mixSeq: A Simple Data Augmentation Method for Neural Machine Translation

Xueqing Wu [1], **Yingce Xia** [2], **Jinhua Zhu** [1], **Lijun Wu** [2], **Shufang Xie** [2], **Tao Qin** [2]
[1] University of Science and Technology of China [2]Microsoft Research Asia
jwuwuwu24@gmail.com, teslazhu@mail.ustc.edu.cn
{yingce.xia,lijuwu,shufxi,taoqin}@microsoft.com

Abstract

Data augmentation, which refers to manipulating the inputs (e.g., adding random noise, masking specific parts) to enlarge the dataset, has been widely adopted in machine learning. Most data augmentation techniques operate on a single input, which limits the diversity of the training corpus. In this paper, we propose a simple yet effective data augmentation technique for neural machine translation, *mixSeq*, which operates on multiple inputs and their corresponding targets. Specifically, we randomly select two input sequences, concatenate them together as a longer input as well as their corresponding target sequences as an enlarged target, and train models on the augmented dataset. Experiments on nine machine translation tasks demonstrate that such a simple method boosts the baselines by a non-trivial margin. Our method can be further combined with single-input based data augmentation methods to obtain further improvements.

1 Introduction

Data augmentation, which enlarges the training corpus by manipulating the inputs through given rules, has been widely used in machine learning tasks. For image classification, there are various data augmentation methods, including cropping, flipping, rotating,cut-out (DeVries and Taylor, 2017), etc. For natural language processing (briefly, NLP), similar data augmentation methods also exist, like randomly swapping words (Lample et al., 2018a), dropping words (Iyyer et al., 2015), and masking specific words (Xie et al., 2017). With data augmentation, the main content of the input is not affected but the noise is introduced so as to increase the diversity of the training set. The effectiveness of above data augmentation methods has been verified by their strong performance improvements in both image processing and NLP tasks. For example, with the combination of data augmentation and meta-learning, state-of-the-art result of image classification is achieved (Cubuk et al., 2019).

Most existing data augmentation methods take one sample from the training set as input, which might limit the scope and diversity of the training corpus. Mixup (Zhang et al., 2018) is a recently proposed data augmentation method, where two samples from the training corpus are leveraged to build a synthetic sample. Specifically, let x_1, x_2 denote two images from the training set, and y_1, y_2 denote their corresponding labels. The synthetic data $(\lambda x_1 + (1-\lambda)x_2, \lambda y_1 + (1-\lambda)y_2)$ is introduced to the augmented dataset, where λ is randomly generated. Such a strategy is further enhanced in follow-up works (Zhang et al., 2019; Berthelot et al., 2019). Pair sampling (Inoue, 2018) is another data augmentation method where the synthetic sample is built as $(0.5x_1 + 0.5x_2, y_1)$. In comparison, according to our knowledge, such ideas are not leveraged in NLP tasks (e.g., machine translation). Therefore, in this work, we explore along this direction to see whether augmenting data through mixing multiple sentences is helpful.

In sequence learning tasks, two inputs x_1 and x_2 might contain different numbers of units (e.g., words or subwords). Besides, for sequence generation tasks, their labels y_1 and y_2 are of different lengths. Therefore, it is not practical to sum them up directly. Instead, we choose to concatenate two inputs and the two labels to get the synthetic data. We find that it is important to use a special token to separate the two sentences in a synthetic data. We name our proposed method as *mixSeq*.

mixSeq is a simple yet very efficient and effective data augmentation method. We conduct experiments on 9 machine translation tasks and find that *mixSeq* can boost the baseline by 0.66 BLEU on average. Specifically, on FLORES Sinhala↔English, our method can improve the baseline by 1.03 points. *mixSeq* can be further combined with data augmen-

Proceedings of the 18th International Conference on Spoken Language Translation, pages 192–197
Bangkok, Thailand (Online), August 5–6, 2021.

tation methods working on a single input, e.g., randomly dropping, swapping or masking words, to further improve the performance (see Table 3).

Normally, *mixSeq* randomly samples the two concatenated sequences. However, if the two concatenated sequences are contextually related, we can enhance our *mixSeq* to a context-aware version: *ctxMixSeq*, which will result in better performance (see Table 4).

2 Our Method

Notations: Let $\mathcal{X}$ and $\mathcal{Y}$ denote two language spaces, which are collections of sentences in the corresponding languages. The target of neural machine translation (briefly, NMT) is to learn a mapping from $\mathcal{X}$ to $\mathcal{Y}$. Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ denote the bilingual NMT training corpus, where $x_i \in \mathcal{X}, y_i \in \mathcal{Y}$, and N is the number of training samples. Let $\texttt{concat}(\cdots)$ denote the concatenation operation, where the input sequences are merged into a longer one, and each input is segmented by a space.

Training Algorithm: We propose *mixSeq*, a simple yet effective data augmentation method, which generates new samples by operating on two existing samples. The algorithm is shown in Algorithm 1.

Algorithm 1: *mixSeq*

1 *Input*: $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, augmented size $\hat{N}$; sentence border label <sep>;
2 Initialize $\hat{\mathcal{D}} = \{\}$;
3 **for** $k \leftarrow 1$ *to* $\hat{N}$ **do**
4 Sample two indices i and j from $\texttt{SamplingFunc}(N)$;
5 $\tilde{x}_k = \texttt{concat}(x_i, \texttt{<sep>}, x_j)$; $\tilde{y}_k = \texttt{concat}(y_i, \texttt{<sep>}, y_j)$;
6 $\hat{\mathcal{D}} = \hat{\mathcal{D}} \cup \{(\tilde{x}_k, \tilde{y}_k)\}$;
7 **end**
8 Upsample or downsample $\mathcal{D}$ to size $\hat{N}$ and get a new dataset $\tilde{\mathcal{D}}$; train an NMT model on $\tilde{\mathcal{D}} \cup \hat{\mathcal{D}}$, which is of size $2\hat{N}$.

In *mixSeq*, the most important step is to build an augmented dataset $\hat{\mathcal{D}}$. As shown from line 3 to line 7 in Algorithm 1, we first sample two aligned sequence pairs (x_i, y_i) and (x_j, y_j) (the design of sampling rule `SamplingFunc` is left to the next part). Then we concatenate their source sentences and the target sentences respectively with a special label <sep> separating two samples, and get two longer sequences, $\tilde{x}_k$ and $\tilde{y}_k$ (line 5 in Algorithm 1). We eventually obtain the augmented dataset $\hat{\mathcal{D}}$ with size $\hat{N}$. After that, we upsample or downsample $\mathcal{D}$ to the same size as $\hat{N}$ and obtain $\tilde{\mathcal{D}}$. Finally, we train our translation models on $\tilde{\mathcal{D}} \cup \hat{\mathcal{D}}$.

Design of `SamplingFunc`: We have two forms of `SamplingFunc`, which corresponds to two variants of our algorithm:

(1) In general cases, `SamplingFunc` randomly samples i and j from $\{1, 2, \cdots, N\}$. For ease of reference, we still use *mixSeq* to denote this variant.

(2) When contextual information is available, i.e., the parallel data is extracted from a pair of aligned document, `SamplingFunc` only samples consecutive sequences in a given document. Assume x_i/y_i represent the i-th sentence in the document, then `SamplingFunc` only samples $(i, i+1)$ index pairs.We use *ctxMixSeq* to denote this variant. *ctxMixSeq* is related to context-aware machine translation (Tiedemann and Scherrer, 2017). The difference is that, during inference, *ctxMixSeq* uses a single sequence as the input, while Tiedemann and Scherrer (2017) uses multiple sequences including the contextual information.

Discussions: *mixSeq* operates on two sequences, while previous data augmentation methods like randomly dropping, swapping or masking words usually operate on a single sequence. These methods can be combined with *mixSeq* to bring further improvements (see Table 3).

3 Experiments

We conduct experiments on the following machine translation tasks to evaluate our method: IWSLT'14 German↔English and Spanish↔English; FLORES English↔Nepali and English↔Sinhala; and WMT'14 English→German. We abbreviate English, German, Spanish, Nepali and Sinhala as En, De, Es, Ne and Si.

3.1 Setup

Datasets: For IWSLT'14 De↔En, following Edunov et al. (2018), we lowercase all words, tokenize them, and apply BPE with $10k$ merge operations (Sennrich et al., 2016) to obtain of the subword representations[1]. The validation set is split from the training set and the test set is the concatenation of *tst2010, tst2011, tst2012, dev2010*

[1]Preprocessing script: https://github.com/pytorch/fairseq/blob/master/examples/translation/prepare-iwslt14.sh.

and *dev2012*. For IWSLT'14 Es↔En, the preprocessing is the same as that for De↔En without lowercasing the words. We use *tst2013* and *tst2014* as the validation and test sets respectively. For FLORES En↔Ne and En↔Si datasets, we used the BPE version of dataset provided by Guzmán et al. (2019). For WMT'14 En→De, we concatenate *newstest2012* and *newstest2013* as the validation set and use *newstest2014* as the test set. The statistics of the datasets are shown in Table 1. On all tasks, the vocabulary is shared between the source language and the target language.

Task	Training	Validation	Test
De↔En	$160k$	$7.3k$	$6.8k$
Es↔En	$184k$	$1.2k$	$1.3k$
En↔Ne	$563k$	$2.6k$	$2.8k$
En↔Si	$405k$	$2.9k$	$2.8k$
WMT	4.5M	$3k$	$3k$

Table 1: The number of sentences in the training, validation and test sets of IWSLT De↔En, Es↔En, FLORES En↔Ne, En↔Si, and WMT datasets.

Models and Training Strategy: For *mixSeq*, we set $\hat{N}$ as $5N$; for *ctxMixSeq*, we set $\hat{N}$ as N. We choose Transformer (Vaswani et al., 2017) as our translation model. For IWSLT tasks, the dimensions of the embedding, feed-forward network and number of layers of the Transformer models are 256, 1024 and 6 respectively. The dropout rate is 0.3. The batch size is 6000 tokens, and we train the models for $300k$ steps. For FLORES tasks, we use exact the same architecture and training strategy as those in (Guzmán et al., 2019) for fair comparison. The model is a 5-layer Transformer with embedding dimension and feed-forward network dimension 512 and 2048. The batch size is $16k$. The baseline model is trained for 100 epochs, while *mixSeq* is trained for 10 epochs considering our enlarged dataset is 10 times larger than the original dataset. For WMT task, the dimensions of the embedding, feed-forward network and number of layers of the Transformer models are 1024, 4096 and 6 respectively. The batch size is 4096 tokens per GPU. We train on eight V100 GPUs and accumulate the gradients for 16 times before updating. For all models, we use Adam with learning rate 5×10^{-4} and the `inverse_sqrt` learning rate scheduler to optimize the models. All models are trained until convergence.

Evaluation: We use beam search with beam width of 5 and length penalty of 1.0 to generate sequences. The generation quality is evaluated by BLEU score.

3.2 Results

The results of standard Transformer and *mixSeq* on small-scale datasets are shown in the first section of Table 2. We adopt another baseline, pair sampling (Inoue, 2018) into NMT for comparison, which can produce a synthetic dataset $\tilde{\mathcal{D}}_{ps}$ made up of pairs $(\texttt{concat}(x_1, \texttt{<sep>}, x_2), y_1)$, $(x_1, y_1) \in \mathcal{D}$, $(x_2, y_2) \in \mathcal{D}$. The results of pair sampling (briefly, PS) are in the third column of Table 2. *mixSeq* generally brings good improvements and significantly outperforms the baseline on all tasks except for two (En→De and En→Si). The pair sampling baseline performs poorly on all tasks. This is because pair sampling requires the translation model to translate the first part of the input (i.e., x_1) while ignoring the second part (i.e., x_2), which is against the goal of NMT. It is also worth noting that the time and number of steps required to converge on the augmented dataset and the original dataset are similar.

Task	Transformer	*mixSeq*	PS
En→De	29.18	29.46	29.09
De→En	34.96	35.78‡	35.22
En→Es	39.61	40.30†	38.95
Es→En	40.94	41.39†	40.80
En→Ne	4.28	5.26‡	4.20
Ne→En	7.68	8.38‡	7.51
En→Si	1.21	1.49	0.88
Si→En	6.68	7.71‡	6.02
WMT	29.15	29.61	-

Table 2: BLEU scores of IWSLT De↔En, Es↔En, FLORES En↔Ne, En↔Si, and WMT En→De. ‡ and † indicate that *mixSeq* outperforms Transformer in the significance test with $p < 0.01$ and $p < 0.05$, respectively.

We also evaluate *mixSeq* on a large-scale dataset, WMT'14 En→De, and the results are shown in the second section of Table 2. Due to resource limitation, we do not try pair sampling. Our method improves the BLEU score by 0.46, which shows that *mixSeq* is a generally effective method for NMT.

We further compare and combine our method with data augmentation methods on one sequence, including randomly dropping, masking and swapping words. We conduct experiments on IWSLT'14 De↔En. As shown in Table 3, our method brings

further improvement when combined with existing data augmentation method on a single sequence. The baseline is improved by up to 0.82 BLEU.

Method	De→En	En→ De
Transformer	34.96	29.18
mixSeq	35.78	29.46
Drop	35.30	29.03
Drop + *mixSeq*	36.01	29.22
Swap	34.52	28.73
Swap + *mixSeq*	34.73	28.98
Mask	35.78	29.49
Mask + *mixSeq*	36.63	30.00

Table 3: Comparison and combination with data augmentation on single sequences.

To verify the effectiveness of *ctxMixSeq*, we conduct experiments on IWSLT'14 En↔De, where contextual information is available. As discussed in Section 2, Tiedemann and Scherrer (2017) is similar with *ctxMixSeq*, except that it takes two sequences $\texttt{concat}(x_{t-1}, \texttt{<sep>}, x_t)$ as the input during inference. We denote this inference method as *2in* (two inputs). Another baseline proposed in Tiedemann and Scherrer (2017) is that the NMT model is trained on dataset $\mathcal{D} \cup \tilde{\mathcal{D}}_a$, where $\tilde{\mathcal{D}}_a = \{\texttt{concat}(x_{t-1}, \texttt{<sep>}, x_t), y_t\}_{t=2}^{N}$. This can be seen as a context-aware version of pair sampling and we briefly denote it as *ctxPS*. The results are in Table 4. *ctxMixSeq* outperforms all baselines proposed by Tiedemann and Scherrer (2017). Compared to *mixSeq*, *ctxMixSeq* brings consistent improvements, especially when combined with *mixSeq*.

Method	En→De	De→En
mixSeq	29.46	35.78
ctxMixSeq	29.65	35.96
ctxMixSeq + 2in	29.50	35.79
ctxPS	29.26	35.48
ctxPS + 2in	29.29	35.78
ctxMixSeq + mixSeq	29.74	36.09

Table 4: Results of context-aware versions of *mixSeq* on IWSLT'14 En↔De.

With *mixSeq*, we find that the alignment is enhanced. We visualize the source-target attention maps obtained by our method. Given $(x_i, \texttt{<sep>}, x_j)$ and the corresponding translation $(y_i, \texttt{<sep>}, y_j)$, we find that most attention weight of y_i is assigned to x_i, with little assigned to x_j. Similar phenomena is observed for y_j. In this way, the attention mechanism is enhanced, which might explain the performance improvements.

	En→Es	Es→En	En→Ne	Ne→En
mixSeq	40.3	41.4	5.26	8.28
No <sep>	38.9	41.1	4.83	8.10

Table 5: Result of *mixSeq* with/without <sep>.

3.3 Analysis

In this section, we conduct ablation study on the usage of <sep> and the effect of concatenating more than two sequences.

Ablation Study of the Usage of <sep>

To evaluate the effect of <sep> token, we remove the <sep> from sequences as another baseline. We conduct the experiments on IWSLT En↔Es and FLORES En↔Ne datasets, and report the results in Table 5. We find that our method performs poorly without <sep>, sometimes even worse than the Transformer. Our conjecture is that <sep> helps the model learn to align each part of the input to the corresponding part of the output, which can improve the representation learning.

Concatenating More Sequences

We wonder whether the BLEU scores can be further boosted by concatenating more sequences. We move a step forward by randomly concatenating three sequences, and build a synthetic dataset $\hat{\mathcal{D}}_3$ with $\hat{N}_3$ examples. Experiments are conducted on FLORES En↔{Ne, Si} datasets, and results are shown in Table 6. In the third and fourth rows, $\hat{N} = \hat{N}_3 = 5N$. In the last row, we set $\hat{N} = \hat{N}_3 = 2.5N$ to ensure the number of synthetic data remains the same.

Dataset	En→Ne	Ne→En	En→Si	Si→En
$\mathcal{D}$	4.28	7.68	1.21	6.68
$\mathcal{D} \cup \hat{\mathcal{D}}$	5.26	8.38	1.49	**7.71**
$\mathcal{D} \cup \hat{\mathcal{D}}_3$	5.39	**8.88**	2.08	7.50
$\mathcal{D} \cup \hat{\mathcal{D}} \cup \hat{\mathcal{D}}_3$	**5.43**	8.25	**2.21**	7.47

Table 6: Results of concatenating various numbers of sequences.

The results show that, although both $\mathcal{D} \cup \hat{\mathcal{D}}_3$ and $\mathcal{D} \cup \hat{\mathcal{D}} \cup \hat{\mathcal{D}}_3$ settings can bring some improvements, the improvements are not consistent among different datasets. Further work is needed on how to use

more samples for data augmentation.

4 Related Work

Most existing data augmentation methods in NMT operate on one single input. Fadaee et al. (2017) replaced common words with rare words under the guidance of language models to improve the translation of rare words. In unsupervised learning, Lample et al. (2018b) proposed to randomly drop, swap, or mask words. Gao et al. (2019) verifies the effectiveness of such methods in supervised NMT. RAML (Norouzi et al., 2016) randomly inserted, deleted or substituted words in the target sequence with probability exponentially decreasing with the edit distance. SwitchOut (Wang et al., 2018) extended RAML by both manipulating on the source side and the target side. Gao et al. (2019) proposed to "softly replace" words by replacing the one-hot representation of words with a distribution on the vocabulary. A concurrent work similar to ours is (Kondo et al., 2021), where <sep> is not leveraged. In other fields, data augmentation methods operating on multiple samples have been proposed. Mixup (Zhang et al., 2018) generated a synthetic sample by averaging two inputs and the two labels. It is further applied to semi-supervised learning to enlarge the dataset (Berthelot et al., 2019). Pair sampling (Inoue, 2018) only averaged the two inputs but not the labels.

5 Conclusion and Future Work

In this work, we proposed a simple yet effective data augmentation method for NMT, which randomly concatenates two training samples to enlarge the datasets. Experiments on nine machine translation tasks demonstrate the effectiveness of our method. For future work, there are a few directions to explore. First, we will apply our method to more NLP tasks. Second, we will theoretically analyze when and why it works. Third, we will study and design more effective data augmentation methods.

References

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. 2019. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems 32*, pages 5049–5059.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. 2019. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 113–123.

Terrance DeVries and Graham W Taylor. 2017. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*.

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Classical structured prediction losses for sequence to sequence learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 355–364, New Orleans, Louisiana. Association for Computational Linguistics.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. *arXiv preprint arXiv:1705.00440*.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. Soft contextual data augmentation for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5539–5544.

Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. 2019. The flores evaluation datasets for low-resource machine translation: Nepali–english and sinhala–english. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6100–6113.

Hiroshi Inoue. 2018. Data augmentation by pairing samples for images classification. *CoRR*, abs/1801.02929.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1681–1691, Beijing, China. Association for Computational Linguistics.

Seiichiro Kondo, Kengo Hotate, Masahiro Kaneko, and Mamoru Komachi. 2021. Sentence concatenation approach to data augmentation for neural machine translation. *arXiv preprint arXiv:2104.08478*.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018a. Unsupervised machine translation using monolingual corpora only. *ICLR*.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018b. Phrase-based & neural unsupervised machine translation. *EMNLP*.

Mohammad Norouzi, Samy Bengio, Navdeep Jaitly, Mike Schuster, Yonghui Wu, Dale Schuurmans, et al. 2016. Reward augmented maximum likelihood for neural structured prediction. In *Advances In Neural Information Processing Systems*, pages 1723–1731.

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. Association for Computational Linguistics.

Jörg Tiedemann and Yves Scherrer. 2017. Neural machine translation with extended context. In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 82–92, Copenhagen, Denmark. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018. Switchout: an efficient data augmentation algorithm for neural machine translation. *arXiv preprint arXiv:1808.07512*.

Ziang Xie, Sida I Wang, Jiwei Li, Daniel Lévy, Aiming Nie, Dan Jurafsky, and Andrew Y Ng. 2017. Data noising as smoothing in neural network language models. *ICLR*.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. *ICLR*.

Hongyi Zhang, Yann N Dauphin, and Tengyu Ma. 2019. Fixup initialization: Residual learning without normalization. *ICLR*.