# Supplementary Materials: Understanding the Failure of Batch Normalization for Transformers in NLP

**Jiaxi Wang**[1], **Ji Wu**[1,2], **Lei Huang**[3]
[1]Department of Electronic Engineering, Tsinghua University
[2]Institute for Precision Medicine, Tsinghua University
{wjx20@mails, wuji_ee@mail}.tsinghua.edu.cn
[3]SKLSDE, Institute of Artificial Intelligence, Beihang University
huangleiAI@buaa.edu.cn

## A Experimental Details

In the main paper, we sketch the experimental settings. Here, we report the details. Our WMT16 experiments require two NVIDIA 3090 GPUs with 24GB memory each, and other experiments are implemented with one NVIDIA 3090 GPU.

### Neural Machine Translation

**Datasets** We use two widely used datasets: IWSLT14 German-to-English (De-En) dataset and WMT16 English-to-German (En-De) dataset. IWSLT14/WMT16 contains 0.15M/4.5M sentence pairs.

**Configurations** Our code is based on *fairseq* [1][1]. We use six encoder layers and six decoder layers for both datasets. We keep the Transformer decoder unchanged and only modify normalization layers in Transformer encoder throughout NMT experiments. We follow the model settings in [2] for IWSLT14 and apply Transformer$_{base}$ model [3] for WMT16. At test phase, we averaged the last six checkpoints and measure case sensitive tokenized BLEU [4] with beam size 4/5 and length penalty 0.6/1.0 for ISWLT14/WMT16. We use Adam with $(\beta_1, \beta_2) = (0.9, 0.98)$, an inverse square root learning rate scheduler, and a warmup stage with 8000 steps. We apply labeling smoothing $\epsilon_{ls} = 0.1$. For IWSLT14, we set num-tokens=4096, max-epochs=60, dropout=0.3, attention dropout=0.1, activation dropout=0.1 and lr=$5e^{-4}$/$1.5e^{-3}$ for Post-Norm/Pre-Norm Transformer. For WMT16, we set num-tokens=8192, update-freq=4, max-epochs=20, dropout=0.1, attention dropout=0.1, activation dropout=0.1 and lr=$7e^{-4}$/$2e^{-3}$ for Post-Norm/Pre-Norm Transformer.

### Language Modeling

**Datasets** We conduct experiments on PTB (0.93M tokens) [5] and WikiText-103 (WT103)(100M tokens) [6]. We follow the evaluation scheme in [7] and use perplexity (PPL) of test set to compare the model performance.

**Configurations** Following the experimental settings in [2][8], we use three and six layers tensorized transformer core-1 for PTB and Wikitext-103 separately. We use the same hyperparameters for Post-Norm and Pre-Norm Transformer. We use Adam optimizer and set lr=$2.5e^{-4}$ with linear decay. For PTB, we use dropout=0.3, batch size=120, max-steps=20000. For PTB, we use dropout=0.1, batch size=60, max-steps=200000.

[1]https://github.com/pytorch/fairseq. MIT license.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

### Named Entity Recognition

**Datasets** We choose two widely used NER datasets: CoNLL2003 (English) [9] and Resume (Chinese) [10]. CoNLL2003/Resume contains four/eight kinds of named entities. CoNLL2003 contains 14.0k/3.2k/3.5k sentences for train/dev/test split while Resume includes 3.8k/0.5k/0.5k sentences for train/dev/test set. We use F1 score to measure the model performance.

**Configurations** We mainly follow the experimental settings in [11]. We use two and four layers transformer encoder for CoNLL2003 and Resume, respectively. An additional CRF layer [12] is added to model the transition probability. We use SGD optimizer with 0.9 momentum and 1% total steps as warmup steps. We set the learning rate to be $9e^{-4}$ and $7e^{-4}$ for CoNLL2003 and Resume.

### Text Classification

**Datasets** For text classification, we use the code[2] and most configurations in [13]. We select one small scale dataset (IMDB [14]) and three large scale datasets (Yelp, DBPedia, Sogou News), including two sentiment classification tasks (IMDB, Yelp) and two topic classification tasks (DBPedia, Sogou News).

**Configurations** We use six Transformer encoder layers with a CLS token at the beginning of each sentence for classification. We extract 30% of training data as validation set for all datasets. The best checkpoint for the validation is applied in the test phase. We run three times with different random seeds for each setting and report the mean accuracy.

## B Optimal Hyperparameters

For RBN, we choose $\lambda, \nu$ both from $\{0, 0.01, 0.1, 1\}$ by validation loss. We empirically find that increasing the mean penalty on NMT for Post-Norm Transformer can further improve the BLEU scores. Thus, we apply $(\lambda, \nu) = (10, 0)$ / $(100, 0)$ on ISWLT14/WMT16 for Post-Norm Transformer exceptionally.

Table 1: Optimal hyperparameters $(\lambda, \nu)$ for each experimental setting. $(\lambda, \nu)$ are mean and variance penalty coefficients separately.

| Task | NMT | | LM | | NER | | TextCls | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Datasets | IWSLT14 | WMT16 | PTB | WT103 | Resume | CoNLL | IMDB | Sogou | DBPedia | Yelp |
| Post-Norm | (10,0) | (100,0) | (0.1,0.01) | (0.1,0.01) | (0.01,0) | (0.01,0) | (0.1,0) | (0.1,0.01) | (0.1,0.01) | (0,0.1) |
| Pre-Norm | (0.1,0.01) | (0.1,0) | (0.01,0) | (0.1,0) | (0.01,0) | (0.01,0) | (0,0.01) | (0.1,0.01) | (0.1,0.01) | (0.1,0.01) |

## C Average TID of BN and RBN

In the main paper, we have shown the TID of the last BN (RBN) layer on various NLP datasets with Post-Norm or Pre-Norm Transformer. Here, we report the average TID of all BN (RBN) layers (Table 2). RBN decreases the average TID of BN.

## D Figures of TID Through Training

We have shown the average mean and variance TID on WMT16/CoNLL/IMDB/WT103 for Pre-Norm Transformer with BN and RBN in the main paper. Here, we plot the average mean and variance TID on other datasets with Pre-Norm and Post-Norm Transformers in Figures 1 to 4. RBN reduces the TID of BN effectively.

[2]https://github.com/declare-lab/identifiable-transformers. Apache-2.0 license.

Table 2: Average TID of all BN/RBN layer in Post-Norm and Pre-Norm Transformers on various natural language tasks at the end of training. RBN reduces the TID of BN effectively.

| Task | NMT | | LM | | NER | | TextCls | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Datasets | IWSLT14 | WMT16 | PTB | WT103 | Resume | CoNLL | IMDB | Sogou | DBPedia | Yelp |
| **Post-Norm Transformer** | | | | | | | | | | |
| Mean TID of BN_avg | 2.8% | 3.0% | 1.0% | 0.9% | 5.0% | 9.9% | 1.5% | 1.9% | 1.3% | 2.7% |
| Mean TID of RBN_avg | 0.8% | 1.5% | 1.0% | 1.0% | 4.4% | 5.6% | 0.4% | 0.4% | 0.4% | 0.4% |
| Var TID of BN_avg | 7.1% | 4.9% | 1.1% | 2.2% | 4.2% | 9.8% | 3.1% | 2.1% | 1.4% | 3.9% |
| Var TID of RBN_avg | 2.6% | 2.8% | 1.0% | 1.0% | 4.0% | 6.5% | 1.7% | 0.3% | 0.3% | 0.2% |
| **Pre-Norm Transformer** | | | | | | | | | | |
| Mean TID of BN_avg | 3.2% | 7.6% | 1.6% | 2.1% | 10.3% | 10.8% | 2.1% | 4.2% | 2.5% | 4.2% |
| Mean TID of RBN_avg | 3.0% | 1.6% | 1.6% | 2.0% | 6.7% | 5.9% | 0.8% | 1.2% | 1.2% | 1.1% |
| Var TID of BN_avg | 5.5% | 12.8% | 1.6% | 1.9% | 8.1% | 7.4% | 3.6% | 3.7% | 2.1% | 5.1% |
| Var TID of RBN_avg | 1.6% | 5.6% | 1.6% | 1.8% | 7.5% | 6.1% | 1.9% | 0.5% | 0.5% | 0.5% |

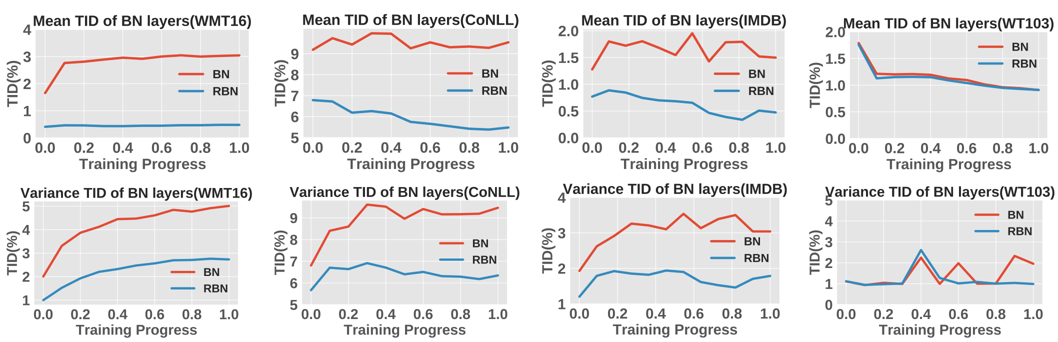


Figure 1: Averaged Mean and Variance TID on WMT16/CoNLL/IMDB/WT103 for Post-Norm Transformer with BN and RBN.

## E Other settings of BN in Test Stage

In the main paper, we test BN (RBN) with population statistics estimated by Exponential Moving Average (EMA). Here, we test two other settings of BN on IWSLT14 dataset. In the first setting (Table 3), we reestimate the population statistics of BN by running two more epochs with zero learning rate. In the second setting (Table 4), we test BN with batch statistics of different effective batch size (max-tokens).

From Table 3, we can see that reestimating the population statistics leads to similar results as EMA. However, using batch statistics instead of population statistics boosts the performance of Post-Norm Transformer$_{BN}$, but hurts the performance of Pre-Norm Transformer$_{BN}$ and Post-Norm Transformer$_{RBN}$ (Table 4). Pre-Norm Transformer$_{RBN}$ is robust to different max-tokens. Note that Transformer$_{LN}$ achieves 35.5 BLEU in both Pre-Norm and Post-Norm settings. BN can not match the performance of LN by changing the test settings.

Table 3: EMA: Use population statistics estimated by EMA. Reestimation: Run two more epochs with zero learning rate to update population statistics. The performance metric is BLEU.

| | Post-BN | Post-RBN | Pre-BN | Pre-RBN |
|---|---|---|---|---|
| EMA | 34.0 | 35.5 | 34.8 | 35.6 |
| Reestimation | 33.8 | 35.6 | 34.9 | 35.4 |

## F Configurations of BN's Variants

We compare the performance of RBN with Power Normalization (PN) [2], Batch Renormalization (BRN) [15] and Moving Averaing Batch Normaliazation (MABN) [16] in the main paper. We mainly

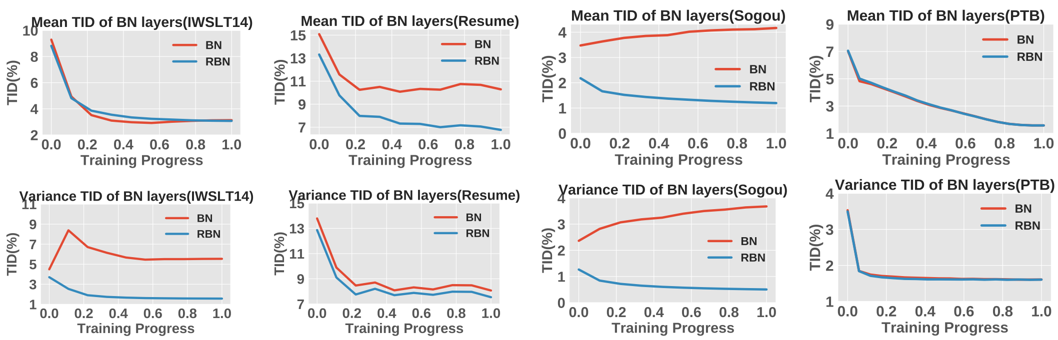


Figure 2: Averaged Mean and Variance TID on IWSLT14/Resume/Sogou/PTB for Pre-Norm Transformer with BN and RBN.

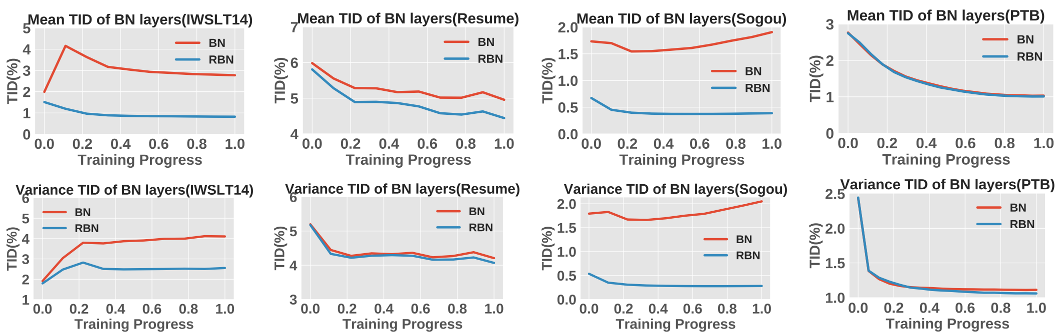


Figure 3: Averaged Mean and Variance TID on IWSLT14/Resume/Sogou/PTB for Post-Norm Transformer with BN and RBN.

follow the hyperparameters in their papers. For PN, we use 4000 warmups, and set foward and backward momentum as $0.9$. For BRN, we use one epoch BN as warmup and linearly increase $r$ to 3 and $d$ to 5. $r$ and $d$ are renormalizing factors. For MABN, we use $16$ mini-batches to compute simple moving average statistics and momentum $\alpha = 0.98$ to compute exponential moving average statistics.

## G Potential Negative Societal Impact

We spend many GPU hours on running experiments which may negatively impact the environment.

## References

[1] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, “fairseq: A fast, extensible toolkit for sequence modeling,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*. Association for Computational Linguistics, 2019, pp. 48–53.

[2] S. Shen, Z. Yao, A. Gholami, M. Mahoney, and K. Keutzer, “Powernorm: Rethinking batch normalization in transformers,” in *ICML*, 2020.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[4] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu: A method for automatic evaluation of machine translation,” in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ser. ACL ’02. Association for Computational Linguistics, 2002, p. 311–318.

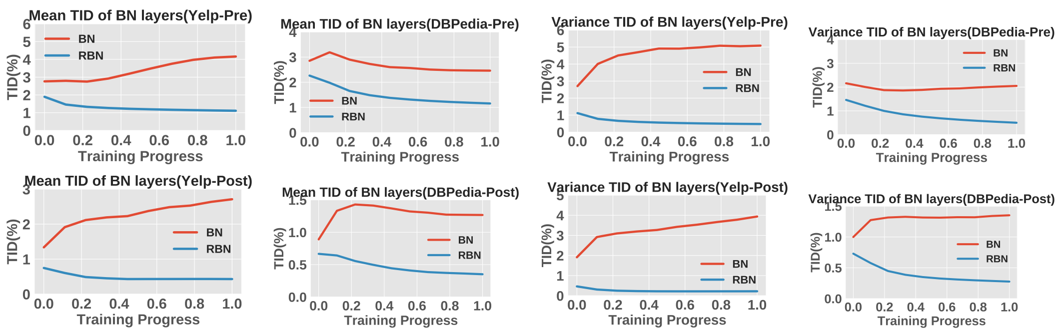


Figure 4: Averaged Mean and Variance TID on Yelp/DBPedia for Pre-Norm Transformer (upper) and Post-Norm Transformer (bottom) with BN and RBN.

Table 4: Testing BN (RBN) with batch statistics of different max-tokens.

| Max-tokens | Post-BN | Post-RBN | Pre-BN | Pre-RBN |
|---|---|---|---|---|
| EMA | 34.0 | 35.5 | 34.8 | 35.6 |
| 512 | 34.8 | 34.9 | 27.2 | 35.6 |
| 1024 | 35.1 | 34.9 | 30.9 | 35.6 |
| 2048 | 35.2 | 34.9 | 32.3 | 35.6 |
| 4096 | 35.2 | 34.9 | 32.7 | 35.6 |
| 8192 | 35.2 | 34.9 | 32.8 | 35.6 |
| 16384 | 35.2 | 34.8 | 32.9 | 35.6 |

[5] T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. H. Cernocky, "Empirical evaluation and combination of advanced language modeling techniques," in *Interspeech*. ISCA, August 2011.

[6] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," *CoRR*, vol. abs/1609.07843, 2016.

[7] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-XL: Attentive language models beyond a fixed-length context," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 2978–2988.

[8] X. Ma, P. Zhang, S. Zhang, N. Duan, Y. Hou, M. Zhou, and D. Song, "A tensorized transformer for language modeling," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019.

[9] E. T. K. Sang and F. D. Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," in *CoNLL*, 2003.

[10] Y. Zhang and J. Yang, "Chinese ner using lattice lstm," in *ACL*, 2018.

[11] H. Yan, B. Deng, X. Li, and X. Qiu, "Tener: Adapting transformer encoder for named entity recognition," 2019.

[12] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *ArXiv*, vol. abs/1508.01991, 2015.

[13] R. Bhardwaj, N. Majumder, S. Poria, and E. Hovy, "More identifiable yet equally performant transformers for text classification," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, 2021, pp. 1172–1182.

[14] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2011, pp. 142–150.

[15] S. Ioffe, "Batch renormalization: Towards reducing minibatch dependence in batch-normalized models," in *NeurIPS*, 2017.

[16] J. Yan, R. Wan, X. Zhang, W. Zhang, Y. Wei, and J. Sun, "Towards stabilizing batch statistics in backward propagation of batch normalization," in *ICLR*, 2020.