Bi-SimCut: A Simple Strategy for Boosting Neural Machine Translation

Pengzhi Gao, Zhongjun He, Hua Wu, and Haifeng Wang
Baidu Inc. No. 10, Shangdi 10th Street, Beijing, 100085, China
{gaopengzhi,hezhongjun,wu_hua,wanghaifeng}@baidu.com

Abstract

We introduce Bi-SimCut: a simple but effective training strategy to boost neural machine translation (NMT) performance. It consists of two procedures: bidirectional pretraining and unidirectional finetuning. Both procedures utilize SimCut, a simple regularization method that forces the consistency between the output distributions of the original and the cutoff sentence pairs. Without leveraging extra dataset via back-translation or integrating large-scale pretrained model, Bi-SimCut achieves strong translation performance across five translation benchmarks (data sizes range from 160K to 20.2M): BLEU scores of 31.16 for en → de and 38.37 for de → en on the IWSLT14 dataset, 30.78 for en → de and 35.15 for de → en on the WMT14 dataset, and 27.17 for zh → en on the WMT17 dataset. SimCut is not a new method, but a version of Cutoff (Shen et al., 2020) simplified and adapted for NMT, and it could be considered as a perturbation-based method. Given the universality and simplicity of SimCut and Bi-SimCut, we believe they can serve as strong baselines for future NMT research.

1 Introduction

The state of the art in machine translation has been dramatically improved over the past decade thanks to the neural machine translation (NMT) (Wu et al., 2016), and Transformer-based models (Vaswani et al., 2017) often deliver state-of-the-art (SOTA) translation performance with large-scale corpora (Ott et al., 2018). Along with the development in the NMT field, consistency training (Bachman et al., 2014) has been widely adopted and shown great promise to improve NMT performance. It simply regularizes the NMT model predictions to be invariant to either small perturbations applied to the inputs (Sato et al., 2019; Shen et al., 2020) and hidden states (Chen et al., 2021) or the model randomness and variance existed in the training procedure (Liang et al., 2021).

Specifically, Shen et al. (2020) introduce a set of cutoff data augmentation methods and utilize Jensen-Shannon (JS) divergence loss to force the consistency between the output distributions of the original and the cutoff augmented samples in the training procedure. Despite its impressive performance, finding the proper values for the four additional hyper-parameters introduced in cutoff augmentation seems to be tedious and time-consuming if there are limited resources available, which hinders its practical value in the NMT field.

In this paper, our main goal is to provide a simple, easy-to-reproduce, but tough-to-beat strategy for training NMT models. Inspired by cutoff augmentation (Shen et al., 2020) and virtual adversarial regularization (Sato et al., 2019) for NMT, we firstly introduce a simple yet effective regularization method named SimCut. Technically, SimCut is not a new method and can be viewed as a simplified version of Token Cutoff proposed in Shen et al. (2020). We show that bidirectional backpropagation in Kullback-Leibler (KL) regularization plays a key role in improving NMT performance. We also regard SimCut as a perturbation-based method and discuss its robustness to the noisy inputs. At last, motivated by bidirectional training (Ding et al., 2021) in NMT, we present Bi-SimCut, a two-stage training strategy consisting of bidirectional pretraining and unidirectional finetuning equipped with SimCut regularization.

The contributions of this paper can be summarized as follows:

- We propose a simple but effective regularization method, SimCut, for improving the generalization of NMT models. SimCut could be regarded as a perturbation-based method and serves as a strong baseline for the approaches of robustness. We also show the compatibility of SimCut with the pretrained language models such as mBART (Liu et al., 2020).

Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3938 - 3948
July 10-15, 2022 ©2022 Association for Computational Linguistics

- We propose Bi-SimCut, a training strategy for NMT that consists of bidirectional pretraining and unidirectional finetuning with SimCut regularization.

- Our experimental results show that NMT training with Bi-SimCut achieves significant improvements over the Transformer model on five translation benchmarks (data sizes range from 160K to 20.2M), and outperforms the current SOTA method BiBERT (Xu et al., 2021) on several benchmarks.

2 Background

2.1 Neural Machine Translation

The NMT model refers to a neural network with an encoder-decoder architecture, which receives a sentence as input and returns a corresponding translated sentence as output. Assume $\mathbf{x} = x_1, ..., x_I$ and $\mathbf{y} = y_1, ..., y_J$ that correspond to the source and target sentences with lengths I and J respectively. Note that y_J denotes the special end-of-sentence symbol $\langle eos \rangle$. The encoder first maps a source sentence $\mathbf{x}$ into a sequence of word embeddings $e(\mathbf{x}) = e(x_1), ..., e(x_I)$, where $e(\mathbf{x}) \in \mathbb{R}^{d \times I}$, and d is the embedding dimension. The word embeddings are then encoded to the corresponding hidden representations $\mathbf{h}$. Similarly, the decoder maps a shifted copy of the target sentence $\mathbf{y}$, i.e., $\langle bos \rangle, y_1, ..., y_{J-1}$, into a sequence of word embeddings $e(\mathbf{y}) = e(\langle bos \rangle), e(y_1), ..., e(y_{J-1})$, where $\langle bos \rangle$ denotes a special beginning-of-sentence symbol, and $e(\mathbf{y}) \in \mathbb{R}^{d \times J}$. The decoder then acts as a conditional language model that operates on the word embeddings $e(\mathbf{y})$ and the hidden representations $\mathbf{h}$ learned by the encoder.

Given a parallel corpus $\mathcal{S} = \{\mathbf{x}^i, \mathbf{y}^i\}_{i=1}^{|\mathcal{S}|}$, the standard training objective is to minimize the empirical risk:

$$\mathcal{L}_{ce}(\theta) = \mathop{\mathbb{E}}_{(\mathbf{x},\mathbf{y}) \in \mathcal{S}} [\ell(f(\mathbf{x}, \mathbf{y}; \theta), \ddot{\mathbf{y}})], \quad (1)$$

where ℓ denotes the cross-entropy loss, θ is a set of model parameters, $f(\mathbf{x}, \mathbf{y}; \theta)$ is a sequence of probability predictions, i.e.,

$$f_j(\mathbf{x}, \mathbf{y}; \theta) = P(y|\mathbf{x}, \mathbf{y}_{<j}; \theta), \quad (2)$$

and $\ddot{\mathbf{y}}$ is a sequence of one-hot label vectors for $\mathbf{y}$.

2.2 Cutoff Augmentation

Shen et al. (2020) introduce a set of cutoff methods which augments the training by creating the partial views of the original sentence pairs and propose Token Cutoff for the machine translation task. Given a sentence pair $(\mathbf{x}, \mathbf{y})$, N cutoff samples $\{\mathbf{x}^i_{\text{cut}}, \mathbf{y}^i_{\text{cut}}\}_{i=1}^N$ are constructed by randomly setting the word embeddings of $x_1, ..., x_I$ and $y_1, ..., y_J$ to be zero with a cutoff probability p_{cut}. For each sentence pair, the training objective of Token Cutoff is then defined as:

$$\mathcal{L}_{tokcut}(\theta) = \mathcal{L}_{ce}(\theta) + \alpha \mathcal{L}_{cut}(\theta) + \beta \mathcal{L}_{kl}(\theta), \quad (3)$$

where

$$\mathcal{L}_{ce}(\theta) = \ell(f(\mathbf{x}, \mathbf{y}; \theta), \ddot{\mathbf{y}}), \quad (4)$$

$$\mathcal{L}_{cut}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \ell(f(\mathbf{x}^i_{\text{cut}}, \mathbf{y}^i_{\text{cut}}; \theta), \ddot{\mathbf{y}}), \quad (5)$$

$$\begin{aligned} \mathcal{L}_{kl}(\theta) = & \frac{1}{N+1} \{ \sum_{i=1}^{N} \text{KL}(f(\mathbf{x}^i_{\text{cut}}, \mathbf{y}^i_{\text{cut}}; \theta) \| p_{\text{avg}}) \\ & + \text{KL}(f(\mathbf{x}, \mathbf{y}; \theta) \| p_{\text{avg}}) \}, \end{aligned} \quad (6)$$

$$\begin{aligned} p_{\text{avg}} = & \frac{1}{N+1} \{ \sum_{i=1}^{N} f(\mathbf{x}^i_{\text{cut}}, \mathbf{y}^i_{\text{cut}}; \theta) \\ & + f(\mathbf{x}, \mathbf{y}; \theta) \}, \end{aligned} \quad (7)$$

in which $\text{KL}(\cdot \| \cdot)$ denotes the Kullback-Leibler (KL) divergence of two distributions, and α and β are the scalar hyper-parameters that balance $\mathcal{L}_{ce}(\theta)$, $\mathcal{L}_{cut}(\theta)$ and $\mathcal{L}_{kl}(\theta)$.

3 Datasets and Baseline Settings

In this section, we describe the datasets used in experiments as well as the model configurations. For fair comparisons, we keep our experimental settings consistent with previous works.

Datasets We initially consider a low-resource (IWSLT14 en↔de) scenario and then show further experiments in standard (WMT14 en↔de) and high (WMT17 zh→en) resource scenarios in Sections 5 and 6. The detailed information of the datasets are summarized in Table 1. We here conduct experiments on the IWSLT14 English-German dataset[1], which has 160K parallel bilingual

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

	IWSLT	WMT	
	en↔de	en↔de	zh→en
train	160239	4468840	20184941
valid	7283	6003	2002
test	6750	3003	2001

Table 1: Number of sentence pairs used in our machine translation experiments.

sentence pairs. Following the common practice, we lowercase all words in the dataset. We build a shared dictionary with 10K byte-pair-encoding (BPE) (Sennrich et al., 2016) types.

Settings We implement our approach on top of the Transformer (Vaswani et al., 2017). We apply a Transformer with 6 encoder and decoder layers, 4 attention heads, embedding size 512, and FFN layer dimension 1024. We apply cross-entropy loss with label smoothing rate 0.1 and set max tokens per batch to be 4096. We use Adam optimizer with Beta $(0.9, 0.98)$, 4000 warmup updates, and inverse square root learning rate scheduler with initial learning rates $5e^{-4}$. We use dropout rate 0.3 and beam search decoding with beam size 5 and length penalty 1.0. We apply the same training configurations in both pretraining and finetuning stages which will be discussed in the following sections. We use `multi-bleu.pl`[2] for BLEU (Papineni et al., 2002) evaluation. We train all models until convergence on a single NVIDIA Tesla V100 GPU. All reported BLEU scores are from a single model. For all the experiments below, we select the saved model state with the best validation performance.

4 Bi-SimCut

In this section, we formally propose Bidirectional Pretrain and Unidirectional Finetune with Simple Cutoff Regularization (Bi-SimCut), a simple but effective training strategy that can greatly enhance the generalization of the NMT model. Bi-SimCut consists of a simple cutoff regularization and a two-phase pretraining and finetuning strategy. We introduce the details of each part below.

4.1 SimCut: A Simple Cutoff Regularization for NMT

Despite the impressive performance reported in Shen et al. (2020), finding the proper hyper-parameters $(p_{\text{cut}}, \alpha, \beta, N)$ in Token Cutoff seems to be tedious and time-consuming if there are limited resources available, which hinders its practical value in the NMT community. To reduce the burden in hyper-parameter searching, we propose SimCut, a simple regularization method that forces the consistency between the output distributions of the original sentence pairs and the cutoff samples.

Our problem formulation is motivated by Virtual Adversarial Training (VAT), where Sato et al. (2019) introduces a KL-based adversarial regularization that forces the output distribution of the samples with adversarial perturbations $\boldsymbol{\delta}_{\mathbf{x}} \in \mathbb{R}^{d\times I}$ and $\boldsymbol{\delta}_{\mathbf{y}} \in \mathbb{R}^{d\times J}$ to be consistent with that of the original samples:

$$\mathrm{KL}(f(e(\mathbf{x}), e(\mathbf{y}); \theta) \| f(e(\mathbf{x}) + \boldsymbol{\delta}_{\mathbf{x}}, e(\mathbf{y}) + \boldsymbol{\delta}_{\mathbf{y}}; \theta)).$$

Instead of generating perturbed samples by gradient-based adversarial methods, for each sentence pair $(\mathbf{x}, \mathbf{y})$, we only generate one cutoff sample $(\mathbf{x}_{\text{cut}}, \mathbf{y}_{\text{cut}})$ by following the same cutoff strategy used in Token Cutoff. For each sentence pair, the training objective of SimCut is defined as:

$$\mathcal{L}_{simcut}(\theta) = \mathcal{L}_{ce}(\theta) + \alpha \mathcal{L}_{simkl}(\theta), \qquad (8)$$

where

$$\mathcal{L}_{simkl}(\theta) = \mathrm{KL}(f(\mathbf{x}, \mathbf{y}; \theta) \| f(\mathbf{x}_{\text{cut}}, \mathbf{y}_{\text{cut}}; \theta)).$$

There are only two hyper-parameters α and p_{cut} in SimCut, which greatly simplifies the hyper-parameter searching step in Token Cutoff. Note that VAT only allows the gradient to be backpropagated through the right-hand side of the KL divergence term, while the gradient is designed to be backpropagated through both sides of the KL regularization in SimCut. We can see that the constraints introduced by $\mathcal{L}_{cut}(\theta)$ and $\mathcal{L}_{kl}(\theta)$ in (3) still implicitly hold in (8):

- $\mathcal{L}_{cut}(\theta)$ in Token Cutoff is designed to guarantee that the output of the cutoff sample should close to the ground-truth to some extent. In SimCut, $\mathcal{L}_{ce}(\theta)$ requires the outputs of the original sample close to the ground-truth, and $\mathcal{L}_{simkl}(\theta)$ requires the output distributions of the cutoff sample close to that of the original sample. The constraint introduced by $\mathcal{L}_{cut}(\theta)$ then implicitly holds.

- $\mathcal{L}_{kl}(\theta)$ in Token Cutoff is designed to guarantee that the output distributions of the original sample and N different cutoff samples

[2] https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl

Method	en→de	de→en
Transformer	28.70	34.99
VAT	29.45	35.52
R-Drop	30.73	37.30
Token Cutoff	30.89	37.61
SimCut	**30.98**	**37.81**

Table 2: SimCut achieves the superior or comparable performance on IWSLT14 en ↔ de translation tasks over the strong baselines such as VAT, R-Drop, and Token Cutoff.

should be consistent with each other. In SimCut, $\mathcal{L}_{simkl}(\theta)$ guarantees the consistency between the output distributions of the original and cutoff samples. Even though SimCut only generates one cutoff sample at each time, different cutoff samples of the same sentence pair will be considered in different training epochs. Such constraint raised by $\mathcal{L}_{kl}(\theta)$ still implicitly holds.

4.2 Analysis on SimCut

4.2.1 How Does the Simplification Affect Performance?

We here investigate whether our simplification on Token Cutoff hurts its performance on machine translation tasks. We compare SimCut with Token Cutoff, VAT, and R-Drop (Liang et al., 2021), a strong regularization baseline that forces the output distributions of different sub-models generated by dropout to be consistent with each other. Table 2 shows that SimCut achieves superior or comparable performance over VAT, R-Drop, and Token Cutoff, which clearly shows the effectiveness of our method. To further compare SimCut with other strong baselines in terms of training cost, we summarize the validation BLEU score along the training time on IWSLT14 de→en translation task in Table 3. From the table, we can see that the BLEU score of SimCut continuously increases in the first 1500 minutes. The results on VAT are consistent with the previous studies on adversarial overfitting, i.e., virtual adversarial training easily suffering from overfitting (Rice et al., 2020). Though SimCut needs more training time to converge, the final NMT model is much better than the baseline. For the detailed training cost for each epoch, Token Cutoff costs about 148 seconds per epoch, while SimCut costs about 128 seconds per epoch. Note that the training cost of Token Cutoff is greatly influenced by the hyper-parameter N. We set N to be 1 in our experiments. With the increasing of N, the training time of Token Cutoff will be much longer. Due to the tedious and time-consuming hyper-parameter searching in Token Cutoff, we will not include its results in the following sections and show the results of SimCut directly.

4.2.2 How Does the Bidirectional Backpropagation Affect Performance?

Even though the problem formulation of SimCut is similar to that of VAT, one key difference is that the gradients are allowed to be backpropagated bidirectionally in the KL regularization in SimCut. We here investigate the impact of the bidirectional backpropagation in the regularization term on the NMT performance. Table 4 shows the translation results of VAT and SimCut with or without bidirectional backpropagation. We can see that both VAT and SimCut benefit from the bidirectional gradient backpropagation in the KL regularization.

4.2.3 Performance on Perturbed Inputs

Given the similar problem formulations of VAT and SimCut, it is natural to regard cutoff operation as a special perturbation and consider SimCut as a perturbation-based method. We here investigate the robustness of NMT models on the perturbed inputs. As discussed in Takase and Kiyono (2021), simple techniques such as word replacement and word drop can achieve comparable performance to sophisticated perturbations. We hence include them as baselines to show the effectiveness of our method as follows:

- **UniRep**: Word replacement approach constructs a new sequence whose tokens are randomly replaced with sampled tokens. For each token in the source sentence $\mathbf{x}$, we sample $\hat{x}_i$ uniformly from the source vocabulary, and use it for the new sequence $\mathbf{x}'$ with probability $1 - p'$:

$$x'_i = \begin{cases} x_i, & \text{with probability } p', \\ \hat{x}_i, & \text{with probability } 1 - p'. \end{cases} \quad (9)$$

We construct $\mathbf{y}'$ from the target sentence $\mathbf{y}$ in the same manner. Following the curriculum learning strategy used in Bengio et al. (2015), we adjust p' with the inverse sigmoid decay:

$$p'_t = \max(q, \frac{k}{k + \exp{(\frac{t}{k})}}), \quad (10)$$

where q and k are hyper-parameters. p'_t decreases to q from 1, depending on the training epoch num-

Minutes	10	30	60	90	150	300	600	900	1200	1500
Transformer	11.51	31.20	34.19	34.88	**35.17**	34.86	34.43	34.28	34.23	33.95
VAT	1.87	20.08	31.69	33.95	35.41	35.78	**35.81**	35.63	35.17	34.99
R-Drop	2.11	26.32	32.81	34.25	35.88	36.91	37.18	37.43	**37.52**	37.43
Token Cutoff	2.16	28.88	32.82	34.61	35.90	36.84	37.70	37.81	**37.93**	37.83
SimCut	1.99	25.12	32.21	33.66	34.93	36.37	37.31	37.62	37.89	**38.10**

Table 3: On the IWSLT14 de→en validation set, the BLEU score increases over time in model training using SimCut. In contrast, the BLEU scores of the other strong baselines all stop increasing before 1500 minutes. The results suggest that the use of SimCut can effectively alleviate the model training from overfitting.

Method	en→de	de→en
VAT	29.45	35.52
+ Bi-backpropagation	29.69	36.26
SimCut	30.98	37.81
- Bi-backpropagation	30.29	36.91

Table 4: Bidirectional backpropagation achieves better performance on IWSLT14 en ↔ de translation tasks compared with unidirectional backpropagation in the KL regularization.

Method	probability			
	0.00	0.01	0.05	0.10
Transformer	34.99	34.01	30.38	25.70
UniRep	35.67	34.91	31.54	27.24
WordDrop	35.65	34.73	31.22	26.46
VAT	35.52	34.65	30.48	25.44
R-Drop	37.30	36.24	32.27	27.19
SimCut	**37.81**	**36.94**	**33.16**	**27.93**

Table 5: The model trained by SimCut achieves high robustness on the perturbed test set and high performance on the clean test set. Entries represent BLEU scores on IWSLT14 de→en test set when we inject perturbations to source sentences with different probability.

ber t. We use p'_t as p' in epoch t. We set q and k to be 0.9 and 25 respectively in the experiments.

- **WordDrop**: Word drop randomly applies the zero vector instead of the word embedding $e(x_i)$ or $e(y_i)$ for the input token x_i or y_i (Gal and Ghahramani, 2016). For each token in both source and target sentences, we keep the original embedding with the probability β and set it to be the zero vector otherwise. We set β to be 0.9 in the experiments.

We construct noisy inputs by randomly replacing words in the source sentences based on a predefined probability. If the probability is 0.0, we use the original source sentence. If the probability is 1.0, we use completely different sentences as source sentences. We set the probability to be 0.00, 0.01, 0.05, and 0.10 in our experiments. We randomly replace each word in the source sentence with a word uniformly sampled from the vocabulary. We apply this procedure to IWSLT14 de→en test set. Table 5 shows the BLEU scores of each method on the perturbed test set. Note that the BLEU scores are calculated against the original reference sentences. We can see that all methods improve the robustness of the NMT model, and SimCut achieves the best performance among all the methods on both the clean and perturbed test sets. The performance results indicate that SimCut could be considered as a strong baseline for the perturbation-based method for the NMT model.

As shown in Table 6, the baseline model completely ignores the translation of “in spielen (in games)” due to the replacement of “denken (think)” with “festgelegten (determined)” in the source sentence. In contrast, our model successfully captures the translation of “in spielen” under the noisy input. This result shows that our model is more robust to small perturbations in an authentic context.

4.2.4 Effects of α and p_{cut}

We here investigate the impact of the scalar hyperparameters α and p_{cut} in SimCut. α is a penalty parameter that controls the regularization strength in our optimization problem. p_{cut} controls the percentage of the cutoff perturbations in SimCut. We here vary α and p_{cut} in $\{1, 2, 3, 4, 5\}$ and $\{0.00, 0.05, 0.10, 0.15, 0.20\}$ respectively and conduct the experiments on the IWSLT14 de→en dataset. Note that SimCut is simplified to R-Drop approximately when $p_{\text{cut}} = 0.00$. The test BLEU scores are reported in Figure 1. By checking model performance under different combinations of α and p_{cut}, we have the following observations: 1) A too small α (e.g., 1) cannot achieve as good performance as larger α (e.g., 3), indicating a certain de-

Input	wir denken (festgelegten), dass wir in der realität nicht so gut sind wie in spielen.
Reference	we feel that we are not as good in reality as we are in games.
Vaswani et al. (2017) on Input	we think we’re not as good in reality as we are in games.
on Noisy Input	we realized that we weren’t as good as we were in real life.
SimCut on Input	we think in reality, we’re not as good as we do in games.
on Noisy Input	we realized that we’re not as good in reality as we are in games.

Table 6: SimCut is more robust to small perturbations in an authentic context. SimCut captures the translation of “in spielen” under the noisy input while the vanilla Transformer ignores the translation of “in spielen” due to the replacement of “denken” with “festgelegten”.

gree of regularization strength during NMT model training is conducive to generalization. Meanwhile, an overwhelming regularization ($\alpha = 5$) is not plausible for learning NMT models. 2) When $\alpha = 3$, the best performance is achieved when $p_{\text{cut}} = 0.05$, and $p_{\text{cut}} = 0.00$ performs sub-optimal among all selected probabilities. Such an observation demonstrates that the cutoff perturbation in SimCut can effectively promote the generalization compared with R-Drop.

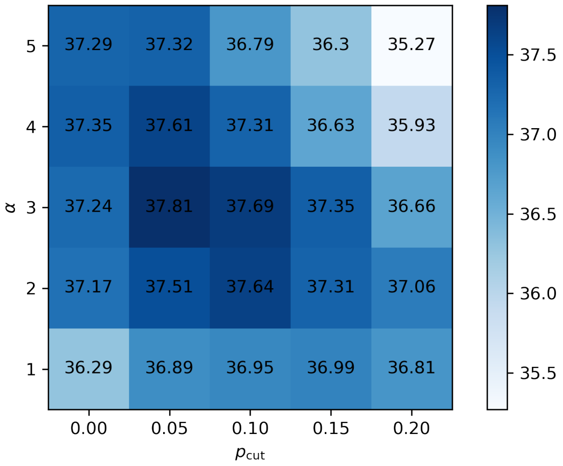

Figure 1: BLEU scores with different α and p_{cut} on IWSLT14 de→en dataset.

Method	de→en
Transformer	32.4
mBART	38.5
mBART with SimCut	**39.3**

Table 7: SimCut achieves better performance on IWSLT14 de→en translation task compared with the standard finetuning approach based on mBART.

4.2.5 Is SimCut Compatible with the Pretrained Language Model?

The multilingual sequence-to-sequence pretrained language models (Song et al., 2019; Liu et al., 2020; Xue et al., 2021) have shown impressive performance on machine translation tasks, where the pretrained models generally learn the knowledge from the large-scale monolingual data. It is interesting to investigate whether SimCut can gain performance improvement based on the pretrained language model. We adopt mBART (Liu et al., 2020) as the backbone model, which is a sequence-to-sequence denoising auto-encoder pretrained on CC25 Corpus[3]. We conduct experiments on IWSLT14 de→en dataset and only remove the duplicated sentence pairs following mBART50 (Tang et al., 2021) in the data preprocessing step. The source and target sentences are jointly tokenized into sub-word units with the 250K SentencePiece (Kudo and Richardson, 2018) vocabulary of mBART. We use case-sensitive sacreBLEU (Post, 2018) to evaluate the translation quality, and the methods applied in the experiments are as follows:

- Transformer: The Transformer model is randomly initialized and trained from scratch. We utilize the same model and training configurations discussed in Section 3.
- mBART: The Transformer model is directly finetuned from mBART. We utilize the default training configurations of mBART.
- mBART with SimCut: The Transformer model is finetuned from mBART with SimCut regularization. We utilize the default training configurations of mBART.

From Table 7 we can see that SimCut could further improve the translation performance of mBART,

[3] https://github.com/pytorch/fairseq/tree/main/examples/mbart

Method	en→de	de→en
Transformer	28.70	34.99
Bi-Pretrain	28.94	35.64
+ Finetune	28.82	35.66
Bi-R-Drop Pretrain	30.30	37.01
+ R-Drop Finetune	30.85	37.55
Bi-SimCut Pretrain	30.57	37.70
+ SimCut Finetune	**31.16**	**38.37**

Table 8: Bidirectional pretrain and unidirectional finetune results on IWSLT14 en ↔ de datasets. Note that the results of bidirectional pretrain are from one model for dual-directional translations.

Method	en→de	de→en	Average
Transformer	28.70	34.99	31.85
VAT	29.45	35.52	32.49
Mixed Rep†	29.93	36.41	33.17
UniDrop†	29.99	36.88	33.44
R-Drop	30.73	37.30	34.02
BiBERT†	30.45	**38.61**	34.53
Bi-SimCut	**31.16**	38.37	**34.77**

Table 9: Our method achieves the superior performance over the existing methods on the IWSLT14 en↔de translation benchmark. † denotes the numbers are reported from the corresponding papers, others are based on our runs.

which again shows the effectiveness and universality of our method.

4.3 Training Strategy: Bidirectional Pretrain and Unidirectional Finetune

Bidirectional pretraining is shown to be very effective to improve the translation performance of the unidirectional NMT system (Ding et al., 2021; Xu et al., 2021). The main idea is to pretrain a bidirectional NMT model at first and use it as the initialization to finetune a unidirectional NMT model. Assume we want to train an NMT model for "English→German", we first reconstruct the training sentence pairs to "English+German→German+English", where the training dataset is doubled. We then firstly train a bidirectional NMT model with the new training sentence pairs:

$$\mathop{\mathbb{E}}_{(\mathbf{x},\mathbf{y})\in\mathcal{S}}[\ell(f(\mathbf{x},\mathbf{y};\theta),\ddot{\mathbf{y}})+\ell(f(\mathbf{y},\mathbf{x};\theta),\ddot{\mathbf{x}})], \quad (11)$$

and finetune the model with "English→German" direction. We follow the same training strategy in Ding et al. (2021) and apply SimCut regularization to both pretraining and finetuning procedures. Table 8 shows that bidirectional pretraining and unidirectional finetuning strategy with SimCut regularization could achieve superior performance compared with strong baseline such as R-Drop.

Comparison with Existing Methods We summarize the recent results of several existing works on IWSLT14 en↔de benchmark in Table 9. The existing methods vary from different aspects, including Virtual Adversarial Training (Sato et al., 2019), Mixed Tokenization for NMT (Wu et al., 2020), Unified Dropout for the Transformer model (Wu et al., 2021), Regularized Dropout (Liang et al., 2021), and BiBERT (Xu et al., 2021). We can see that our approach achieves an improvement of 2.92 BLEU score over Vaswani et al. (2017) and surpass the current SOTA method BiBERT that incorporates large-scale pretrained model, stochastic layer selection, and bidirectional pretraining. Given the simplicity of Bi-SimCut, we believe it could be considered as a strong baseline for the NMT task.

5 Standard Resource Scenario

We here investigate the performance of Bi-SimCut on the larger translation benchmark compared with the IWSLT14 benchmark.

5.1 Dataset Description and Model Configuration

For the standard resource scenario, we evaluate NMT models on the WMT14 English-German dataset, which contains 4.5M parallel sentence pairs. We combine newstest2012 and newstest2013 as the validation set and use newstest2014 as the test set. We collect the pre-processed data from Xu et al. (2021)'s release[4], where a shared dictionary with 52K BPE types is built. We apply a standard Transformer Big model with 6 encoder and decoder layers, 16 attention heads, embedding size 1024, and FFN layer dimension 4096. We apply cross-entropy loss with label smoothing rate 0.1 and set max tokens per batch to be 4096. We use Adam optimizer with Beta $(0.9, 0.98)$, 4000 warmup updates, and inverse square root learning rate scheduler with initial learning rates $1e^{-3}$. We decrease the learning rate to $5e^{-4}$ in the finetuning stage. We select the dropout rate from 0.3, 0.2, and 0.1 based on the validation performance. We

[4] https://github.com/fe1ixxu/BiBERT

Method	en→de	de→en	Average
Transformer + Large Batch† (Ott et al., 2018)	29.30	-	-
Evolved Transformer† (So et al., 2019)	29.80	-	-
BERT Initialization (12 layers)† (Rothe et al., 2020)	30.60	33.60	32.10
BERT-Fuse† (Zhu et al., 2020)	30.75	-	-
R-Drop (Liang et al., 2021)	30.13	34.54	32.34
BiBERT† (Xu et al., 2021)	**31.26**	34.94	**33.10**
SimCut	30.56	34.86	32.71
Bi-SimCut Pretrain	30.10	34.42	32.26
+ SimCut Finetune	30.78	**35.15**	32.97

Table 10: Our method achieves the superior or comparable performance over the existing methods on the WMT14 en↔de translation benchmark. † denotes the numbers are reported from Xu et al. (2021), others are based on our runs.

use beam search decoding with beam size 4 and length penalty 0.6. We train all models until convergence on 8 NVIDIA Tesla V100 GPUs. All reported BLEU scores are from a single model.

5.2 Results

We report test BLEU scores of all comparison methods and our approach on the WMT14 dataset in Table 10. With Bi-SimCut bidirectional pretraining and unidirectional finetuning procedures, our NMT model achieves strong or SOTA BLEU scores on en→de and de→en translation benchmarks. During the NMT training process, we fix p_{cut} to be 0.05 and tune the hyper-parameter α in both R-Drop and SimCut based on the performance on the validation set. Note that the BLEU scores of R-Drop are lower than that reported in Liang et al. (2021). Such gap might be due to the different prepossessing steps used in Liang et al. (2021) and Xu et al. (2021). It is worth mentioning that Bi-SimCut outperforms BiBERT on de→en direction even though BiBERT incorporates bidirectional pretraining, large-scale pretrained contextualized embeddings, and stochastic layer selection mechanism.

6 High Resource Scenario

To investigate the performance of Bi-SimCut on the distant language pairs which naturally do not share dictionaries, we here discuss the effectiveness of Bi-SimCut on the Chinese-English translation task.

6.1 Dataset Description and Model Configuration

For the high resource scenario, we evaluate NMT models on the WMT17 Chinese-English dataset, which consists of 20.2M training sentence pairs, and we use newsdev2017 as the validation set and newstest2017 as the test set. We firstly build the source and target vocabularies with 32K BPE types separately and treat them as separated or joined dictionaries in our experiments. We apply the same Transformer Big model and training configurations used in the WMT14 experiments. We use beam search decoding with beam size 5 and length penalty 1. We train all models until convergence on 8 NVIDIA Tesla V100 GPUs. All reported BLEU scores are from a single model.

Method	share	zh→en
Transformer	x	25.53
Transformer	✓	25.31
SimCut	x	26.86
SimCut	✓	26.74
Bi-SimCut Pretrain	✓	26.13
+ SimCut Finetune	✓	**27.17**

Table 11: Our method achieves strong performance on the WMT17 zh→en translation benchmark. share denotes whether a shared dictionary is applied.

6.2 Results

We report test BLEU scores of the baselines and our approach on the WMT17 dataset in Table 11. Note that share means the embedding matrices for encoder input, decoder input and decoder output are all shared. The NMT models with separated dictionaries perform slightly better than those with the shared dictionary. We can see that our approach significantly improves the translation performance. In particular, Bi-SimCut achieves more than 1.6 BLEU score improvement over Vaswani et al. (2017), showing the effectiveness and universality of our approach on the distant language pair in the NMT task.

7 Related Work

Adversarial Perturbation It is well known that neural networks are sensitive to noisy inputs, and adversarial perturbations are firstly discussed in the filed of image processing (Szegedy et al., 2014; Goodfellow et al., 2015). SimCut could be regarded as a perturbation-based method for the robustness research. In the field of natural language processing, Miyato et al. (2017) consider adversarial perturbations in the embedding space and show its effectiveness on the text classification tasks. For the NMT tasks, Sato et al. (2019) and Wang et al. (2019) apply adversarial perturbations in the embedding space during training of the encoder-decoder NMT model. Cheng et al. (2019) leverage adversarial perturbations and generate adversarial examples by replacing words in both source and target sentences. They introduce two additional language models for both sides and a candidate word selection mechanism for replacing words in the sentence pairs. Takase and Kiyono (2021) compare perturbations for the NMT model in view of computational time and show that simple perturbations are sufficiently effective compared with complicated adversarial perturbations.

Consistency Training Besides perturbation-based methods, our approach also highly relates to a few works of model-level and data-level consistency training in the NMT field. Among them, the most representative methods are R-Drop (Liang et al., 2021) and Cutoff (Shen et al., 2020). R-Drop studies the intrinsic randomness in the NMT model and regularizes the NMT model by utilizing the output consistency between two dropout sub-models with the same inputs. Cutoff considers consistency training from a data perspective by regularizing the inconsistency between the original sentence pair and the augmented samples with part of the information within the input sentence pair being dropped. Note that Cutoff takes the dropout sub-models into account during the training procedure as well. We want to emphasize that SimCut is not a new method, but a version of Cutoff simplified and adapted for the NMT tasks.

8 Conclusion

In this paper, we propose Bi-SimCut: a simple but effective two-stage training strategy to improve NMT performance. Bi-SimCut consists of bidirectional pretraining and unidirectional finetuning procedures equipped with SimCut regularization for improving the generality of the NMT model. Experiments on low (IWSLT14 `en`↔`de`), standard (WMT14 `en`↔`de`), and high (WMT17 `zh`→`en`) resource translation benchmarks demonstrate Bi-SimCut and SimCut's capabilities to improve translation performance and robustness. Given the universality and simplicity of Bi-SimCut and SimCut, we believe: 1) SimCut could be regarded as a perturbation-based method, and it could be used as a strong baseline for the robustness research. 2) Bi-SimCut outperforms many complicated methods which incorporate large-scaled pretrained models or sophisticated mechanisms, and it could be used as a strong baseline for future NMT research. We hope researchers of perturbations and NMT could use SimCut and Bi-SimCut as strong baselines to make the usefulness and effectiveness of their proposed methods clear. For future work, we will explore the effectiveness of SimCut and Bi-SimCut on more sequence learning tasks, such as multilingual machine translation, domain adaptation, text classification, natural language understanding, etc.

Acknowledgements

We would like to thank the anonymous reviewers for their insightful comments.

References

Philip Bachman, Ouais Alsharif, and Doina Precup. 2014. Learning with pseudo-ensembles. In Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada, pages 3365–3373.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada, pages 1171–1179.

Guandan Chen, Kai Fan, Kaibo Zhang, Boxing Chen, and Zhongqiang Huang. 2021. Manifold adversarial augmentation for neural machine translation. In Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 3184–3189, Online. Association for Computational Linguistics.

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4324–4333, Florence, Italy. Association for Computational Linguistics.

Liang Ding, Di Wu, and Dacheng Tao. 2021. Improving neural machine translation by bidirectional training. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 3278–3284, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain, pages 1019–1027.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. 2021. R-drop: Regularized dropout for neural networks. In Advances in Neural Information Processing Systems, volume 34, pages 10890–10905. Curran Associates, Inc.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. Transactions of the Association for Computational Linguistics, 8:726–742.

Takeru Miyato, Andrew M. Dai, and Ian J. Goodfellow. 2017. Adversarial training methods for semi-supervised text classification. In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In Proceedings of the Third Conference on Machine Translation: Research Papers, pages 1–9, Brussels, Belgium. Association for Computational Linguistics.

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. Association for Computational Linguistics.

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

Leslie Rice, Eric Wong, and Zico Kolter. 2020. Overfitting in adversarially robust deep learning. In Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event, volume 119 of Proceedings of Machine Learning Research, pages 8093–8104. PMLR.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. Transactions of the Association for Computational Linguistics, 8:264–280.

Motoki Sato, Jun Suzuki, and Shun Kiyono. 2019. Effective adversarial regularization for neural machine translation. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 204–210, Florence, Italy. 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. Association for Computational Linguistics.

Dinghan Shen, Mingzhi Zheng, Yelong Shen, Yanru Qu, and Weizhu Chen. 2020. A simple but tough-to-beat data augmentation approach for natural language understanding and generation. arXiv preprint arXiv:2009.13818.

David R. So, Quoc V. Le, and Chen Liang. 2019. The evolved transformer. In Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA, volume 97 of Proceedings of Machine Learning Research, pages 5877–5886. PMLR.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: masked sequence to sequence pre-training for language generation. In Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA, volume 97 of Proceedings of Machine Learning Research, pages 5926–5936. PMLR.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings.

Sho Takase and Shun Kiyono. 2021. Rethinking perturbations in encoder-decoders for fast training. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5767–5780, Online. Association for Computational Linguistics.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2021. Multilingual translation from denoising pre-training. In Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 3450–3466, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pages 5998–6008.

Dilin Wang, ChengYue Gong, and Qiang Liu. 2019. Improving neural language modeling via adversarial training. In Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA, volume 97 of Proceedings of Machine Learning Research, pages 6555–6565. PMLR.

Lijun Wu, Shufang Xie, Yingce Xia, Yang Fan, Jian-Huang Lai, Tao Qin, and Tie-Yan Liu. 2020. Sequence generation with mixed representations. In Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event, volume 119 of Proceedings of Machine Learning Research, pages 10388–10398. PMLR.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144.

Zhen Wu, Lijun Wu, Qi Meng, Yingce Xia, Shufang Xie, Tao Qin, Xinyu Dai, and Tie-Yan Liu. 2021. UniDrop: A simple yet effective technique to improve transformer without extra cost. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3865–3878, Online. Association for Computational Linguistics.

Haoran Xu, Benjamin Van Durme, and Kenton Murray. 2021. BERT, mBERT, or BiBERT? a study on contextualized embeddings for neural machine translation. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 6663–6675, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 483–498, Online. Association for Computational Linguistics.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. 2020. Incorporating BERT into neural machine translation. In 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net.