NT5 at WMT 2022 General Translation Task

*Makoto Morishita♠, *Keito Kudo♢, *Yui Oka♠, *Katsuki Chousa♠,
†Shun Kiyono♡, †Sho Takase♣, Jun Suzuki♢♡
♠NTT Communication Science Laboratories ♢Tohoku University
♡RIKEN Center for Advanced Intelligence Project ♣Tokyo Institute of Technology

Abstract

This paper describes the NTT-Tohoku-TokyoTech-RIKEN (NT5) team's submission system for the WMT'22 general translation task. This year, we focused on the English-to-Japanese and Japanese-to-English translation tracks. Our submission system consists of an ensemble of Transformer models with several extensions. We also applied data augmentation and selection techniques to obtain potentially effective training data for training individual Transformer models in the pre-training and fine-tuning scheme. Additionally, we report our trial of incorporating a reranking module and the reevaluated results of several techniques that have been recently developed and published.

1 Introduction

This paper describes an overview of our submission systems for participating in the WMT 2022 general machine translation tasks. Our team, named NT5, is comprised of individuals from four organizations: NTT, Tohoku University, Tokyo Institute of Technology, and RIKEN. This year, we focused on bi-directional translation in a single language pair: English-to-Japanese (En→Ja) and Japanese-to-English (Ja→En) translation tracks.

Our submission system consists of an ensemble of Transformer models (Vaswani et al., 2017) with several recent extensions. We also applied data augmentation and selection techniques to obtain potentially effective training data for training individual Transformer models in the pre-training/fine-tuning scheme. The models were first trained with a large but possibly noisy parallel corpus for pre-training and then with a small but clean parallel corpus for fine-tuning. Additionally, we report our trial of incorporating a reranking module that rescores the n-best lists based on source-to-target, target-to-source, and masked language models.

The following section briefly provides an overview of our entire system and each module in more depth.

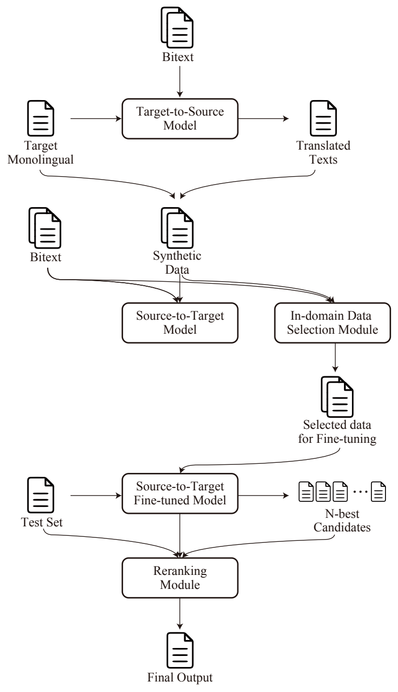

Figure 1: System overview

2 System Overview

Figure 1 shows an overview of our system. Our submissions are for the **constrained track**, which only uses parallel and monolingual data that are provided by the WMT shared-task organizers.

*: Equal contributions. Morishita trained the initial translation model and created a synthetic corpus. Morishita and Kudo tuned the model's hyper-parameters. Oka implemented the relative positional embeddings. Chousa worked on the data filtering and the reranking module. Kiyono and Takase implemented the B2T connections and helped with the experiments. Morishita and Oka implemented the in-domain data selection methods for fine-tuning. Morishita, Kudo, and Suzuki developed an effective fine-tuning strategy. Morishita, Kudo, and Suzuki trained the main translation models. Suzuki organized the team. Everyone contributed to writing this paper.

†: Their current affiliation is LINE Corporation.

We selected Transformer models (Vaswani et al., 2017) as our base translation model and chose a two-step training strategy: pre-training and fine-tuning schemes. We first constructed datasets for the pre-training and fine-tuning.

The pre-training dataset must be as large as possible, even if the data are noisy (Bansal et al., 2022). We first trained the Transformer models using only the provided bitext datasets for both translation directions: En→Ja and Ja→En. We refer to these first trained models as **initial models**. We then generated synthetic datasets for both directions through back-translation (Sennrich et al., 2016), i.e., translating target-side monolingual data using the initial model in the reverse translation direction.

The fine-tuning dataset must be as clean as possible, even if it is relatively small. Indeed, in our previous year's submission (Kiyono et al., 2020), we adapted the models to a news domain in the fine-tuning phase and drastically improved the translation quality. However, this year's task focused on a general domain, i.e., a test set that consisted of sentences from multiple domains. Thus, adaptation by fine-tuning is much more challenging. We tested and combined data selection methods based on sentence embeddings and language models for obtaining fine-tuning data. This process can be viewed as selecting domain adaptation data.

By using these datasets, we pre-trained the Transformers with pre-training configurations and fine-tuned the pre-trained models with the fine-tuning configurations described in Table 2. Finally, we conducted an ensemble of fine-tuned models. A notable characteristic of our system is that we combined the Transformer models with heterogeneous model configurations for the ensembling. Each model configuration primarily differs in its depth and width. Moreover, we applied recent advances in the extensions of Transformer models, such as bottom-to-top connection (Takase et al., 2022) and relative position embedding (Shaw et al., 2018).

Our system also uses a reranking module. We generated the ten best translation lists as reranking candidates using an ensemble of Transformer models. Then we selected the best translations based on the weighted sum of the likelihoods obtained from the source-to-target and target-to-source translation models and the masked language models.

Corpus	w/o Filtering	w/Filtering
JParaCrawl v3.0	25.7 M	25.0 M
WikiMatrix	3.89 M	3.64 M
JESC	2.80 M	2.57 M
Wiki Titles v3	757 K	327 K
KFTT	440 K	371 K
TED Talks	242 K	224 K
NewsCommentary v16	1.9 K	1.8 K

Table 1: Number of sentence pairs in bitext corpus

3 Dataset Construction

3.1 Provided Data

Bitext Corpus We used all the provided bitext corpora: JParaCrawl v3.0, News Commentary v16, Wiki Titles v3, WikiMatrix, Japanese-English Subtitle Corpus (JESC), The Kyoto Free Translation Task (KFTT) Corpus, and TED Talks. We filtered out the potentially noisy pairs using the straightforward parallel corpus filtering methods, as described in Section 3.2. Table 1 shows the size of each dataset without/with filtering.

Monolingual Corpus We also used the following provided monolingual data: News Crawl, News Commentary, and Common Crawl. We back-translated the monolingual sentences with a target-to-source model trained only with the provided parallel data, as described in Section 3.2, and used them as synthetic data (Sennrich et al., 2016).

3.2 Building Pre-training Data

Synthetic Data Construction To augment the training data, we constructed synthetic data by applying the initial translation model trained with bitext to the monolingual data. As a preprocessing step, we truecased[1] both the bitext and monolingual data. We then tokenized the data into subwords using the `Sentencepiece` tool (Kudo and Richardson, 2018) with the unigram language model option. We set the vocabulary size to 32,000 for the initial translation model, which is used for creating synthetic data. For the final submission model, we increased the vocabulary size to 64,000. Our hypothesis argues that a bigger vocabulary is crucial for completely exploiting large synthetic data. In fact, this 64,000-vocabulary model outperformed the 32,000-vocabulary model in our preliminary experiment.

[1]`https://github.com/moses-smt/mosesdecoder/blob/master/scripts/recaser/truecase.perl`

Initial Translation Model	
Subword Size	32,000
Architecture	Transformer (big) with FFN size of 4,096
Optimizer	Adam ($\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 1 \times 10^{-8}$)
Learning Rate Schedule	Inverse square root decay
Warmup Steps	4,000
Max Learning Rate	0.001
Dropout	0.3
Gradient Clip	1.0
Batch Size	1,280,000 tokens
Number of Updates	50,000 steps
Averaging	Save a checkpoint every 200 steps and average the last eight
Implementation	`fairseq` (Ott et al., 2019)
Pre-training Configuration	
Subword Size	64,000
Architecture	(See Table 4)
Optimizer	Adam ($\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 1 \times 10^{-8}$)
Learning Rate Schedule	Inverse square root decay
Warmup Steps	4,000
Max Learning Rate	0.001
Dropout	0.1
Gradient Clip	0.1
Batch Size	1,024,000 tokens
Maximum Number of Updates	100,000 steps
Averaging	Save a checkpoint every 2,000 steps and average the last ten
Implementation	`fairseq` (Ott et al., 2019)
Fine-tuning Configuration	
Subword Size	Identical to Pre-training Configuration
Architecture	(See Table 4)
Optimizer	Adam ($\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 1 \times 10^{-8}$)
Learning Rate Schedule	Fixed
Warmup Steps	N/A
Max Learning Rate	0.00001
Dropout	0.2
Gradient Clip	1.0
Batch Size	14,400 tokens
Number of Updates	Tuned for each model (See Secsion 4.3)
Averaging	Save a checkpoint every ten steps and average the last ten
Implementation	`fairseq` (Ott et al., 2019)

Table 2: List of hyper-parameters: We used initial translation model for creating synthetic data, pre-training configuration to construct pre-training models described in Section 4.2, and fine-tuning configuration to construct models for submission. We used several different model configurations for ensembling. See Table 4 for more details.

As the initial translation data, we trained the Transformer-big model defined in the original Transformer paper (Vaswani et al., 2017) for both language directions (English-to-Japanese and Japanese-to-English translations) only with the provided bitext data. The detailed hyper-parameters are described in the initial translation model section of Table 2. Finally, we respectively translated 1.4B and 1.2B monolingual sentences for English and Japanese.

	#sent. pairs	#subwords (JA)	#subwords (EN)
En→Ja	579 M	11.6 B	13.1 B
Ja→En	724 M	15.5 B	16.7 B

Table 3: Statistics of synthetic data used for pre-training

Data Cleaning For both the provided bitext and synthetic data, we carried out cleaning based on a combination of sentence embeddings and hand-crafted rules.

For both the bitext and synthetic data, we removed the too-long sentences whose length exceeded 500 characters. We also removed the sentences that were identified as not being written in English or Japanese with the `langid`[2] toolkit.

For the synthetic data, we further applied a sentence embedding-based filtering approach. We took advantage of LaBSE (Feng et al., 2022) to embed the Japanese and English sentences into the same embedding space. We then scored and ranked the parallel sentence pairs based on the cosine similarity of their sentence embeddings. Subsequently, we filtered out the following items from the synthetic data:

- duplicated sentence pairs
- sentences over 150 words[3] or single words with over 40 characters
- sentences whose ratio between word and character count is greater than 12
- sentences that contain invalid Unicode characters
- sentence pairs whose source/target word ratio exceeds 4
- sentence pairs whose source/target length ratio exceeds 6
- sentence pairs whose source and target sentences are identical
- sentence pairs whose cosine similarity is greater than 0.96[4]

[2]`https://github.com/saffsd/langid.py`

[3]We tokenized the Japanese sentences by MeCab (Kudo, 2006) with the IPA dictionary. Note that this tokenization is for cleaning purpose only.

[4]We found that sentence pairs with high cosine similarities might be noisy; for example, the source and target sentences

Finally, we respectively selected approximately the top 579 M and 724 M sentences from the translated 1.2B and 1.4B monolingual sentences as the synthetic data of En→Ja and Ja→En in the rank orders. Table 3 shows the statistics of synthetic data used for our pre-training.

3.3 Building Fine-tuning Data

As for the fine-tuning data, we prepared two types of data: *news* and *general*. The news data consist of the dev and test sets of the WMT'20 news translation task, which has 3,991 sentences. General data were created by selecting parallel sentences in the target domain. We used the n-gram language model-based method proposed by Moore and Lewis (2010) and selected the top 20,000 scored sentences from the synthetic corpus. We also used sentence embeddings to select the general domain data. We used an unsupervised SimSCE (Gao et al., 2021) as the English sentence embedding and SentenceTransformers (Reimers and Gurevych, 2019) as the Japanese sentence embedding[5]. We searched for the nearest 4,000 sentences to the target domain using `faiss` (Johnson et al., 2019) and combined the sentences selected by both the language model-based and sentence embeddings. As a result, our general domain data contained 24,000 sentences.

4 Primary Translation Module

4.1 Model Configuration

We trained several Transformer models for the model ensembling in the decoding phase. We independently trained models with different sizes due to the restrictions on computational resources at hand. We pre-trained and fine-tuned each model with the configurations shown in Table 2. The details of the model configurations are summarized in Table 4.

Our configuration has three notable characteristics: a bottom-to-top (B2T) connection (Takase et al., 2022), relative position embedding, and a larger batch size.

B2T Connection Transformer architectures can be categorized into two types based on the position of the layer normalizations: Post-LN and Pre-LN. Previous studies (Xiong et al., 2020; Liu et al., 2020; Takase et al., 2022) indicated that training a deep Post-LN Transformer[6] is unstable due to the vanishing gradient problem. However, Takase et al. (2022) argued that Post-LN Transformers outperform Pre-LN Transformers if their trainings are successful. Thus, we want to exploit the advantage of Post-LN Transformers. In addition, we want to make our Transformers as deep (and wide) as possible to make a full use of large synthetic data.

Several studies proposed techniques that stabilize the trainings of Post-LN Transformers while retaining their performance advantages (Liu et al., 2020; Takase et al., 2022). In this study, we used the B2T connection proposed by Takase et al. (2022), which has an additional residual connection from an input to an output in each layer. The B2T connection is easy to implement and can be incorporated with a tiny amount of extra computational cost.

Relative Position Embedding A Transformer model was originally equipped with Absolute Position Embedding (APE) (Gehring et al., 2017) for position representation. However, several recent studies (Raffel et al., 2020; Narang et al., 2021) report that Relative Position Embedding (RPE) (Shaw et al., 2018) outperforms APE, especially for sentences whose lengths are unseen during the training (Kiyono et al., 2021). Thus, for the Transformer encoder, we replaced APE with RPE. Following Shaw et al. (2018), we set clipping distance k to 16.

Larger Batch Size Ott et al. (2019) demonstrated that a large batch size improves performance. The recent development of large language models also indicates this tendency (Hoffmann et al., 2022). Given this knowledge, we followed the setting of T5 (Raffel et al., 2020) and selected a token batch size of approximately 1M. Note that this is much larger than the batch size used by Ott et al. (2019).

4.2 Pre-training

We trained each model described in Table 4 with the filtered bitext and synthetic data described in Section 3.2. We set the maximum number of updates to 100,000 and used early stopping based on the validation set performance. In this phase, we used the Pre-training configuration of Table 2. Since the synthetic data is extremely larger than the

are sometimes identical. Thus we removed them from the training data.

[5]We used `stsb-xlm-r-multilingual`.

[6]When we used the dimension sizes described in Table 4, the trainings of nine or more layers of Post-LN Transformers diverged.

Configuration	#Models	#Params.	Encoder				Decoder			
			Layer	d_{model}	d_{ffn}	Attention Heads	Layer	d_{model}	d_{ffn}	Attention Heads
`NTT-Base`	2	547M	9	1024	8192	16	9	1024	8192	16
`ABCI-Base`	2	622M	9	1024	16384	16	9	1024	4096	16
`ABCI-EncBig`	1	2.0B	12	1024	65536	16	9	1024	8192	16
`ABCI-EncDeep`	1	736M	18	1024	8192	16	9	1024	8192	16
`Failab-EncBig`	1	1.7B	9	1024	61440	16	9	1024	16384	16
`Failab-DecBig`	1	1.7B	9	1024	16384	16	9	1024	61440	16

Table 4: List of model configurations used in final system: d_{model} and d_{ffn} respectively denote sizes of embedding and feedforward layers. In En→Ja, `Failab-EncBig` and `Failab-DecBig` did not fit in the GPU memory. Therefore, we set d_{ffn} to 58368 instead of 61440, which is the largest value that successfully worked.

bitext, we upsampled the bitext until it reaches to a 1:1 ratio to the synthetic data. In addition, we used the tagged back-translation technique (Caswell et al., 2019). In detail, we attached a special token $\langle$`BT`$\rangle$ to the beginning of source sentences in synthetic data.

To improve the performance, we tried using several perturbation methods described in Takase and Kiyono (2021) in this training phase. However, they did not positively affect the performance. Since the number of sentences in our training data is far greater than in their study, regularization by perturbations might be ineffective.

4.3 Fine-tuning

We fine-tuned the pre-trained translation models with the fine-tuning dataset described in Section 3.3 and used the configurations described in Table 2. We set the maximum number of updates to 600, and used early-stopping according to the performance on the test data of WMT'21 (wmt21test).

4.4 Ensemble

We ensembled the fine-tuned models described in Table 4[7]. How we ensembled the Transformer models trained in different model configurations is another unique characteristic of our system compared with the standard configurations used in the WMT submission systems.

5 Post-processing

5.1 Reranking

We tried to apply a reranking method to select the most likely candidate from a set of candidates and input. We scored the candidate with several models and unified these scores with Minimum Error Rate Training (MERT) (Och, 2003), which is often used in Statistical Machine Translation (SMT).

Suppose we have set of candidate output sentences C_i for each source sentence s_i, where $i \in \{1, \dots, I\}$. In our case, we generated n-best candidates using the submission model with the beam-search algorithm.

Hereafter, $P_j(s_i, e) \in [0, 1]$ denotes candidate score $e \in C_i$ for i-th input s_i from the j-th model, where $j \in \{1, \dots, J\}$, and $\boldsymbol{w} = (w_1, \dots, w_J)$ denotes the vector representation of the model weights. Given weights $\boldsymbol{w}$, the most likely candidate $\hat{e}_i^{\boldsymbol{w}}$ from C_i is obtained by maximizing the weighted sum of P_j:

$$\hat{e}_i^{\boldsymbol{w}} = \operatorname*{argmax}_{e \in C_i} \left\{ \sum_{j=1}^{J} w_j P_j(s_i, e) \right\}. \quad (1)$$

Finally, we explored $\hat{\boldsymbol{w}}$ for the parameter estimation of $\boldsymbol{w}$ by solving the following optimization problem:

$$\hat{\boldsymbol{w}} = \operatorname*{argmax}_{\boldsymbol{w} \in [0,1]^J} \left\{ \texttt{corpus\_bleu}(\hat{\mathcal{E}}^{\boldsymbol{w}}) \right\}, \quad (2)$$

where $\hat{\mathcal{E}}^{\boldsymbol{w}} = \left(\hat{e}^i_{\boldsymbol{w}}\right)_{i=1}^{I}$.

For the candidate's score, we used the following models to compute $P_j(s_i, e)$.

L2R Forward and Backward Translation Models The left-to-right (L2R) forward and backward translation models are identical as those used for the candidate generation of En→Ja and Ja→En. For each direction, we trained two models with two different training data; these four models computed the score by force-decoding a candidate from their input.

R2L Forward and Backward Translation Models The right-to-left (R2L) forward and backward translation models generate a translation in reverse

[7]We trained two models with both the `NTT-base` and `ABCI-base` configurations with different random seeds.

ID	Model	En→Ja			Ja→En		
		wmt20dev	wmt21test	wmt22test	wmt20dev	wmt21test	wmt22test
(a)	`NTT-Base` (bitext only)	22.5	-	-	22.7	-	-
(b)	`NTT-Base` (Seed#1)	23.9	25.6	-	24.1	21.5	-
(c)	`NTT-Base` (Seed#2)	23.7	25.5	-	24.0	21.5	-
(d)	`ABCI-Base` (Seed#1)	25.4	27.6	-	**25.5**	**23.4**	-
(e)	`ABCI-Base` (Seed#2)	**26.0**	**28.3**	-	**25.5**	23.2	-
(f)	`ABCI-EncBig`	24.7	26.7	-	**25.5**	22.6	-
(g)	`ABCI-EncDeep`	24.8	26.5	-	25.3	22.8	-
(h)	`Failab-EncBig`	24.6	26.3	-	23.7	20.4	-
(i)	`Failab-DecBig`	23.5	25.4	-	23.0	20.9	-
(j)	(b), finetuned on news	-	28.6	26.3	-	25.6	24.9
	(c), finetuned on news	-	29.0	26.2	-	26.0	25.1
	(d), finetuned on news	-	28.9	26.6	-	25.8	25.4
	(e), finetuned on news	-	28.5	26.6	-	25.8	25.0
	(f), finetuned on news	-	**29.4**	26.5	-	**27.0**	25.5
	(g), finetuned on news	-	28.8	**26.7**	-	26.4	**25.6**
	(h), finetuned on news	-	29.2	**26.7**	-	25.6	25.2
	(i), finetuned on news	-	28.6	26.4	-	25.9	25.1
(k)	(b), finetuned on news+general	-	27.8	25.4	-	24.9	23.9
	(c), finetuned on news+general	-	28.0	24.8	-	24.5	23.8
	(d), finetuned on news+general	-	28.4	25.3	-	25.0	24.7
	(e), finetuned on news+general	-	28.0	25.3	-	24.9	24.6
	(f), finetuned on news+general	-	28.0	25.0	-	25.8	24.8
	(g), finetuned on news+general	-	28.5	25.6	-	25.3	24.5
	(h), finetuned on news+general	-	28.6	25.1	-	25.2	24.2
	(i), finetuned on news+general	-	27.9	24.7	-	25.1	23.9
(l)	Ensemble of (j)	-	**30.6**	**27.6**	-	27.8	**26.6**
(m)	Ensemble of (k)	-	29.6	25.8	-	26.8	25.4
(n)	Ensemble of (l) and (m)	-	**30.6**	27.2	-	**27.9**	**26.6**
(o)	(n) + reranking	-	-	25.7	-	-	25.0

Table 5: Performance comparison of models trained for submission: Models (b)-(i) are pre-trained models (details in Section 4.2). Models (j)-(o) do not contain wmt20dev result because the dataset is in news fine-tuning dataset. We chose model (l) for the final submission. Note that the wmt22test results were computed as the post-evaluation after the wmt22 test data was released.

word order. We trained the model of both directions with all the provided bitext datasets and computed the scores with the same procedure as was used for the L2R models.

Masked Language Models We also used the masked language models to compute the likelihood of the decoded target sentences. Specifically, we used the pre-trained models of DeBERTa (He et al., 2021)[8] for English and RoBERTa (Liu et al., 2019)[9] for Japanese. To score the candidate, we adopted *pseudo-log-likelihood scores* (PLLs), computed by masking tokens one by one, as proposed by Salazar et al. (2020) Finally, we normalized the PLLs by dividing them by token length.

5.2 Rule-base Formatting

We also applied language-specific post-processing.

Ja→En We detokenized and detruecased the sentences and removed all the unknown tokens from the outputs. Since some placeholders were tokenized into two or more tokens, we fixed them to a single token.

En→Ja We removed the spaces in the English proper nouns of two characters or fewer, the spaces before and after such special symbols as "/", "-" or "#PRS/ORG#". We replaced English style commas "," and periods "." with the Japanese styles: "，" and "。".

6 Results

Table 5 shows the performance of both the intermediate models and the final model for our submission. Our result highlights the effectiveness of the

[8] `https://huggingface.co/microsoft/deberta-v2-xxlarge`

[9] `https://huggingface.co/nlp-waseda/roberta-large-japanese-seq512`

techniques incorporated in our system.

Effectivenss of Fine-tuning Data We expected model (m), which was fine-tuned on the general domain, to achieve the best result. However, the model (l), which was fine-tuned on the news domain, achieved the higher BLEU score on both wmt21test and wmt22test. We suspect this is because the data used for the news fine-tuning are cleaner than those of the general domain. Since the fine-tuning data for the news domain consists of the previous years' dev/test sets that were translated by professionals, the news domain data are clean while the general domain data were chosen mainly from synthetic data. We will analyze the relationship between the translation accuracy and the cleanliness of the fine-tuned data in the future.

Negative Result on Reranking In Table 5, the performance of model (n) and (o) demonstrate that the reranking technique (Section 5.1) did not improve the performance over the ensemble models on wmt22test. We suspect that this performance degradation comes from the domain difference between the datasets used for MERT and the evaluation. For MERT, We used wmt21test, whose domain is news, to optimize the model weights; however, this year's test set, wmt22test, contains sentences from multiple domains. Thus, we chose model (l), which is the model without reranking, for our final submission.

7 Conclusion

We described the submission of our joint team (NTT, Tohoku, TokyoTech, and RIKEN) to the WMT'22 general translation task. We participated in the En↔Ja translation. Our system mainly consists of an ensemble of Transformer models with several recent extensions. We also applied data augmentation and selection techniques to train individual Transformer models in our pre-training/fine-tuning training scheme.

Acknowledgments

We thank two anonymous reviewers who provided valuable feedback. This work was mainly done under the NTT-Tohoku University collaborative research agreement. The work of Sho Takase was partly supported by JSPS KAKENHI Grant Number JP21K17800 and JST ACT-X Grant Number JPMJAX200I. The work of Jun Suzuki was partly supported by JST Moonshot R&D Grant Number JPMJMS2011 (fundamental research).

References

Yamini Bansal, Behrooz Ghorbani, Ankush Garg, Biao Zhang, Colin Cherry, Behnam Neyshabur, and Orhan Firat. 2022. Data scaling laws in NMT: The effect of noise and architecture. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pages 1466–1482.

Isaac Caswell, Ciprian Chelba, and David Grangier. 2019. Tagged Back-Translation. In *Proceedings of the Fourth Conference on Machine Translation (WMT)*, pages 53–63.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 878–891.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6894–6910.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with disentangled attention. In *Proceedings of 9th International Conference on Learning Representations (ICLR)*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Shun Kiyono, Takumi Ito, Ryuto Konno, Makoto Morishita, and Jun Suzuki. 2020. Tohoku-AIP-NTT at WMT 2020 news translation task. In *Proceedings of the Fifth Conference on Machine Translation (WMT)*, pages 145–155.

Shun Kiyono, Sosuke Kobayashi, Jun Suzuki, and Kentaro Inui. 2021. SHAPE: Shifted absolute position embedding for transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3309–3321.

Taku Kudo. 2006. MeCab: yet another part-of-speech and morphological analyzer. http://mecab.sourceforge.net.

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 (EMNLP)*, pages 66–71.

Liyuan Liu, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, and Jiawei Han. 2020. Understanding the difficulty of training transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5747–5763.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 220–224.

Sharan Narang, Hyung Won Chung, Yi Tay, Liam Fedus, Thibault Fevry, Michael Matena, Karishma Malkan, Noah Fiedel, Noam Shazeer, Zhenzhong Lan, Yanqi Zhou, Wei Li, Nan Ding, Jake Marcus, Adam Roberts, and Colin Raffel. 2021. Do transformer modifications transfer across implementations and applications? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5758–5773.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 160–167.

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 *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 48–53.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research (JMLR)*, 21(140):1–67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2699–2712.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving Neural Machine Translation Models with Monolingual Data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 86–96.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 464–468.

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 (NAACL-HLT)*, pages 5767–5780.

Sho Takase, Shun Kiyono, Sosuke Kobayashi, and Jun Suzuki. 2022. On layer normalizations and residual connections in transformers. *arXiv preprint arXiv:2206.00330*.

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 31 (NIPS)*, pages 5998–6008.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. 2020. On layer normalization in the transformer architecture. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 10524–10533.