Tohoku-AIP-NTT at WMT 2020 News Translation Task

Shun Kiyono[♠♢] **Takumi Ito**[♢♠] **Ryuto Konno**[♢♠] **Makoto Morishita**[♡♢] **Jun Suzuki**[♢♠*]
[♠]RIKEN Center for Advanced Intelligence Project [♢]Tohoku University
[♡]NTT Communication Science Laboratories
shun.kiyono@riken.jp;
{t-ito, ryuto, jun.suzuki}@ecei.tohoku.ac.jp;
makoto.morishita.gr@hco.ntt.co.jp

Abstract

In this paper, we describe the submission of Tohoku-AIP-NTT to the WMT'20 news translation task. We participated in this task in two language pairs and four language directions: English↔German and English↔Japanese. Our system consists of techniques such as back-translation and fine-tuning, which are already widely adopted in translation tasks. We attempted to develop new methods for both synthetic data filtering and reranking. However, the methods turned out to be ineffective, and they provided us with no significant improvement over the baseline. We analyze these negative results to provide insights for future studies.

1 Introduction

The joint team of Tohoku University, RIKEN AIP, and NTT (Tohoku-AIP-NTT) participated in the WMT'20 shared news translation task in two language pairs and four language directions: English→German (En→De), German→English (De→En), English→Japanese (En→Ja), and Japanese→English (Ja→En).

At the very beginning of this year's shared task, we planned to employ the following two enhancements at the core of our system. The first enhancement is the noisy synthetic data filtering (Koehn et al., 2018) to better utilize the millions of back-translated synthetic data. However, as we analyze in Section 5.1, this filtering turned out to be ineffective. The second enhancement is the reranking of n-best candidates generated a the model.

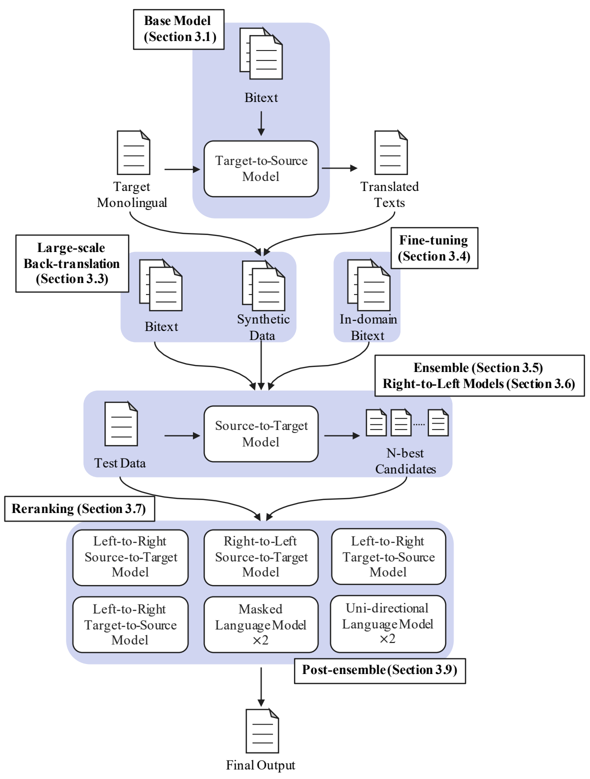
Base Model (Section 3.1)
Bitext
Target Monolingual
Target-to-Source Model
Translated Texts
Large-scale Back-translation (Section 3.3)
Fine-tuning (Section 3.4)
Bitext
Synthetic Data
In-domain Bitext
Ensemble (Section 3.5) Right-to-Left Models (Section 3.6)
Test Data
Source-to-Target Model
N-best Candidates
Reranking (Section 3.7)
Left-to-Right Source-to-Target Model
Right-to-Left Source-to-Target Model
Left-to-Right Target-to-Source Model
Left-to-Right Target-to-Source Model
Masked Language Model ×2
Uni-directional Language Model ×2
Post-ensemble (Section 3.9)
Final Output

Figure 1: Overview of our system.

Given a collection of scores from multiple generative/translation models, our reranking module selects the best candidate. We attempted to develop sophisticated machine learning based methods for optimizing the weight of each score. However, we found that those methods are not as effective as the simple grid search on the BLEU score (details in Section 3.7 and Section 5.3).

Eventually, we designed our system as a combination of techniques that are already widely adopted in the shared task, such as back-translation and fine-tuning. The overview of our system is shown in Figure 1. We achieved the first place in De→En on automatic evaluation and obtained strong results in other language directions.

[*] Shun conducted most of the experiments for both En↔De and En↔Ja. Takumi preprocessed En↔Ja data. Ryuto trained fasttext word vectors and implemented the post-ensemble method. Takumi and Ryuto worked on synthetic data filtering approaches. Makoto back-translated monolingual corpus for all language directions. Shun, Makoto, and Takumi developed an effective fine-tuning strategy. Jun implemented the entire reranking module and organized the team. Everyone contributed to writing this paper.

2 Dataset and Preprocessing

2.1 Bitext

For both En↔De and En↔Ja, we used all bitexts that are available for a constrained system.

En↔De Following Ng et al. (2019), we applied language identification filtering (`langid`)[1] to the bitext. In this filtering, sentence pairs were removed if a supposedly English/German sentence is identified as a non-English/German sentence. Then, we applied the `clean-corpus-n` script available in the Moses toolkit (Koehn et al., 2007) and removed sentence pairs that are either too long and/or their length ratio is too large[2]. These two filtering processes provided us with approximately 44M sentence pairs. Then, we trained and applied the Moses `truecaser` independently for each language. We also trained byte-pair encoding (BPE) (Sennrich et al., 2016c) models using the `sentencepiece` (Kudo and Richardson, 2018) implementation. For BPE training, we used only a subset of the parallel corpus (Europarl, NewsCommentary, and RAPID) to prevent extremely rare characters from contaminating the vocabulary and the subword segmentation.

En↔Ja Similar to En↔De, we applied `langid` to clean bitext, but we did not use `clean-corpus-n` since the Japanese text is not segmented. Instead, we simply removed sentence pairs in which the English sentence is longer than 500 tokens. Eventually, we obtained about 17M sentence pairs. We used `truecaser` for the English side only, because case information does not exist in the Japanese language. We independently trained the BPE merge operation on the bitext. We set the character coverage option[3] of `sentencepiece` to 1.0 and 0.9998 for English and Japanese, respectively.

2.2 Monolingual Corpus

The origins of the monolingual corpus in our system are the Europarl, NewsCommentary, and entire NewsCrawl (2008-2019) corpora for English and German, and the Europarl, NewsCommentary and CommonCrawl corpora for Japanese. Similarly to bitext preprocessing in Section 2.1, we applied `langid` filtering to all monolingual corpora. These corpora are used for large-scale back-translation (Section 3.3).

3 System Overview

3.1 Base Model and Hyperparameter

The well-known Transformer model (Vaswani et al., 2017) is our base Encoder Decoder model. Specifically, we started with the "Transformer (big)" setting described by Vaswani et al. (2017) and increased the feed-forward network (FFN) size from 4,096 to 8,192. Ng et al. (2019) reported that this larger FFN setting slightly improves the performance; we also confirmed it in our preliminary experiment.

Table 1 shows a list of hyperparameters for model optimization. We employed an extremely large mini-batch size of 512,000 tokens using the delaying gradient update technique (Bogoychev et al., 2018; Ott et al., 2018). This is because previous studies showed that a large mini-batch size leads to a faster convergence (Ott et al., 2018) and a better generalization (Popel and Bojar, 2018; Bawden et al., 2019; Morishita et al., 2019). We also used a large learning rate of 0.001 to further accelerate the convergence (Goyal et al., 2017; Ott et al., 2018; Liu et al., 2019). We use the `fairseq` toolkit (Ott et al., 2019) for the entire set of experiments. Every reported BLEU score is measured using `SacreBLEU` (Post, 2018).

3.2 Subword Size

For En↔De, we used the subword size of 32,000, which is commonly used in previous studies (Vaswani et al., 2017; Ng et al., 2019). For En↔Ja, we conducted a hyperparameter search for a suitable subword size; Morishita et al. (2019) empirically showed that a small subword size (e.g., 4,000) is superior to those commonly adopted in the literature (e.g., 16,000 and 32,000). Given their findings, we searched for the subword size in the following range: $\{4000, 8000, 16000, 32000\}$.

Table 2 shows that the largest subword size achieves the best performance, which is inconsistent with the result of Morishita et al. (2019). One explanation for this result is that Morishita et al. (2019) conducted an experiment on the ASPEC corpus, whose size (approx. 3M) is much smaller than that of the bitext available for the En↔Ja task. That is, the bitext available for the En↔Ja task is sufficiently large for the model to learn a meaningful representation for each subword unit that is

[1] https://github.com/saffsd/langid.py

[2] We set the minimum length to 1, the maximum length to 250, and the maximum ratio to 3.0.

[3] `--character_coverage`

Base Model	
Architecture	Transformer (big) with FFN size of 8,192
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 Clipping	1.0
Label Smoothing	$\epsilon_{ls} = 0.1$ (Szegedy et al., 2016)
Mini-batch Size	512,000 tokens
Number of Updates	40,000 steps for En↔De and 80,000 steps for En↔Ja
Averaging	Save checkpoint for every 2,000 steps and take an average of last 10 checkpoints
Uni-directional Language Model	
Architecture	`transformer_lm_big` setting available in `fairseq`
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	0.0005
Dropout	0.1
Gradient Clipping	1.0
Weight Decay	0.0
Mini-batch Size	512,000 tokens
Number of Updates	50,000 steps
Masked Language Model	
Architecture	RoBERTa-base (Liu et al., 2019)
Optimizer	Adam ($\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 1 \times 10^{-8}$)
Learning Rate Schedule	Polynomial decay
Warmup Steps	10,000
Max Learning Rate	0.0005
Dropout	0.1
Gradient Clipping	1.0
Weight Decay	0.01
Mini-batch Size	2,048 sentences
Number of Updates	125,000 steps

Table 1: List of hyperparameters for each model.

close to the word level. Thus, we also used the subword size of 32,000 for En↔Ja.

3.3 Large-scale Back-translation

We used the back-translation technique (Sennrich et al., 2016b) to generate large-scale synthetic data. First, we trained models on the bitext for all language pairs. Second, for each language, we fed the monolingual corpus (Section 2.2) to the model. Here, we used the beam search of width 6 and length penalty of 1.0. Finally, we applied length and ratio filtering to the model outputs[4]. The size

Subword Size	En→Ja
4,000	19.2
8,000	19.6
16,000	19.4
32,000	19.7

Table 2: Effectiveness of different subword sizes on the validation set of En↔Ja task.

	En→De	De→En	En→Ja	Ja→En
No filtering	336M	236M	1777M	236M
After filtering	328M	230M	235M	230M

Table 3: Number of sentence pairs in the synthetic data of each language pair

of the synthetic data that we generated for each language direction is shown in Table 3. The size of the synthetic data for En→Ja, which is generated from CommonCrawl, is extremely large. Thus, we randomly subsampled the synthetic data of En→Ja so that its size roughly matches those of De→En and Ja→En.

We searched for an effective setting for incorporating the synthetic data. As the most straightforward starting point, we simply combined bitext and synthetic data and trained the model. Here, we upsampled the bitext so that the model sees the bitext and synthetic data at a 1:1 ratio (Ng et al., 2019). Table 4 shows the result. Here, naively using the synthetic data (BASE+BT) decreased the performance of the model trained with the bitext only (BASE). Given this result, we considered the following two enhancements:

Tagged Back-translation We used the tagged back-translation technique (Tagged-BT) (Caswell et al., 2019), which prepends a special tag token (e.g., $\langle$`BT`$\rangle$) to the source sentence of synthetic data. This simple technique can inform the model about the origin of the given training data, i.e., whether the sentence pair is back-translated. Marie et al. (2020) empirically demonstrated that the model trained with such tagged data can avoid overfitting to the synthetic data. In Table 4, the Tagged-BT (BASE+TAGGED-BT) successfully improves the performance from BASE except for the newstest2019. We suspect that the performance does not improve on newstest2019 because it does not contain the "translationese" text, i.e., human-generated translations, which are reported to be the main source of improvement of back-

[4]For En↔De, we removed sentence pairs that contain sentences longer than 250 tokens. For En↔Ja, we removed sentence pairs such that the English sentence is longer than 250 tokens, or the Japanese sentence is longer than 500 characters.

Setting	newstest 2014	newstest 2018	newstest 2019
BASE	32.2	47.3	42.2
BASE+BT	32.1	45.9	38.8
BASE+TAGGED-BT	33.0	48.0	42.0
BASE (l = 9)+TAGGED-BT	33.1	49.6	42.7
BASE (l = 12)+TAGGED-BT	33.4	49.4	42.3

Table 4: Effectiveness of using the synthetic data on En→De

translation (Bogoychev and Sennrich, 2019; Marie et al., 2020).

Deeper Model We also considered increasing the model size to take advantage of a massive amount of training data. Specifically, we increased the number of layers l from 6 to 9 and 12 (Wang et al., 2019). Table 4 shows that the performances of BASE ($l = 9$)+TAGGED-BT and BASE ($l = 12$)+TAGGED-BT are almost comparable. We determined that BASE ($l = 9$)+TAGGED-BT is the best option by considering the model performance and training efficiency regarding the GPU memory constraints.

3.4 Fine-tuning

Fine-tuning the model with an in-domain news corpus is acknowledged as an extremely important technique for boosting the performance (Sennrich et al., 2016b; Junczys-Dowmunt, 2019; Ng et al., 2019; Bawden et al., 2019). We fine-tuned our models as follows:

En↔De For En↔De, we fine-tuned the model with a collection of newstest2008-2018 and evaluated its performance on newstest2019. For En→De, we only used sentence pairs whose source sentence is originally written in English, i.e., we never used texts with translationese on the source side for fine-tuning. Similarly, for De→En, we used sentence pairs whose source sentence is originally written in German. This way, we ensured that our model does not overfit to the translationese texts; since newstest2019 does not contain translationese texts (Barrault et al., 2019), we expected that newstest2020 does not contain translationese either.

We fine-tuned the model for 200 iterations with a mini-batch size of 20,000 tokens. During the fine-tuning, we fixed the learning rate to 1e-06 for De→En and 1e-05 for En→De. We saved the model every 20 iterations and took an average of the last eight saved models for decoding.

En↔Ja For fine-tuning, we used the Kyoto Free Translation Task (KFTT) corpus and NewsCommentary as the *clean* bitext and NewsCommentary as the *news* bitext. We fine-tuned the models by a two-step procedure, that is, we first fine-tuned with the *clean* bitext for 2,000 steps. Then we fine-tuned with the *news* bitext for 200 steps. We found that the validation performance of this two-step procedure is slightly better than that of the fine-tuning with the *news* bitext only.

3.5 Ensemble

We used the model ensemble method to improve the performance. First, we trained four models with different random seeds. These models were then simultaneously used for computing the score of each candidate during the beam search decoding.

3.6 Right-to-Left Models

We used Right-to-Left (R2L) models for reranking the n-best candidates from Left-to-Right (L2R) models. R2L models generate sentences in reverse order. Suppose that conventional L2R models generate sentences from the beginning-of-the-sentence (BOS) to the end-of-the-sentence (EOS); R2L models generate from EOS to BOS. This reranking technique was independently proposed by Liu et al. (2016) and Sennrich et al. (2016a) to mitigate the search error of L2R models, which may occur around EOS. We trained four R2L models and used their scores for reranking the n-best candidates generated by L2R models (Section 3.5). Specifically, we computed the score of each candidate with both L2R models and R2L models. Then, we took the sum of the two scores and obtained the final score. We sorted this final score and then selected the candidate with the highest score.

3.7 Reranking

We also applied a reranking method based on the scores of several translation (or generative) models, which is closely related to one iteration of Minimum Error Rate Training (MERT) (Och, 2003) often used in Statistical Machine Translation (SMT). The underlying idea is to find the balance of likelihood independently computed from the models.

Suppose we have a set of candidate output sentences for each input in either the validation (training phase) or the test (evaluation phase) sets. In our case, we independently generated n-best candidates using the L2R and R2L models, and obtained $2n$ candidates in total for each. Here, let $\mathcal{C}_i$ represent the set of the obtained $2n$ candidates of the

i-th input.

Next, $P_j(e) \in [0, 1]$ denotes the score of the candidate $e \in \mathcal{C}_i$ obtained from the j-th model, where $j \in \{1, \ldots, J\}$. Let $w_j \in [0, 1]$ be a weighting factor of the j-th model, and $\boldsymbol{w} = (w_1, \ldots, w_J)$ be the vector representation of the weighting factor. We then obtained the most likely candidate $\hat{e}_{i,\boldsymbol{w}}$ from $\mathcal{C}_i$ given the i-th input and $\boldsymbol{w}$ as follows:

$$\hat{e}_{i,\boldsymbol{w}} = \underset{e \in \mathcal{C}_i}{\operatorname{argmax}} \left\{ \sum_{j=1}^{J} w_j \log(P_j(e)) \right\}. \qquad (1)$$

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

$$\widehat{\boldsymbol{w}} = \underset{\boldsymbol{w} \in \mathcal{G}_{\boldsymbol{w}}}{\operatorname{argmax}} \{ \texttt{SacreBLEU}(\widehat{\mathcal{E}}_{\boldsymbol{w}}) \}, \qquad (2)$$

where $\hat{\mathcal{E}}_{\boldsymbol{w}} = (\hat{e}_{i,\boldsymbol{w}})_{i=1}^{I}$ and $\mathcal{G}_{\boldsymbol{w}}$ represent a set of values that w_j can take, namely, $[0, 1]^J$.

For the reranking experiment, we prepared the following generative and translation models to compute $P_j(e)$.

Source-to-Target L2R and R2L Model The Source-to-Target L2R and R2L models are the same as that used for the candidate generation; the ensemble of four L2R models and four R2L models compute the score of each candidate.

Target-to-Source L2R and R2L Model The Target-to-Source (T2S) model translates a sequence in a reverse direction, that is, it translates a given target sequence to a source sequence. For example, if a candidate sentence is generated by the En→De model, we use the De→En model for computing the T2S score.

Uni-directional Language Model We used the uni-directional language model (UniLM) to compute the likelihood of the decoded target sequence. To do this, we trained the Transformer-based language model (Baevski and Auli, 2019) for all languages on monolingual data. We obtained two distinct scores from two normalization methods: (1) simply dividing by the target sequence length (Yee et al., 2019) and (2) *SLOR* (Pauls and Klein, 2012; Lau et al., 2020). A list of hyperparameters is shown in Table 1.

Masked Language Model We also used the pretrained masked language model (MLM) (Devlin et al., 2019) for computing the score. Specifically, we trained the RoBERTa-base (Liu et al., 2019) setting available in `fairseq` on monolingual data. First, we computed the unnormalized log-probabilities by the method described by Wang and Cho (2019). Then, we normalized the probability by (1) dividing by the sequence length and (2) *PenLP* (Vaswani et al., 2017; Lau et al., 2020). A list of hyperparameters is shown in Table 1.

Because the uni-directional language model and MLM both have two distinct variations, we used a total of six models, namely, $J = 6$.

3.8 Post-processing

We converted the decoded target sequence from a sequence of subwords to tokens. Then we applied the Moses `detruecaser` to English and German sequences. We also applied language-specific post-processing as follows:

En↔De We observed that the rare tokens such as Greek letters in the source sequence are sometimes translated into ⟨`UNK`⟩. We handled ⟨`UNK`⟩ in the decoded sequence by copying the corresponding token from the source sequence. We determined the corresponding token by finding the token that does not exist in one of the source-side or target-side vocabularies.

En→Ja We did not take any special measures for ⟨`UNK`⟩[5]. We replaced the English style comma “, ” and period “. ” with the Japanese style “、” and “。” respectively.

Ja→En We observed that the model translates the Japanese vertical bar “｜” to ⟨`UNK`⟩. Thus, we replaced all ⟨`UNK`⟩ with “|”.

3.9 Post-ensemble

Kobayashi (2018) proposed the method of taking the ensemble of multiple models *after* decoding the sequence, namely, post-ensemble (PostEnsemble). The underlying idea of PostEnsemble is to choose “majority-like” candidates by comparing the similarities among candidates. He applied PostEnsemble to the abstractive summarization task and reported that the performance is superior to that of the conventional ensemble.

We used PostEnsemble in En→Ja[6]. Specifically, we adopted the `PostCosE` variant in which the cosine similarity is used as a similarity metric. We created 300 dim fasttext word vectors (Bojanowski et al., 2017) on the Japanese monolingual corpus.

[5] In fact, we never observed ⟨`UNK`⟩ in the decoded test set.

[6] Kobayashi (2018) introduced PostEnsemble as the method that *replaces* the conventional ensemble. Instead, we used two ensemble methods simultaneously.

4 Results

Performance on the Validation Set We show the validation performance of our system in Table 5. We used newstest2019 and the official validation set for En↔De and En↔Ja, respectively, for the validation data. The table shows the effectiveness of incorporating each technique described in Section 3. Each technique consistently improves the performance in most cases. In addition, it is noteworthy that both En→De and De→En models significantly outperform the performance of the best system from last year's shared task (WMT'19).

Performance on the Test Set We show the test set performance that we measured in the OCELoT system[7] in Table 6. The system provides us with the SacreBLEU score and the chrF score (Popović, 2015).

We used the following models for POSTENSEMBLE of Ja→En: (1) model (f) (Table 5), (2) Model (f) with the ensemble of eight models, in which four models are fine-tuned with the *clean* bitext and the other four models are fine-tuned with the *news* bitext, and (3) Model (2) without n-best candidates from the R2L model.

The performance of En→Ja appears significantly better than the validation performance reported in Table 5; this is because OCELoT computes the BLEU score with character-level segmentation, whereas we used the MeCab-based word-level segmentation[8]. We also computed the BLEU score with the MeCab-based segmentation for reference and obtained 25.8 points.

5 Analysis

In this section, we introduce several negative results from our preliminary experiments. Our attempts include the following: (1) filtering synthetic data, (2) incorporating forward-translation, and (3) developing a more sophisticated reranking method. We also analyze the issue regarding the use of brackets in the En→Ja task.

5.1 Negative Results on Synthetic Data Filtering

We applied corpus filtering to the synthetic data created in Section 3.3. The goal of this filtering is to extract and utilize the "clean" subset of synthetic data that may contribute to the model performance. For each of the sentence pairs in the synthetic data, we assigned scores that represent the likelihood of being a sentence pair (Section 5.1.1). Then, we regarded these scores as features for classification; we trained a model classifying clean and noisy sentence pairs (Section 5.1.2). Finally, on the basis of the confidence scores of the classifier, we extracted the presumably clean subset of the synthetic data.

5.1.1 Features

Pointwise HSIC We computed the score for each sentence pair using the pointwise Hilbert-Schmidt independence criterion (PHSIC) (Yokoi et al., 2018), which is a kernel-based co-occurrence measure. Given a set of sentence pairs, PHSIC can assign a high score to a sentence pair that is consistent with the rest of the sentence pairs. To do this, PHSIC utilizes kernel functions and calculates the sentence similarity. Yokoi et al. (2018) applied PHSIC to machine translation corpus filtering and reported promising results. Thus, we also employed PHSIC for synthetic data filtering.

First, we learned the parameters of the PHSIC matrix with a cosine kernel by using all sentence pairs in the bitext, which are represented as sentence embeddings. Then, we used this trained matrix to compute the scores for the synthetic data. We used the following two methods for computing the sentence embeddings: (1) the weighted sum of fasttext vectors (Bojanowski et al., 2017) by smoothed inverse frequency (SIF) weighting (Arora et al., 2017) and (2) the average of final hidden states of the pre-trained MLM. Here, the fasttext vector is the same as the one used for post-ensemble (Section 3.9), and MLM is the one from the reranking (Section 3.7). The word frequency for SIF weighting is calculated from the monolingual corpus.

Cross-entropy from T2S Model We computed the word-normalized conditional cross-entropy using the T2S translation model. For example, the synthetic data generated using the En→De model are scored using the De→En model.

5.1.2 Training a Classifier

We trained a linear support vector machine model that classifies clean and noisy sentence pairs. To train the classifier, we used newstest2009-2019 and the official validation set as clean sentence pairs for En↔De and En↔Ja, respectively. We generated the noisy sentence pairs by randomly adding the noise presented by Wang et al. (2018) to the clean sentence pairs.

[7] https://ocelot.mteval.org/

[8] The use of the MeCab-based segmentation is recommended by SacreBLEU.

ID	Setting	En→De	De→En	En→Ja	Ja→En
(a)	BASE (Section 3.1)	42.4	42.0	19.7	21.6
(b)	BASE ($l = 9$)+TAGGED-BT (Section 3.3)	42.7	42.5	22.0	23.9
(c)	(b) + fine-tuning (Section 3.4)	44.9	42.3	23.1	24.4
(d)	(c) $\times$ 4 (Section 3.5)	45.5	42.8	23.9	25.4
(e)	(d) + 4 $\times$ (c)-R2L (Section 3.6)	45.4	43.6	24.2	25.9
(f)	(e) + reranking (Section 3.7)	45.7	43.8	24.9	26.2
-	The best system in WMT'19	44.9	42.8	-	-

Table 5: Effectiveness of each technique: we use newstest2019 and official validation set for En↔De and En↔Ja respectively. The best result from WMT'19 is unavailable for En↔Ja, because this task has newly appeared this year.

Direction	Setting / ID	BLEU	chrF
En→De	(f) (Table 5)	37.5	0.647
De→En	(f) (Table 5)	43.8	0.690
En→Ja	(f) (Table 5)	40.1	0.343
Ja→En	POSTENSEMBLE	25.5	0.536

Table 6: Performance on WMT'20 Test Set: refer to Table 5 for model ID.

	newstest		
Amount of Synthetic Data Used: r (%)	2014	2018	2019
100	33.0	48.0	42.0
50	32.9	48.4	42.3
33	33.1	47.9	42.2
25	32.9	48.5	42.4

Table 7: Effectiveness of corpus filtering on En→De.

	newstest		
Setting	2014	2018	2019
BASE	32.2	47.3	42.2
BASE+TAGGED-BT	33.0	48.0	42.0
BASE+TAGGED-FT	31.7	46.7	42.1
BASE+TAGGED-BT+TAGGED-FT	33.1	48.3	42.4

Table 8: Effectiveness of incorporating forward-translation and back-translation on En→De.

After training, we classified each sentence pair in the synthetic data. The confidence score of the classifier was used as an overall score that represents the "cleanness" (i.e., quality) of the sentence pair.

5.1.3 Results

We investigated the effectiveness of the synthetic data filtering. First, we sorted the synthetic data according to the score computed with the classifier (Section 5.1.2). Then, we used the top r% of synthetic data for training.

Table 7 shows the results of synthetic data filtering with varying r. We trained the En→De model using the BASE+TAGGED-BT setting. The results showed that our filtering does not seem to improve the performance over the baseline ($r = 100$). One of the possible reasons for this ineffectiveness is the quality of the sentence embeddings used for PHSIC. That is, the use of fasttext and pretrained MLM might be inappropriate. Utilizing more powerful sentence encoders such as Sentence-BERT (Reimers and Gurevych, 2019) and Universal Sentence Encoder (Cer et al., 2018) is an interesting option to explore in the future; however, the methods of acquiring such resources in the constrained setting is not trivial.

5.2 Effectiveness of Incorporating Forward-Translation

Forward-translation (Burlot and Yvon, 2018) is a technique similar to back-translation; the difference is that while back-translation uses the target-side monolingual data, forward-translation uses the source-side monolingual data to generate synthetic data. Bogoychev and Sennrich (2019) reported that forward-translation is effective for improving the translation of texts that are originally written in the source language (i.e., non-translationese texts).

To determine if we can take the best of the two techniques, namely, forward-translation and back-translation, we combined the synthetic data and trained the model. As described in Section 3.3, we prepended a distinct tag to each data source: ⟨FT⟩ and ⟨BT⟩ for data generated by forward-translation and back-translation respectively. Then, we upsampled the bitext, so that the model is fed with the bitext and synthetic data at a 1:0.5:0.5 ratio.

Table 8 shows the result. The model in-

Input	Only one member of the family, then 15-year-old Cassidy Stay, survived.
Reference	家族の中で、ただ一人、当時15歳だったカシディ・ステイさんだけが一命を取り留めた。
Model Output	当時15歳のキャシディ・ステイ(Cassidy Stay)だけが生き残った。
Input	Madam Needjan, pledged the association's support to the hospital and called on other associations to emulate the gesture.
Reference	マダム・ニージャンは、協会の当病院への支援を約束し、他の団体もこうした行為に追随するよう呼びかけた。
Model Output	マダム・ニージャン(Madam Needjan)は、協会が病院を支援することを約束し、他の協会にこのジェスチャーを模倣するよう求めた。

Figure 2: Error analysis of En→Ja translation.

corporating both back-translation and forward-translation (BASE+TAGGED-BT+TAGGED-FT) achieves the best result, however, the improvement was marginal. In addition, the performance of the model with forward-translation only (BASE+TAGGED-FT) was worse than that of the baseline (BASE) in all datasets. Given this result, we only used back-translation and kept the training procedure as simple as possible in our final system.

5.3 Negative Result on Reranking

We actually investigated several different types of reranking algorithms other than the standard grid search described in Section 3.7. For example, we experiment withed optimizing model weights by machine learning based methods such as those using support vector machines, XGBoost (Chen and Guestrin, 2016), and deep neural networks. Unfortunately, none of them worked well. In this competition, we only used the model scores for the reranking. This setting immediately leads the overfitting to the development sets, and hard to extract meaningful generalized weights (rules) that also work well for unseen test data. The development of the methods that can further and consistently improve the quality of translations is our future work for the next year.

5.4 Japanese Text and Brackets

Figure 2 shows examples from the validation set of the En→Ja task. These examples illustrate the weakness of our model, in which the named entities are often inappropriately translated. According to the references in the figure, the named entities must be translated from alphabetical characters to *katakana* (カタカナ), e.g., *Cassidy Stay* to カシディ・ステイ. Although our model successfully translates the named entities in most of the cases, the model also copies original alphabetical characters into the brackets. For example, the model translates *Madam Needjan* to マダム・ニージャン*(Madam Needjan)*. These alphabetical characters damage the BLEU score. We can remove the extra brackets by the rule-based post-processing; however, we find that this naive operation hurts the brevity penalty.

This extra bracket problem seems to reflect the way that the named entities are written in the En↔Ja training data such as KFTT. We should have considered special preprocessing measures in advance to alleviate this problem.

6 Conclusion

In this paper, we described the submission of the joint team of Tohoku, AIP, and NTT (Tohoku-AIP-NTT) to the WMT'20 news translation task. We participated in the En↔De and En↔Ja translation. In preliminary experiments, we attempted new techniques such as synthetic data filtering, forward-translation, and sophisticated reranking. However, none of them was effective. In the submission, we used several standard techniques such as back-translation and fine-tuning. As a result, we achieved the best BLEU score on De→En and strong results in other directions.

Acknowledgments

We would like to thank the two anonymous reviewers for their valuable feedback.

References

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. In *Proceedings of the 5th International Conference on Learning Representations (ICLR 2017)*.

Alexei Baevski and Michael Auli. 2019. Adaptive Input Representations for Neural Language Modeling. In *Proceedings of the 7th International Conference on Learning Representations (ICLR 2019)*.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 Conference on Machine Translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (WMT 2019)*, pages 1–61.

Rachel Bawden, Nikolay Bogoychev, Ulrich Germann, Roman Grundkiewicz, Faheem Kirefu, Antonio Valerio Miceli Barone, and Alexandra Birch. 2019. The University of Edinburgh's Submissions to the WMT19 News Translation Task. In *Proceedings of the Fourth Conference on Machine Translation (WMT 2019)*, pages 103–115.

Nikolay Bogoychev, Kenneth Heafield, Alham Fikri Aji, and Marcin Junczys-Dowmunt. 2018. Accelerating Asynchronous Stochastic Gradient Descent for Neural Machine Translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 2991–2996.

Nikolay Bogoychev and Rico Sennrich. 2019. Domain, Translationese and Noise in Synthetic Data for Neural Machine Translation. *arXiv preprint arXiv:1911.03362*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics (TACL 2017)*, 5:135–146.

Franck Burlot and François Yvon. 2018. Using Monolingual Data in Neural Machine Translation: a Systematic Study. In *Proceedings of the Third Conference on Machine Translation (WMT 2018)*, pages 144–155.

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

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal Sentence Encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174.

Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2019)*, pages 4171–4186.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. 2017. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour. *arXiv preprint arXiv:1706.02677*.

Marcin Junczys-Dowmunt. 2019. Microsoft Translator at WMT 2019: Towards Large-Scale Document-Level Neural Machine Translation. In *Proceedings of the Fourth Conference on Machine Translation (WMT 2019)*, pages 225–233.

Hayato Kobayashi. 2018. Frustratingly Easy Model Ensemble for Abstractive Summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 4165–4176.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180.

Philipp Koehn, Huda Khayrallah, Kenneth Heafield, and Mikel L. Forcada. 2018. Findings of the WMT 2018 Shared Task on Parallel Corpus Filtering. In *Proceedings of the Third Conference on Machine Translation (WMT 2018)*, pages 726–739.

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.

Jey Han Lau, Carlos Armendariz, Matthew Purver, Chang Shu, and Shalom Lappin. 2020. How Furiously Can Colourless Green Ideas Sleep? Sentence Acceptability in Context. *Transactions of the Association for Computational Linguistics*, 8:296–310.

Lemao Liu, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. 2016. Agreement on Target-bidirectional Neural Machine Translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2016)*, pages 411–416.

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*.

Benjamin Marie, Raphael Rubino, and Atsushi Fujita. 2020. Tagged Back-translation Revisited: Why Does It Really Work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 5990–5997.

Makoto Morishita, Jun Suzuki, and Masaaki Nagata. 2019. NTT Neural Machine Translation Systems at WAT 2019. In *Proceedings of the 6th Workshop on Asian Translation (WAT 2019)*, pages 99–105.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook FAIR's WMT19 News Translation Task Submission. In *Proceedings of the Fourth Conference on Machine Translation (WMT 2019)*, pages 314–319.

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 2003)*, 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 (Demonstrations)*, pages 48–53.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling Neural Machine Translation. In *Proceedings of the Third Conference on Machine Translation (WMT 2018)*, pages 1–9.

Adam Pauls and Dan Klein. 2012. Large-Scale Syntactic Language Modeling with Treelets. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, pages 959–968.

Martin Popel and Ondřej Bojar. 2018. Training Tips for the Transformer Model. *The Prague Bulletin of Mathematical Linguistics*, 110(1):43–70.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395.

Matt Post. 2018. A Call for Clarity in Reporting BLEU Scores. In *Proceedings of the Third Conference on Machine Translation (WMT 2018)*, pages 186–191.

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 2019)*, pages 3982–3992.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Edinburgh Neural Machine Translation Systems for WMT 16. In *Proceedings of the First Conference on Machine Translation (WMT 2016)*, pages 371–376.

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

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016c. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 1715–1725.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016)*, pages 2818–2826.

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 2017)*, pages 5998–6008.

Alex Wang and Kyunghyun Cho. 2019. BERT has a Mouth, and It Must Speak: BERT as a Markov Random Field Language Model. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 30–36.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. 2019. Learning Deep Transformer Models for Machine Translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 1810–1822.

Rui Wang, Benjamin Marie, Masao Utiyama, and Eiichiro Sumita. 2018. NICT's Corpus Filtering Systems for the WMT18 Parallel Corpus Filtering Task. In *Proceedings of the Third Conference on Machine Translation (WMT 2018)*, pages 963–967.

Kyra Yee, Yann Dauphin, and Michael Auli. 2019. Simple and Effective Noisy Channel Modeling for Neural Machine Translation. 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 2019)*, pages 5696–5701.

Sho Yokoi, Sosuke Kobayashi, Kenji Fukumizu, Jun Suzuki, and Kentaro Inui. 2018. Pointwise HSIC: A Linear-Time Kernelized Co-occurrence Norm for Sparse Linguistic Expressions. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 1763–1775.