NAIST-NICT WMT'23 General MT Task Submission

Hiroyuki Deguchi[1,2] **Kenji Imamura**[2] **Yuto Nishida**[1]
Yusuke Sakai[1] **Justin Vasselli**[1] **Taro Watanabe**[1]
[1]Nara Institute of Science and Technology
[2]National Institute of Information and Communications Technology
{deguchi.hiroyuki.db0, nishida.yuto.nu8,sakai.yusuke.sr9,
vasselli.justin_ray.vk4, taro}@is.naist.jp
kenji.imamura@nict.go.jp

Abstract

In this paper, we describe our NAIST-NICT submission to the WMT'23 English ↔ Japanese general machine translation task. Our system generates diverse translation candidates and reranks them using a two-stage reranking system to find the best translation. First, we generated 50 candidates each from 18 translation methods using a variety of techniques to increase the diversity of the translation candidates. We trained seven models per language direction using various combinations of hyperparameters. From these models we used various decoding algorithms, ensembling the models, and using kNN-MT (Khandelwal et al., 2021). We processed the 900 translation candidates through a two-stage reranking system to find the most promising candidate. In the first step, we compared 50 candidates from each translation method using DrNMT (Lee et al., 2021) and returned the candidate with the best score. We ranked the final 18 candidates using COMET-MBR (Fernandes et al., 2022) and returned the best score as the system output. We found that generating diverse translation candidates improved translation quality using the well-designed reranker model.

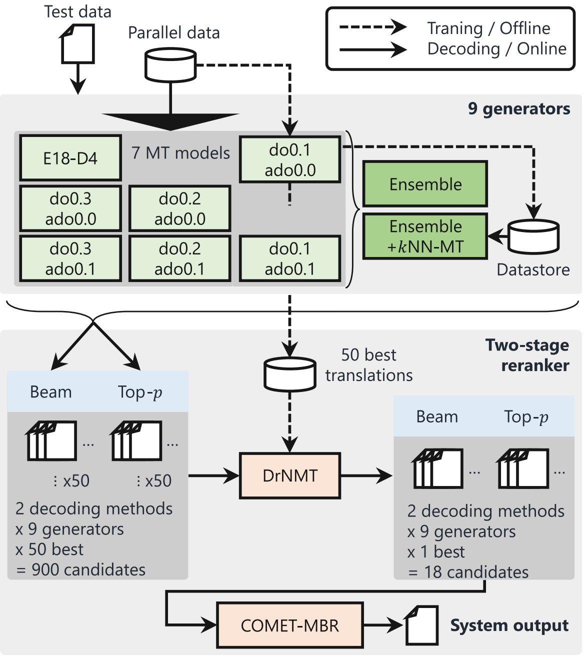

Figure 1: Overview of our system. "E18-D4" denotes "18-layer encoder and 4-layer decoder", and "do" and "ado" denote "dropout" and "dropout after applying attention softmax", respectively.

1 Introduction

We participated in the WMT'23 general machine translation task for English-to-Japanese (En-Ja) and Japanese-to-English (Ja-En) translation. Our team aimed to improve translation performance using only the provided parallel data. Our system generates diverse translation candidates and reranks them using a two-stage reranking system to find the best translation.

Figure 1 shows an overview of our system. We trained 7 Transformer (Vaswani et al., 2017) NMT models per language direction using various combinations of hyperparameters. The translation generator consists of 9 instances: 7 MT models, the ensemble model, and a kNN-MT (Khandelwal et al., 2021) system that interpolates tokens from retrieved examples using the ensemble model. The generator generates the 50-best translations each from two decoding methods: beam search and top-p sampling. This combination allows the generator to find diverse translation candidates. Next, the 900 candidates (9 generators × 2 decoding methods × 50 best) are passed to our two-stage reranker to find the best translation. The first step of reranking uses DrNMT (Lee et al., 2021) to rerank the 50-best translation candidates to select the 1-best translation from each of the 18 generator and decoding method combinations. DrNMT is trained to maximize the BLEU (Papineni et al., 2002) score, whereas we use the second step reranking to find the highest COMET (Rei et al., 2020) score expectation from the remaining candidates. The 18

Proceedings of the Eighth Conference on Machine Translation (WMT), pages 110–118
December 6–7, 2023. ©2023 Association for Computational Linguistics

candidates from the first step are reranked using COMET-MBR (Fernandes et al., 2022) to select the best translation that is returned by the system.

Our experiments show that our two-stage reranker outperforms the BLEU, chrF, and COMET scores by DrNMT alone, and the BLEU and chrF scores by COMET-MBR alone in both En-Ja and Ja-En translation tasks on wmttest2022 (Kocmi et al., 2022).

2 Preprocessing

For the training data, we used the provided bilingual parallel data, which included JParaCrawl v3 (Morishita et al., 2020), News Commentary v18.1, Wiki Titles v3, WikiMatrix, the Japanese-English Subtitle Corpus (Pryzant et al., 2017), the Kyoto Free Translation Task Corpus (Neubig, 2011), and the Web Inventory of Transcribed and Translated Talks (Cettolo et al., 2012). We did not backtranslate the monolingual data due to resource constraints for training MT models and a reranker model.

As the English translation of the Japanese-English Subtitle Corpus was only available in lowercase, we trained a Moses truecaser (Koehn et al., 2007) using the other corpora to add capitalization to the subtitle corpus. After truecasing, the first letter of each sentence was capitalized using detruecasing to produce sentence-case English text that matched the casing in the other corpora.

We cleaned the data by removing duplicate lines and applying language filtering. Because much of the training data were crawled from the internet, we used fasttext (Joulin et al., 2016a,b) to predict the language of each sentence and removed sentences that were not predicted to be in the correct language. This helped to reduce noise in the dataset by removing sentences with garbage tokens.

We tokenized text into subword units using sentencepiece (Kudo and Richardson, 2018). Since our system generates many candidates using multiple models, we preliminary measured the generation speed and selected the number of vocabulary with the fastest decoding. Our initial experiments demonstrated that when the target language was Japanese, a vocabulary size of 32k resulted in fewer tokens needing to be generated, which increased the translation speed. However, when the target language was English, a vocabulary size of 16k was faster than an English vocabulary of 32k. Therefore, we trained separate dictionaries for English and Japanese, with the English-side dictionary containing nearly 16k tokens and the Japanese-side containing nearly 32k tokens. The character coverage of the tokenizers also varied between languages. We trained the English tokenizer with 100% character coverage, whereas character coverage for Japanese was 99.98%.

	#sentence pairs
No filter	33,875,242
+ deduplicate	29,940,444
++ language filter	29,279,161
+++ length filter	27,880,378

Table 1: Number of sentence pairs in the training data after each preprocessing step.

Generator: MT model	
Architecture	Transformer big
Embedding dimension	1,024
FFN inner dimension	8,192
Dropout (do)	0.1
Attention dropout (ado)	0.0
Loss function	label smoothed cross entropy
Label smoothing	$\epsilon = 0.1$
Optimizer	Adam ($\beta_1 = 0.9, \beta_2 = 0.98$)
Learning rate (LR)	1e-3
LR scheduler	inverse square root
Warm-up steps	4,000
Global batch size	Roughly 512,000 tokens
Training steps	60,000
Reranker: DrNMT	
Architecture	XLM-R large
Classifier dropout	0.2
Loss function	(Section 3.2.1)
Optimizer	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.98$)
Learning rate (LR)	5e-5
LR scheduler	polynomial decay
Warm-up steps	8,000
Global batch size	512 sentences * 50 hypotheses

Table 2: Hyperparameters of the models we trained.

After subword segmentation, we removed all sentences shorter than one token or longer than 250 tokens. We also removed all sentences in which the number of tokens in one language was more than double the number of tokens in the translation, i.e., the ratio of tokens between the source and target was >2.0. The number of sentence pairs before/after preprocessing is shown in Table 1.

3 Translation System

3.1 Generator

The generator generates diverse translation candidates from multiple models and multiple decoding methods. The generator consists of seven MT models, an ensemble of the seven models, and the ensemble enhanced with kNN-MT (Khandelwal et al., 2021) for a total of 9 instances.

3.1.1 MT models

The 7 MT models are trained from the provided parallel data. Our MT model with the default setting is shown in Table 2. Six of the seven models vary from the default setting only in dropout and attention dropout, while the last varies the number of layers. Our model has two types of dropouts whose values are varied: “dropout (do)” and “attention dropout (ado)”. The dropout (do) is applied to the token embedding layer and the outputs of the sub-layers within each layer, i.e., the outputs of the attention layers and feed-forward network. The attention dropout (ado) is applied after softmax to the attention weights, i.e., before multiplying the values. Six models are trained with varying dropouts, one for each combination of do $= \{0.1, 0.2, 0.3\}$ and ado $= \{0.0, 0.1\}$. In addition to the models that vary dropout, we trained a deep-shallow model (Kasai et al., 2021), which has 18 encoder layers and 4 decoder layers. For each model, we averaged the parameters of the last 10 checkpoints (10,000 training steps).

3.1.2 kNN-MT

Datastore construction kNN-MT (Khandelwal et al., 2021) requires a datastore to be constructed to store the translation examples to be accessed during decoding. Let $\boldsymbol{x} = (x_1, \ldots, x_{|\boldsymbol{x}|}) \in \mathcal{V}_X^{|\boldsymbol{x}|}$ and $\boldsymbol{y} = (y_1, \ldots, y_{|\boldsymbol{y}|}) \in \mathcal{V}_Y^{|\boldsymbol{y}|}$ denote a source sentence and target sentence, respectively, where $|\cdot|$ is the length of the sequence, and $\mathcal{V}_X$ and $\mathcal{V}_Y$ are the vocabularies of the source language and target language, respectively. The datastore for kNN-MT consists of translation examples in the form of key–value pairs, as shown in Figure 2. Each target token y_t from the translation examples is stored in the datastore with a d-dimensional key ($\in \mathbb{R}^d$), which is the representation of the translation context $(\boldsymbol{x}, \boldsymbol{y}_{<t})$ obtained from the decoder of the pre-trained NMT model. The datastore $\mathcal{M} \subseteq \mathbb{R}^d \times \mathcal{V}_Y$ is formally defined as a set of tuples as follows:

$$\mathcal{M} = \{(f(\boldsymbol{x}, \boldsymbol{y}_{<t}), y_t) \mid (\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}, 1 \leq t \leq |\boldsymbol{y}|\}, \tag{1}$$

where $\mathcal{D}$ denotes parallel data and $f : \mathcal{V}_X^{|\boldsymbol{x}|} \times \mathcal{V}_Y^{t-1} \rightarrow \mathbb{R}^d$ returns the intermediate representation of the final decoder layer from the source sentence and prefix target tokens. The representation used as the key vector is the vector that is passed into the final feed-forward layer (Khandelwal et al., 2021).

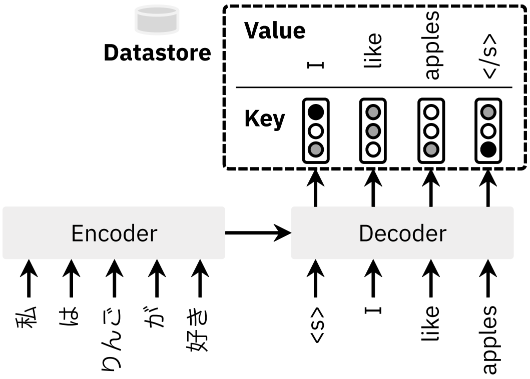

Figure 2: Datastore construction.

kNN index	
Implementation	FAISS
Index	`IndexIVFPQ`
# of entries	
Ja (En-Ja)	732,222,393
En (Ja-En)	836,254,078
# of centroids	131,072
# of bits in PQ	8 bits
# of sub-vectors in PQ	$M = 64$
Vector pre-transform	OPQ (Ge et al., 2014)
Decoding	
# of retrieved tokens	$k = 64$
Temperature of $p_{k\mathrm{NN}}$	$\tau = 100$
Weight for $p_{k\mathrm{NN}}$	$\lambda = 0.1$
# of probed clusters	32

Table 3: Hyperparameters of our kNN indexes and kNN-MT.

In our system, we used the model trained with the default settings (as seen in Table 2) to obtain the keys for the datastore.

kNN index To search the k-nearest-neighbor tokens efficiently, we used FAISS (Johnson et al., 2019). For the kNN indexes, we used `faiss.IndexIVFPQ` which consists of an inverted file index (IVF) that performs k-means clustering and product quantization (PQ) (Jégou et al., 2011) which divides a vector into M sub-vectors and performs vector quantization in each subspace. Note that in IVFPQ, the codewords of PQ are learned from the residual vectors from the centroids of the IVF. Additionaly, we used optimized PQ (OPQ) (Ge et al., 2014) to reduce the quantization error of PQ. The hyperparameters of our kNN indexes are summarized in Table 3.

Decoding During decoding, kNN-MT retrieves the k-nearest-neighbor key–value pairs $\{(\boldsymbol{k}_i, v_i)\}_{i=1}^{k} \subseteq \mathbb{R}^d \times \mathcal{V}_Y$ from the datastore $\mathcal{M}$ using the query vector $f(\boldsymbol{x}, \boldsymbol{y}_{<t})$ at timestep t. Next, $p_{k\mathrm{NN}}$ is calculated as follows:

$$p_{k\mathrm{NN}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}) \propto \sum_{i=1}^{k} \mathbb{1}_{y_t=v_i} \exp \frac{-\|\boldsymbol{k}_i - f(\boldsymbol{x}, \boldsymbol{y}_{<t})\|_2^2}{\tau}, \quad (2)$$

where τ is the temperature parameter for $p_{k\mathrm{NN}}$. Then, kNN-MT generates the output probability by computing the linear interpolation between the kNN and MT probabilities, $p_{k\mathrm{NN}}$ and p_{MT}, respectively:

$$P(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}) = \lambda p_{k\mathrm{NN}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}) + (1-\lambda) p_{\mathrm{MT}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}). \quad (3)$$

kNN-MT with the ensemble model kNN-MT is typically used with a single model, whereas in our system, we obtain the output probability for each token by interpolating between the kNN probability and the probability from the ensemble model. The output probability from the ensemble kNN-MT is formulated by defining p_{MT} in Equation 3 as follows:

$$p_{\mathrm{MT}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}; \boldsymbol{\theta}) = \frac{1}{|\boldsymbol{\theta}|}(p_{\mathrm{MT}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}; \theta_1) + \ldots + p_{\mathrm{MT}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}; \theta_{|\boldsymbol{\theta}|}), \quad (4)$$

where $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_{|\boldsymbol{\theta}|}\}$ denotes the parameters of the trained MT models; $|\boldsymbol{\theta}| = 7$ in our system. The kNN-MT decoding interpolated between the token distribution of the retrieved translation context tokens and the full ensemble of models. As such, the weight assigned to the kNN token distribution was kept small so as not to overpower the information from the ensemble. We used $\lambda = 0.1$ and $\tau = 100$ in the kNN-MT decoding shown in Table 3.

3.1.3 Decoding algorithms

From each model, we output the 50 best hypotheses generated using beam search with a beam width of 50. For diversity, we generated another 50 hypotheses using top-p sampling with $p = 0.7$ and a beam width of 50. We formed an ensemble of models to produce two more sets of 50 hypothesis sentences from beam search and top-p sampling.

3.2 Reranker

We use a two-stage reranker consisting of an intra-system reranker, which selects the best of the 50 hypotheses from each system, and an inter-system reranker, which selects the best hypothesis from the 18 remaining candidate translations.

3.2.1 DrNMT

Discriminative reranking for NMT (DrNMT) (Lee et al., 2021) is a discriminative model that learns to predict the distributions of the evaluation scores of a set of translation hypotheses given a source sentence. DrNMT is similar to a quality estimation model (Zerva et al., 2022), but it is optimized to distinguish the better translation from hypotheses generated from a single system. In addition, it cannot be used for comparing inter-systems because the weights for features are tuned using the translation hypotheses of the development set. We used BLEU (Papineni et al., 2002) as the evaluation metric for this first-stage reranker.

Model The DrNMT model takes as input a source sentence $\boldsymbol{x} \in \mathcal{V}_X^{|\boldsymbol{x}|}$ concatenated with a hypothesis translation $\boldsymbol{y}^{(j)} \in \mathcal{V}_Y^{|\boldsymbol{y}^{(j)}|}$. The DrNMT model passes this into XLM-R (Conneau et al., 2020), which is a multilingual pre-trained encoder. The hidden state of the [CLS] token then represents the combination of the source and hypothesis and is converted into a scalar score by the classification head of RoBERTa (Liu et al., 2019). We used an input dimension of 1,024, a hidden dimension of 768, and output dimension of 1. The activation function for the classification head is tanh.

Objective The objective function minimizes the KL-divergence between the DrNMT model distribution and the distribution of BLEU scores of the n-best hypotheses; that is, the objective function $\mathcal{L}(\theta)$ is as follows:

$$\mathcal{L}(\theta) = \mathrm{KL}[p_T \parallel p_M] = -\sum_{j=1}^{n} p_T\left(\boldsymbol{y}^{(j)}, \boldsymbol{y}^*\right) \log p_M\left(\boldsymbol{y}^{(j)}|\boldsymbol{x}; \theta\right), \quad (5)$$

where n denotes the number of translation hypotheses, and p_M and p_T denote the distributions of the DrNMT model and BLEU scores, respectively. $\boldsymbol{y}^*$ denotes the reference translation of $\boldsymbol{x}$. The BLEU scores are normalized using min-max scaling and the distribution of the BLEU scores is emphasized using the temperature coefficient T. In this paper, we use $T = 0.5$.

Training We trained the DrNMT model using the 50 best translation hypotheses generated by the model with the default configuration for each source sentence over the entire training set, i.e., 28M source sentences. The model is trained using early stopping, which selects the checkpoint with the maximum BLEU score in the validation set.

Tuning The score of the first-stage reranker is a weighted sum of the DrNMT model score, translation model score, and length penalty. This combination of scores is similar to minimum error rate training (Och, 2003). The weights that maximize the BLEU score of the validation set were learned and used.

Implementation We used the implementation published in FAIRSEQ[1]. Note that this implementation uses SACREBLEU (Post, 2018) to compute the BLEU scores. We modified the published code of DrNMT to change the SACREBLEU tokenizers according to the target language because the published implementation always calls the English tokenizer.

3.2.2 COMET-MBR

COMET-MBR (Fernandes et al., 2022) performs minimum Bayes risk (MBR) decoding (Kumar and Byrne, 2004; Eikema and Aziz, 2020; Müller and Sennrich, 2021; Eikema and Aziz, 2022) using a COMET (Rei et al., 2020, 2022) model trained on direct assessments. A translation $\hat{\boldsymbol{y}}^{\text{MAP}} \in \mathcal{V}_Y^{|\boldsymbol{y}^*|}$ is typically generated using maximum-a-posteriori (MAP) decoding as follows:

$$\hat{\boldsymbol{y}}^{\text{MAP}} = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} \log p(\boldsymbol{y}|\boldsymbol{x}), \tag{6}$$

where $\mathcal{Y} \subseteq \bigcup_{i=1}^{\infty} \mathcal{V}_Y^i$ is the search space of target sentences. In MBR decoding, instead of finding the most probable translation, the goal is to find the translation that minimizes the Bayes risk as follows:

$$\begin{aligned} \hat{\boldsymbol{y}}^{\text{MBR}} &= \operatorname*{argmax}_{\boldsymbol{h} \in \bar{\mathcal{Y}}} \underbrace{\mathbb{E}_{\boldsymbol{y}' \sim p(\boldsymbol{y}|\boldsymbol{x})}[u(\boldsymbol{y}', \boldsymbol{h})]}, \\ &\approx \frac{1}{m} \sum_{j=1}^{m} u(\boldsymbol{y}^{(j)}, \boldsymbol{h}) \end{aligned} \tag{7}$$

where $\bar{\mathcal{Y}} \subseteq \mathcal{Y}$ is a set of translation hypotheses and $u : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is the utility function. In this paper, we used COMET[2] (Rei et al., 2020, 2022) as utility function u. Note that we share the hypotheses $\bar{\mathcal{Y}}$ and the sample set for expectation estimation $\{\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(m)}\}$, except for $\boldsymbol{h}$, i.e., $\{\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(m)}\} = \bar{\mathcal{Y}} \setminus \{\boldsymbol{h}\}$. Thus, given a candidate set, the computational complexity of MBR decoding is in the order of $\mathcal{O}(m^2)$, which results in a slower inference speed when m is large.

3.2.3 Two-stage reranking

We applied two-stage reranking with DrNMT and COMET-MBR, which allowed us to use each model for the task it was trained to handle best, to optimize for two metrics and to reduce the inference speed of reranking.

In the first stage, DrNMT (Lee et al., 2021) is used to prune the 50 candidates for each candidate set generated from each of the 18 combinations of decoding methods and generators. As DrNMT is trained to rerank the n-best candidates from a single model, it is ideally suited to the task of reranking the candidates generated with the same combination of model and decoding method, i.e., within a system. In the second stage, COMET-MBR (Fernandes et al., 2022) is used to select the system output from the 18 candidate translations selected by DrNMT.

We use COMET-MBR to rerank the best outputs of each system because COMET was trained on translation scores from the output of various models from previous WMT translation tasks, making it well suited to inter-system comparisons. Each of the two stages is trained to optimize a different metric: Stage one uses BLEU, which evaluates surface forms, whereas stage two uses COMET, which evaluates semantics. Additionally, the inference speed of COMET-MBR makes it time-consuming for large candidate sets, but pruning with DrNMT, which performs inference in a single forward computation, reduces the computational cost.

4 Experimental Results

We evaluated the translation performance of our system on wmttest2022 (Kocmi et al., 2022). We measured the BLEU and chrF scores using SACREBLEU, and the COMET score using `Unbabel/wmt22-comet-da`. The models of our

[1] https://github.com/facebookresearch/fairseq/tree/main/examples/discriminative_reranking_nmt

[2] https://huggingface.co/Unbabel/wmt22-comet-da

		En-Ja			Ja-En		
Method	# of cands.	BLEU	chrF	COMET	BLEU	chrF	COMET
1-best of the ensemble	1	25.5	34.0	86.4	23.1	48.0	80.9
DrNMT	50	26.7	34.7	86.6	23.7	48.4	81.1
COMET-MBR	900	26.1	35.4	**90.5**	22.0	48.0	**84.1**
DrNMT+COMET-MBR (ours)	900	**27.1**	**35.6**	88.4	**24.4**	**49.3**	82.4
DrNMT+Oracle-COMET-DA	900	30.5	39.1	90.2	29.0	53.7	85.5

Table 4: Experimental results of our system on wmttest2022. "# of cands." denotes the number of candidates generated by the translation generator. The **bold scores** indicate the best scores in each translation direction.

generator were trained using FAIRSEQ (Ott et al., 2019). We used KNN-SEQ[3] (Deguchi et al., 2023) for kNN-MT generation built on top of FAIRSEQ. The first stage of our reranker, DrNMT, was also built using FAIRSEQ, whereas COMET-MBR was built using COMET (Rei et al., 2020).

Table 4 shows the results of our system. In the table, the translation candidates of "1-best of the ensemble" were generated using the ensemble model without kNN-MT using beam search decoding. The candidates of "DrNMT" were generated using the ensemble model and the 50-best translations were obtained using beam search decoding. As DrNMT uses the log probability of an MT model for inference, it cannot compare candidates generated by different MT models or generation methods. The results show that DrNMT not only improved the BLEU scores but also the chrF and COMET scores from the 1-best translation, despite only being trained to maximize the BLEU score. "COMET-MBR" reranks all candidates, i.e., 900 translations (= 9 generators × 2 decoding methods × 50 best candidates). COMET-MBR achieved the highest COMET scores for both En-Ja and Ja-En, but the BLEU and chrF scores were not improved for Ja-En, and the inference speed of COMET-MBR with 900 translation candidates was slow. Our primary system used "DrNMT+COMET-MBR" described in Section 3.2.3. This method obtained higher scores for all metrics compared with using DrNMT alone in both translation directions, in addition to the highest BLEU and chrF scores overall. To summarize, our results show that using the rerankers appropriately as intra- and inter-system rerankers is effective for improving translation quality. DrNMT+Oracle-COMET-DA is the oracle performance of the second stage reranker, i.e., the score computed by the largest COMET-DA score for candidates after reranking the 50-best of each system using DrNMT (first stage reranker). Our DrNMT+COMET-MBR scores underperformed the oracle performance, and we leave its improvement for future work.

In addition, we investigated which hypothesis was selected as the system output in DrNMT+COMET-MBR. Figure 3 shows the percentages of counts selected as the system output. In the figure, when the system output comes from multiple hypotheses, i.e., duplicated hypotheses are selected, each hypothesis is counted as selected. The results show that the hypotheses generated by beam search of the ensemble and ensemble+kNN-MT models were selected as the system outputs roughly 40% in En-Ja and 50% in Ja-En. Thus, half of the system outputs were not selected from hypotheses generated from the ensemble model using beam search. Therefore, it can be said that "DrNMT+COMET-MBR" outperformed "DrNMT" by selecting from the hypotheses generated by various generators and various decoding methods.

5 Conclusion

In this paper, we described our submission as a joint team of NAIST and NICT (NAIST-NICT) to the WMT'23 general MT task. We participated in this task in the En-Ja and Ja-En translation directions. We built our system using a diverse translation generator and two-stage reranker. In future work, we will investigate qualitatively how translation diversity contributes to translation quality.

Limitations

A limitation of our system is its reliance on large computation resources. As our system generates 50 candidates using two decoding methods from

[3] https://github.com/naist-nlp/knn-seq

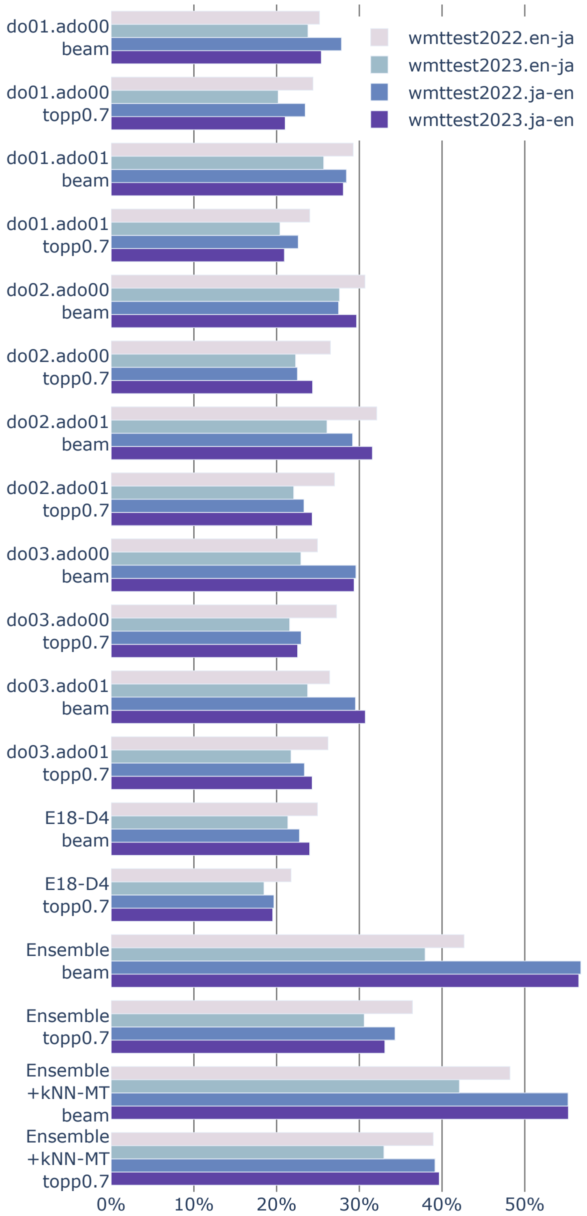

Figure 3: Percentages of counts selected as the system output by COMET-MBR.

each of the nine generators, it requires significant resources. The beam size of 50 is larger than most machine translators and requires more computing power (memory and time).

Note that the reranking approach cannot output translations of higher quality than those translated by the generators.

Ethics Statement

Our system did not restrict the training data and the translator's outputs. Therefore, similar to other translation systems, it may generate factually inaccurate translations.

References

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. WIT3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation*, pages 261–268, Trento, Italy. European Association for Machine Translation.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Hiroyuki Deguchi, Hayate Hirano, Tomoki Hoshino, Yuto Nishida, Justin Vasselli, and Taro Watanabe. 2023. knn-seq: Efficient, extensible knn-mt framework. *arXiv preprint arXiv:2310.12352*.

Bryan Eikema and Wilker Aziz. 2020. Is MAP decoding all you need? the inadequacy of the mode in neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4506–4520, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Bryan Eikema and Wilker Aziz. 2022. Sampling-based approximations to minimum Bayes risk decoding for neural machine translation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10978–10993, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Patrick Fernandes, António Farinhas, Ricardo Rei, José G. C. de Souza, Perez Ogayo, Graham Neubig, and Andre Martins. 2022. Quality-aware decoding for neural machine translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1396–1412, Seattle, United States. Association for Computational Linguistics.

Tiezheng Ge, Kaiming He, Qifa Ke, and Jian Sun. 2014. Optimized product quantization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):744–755.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid. 2011. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128.

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

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016a. Fasttext.zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016b. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Jungo Kasai, Nikolaos Pappas, Hao Peng, James Cross, and Noah Smith. 2021. Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation. In *International Conference on Learning Representations*.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. In *International Conference on Learning Representations (ICLR)*.

Tom Kocmi, Rachel Bawden, Ondřej Bojar, Anton Dvorkovich, Christian Federmann, Mark Fishel, Thamme Gowda, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Rebecca Knowles, Philipp Koehn, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Michal Novák, Martin Popel, and Maja Popović. 2022. Findings of the 2022 conference on machine translation (WMT22). In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 1–45, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

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, Prague, Czech Republic. Association for Computational Linguistics.

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

Shankar Kumar and William Byrne. 2004. Minimum Bayes-risk decoding for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 169–176, Boston, Massachusetts, USA. Association for Computational Linguistics.

Ann Lee, Michael Auli, and Marc'Aurelio Ranzato. 2021. Discriminative reranking for neural machine translation. 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)*, pages 7250–7264, Online. Association for Computational Linguistics.

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.

Makoto Morishita, Jun Suzuki, and Masaaki Nagata. 2020. JParaCrawl: A large scale web-based English-Japanese parallel corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 3603–3609, Marseille, France. European Language Resources Association.

Mathias Müller and Rico Sennrich. 2021. Understanding the properties of minimum Bayes risk decoding in neural machine translation. 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)*, pages 259–272, Online. Association for Computational Linguistics.

Graham Neubig. 2011. The Kyoto free translation task. http://www.phontron.com/kftt.

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*, pages 160–167, Sapporo, Japan. Association for Computational Linguistics.

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, Minneapolis, Minnesota. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

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

Reid Pryzant, Yongjoo Chung, Dan Jurafsky, and Denny Britz. 2017. JESC: japanese-english subtitle corpus. *CoRR*, abs/1710.10639.

Ricardo Rei, José G. C. de Souza, Duarte Alves, Chrysoula Zerva, Ana C Farinha, Taisiya Glushkova, Alon Lavie, Luisa Coheur, and André F. T. Martins. 2022. COMET-22: Unbabel-IST 2022 submission for the metrics shared task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 578–585, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I Guyon, U V Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Chrysoula Zerva, Frédéric Blain, Ricardo Rei, Piyawat Lertvittayakumjorn, José G. C. de Souza, Steffen Eger, Diptesh Kanojia, Duarte Alves, Constantin Orăsan, Marina Fomicheva, André F. T. Martins, and Lucia Specia. 2022. Findings of the WMT 2022 shared task on quality estimation. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 69–99, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.