Token Drop mechanism for Neural Machine Translation

Huaao Zhang
Soochow University / Suzhou China
hazhang@stu.suda.edu.cn

Shigui Qiu
Soochow University / Suzhou China
sgqiu@stu.suda.edu.cn

Xiangyu Duan*
Soochow University / Suzhou China
xiangyuduan@suda.edu.cn

Min Zhang
Soochow University / Suzhou China
minzhang@suda.edu.cn

Abstract

Neural machine translation with millions of parameters is vulnerable to unfamiliar inputs. We propose Token Drop to improve generalization and avoid overfitting for the NMT model. Similar to word dropout, whereas we replace dropped token with a special token instead of setting zero to words. We further introduce two self-supervised objectives: Replaced Token Detection and Dropped Token Prediction. Our method aims to force model generating target translation with less information, in this way the model can learn textual representation better. Experiments on Chinese-English and English-Romanian benchmark demonstrate the effectiveness of our approach and our model achieves significant improvements over a strong Transformer baseline[1].

1 Introduction

Neural machine translation (NMT) achieved enormous success in advancing the quality of translation (Bahdanau et al., 2015; Vaswani et al., 2017; Gehring et al., 2017). In spite of the impressive performance, NMT models are still vulnerable to perturbations in the input sentences (Belinkov and Bisk, 2018; Cheng et al., 2019) , i.e. a tiny perturbation will affect hidden representation and lead to low quality of translation (Zhao et al., 2018). Moreover, NMT commonly consists of millions of parameters, which making it prone to overfitting especially in low resource scene.

A natural way to improve generalization is synthesizing natural noise (Karpukhin et al., 2019) or adopting arbitrary noise (Cheng et al., 2018; Ebrahimi et al., 2018). Another way is exploring regularization techniques to avoid overfitting (Miceli et al., 2017), making model robust to unseen or unfamiliar inputs. However, as discrete data, the text is hard to retain the semantic information after corruption.

In this paper, we propose Token Drop to prevent overfitting and improve generalization. Different from standard dropout (Srivastava et al., 2014) that drops neurons in network randomly, we drop tokens of the input sentences. In order to retain semantic information, we replace tokens with a special symbol $< unk >$. This allows model learn hidden representation from rest token's context, and predict target translation condition on latent variable. On the one hand, our method allows model meeting exponentially different sentences can be explained as data augmentation; On the other hand, our method corrupts input sentences with natural noise can be seen as regularization term for NMT.

We investigate two self-supervised objectives: Replaced Token Detection and Dropped Token Prediction. Considering our Token Drop method regularize parameters by weakening model inputs, making NMT suitable for applying self-supervised objective. During training: (1) use a discriminator to detect whether input tokens are dropped or not; (2) leverage hidden state to predict original tokens of dropped tokens inspired by *Cloze* task (Devlin et al., 2019). Both of them guide model to generate semantically similar representation, leading to a better generalization capacity.

* Corresponding author.

[1] Our code is released at https://github.com/zhajiahe/Token_Drop

Proceedings of the 28th International Conference on Computational Linguistics, pages 4298–4303
Barcelona, Spain (Online), December 8-13, 2020

2 Token Drop Training

Standard dropout prevents overfitting by setting input neurons or hidden neurons to zero with a certain probability p (Hinton et al., 2012; Srivastava et al., 2014). whereas we consider the input sequences of machine translation models instead of the network's neurons, which named Token Drop. Given a input tokens sequence of sentence $X = \{x_1, x_2, ..., x_n\}$ and posit a $|X|$ independent drop rate p. The token x_i in X will be dropped if m_i is 1 . This process as Equation 1:

$$\mathbf{m} \sim Bernoulli(p) \qquad \hat{X} = \textbf{REPLACE}(X, \mathbf{m}, < \textit{unk} >) \tag{1}$$

$$\mathcal{L}_M = -\log \sum P(Y|\hat{X}, \theta_M) \tag{2}$$

The Token Drop can be interpreted as data augmentation and regularization technique for NMT. Seeing that NMT model commonly adopts encoder and decoder architecture, therefore our method drops tokens for both source and target inputs. For the source side, model encoder learns intermediate representation by exponentially different incomplete sentences. For the target side, model decoder generates target translation condition on latent variable, weakening the constraint caused by teacher forcing. Both of them receives incomplete information from inputs, simulating the real situation (e.g unknown or unfamiliar data) at test time.

2.1 Token Drop methods

We adopt three drop strategy for Token Drop:

Zero-Out is introduced by Sennrich et al. (2016a), different from the standard dropout, the method drops full word by setting zero to word embedding during training. The deficiency is zero vector can not learn representation from its context in the self-attention layer.

Drop-Tag (Kågebäck and Salomonsson, 2016) replaces token with a $<$ *dropped* $>$ tag. The tag is subsequently treated just like any other word in the vocabulary and has a corresponding word embedding that is trained. We adopt this technique for NMT to learn better feature representation.

Unk-Tag replaces token with generic unknown word token $<$ *unk* $>$. Bowman et al. (2016) and Yang et al. (2017) apply it to RNN decoder to force model make prediction by latent variable. We found this perfectly suits for NMT system especially on self-attention layers. Better than Drop-Tag method, it need not to add an extra token as well as parameters.

2.2 Replaced Token Detection

We propose the Replaced Token Detection task to promote generalization ability of the model encoder. We regard dropped information as self-supervised label, following Clark et al. (2020), we train a discriminator $D(G(x))$ to detect whether tokens are dropped or not. On account of our dropped tokens are obvious to distinguish, so we add a simple linear classifier to detect replaced tokens. The objective is :

$$\mathcal{L}_{RTD} = \mathbb{E}_{x\sim\mathcal{S}}[-\log D(G(\mathbf{x}))] + \mathbb{E}_{\hat{x}\sim d(x)}[1 - \log D(G(\hat{x}))] \tag{3}$$

Where $d(x)$ denotes the dropped tokens. In our model, the encoder serves as a generator G, which generates hidden state of input tokens. The discriminator D tries to distinguish whether a token is dropped or not, while the generator G has to produce similar representation for x and $\hat{x}$, making the model robust to noisy and unknown inputs.

2.3 Dropped Token Prediction

In consideration of our Token Drop model randomly replaces tokens of input sentence, similar to Masked Language Model (Devlin et al., 2019), which masks then predicts masked tokens by the rest of the tokens, making use of contextual information. Accordingly, we propose Dropped Token Prediction (DTP), predicting dropped tokens by their hidden states. The DTP objective is :

$$\mathcal{L}_{DTP} = -\log \sum_{\hat{x}\in d(x)} P(\hat{x}|X_{/d(x)}, \theta_M) \tag{4}$$

$$P(\hat{x}|X_{/d(x)}) = E(G(X_{/d(x)})) \tag{5}$$

Where $d(x)$ and $X_{/d(x)}$ denote the dropped tokens and the rest tokens respectively. G is model encoder, $E(.)$ is prediction layer. In our implementation of DTP, we adopt *weight tying* (Press and Wolf, 2017) , that is to share the same weight matrix between embedding layer and token prediction classifier.

At the end we train our model jointly with DTP and RTD objective:

$$\mathcal{L} = \mathcal{L}_M + \alpha\mathcal{L}_{RTD} + \beta\mathcal{L}_{DTP} \tag{6}$$

3 Experiment

We conduct our approach on two machine translation benchmarks: LDC (ZH-EN) and WMT16 (EN-RO)[2]

3.1 Dataset and Evaluation

For ZH-EN translation, we used 1.25M sentence pairs extract from LDC corpus. Byte-pair encoding is employed separate vocabulary of about 42K and 31K tokens with 32K merge operations (Sennrich et al., 2016b). we chose NIST06 as the valid set and NIST02, NIST03, NIST04, NIST05, NIST08 as the test set, which contains 878, 919, 1788, 1082, 1357 sentence pairs respectively. We measure BLEU score with *multi-bleu.pl*[3]. For WMT16 EN-RO data which consists of 610K pairs, we adopt Lee et al. (2018)'s preprocessing. The vocabulary is 35K joint source and target subwords (Sennrich et al., 2016b). we use newstest-2016 as test set and report tokenized BLEU score.

3.2 Models and Settings

We adopt the Transformer model (Vaswani et al., 2017) implemented in PyTorch in the *fairseq-py* toolkit (Ott et al., 2019). We closely followed settings by Vaswani et al. (2017)($d_{model} = 512, d_{hidden} = 512, d_{FFN} = 2048, n_{layer} = 6, n_{head} = 8$) and used dropout of $p_{dropout} = 0.3$. As for our approach, we set drop rate $p_s = 0.15$ and $p_t = 0.3$ respectively for ZH-EN task. For EN-RO, we set $p_s = 0.15$ and $p_t = 0.2$. To train with DTP and RTD objective, we simply set $\alpha, \beta = 1$.

4 Result

Models	LDC ZH-EN						WMT EN-RO	
	NIST02	NIST03	NIST04	NIST05	NIST08	AVG	EN→RO	RO→ EN
Transformer	47.17	46.78	47.46	47.97	38.01	45.47	34.60	33.96
Zero-Out	47.86	47.12	48.86	47.90	38.65	46.08	35.21	35.00
Drop-Tag	48.99	48.50	49.81	49.60	39.31	47.24	35.34	35.19
Unk-Tag	48.96	48.52	49.50	49.49	39.31	47.16	35.28	35.13
+ RTD	49.00	**49.36**	49.43	49.25	39.92	47.39	35.49	35.38
+ DTP	49.16	49.19	49.99	**49.83**	40.20	47.75	**35.75**	35.38
+ RTD&DTP	**49.52**	49.29	**50.17**	49.82	**40.38**	**47.84**	35.67	**35.69**

Table 1: BLEU scores on test set for LDC Chinese-English and WMT16 English-Romanian tasks

The results of our experiment on NIST Chinese-English and WMT16 English-Romanian tasks are shown in Table 1. We first conduct Token Drop through three drop methods (Zero-out, Drop-Tag, Unk-Tag), the results show that Token Drop model significantly outperform baseline on two languages. Furthermore, we combine Unk Tag method with DTP and RTD training objective, the results show that both DTP and RTD provide a further improvement on Token Drop training. Overall, we get a gain of 2.37, 1.15 and 1.73 BLEU score on three tasks respectively.

[2]http://www.statmt.org/wmt16/

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

Method	0.00	0.05	0.10	0.15
Beseline	47.44	39.26	30.83	23.20
(Cheng et al., 2019)	46.95	44.20	41.71	39.89
Ours	**48.75**	**46.55**	**44.12**	**41.64**

Table 2: Result on incomplete inputs with different p_s for ZH-EN valid set (NIST06).

In order to demonstrate the generalization capacity of our model on real situation. We constrain input information by replacing words with generic unknown symbol $< unk >$. For each sentence, we generate 100 noisy sentences then report average BLEU score. We also compare our method with Cheng et al. (2019), who introduced white-box adversarial noisy inputs to improve robustness. Table 2 reports our result on incomplete inputs, from where we can see our approach outperforms previous method. This improvement confirms that our model obtains larger generalization capacity over baseline.

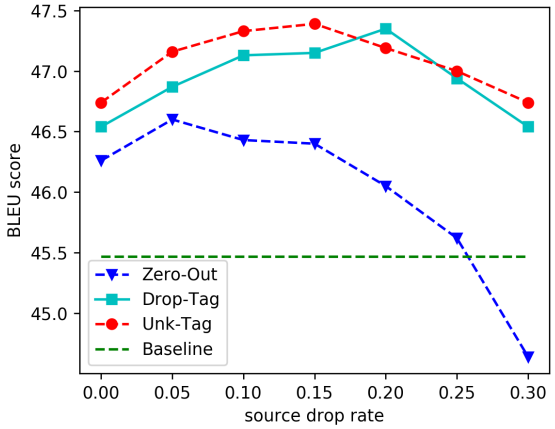

Figure 1: Effect of different source drop rate p_s ($p_t = 0.3$) on LDC ZH-EN translation task

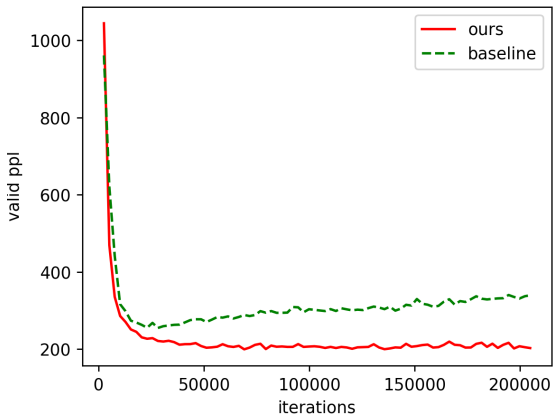

Figure 2: Learning curves of baseline model of our Token Drop model

To examine the impact of Token Drop, we train our model with different source drop rate $p_s = [0.0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3]$. From Figure 1 we can see that model training with a moderate drop rate p would advanced in performance significantly. Drop-Tag and Unk-Tag are quite similar, both of them outperforms Zero-Out method. We plot learning curves of baseline model and our Token Drop model. Figure 2 shows that with the increase of training iterations, our model achieves lower and more stable perplexity than baseline, demonstrating the effectiveness of our approach to prevent overfitting and improve translation quality.

5 Related Work

Word Dropout Iyyer et al. (2015) proposed *word dropout* as feature extractor for text classification task. Bowman et al. (2016) and Xie et al. (2017) applied *word dropout* to RNN decoder can be regard as a smoothing technique. For machine translation task, Sennrich et al. (2016a) randomly set zero to words of input sentence to prevent overfitting, advancing in considerable performance on noisy dataset. In this paper, we explain word dropout as data augmentation (i.e. allows model meeting exponentially different sentences) and a regularization technique (i.e. weakens the encoder and decoder, obtaining better intermediate representations).

6 Conclusion

In this paper, we have proposed Token Drop mechanism for neural machine translation task. Inspired by self-supervised learning, we introduced Replaced Token Detection and Dropped Token Prediction

training objective. We found that NMT model trained with Token Drop gains larger generalization capacity and reduction in overfitting. Even without prior knowledge and additional parameters, our proposed approach reports convincing results on neural machine translation. In future work, we plan to investigate impact of dropping on different words, e.g. word importance and word type.

Acknowledgements

The authors would like to thank the anonymous reviewers for the helpful comments. This work was supported by National Natural Science Foundation of China (Grant No. 61673289, 61525205).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. of ICLR*.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *Proc. of ICLR*.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proc. of SIGNLL*.

Yong Cheng, Zhaopeng Tu, Fandong Meng, Junjie Zhai, and Yang Liu. 2018. Towards robust neural machine translation. In *Proc. of ACL*.

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. In *Proc. of ACL*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *Proc. of ICLR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL*.

Javid Ebrahimi, Daniel Lowd, and Dejing Dou. 2018. On adversarial examples for character-level neural machine translation. In *Proc. of COLING*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proc. of ICML*.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv:1207.0580*.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proc. of ACL*.

Mikael Kågebäck and Hans Salomonsson. 2016. Word sense disambiguation using a bidirectional lstm. *arXiv:1606.03568*.

Vladimir Karpukhin, Omer Levy, Jacob Eisenstein, and Marjan Ghazvininejad. 2019. Training on synthetic noise improves robustness to natural noise in machine translation. In *Proc. of W-NUT*.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proc. of ACL*.

Barone Miceli, Valerio Antonio, Barry Haddow, Ulrich Germann, and Rico Sennrich. 2017. Regularization techniques for fine-tuning in neural machine translation. In *Proc. of EMNLP*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proc. of NAACL*.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proc. of EACL*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Edinburgh neural machine translation systems for WMT 16. In *Proc. of WMT*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proc. of ACL*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*.

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 *Proc. of NIPS*.

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. *arXiv:1703.02573*.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017. Improved variational autoencoders for text modeling using dilated convolutions. In *Proc. of ICML*.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2018. Generating natural adversarial examples. In *Proc. of ICLR*.