]                           | -        | -         | -         |         |          |          |
| Gradient-based Method [55]     | -        | -         | -         |         |          |          |
| BPE-Dropout [56]               | -        | -         | -         |         |          |          |
|                                | Our Implemented Models |          |          |          |          |
| Transformer (small)            | 36.6     | 45.1      | 28.0      | 30.3     | 28.3     |
| Transformer (base)             | -        | -         | -         | -         |          |          |
| SwitchDropout [46]             | 37.0 (+0.3) | 45.5 (+0.4) | 28.2 (+0.2) | 30.5 (+0.4) | 27.3 (+0.4) | 27.5 (+0.2) | 27.9 (+0.2) |
| WordDropout [43]               | 36.6 (+0.0) | 44.9 (+0.2) | 28.4 (+0.4) | 30.4 (+0.1) | 28.3 (+0.0) | 27.7 (+0.4) | 27.4 (+0.3) |
| BLanking [4]                   | 36.7 (+0.1) | 45.3 (+0.2) | 28.3 (+0.2) | 30.2 (+0.1) | 28.0 (+0.3) | 27.2 (+0.1) | 27.0 (+0.1) |
| Dropout [2]                    | 36.7 (+0.1) | 45.1 (+0.0) | 28.2 (+0.2) | 30.5 (+0.2) | 28.4 (+0.1) | 27.6 (+0.3) | 27.2 (+0.1) |
| Replacement [4]                | 36.6 (+0.0) | 45.0 (+0.1) | 28.2 (+0.2) | 30.2 (+0.1) | 28.1 (+0.2) | 27.3 (+0.0) | 27.0 (+0.1) |
| Power function                 |          |          |          |          |          |          |          |
| Our Method (Blanking)          | 37.0 (+2.4) | 45.1 (+1.4) | 29.2 (+1.4) | 30.2 (+1.0) | 28.8 (+0.8) | 28.4 (+0.3) | 28.8 (+0.3) |
| Our Method (Dropout)           | 36.8 (+0.2) | 45.6 (+0.5) | 29.3 (+1.3) | 31.0 (+0.7) | 28.9 (+0.6) | 27.5 (+0.2) | 27.3 (+0.2) |
| Our Method (Replacement)       | 37.0 (+0.4) | 45.4 (+0.3) | 29.2 (+1.2) | 30.9 (+0.6) | 28.8 (+0.5) | 27.6 (+0.3) | 27.3 (+0.2) |
| Linear function                |          |          |          |          |          |          |          |
| Our Method (Blanking)          | 37.1 (+0.5) | 45.3 (+0.2) | 28.9 (+0.9) | 30.5 (+0.2) | 28.4 (+0.1) | 28.1 (+0.8) | 28.0 (+0.9) |
| Our Method (Dropout)           | 36.7 (+0.1) | 45.0 (+0.1) | 29.2 (+1.2) | 31.0 (+0.7) | 28.7 (+0.4) | 27.4 (+0.1) | 27.1 (+0.0) |
| Our Method (Replacement)       | 36.8 (+0.2) | 44.9 (+0.2) | 28.9 (+0.9) | 31.0 (+0.7) | 28.6 (+0.3) | 27.2 (+0.1) | 26.9 (+0.2) |
| Logarithmic function           |          |          |          |          |          |          |          |
| Our Method (Blanking)          | 37.3 (+0.7) | 45.4 (+0.3) | 28.8 (+0.8) | 31.0 (+0.7) | 28.5 (+0.2) | 27.9 (+0.6) | 27.7 (+0.6) |
| Our Method (Dropout)           | 36.8 (+0.2) | 45.2 (+0.1) | 28.9 (+0.9) | 31.0 (+0.7) | 28.6 (+0.3) | 27.4 (+0.1) | 27.2 (+0.1) |
| Our Method (Replacement)       | 37.2 (+0.6) | 45.0 (+0.1) | 29.1 (+1.1) | 30.9 (+0.6) | 28.7 (+0.4) | 27.5 (+0.2) | 27.3 (+0.2) |

Table VI

| Function | Power | Linear | Logic |
|----------|-------|--------|-------|
| De-En    | 4.25  | 3.40   | 4.50  | 3.63  |
| En-Fr    | 5.17  | 3.85   | 5.21  | 4.14  |
| En-Vi    | 5.17  | 3.86   | 5.23  | 4.15  |
| En-De    | 6.96  | 4.62   | 6.90  | 5.16  |

Values in parentheses are the average depths of a dependency parsing tree.
On En-De task, the average depth of selected words by Linear function is still the largest. However, it is smaller than the average depth of the dependency parsing tree and limits the negative impact on the performance. Compared with low-resource translation tasks, En-De has 4.5 M sentence pairs for training, which makes our method select enough words with low depths. The result demonstrates that machine translation and data augmentation should focus on important words to improve the performance while other words should not be ignored.

VI. ABLATION STUDY

A. Effect of Different Training Stages

To evaluate the performance of our method in different training stages, we do data augmentation Blanking with Power function before training and during training respectively on En-De. Table VII indicates no significant performance difference between the two results, suggesting that our method can be effectively applied in different stages of training.

B. Effect of Dependency Parsing Tree Depths

To evaluate the effect of dependency parsing tree depths, we introduce a hyper-parameter $\tau$ to (3) for the Power function and get

$$q_i = \frac{\tau}{1 - 2^{d_i - 1}},$$

where $0 \leq \tau \leq 1$. As the $\tau$ increases, the effect of depths on the probability of word selection also increases. We evaluate our model on En-Vi dataset with the 3rd test set. Table VIII shows the average depths of selected words and the BLEU scores with different $\tau$. The results in Table VIII demonstrates that our method performs better with a larger $\tau$, which indicates that larger depths of dependency parsing trees do benefit the performance of our method.

C. The Speed of Convergence With Different Methods

To evaluate the convergence performance of our method, we conduct experiments on De-En tasks using our method with Power function and calculate the BLEU score of the first 45 epochs. Fig. 5 shows that all our methods can converge faster than the baseline, which indicates that our method can achieve better performance at early stages of training and improve the speed of convergence.

D. Effect of Hyper-Parameter $\alpha$

To investigate the effect of hyper-parameter $\alpha$ in (5), we evaluate our methods with Power on En-Fr task. Table IX shows the results of our methods with different $\alpha$ and indicates that our method is not sensitive to the hyper-parameter $\alpha$. The performances with $\alpha < 0.3$ are similar and drop with the increase of $\alpha$, which means that $\alpha = 0.1$ can perform best compared to other values of $\alpha$. Table IX also shows that $\alpha$ has little influence on the performance and an optimal $\alpha$ can be easily found only by a few times of checks.

E. Effect of Source and Target

To evaluate the performance of our method in source and target, we conduct data augmentation with our method Power function and Blanking in source, target and both sides on De-En, En-Fr and En-De. Table XI shows that our method can improve
the performance in the source while may hurt the performance in the target, indicating that the decoder is more sensitive to syntactically modified data. Compared to SwitchOut which conducts data augmentation in target, our method selects more words closer to leaves of the dependency parsing tree. It makes the encoder and decoder generate representations that focus on important words. However, all target words have equal statuses during evaluation because the calculation of BLEU score only uses sequences of words but not representations of sentences, and missing any words may hurt the BLEU score.

### F. Effect of the Generality of Our Method

To validate the generality of our method, we evaluate our method on a text summarization (Annotated Gigaword [58]) task. Table XII shows a result of our method with Power function and Blanking which outperforms other methods using our method. The results indicate that our method can be applied to other NLP tasks and achieve better performance.

### G. Effect of the Amounts of Augmented Data

To evaluate the effect of different amounts of augmented data, we evaluate our method on IWSLT14 En-Fr. Table XIII presents the results of experiments conducted on the IWSLT14 En-Fr task, where different amounts of augmented data were utilized. In order to clearly indicate the amount of augmented data, we introduce the ratio $r = \frac{M}{T}$, where $M$ represents the amount of augmented data and $T$ represents the amount of the original training data. Table XIII shows that the model improves performance with an increased amount of augmented data, while the improvement is tiny when $r \leq 0.4$. It reveals that the performance of the model is influenced by the amount of augmented data, and more augmented data leads to improved performance.

### VII. Case Study

Table X gives three cases from IWSLT14 En-Fr corpus. These cases show that our methods tend to select some words which will not change the structure of dependency tree. However, our

| Sentence | Original Method | Blanking | Dropout | Replacement |
|----------|-----------------|----------|---------|-------------|
| i wrote a book about eight incredible people all over this country doing social <BLANK> work. | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". |
| i experienced this firsthand myself when i graduated from barnard college in 2002. | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". |
| they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". |

Note that <BLANK> is the special placeholder is to replace word in method blanking, replaced words are marked as red, and dropout words are crossed out.

| Sentence | Power function | Linear function | Logic function |
|----------|----------------|----------------|---------------|
| i wrote a book about eight incredible people all over this country doing social justice work. | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". |
| i experienced this firsthand myself when i graduated from barnard college in 2002. | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". |
| they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". | they were the co-authors of a book called "manifesta". |

Table X gives three cases from IWSLT14 En-Fr corpus. These cases show that our methods tend to select some words which will not change the structure of dependency tree. However, our
methods also select important words such as the first sentence which is processed by Replacement method with Logarithmic function. It means that our syntax-aware enhancement methods may reduce the probability of word being selected if this word is important while all words can still be selected by our methods which can avoid selecting specific words only.

VIII. DISCUSSION

As mentioned above, our proposed syntax-aware method tends to select words closer to the leaves of dependency tree and avoids selecting words near the root which are more likely important words in sentence. Our experimental results indicate that our proposed method is generally effective for performance improvement, and indicates that NMT indeed has a data augmentation preference of revising those marginal words with less importance. This observation can be mirrored by multiple empirical comparisons in our experimental results.

We assume that the automatically generated data by a data augmentation method may include both useful and harmful parts for machine learning. In the case of NMT, the harmful data may result from the revision of key words in the original sentence. Among all three baseline data augmentation methods, the method Blanking is better than any other, while Replacement is the worst, with a performance gap of 1.0 BLEU scores. The method Blanking gives the least change to a sentence among all the three methods, while the method Replacement may change the meaning and syntactic of the original sentence by adding new words to the sentence. The method Dropout takes a moderate revision over sentence by simply removing partial information of words, which will not alter the original sentence as violently as the method Replacement. Thus, it is not a coincidence that the performance improvement from the three data augmentation methods has the same ranking list as the same as the degree of change to sentence by them. Overall, data augmentation for NMT prefer to those less syntactically and less semantically modified data.

Compared with others, the method Blanking keeps syntactic structure of the original sentence unchanged and does not introduce potentially harmful information as Replacement. New words replaced in the method Replacement may dramatically change the meaning of sentence which makes it perform quite unsatisfactorily. With our method to adjust the probabilities of selecting words to keep those key words of sentence, our method can reduce the negative impacts of the original data augmentation methods.

IX. CONCLUSION

In this work, we have presented a novel syntax-aware enhancement of robustness-oriented data augmentation for NMT, which is capable of setting sentence-specific probability on word selection for word-level revising. In detail, we take the depth of word in dependency parse tree to give the initial clue for word importance measuring, so that less important words to the its sentence may be more likely selected. Our proposed method is conceptually simple, easily implemented, and conveniently incorporated to standard word level data augmentation method, which significantly differs from previous computation only motivated methods and delivers helpful linguistic clues for better data augmentation in NMT. Our method is evaluated on WMT14 En-De dataset, IWSLT14 De-En dataset, IWSLT14 En-Fr dataset and IWSLT15 En-Vi dataset. The result of extensive experiments show our method can outperform strong baselines by effectively enhancing standard data augmentation methods.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in Proc. 26th Annua. Conf. Neural Inf. Process. Syst., 2012, pp. 1106–1114.
[2] M. Iyyer, V. Manjunatha, J. Boyd-Graber, and H. Daumé III, “Deep unordered composition rivals syntactic methods for text classification,” in Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics, 7th Int. Joint Conf. Natural Lang. Process., 2015, pp. 1681–1691.
[3] R. Sennrich, B. Haddow, and A. Birch, “Improving neural machine translation models with monolingual data,” in Proc. 54th Annu. Meeting Assoc. Comput. Linguistics, 2016, pp. 86–96.
[4] Z. Xie et al., “Data noising as smoothing in neural network language models,” in Proc. 5th Int. Conf. Learn. Representations, 2017.
[5] S. Edunov, M. Ott, M. Auli, and D. Grangier, “Understanding back-translation at scale,” in Proc. Conf. Empirical Methods Natural Lang. Process., 2018, pp. 489–500.
[6] M. Artetxe, G. Labaka, E. Agirre, and K. Cho, “Unsupervised neural machine translation,” in Proc. 6th Int. Conf. Learn. Representations, 2018.
[7] F. Gao et al., “Soft contextual data augmentation for neural machine translation,” in Proc. 57th Annu. Meeting Assoc. Comput. Linguistics, 2019, pp. 5539–5544.
[8] M. Xia, X. Kong, A. Anastasopoulos, and G. Neubig, “Generalized data augmentation for low-resource translation,” in Proc. 57th Annu. Meeting Assoc. Comput. Linguistics, 2019, pp. 5786–5796.
[9] Z. Li and L. Specia, “Improving neural machine translation robustness via data augmentation: Beyond back-translation,” in Proc. 5th Workshop Noisy User-generated Text, 2019, pp. 328–336.
[55] Y. Cheng, “Robust neural machine translation with doubly adversarial inputs,” in Proc. 57th Conf. Assoc. Comput. Linguistics, 2019, pp. 4324–4333, doi: 10.18653/v1/p19-1425.

[56] I. Provilkov, D. Emelianenko, and E. Voita, “BPE-dropout: Simple and effective subword regularization,” in Proc. 58th Annu. Meeting Assoc. Comput. Linguistics, 2020, pp. 1882–1892, doi: 10.18653/v1/2020.acl-main.170.

[57] Y. Cheng, L. Jiang, W. Macherey, and J. Eisenstein, “AdvAug: Robust adversarial augmentation for neural machine translation,” in Proc. 58th Annu. Meeting Assoc. Comput. Linguistics, 2020, pp. 5961–5970, doi: 10.18653/v1/2020.acl-main.529.

[58] C. Napoles, M. Gormley, and B. Van Durme, “Annotated gigaword,” in Proc. Joint Workshop Autom. Knowl. Base Construction Web-scale Knowl. Extraction, Jun. 2012, pp. 95–100. [Online]. Available: https://aclanthology.org/W12-3018

Sufeng Duan received the bachelor’s degree in spatial information and digital technology from Wuhan University, Wuhan, China and master’s degree in computer science and technology from Shanghai Jiao Tong University, Shanghai, China, in 2014 and 2017, respectively. Since 2018, he has been a Ph.D. student with the Center for Brain-like Computing and Machine Intelligence of Shanghai Jiao Tong University. His research focuses on natural language processing, especially Chinese word segmentation and machine translation.

Hai Zhao received the B.Eng. degree in sensor and instrument engineering and M.Phil. degree in control theory and engineering degrees from Yanshan University, Qinhuangdao, China, in 1999 and 2000, respectively. He received the Ph.D. degree in computer science from Shanghai Jiao Tong University, Shanghai, China, in 2005. He is currently a Full Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, after he joined the university in 2009. He was a Research Fellow with the City University of Hong Kong, Hong Kong, from 2006 to 2009, Visiting Scholar with Microsoft Research Asia, Beijing, China, in 2011, and Visiting Expert with NICT, Tokyo, Japan in 2012. His research interests include natural language processing and related machine learning, data mining, and artificial intelligence. He is an ACM Professional Member and was the Area Co-Chair with ACL 2017 on tagging, chunking, syntax, and parsing, and the Senior Area chair with ACL 2018, 2019 on phonology, morphology, and word segmentation.

Dongdong Zhang is a Principal Researcher with Microsoft Research Asia. His research interests include neural machine translation, pre-training techniques, and multilingual modeling.