Towards Robust k-Nearest-Neighbor Machine Translation

Hui Jiang[1]*, **Ziyao Lu**[2]*, **Fandong Meng**[2], **Chulun Zhou**[2], **Jie Zhou**[2], **Degen Huang**[3] and **Jinsong Su**[1,4]†
[1]School of Informatics, Xiamen University, China
[2]Pattern Recognition Center, WeChat AI, Tencent Inc, China
[3]Dalian University of Technology, China
[4]Pengcheng Laboratory, China
hjiang@stu.xmu.edu.cn {ziyaolu,fandongmeng,chulunzhou,withtomzhou}@tencent.com
huangdg@dlut.edu.cn jssu@xmu.edu.cn

Abstract

k-Nearest-Neighbor Machine Translation (kNN-MT) becomes an important research direction of NMT in recent years. Its main idea is to retrieve useful key-value pairs from an additional datastore to modify translations without updating the NMT model. However, the underlying retrieved noisy pairs will dramatically deteriorate the model performance. In this paper, we conduct a preliminary study and find that this problem results from not fully exploiting the prediction of the NMT model. To alleviate the impact of noise, we propose a confidence-enhanced kNN-MT model with robust training. Concretely, we introduce the NMT confidence to refine the modeling of two important components of kNN-MT: kNN distribution and the interpolation weight. Meanwhile we inject two types of perturbations into the retrieved pairs for robust training. Experimental results on four benchmark datasets demonstrate that our model not only achieves significant improvements over current kNN-MT models, but also exhibits better robustness. Our code is available at https://github.com/DeepLearnXMU/Robust-knn-mt.

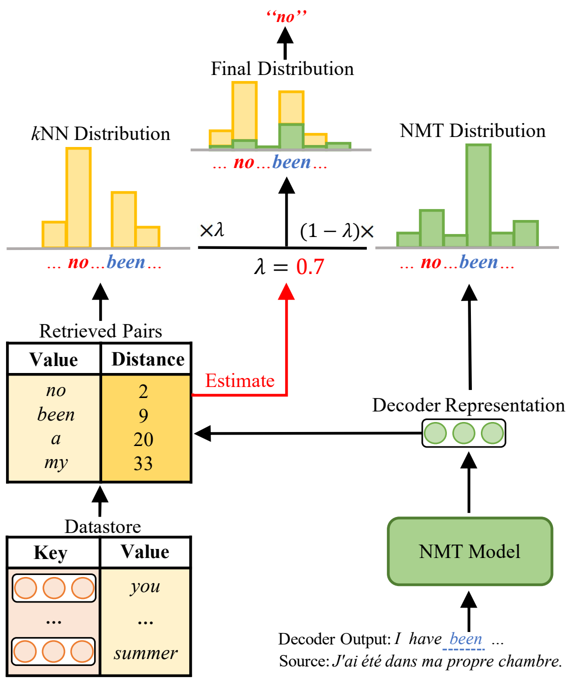


Figure 1: An example of kNN-MT model using dynamically estimation of λ (Zheng et al., 2021a; Jiang et al., 2021).

1 Introduction

As a commonly-used paradigm of retrieval-based neural machine translation (NMT), k-Nearest-Neighbor Machine Translation (kNN-MT) has proven to be effective in many studies (Khandelwal et al., 2021; Zheng et al., 2021a; Jiang et al., 2021; Wang et al., 2022; Meng et al., 2022), and thus attracted much attention in the community of machine translation. The core of kNN-MT is to use an auxiliary datastore containing cached decoder representations and corresponding target tokens. This datastore can flexibly guide the NMT model to make better predictions, especially for domain adaptation. Compared with other retrieval-based paradigm (Tu et al., 2018; Cai et al., 2021), kNN-MT has two advantages: 1) It is more scalable because we can directly improve the NMT model by just manipulating the datastore. 2) It is more interpretable due to its observable retrieved pairs.

Generally, kNN-MT mainly involves two stages: datastore establishment and candidate retrieval. During the first stage, a pre-trained NMT model is used to construct a datastore containing key-value pairs, where the key is the decoder representation and the value is the corresponding target token. At the second stage, given the current decoder representation as a query at each time step, the k nearest key-value pairs are retrieved from the datastore. Then, according to the query-key distances, the

This work is done when Hui Jiang was interning at Pattern Recognition Center, WeChat AI, Tencent Inc, China.
*Equal contribution
†Corresponding author

retrieved values are converted into a translation probability distribution over candidate target tokens, denoted as *kNN distribution*. Finally, the predicted distribution of the NMT model is interpolated by kNN distribution with a hyper-parameter λ. Along this line, many efforts have been made to improve kNN-MT (Zheng et al., 2021a; Jiang et al., 2021). Particularly, as shown in Figure 1, adaptive kNN-MT (Zheng et al., 2021a) uses the query-key distances and retrieved pairs to dynamically estimate λ, exhibiting better performance than most kNN-MT models.

However, we find that existing kNN-MT models often suffer from a serious drawback: the model performance will dramatically deteriorate due to the underlying noise in retrieved pairs. For example, in Figure 1, the retrieved results may contain unrelated tokens such as "no", which leads to a harmful kNN distribution. Besides, for estimating λ, previous studies (Zheng et al., 2021a; Jiang et al., 2021) only consider the retrieved pairs while ignoring the NMT distribution. Back to Figure 1, compared with the kNN distribution, the NMT model gives a much higher probability to the ground-truth token "*been*". Although the kNN distribution is insufficiently accurate, it is still assigned with a greater weight than the NMT distribution. Obviously, this is inconsistent with our intuition that when the NMT model has high confidence in its prediction, it needs less help from others, and thus the kNN distribution should have a lower weight. Moreover, we find that during training, a non-negligible proportion of the retrieved pairs from the datastore do not contain ground-truth tokens. This can cause insufficient training of kNN modules. To sum up, conventional kNN-MT models are vulnerable to noise in datastore, for which we further conduct a preliminary study to validate the above issues. Therefore, dealing with the noise for the kNN-MT model remains to be a significant task.

In this paper, we explore a robust kNN-MT model. In terms of model architecture, we explore how to more accurately estimate the kNN distribution and better combine it with the NMT distribution. Concretely, unlike previous studies (Zheng et al., 2021a; Jiang et al., 2021) that only use retrieved pairs to dynamically estimate λ, we additionally use the confidence of NMT prediction to calibrate the calculation of λ where confidence is the predicted probability on each retrieved token. Meanwhile, we improve the kNN distribution by integrating the confidence to reduce the effect of noise. Besides, we propose to boost the robustness of our model by randomly adding perturbations to retrieved key representations and augmenting retrieved pairs with pseudo ground-truth tokens. By these means, our proposed approach can enhance the kNN-MT model to better cope with the noise in retrieved pairs, thus improving its robustness.

To investigate the effectiveness and generality of our model, we conduct experiments on several commonly-used benchmarks. Experimental results show that our model significantly outperforms the adaptive kNN-MT, which is the state-of-the-art kNN-MT model, across most domains. Moreover, our model exhibits better performance than adaptive kNN-MT on pruned datastores.

2 Related Work

Retrieval-based approaches leveraging auxiliary sentences have shown effectiveness in improving NMT models. Usually, they first retrieve relevant sentences from translation memory and then exploit them to boost NMT models during making a translation. For example, Tu et al. (2018) maintains a continuous cache storing attention vectors as keys and decoder representations as values. The retrieved values are then used to update the decoder representations. Bapna and Firat (2019) preform n-gram retrieval to identify similar source n-grams from the translation memory, where the corresponding target words are then encoded to update decoder representations. Xia et al. (2019) pack the retrieved target sentences into a compact graph which is then incorporated into decoder representations. He et al. (2021) propose several Transformer-based encoding methods to vectorize retrieved target sentences. Cai et al. (2021) propose a cross-lingual memory retriever to leverage target-side monolingual translation memory, showing effectiveness in low-resource and domain adaption scenarios.

Compared with the above studies involving additional training, non-parametric retrieval-augmented approaches (Zhang et al., 2018; Bulté and Tezcan, 2019; Xu et al., 2020) are more flexible and thus attract much attention. According to word alignments, Zhang et al. (2018) retrieve similar source sentences with target words from a translation memory, which are used to increase the probabilities of the collected target words to be translated. Both Bulté and Tezcan (2019) and Xu et al. (2020)

#	Strategy	Vanilla kNN-MT	Adaptive kNN-MT
1	All	45.92	47.88
2	-Random	44.75	46.56
3	-[0%, 20%)	44.18	45.76
4	-[20%, 40%)	44.21	46.96
5	-[40%, 60%)	44.10	46.18
6	-[60%, 80%)	41.78	43.49
7	-[80%, 100%]	42.22	42.54

Table 1: The performance of the models equipped with a datastore, where pairs within different intervals of ranking are individually removed. "-[0%, 20%)" means the top 20% pairs with the highest NMT confidence are removed. "All" means the entire datastore is used, and "-Random" means 20% pairs are randomly removed from the used datastore.

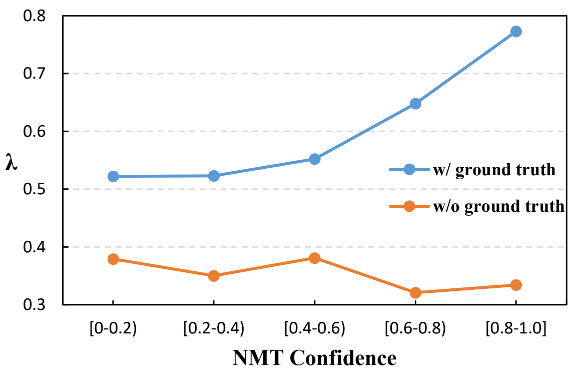


Figure 2: λ with respect to different NMT confidences on IT test set. Y-axis is λ, which is the weight of kNN distribution generated by adaptive kNN-MT. X-axis is the NMT confidence. We calculate the average generated λ in different confidence intervals. The blue curve and orange curve represent the cases where the ground-truth token is retrieved or missed in the kNN distribution, respectively.

retrieve related sentences via fuzzy matching and use the retrieved target sentences as the auxiliary information of the current source sentence.

Recently, a new non-parametric paradigm called kNN-MT (Khandelwal et al., 2021) has been proved to be simpler and more expressive. Typically, it uses the decoder representations as keys and the corresponding target words as values to build a datastore. During inference, based on retrieved results, the predicted distribution of the NMT model is interpolated by the kNN distribution with a hyper-parameter λ. Subsequently, some studies (Zheng et al., 2021a; Jiang et al., 2021) achieve better results by dynamically estimating λ. Meanwhile, there are also some researchers improving the retrieval efficiency of kNN-MT via cluster-based approaches (Wang et al., 2022) or limiting the search space by source tokens (Meng et al., 2022). Besides, Zheng et al. (2021b) presents a framework that uses in-domain monolingual target sentences to construct a datastore for unsupervised domain adaptation.

Finally, it should be noted that there have been many NLP studies (Cheng et al., 2018, 2019; Liu et al., 2020; Miao et al., 2022) on exploring robustness of NLP models. In comparison with the above-mentioned studies, our work is the first to improve the robustness of kNN-MT approaches.

3 Preliminary Study

To investigate the impact of noise on kNN-MT, we conduct a preliminary experiment in this section. We use the *NMT confidence* to represent the predicted probability on the target token from the NMT model. By this way, we remove the pairs of datastore within different confidence intervals to investigate the impact of noisy datastore in different degrees of NMT confidence.

Specifically, during the datastore establishment, besides the key-value pairs, we additionally record the NMT confidence of each target token and use it to rank these pairs. Then we split the datastore into multiple partitions, we alternatively remove each partition of the datastore and observe the performance of vanilla kNN-MT (Khandelwal et al., 2021) and adaptive kNN-MT (Zheng et al., 2021a) on the IT training dataset (Koehn and Knowles, 2017). To ensure the persuasiveness of our experiments, we directly use the setting of adaptive kNN-MT. In this way, we remove the key-value pairs within a specific interval of ranking to see the model performance.

Table 1 lists the performance of the above two models. When removing the partition of the datastore within the interval [80%, 100%], we can observe that the performances of both models significantly degrade (See Row 7 in Table 1), even inferior to "-Random". It is reasonable because when the model has low confidence on its own prediction, it needs the retrieved pairs as supplementary information. Meanwhile, if we remove the high-confidence partition of datastore within the interval [0%, 20%), the performances of both models also decline (See Row 3 in Table 1), underperforming the models with "-Random". Intuitively, removing high-confidence partition should not have such a negative effect, as the NMT model is able to predict them correctly. We conjecture that this is because the retrieved pairs contain much noise after removing the high-confidence partition, harming the kNN distribution which is then used to interpolate the NMT distribution.

Furthermore, since adaptive kNN-MT ignores NMT distribution, it may give a large weight to the incorrect kNN distribution. To verify this, we collect the λ generated by adaptive kNN-MT with respect to different NMT confidences in Figure 2. Looking at the orange curve, we find that when the adaptive kNN-MT fails to retrieve the ground-truth token, it gives a similar weight λ regardless of the NMT confidence. Besides, the blue curve shows the situation when the ground-truth token is successfully retrieved. We can see that adaptive kNN-MT gives a relatively small λ=0.52 even when the NMT model fails to predict the ground-truth (See [0-0.2) interval in Figure 2). Intuitively, the performance of the model would be further improved if it can generate a larger λ when the model has unconfident NMT distribution and high-quality kNN distribution.

The above experimental results indicate that the kNN-MT models are sensitive to the quality of the datastore, which limits their applicability to a noisy datastore. Therefore, it is of great significance to explore robust kNN-MT.

4 Our Model

4.1 Confidence-enhanced kNN-MT

Based on the observations in Section 3, we can find that neglecting the prediction confidence of NMT model makes kNN-MT vulnerable to the noisy datastore. Therefore, we leverage the prediction confidence of NMT model to enhance the robustness of kNN-MT, denoted as confidence-enhanced kNN-MT. Similar to other kNN-MT models (Khandelwal et al., 2021; Zheng et al., 2021a), our model introduces a datastore to assist a pre-trained NMT model, involving two stages: datastore establishment and confidence-enhanced kNN-MT prediction. Next, we give a full description of these two stages.

At the stage of datastore establishment, we adopt the pre-trained NMT model (Vaswani et al., 2017) to translate all training instances in an offline manner. During this process, we record all decoder representations and their corresponding ground-truth

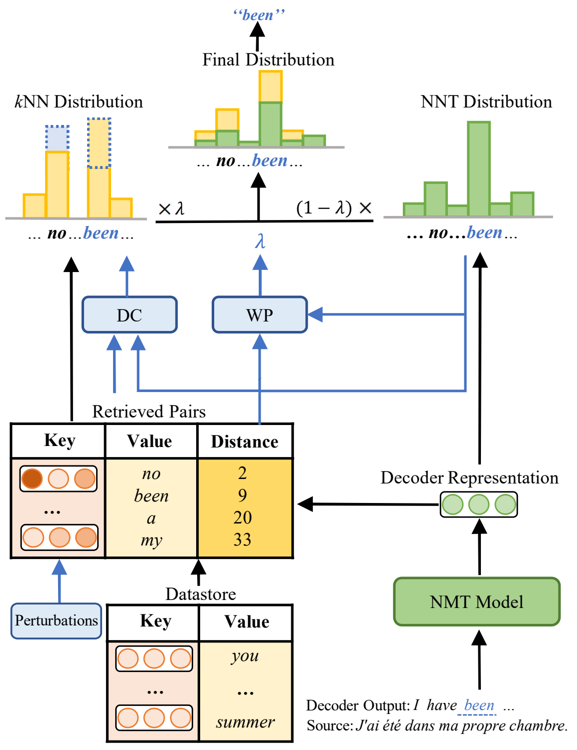


Figure 3: The overview of our confidence-enhanced kNN-MT model. The Distribution Calibration (DC) and Weight Prediction (WP) networks are trained to calibrate the kNN distribution and estimate the weight of kNN distribution, respectively.

target tokens as keys and values, respectively. Formally, given a training set $\{(x, y)\}$, we construct the datastore D as follows:

$$D = \{(h_t, y_t), \forall y_t \in y|(x, y)\}, \tag{1}$$

where the key h_t is the decoder representation of y_t, and the value y_t is the corresponding ground-truth target token with t denoting decoding timestep.

While inference, we firstly obtain the decoder representation $\hat{h}_t$ from the NMT model at the t-th timestep of decoding. Afterwards, as implemented in the conventional kNN-MT (Khandelwal et al., 2021), we convert the retrieved pairs N_t=$\{(h_k, v_k), 1{\le}k{\le}K\}$ into a probability distribution over its values (i.e., kNN distribution). Finally it is used to interpolate the NMT distribution to obtain a better translation. Particularly, as shown in Figure 3, on the basis of previous kNN-MT models, we further introduce **D**istribution **C**alibration (**DC**) network and **W**eight **P**rediction (**WP**) network, which leverage the model confidence to produce better kNN distribution and make more accurate estimation of λ, respectively.

Concretely, we use the retrieved pairs N_t and the decoder representation $\hat{h}_t$ to construct the kNN distribution. Moreover, we propose the DC network to quantify the importance c_k of each retrieved pair (h_k, v_k), which is then used to refine the kNN distribution. Formally, the kNN distribution is constructed in the following way:

$$p_{k\text{NN}}(y_t|\hat{h}_t) \propto \sum_{(h_k,v_k)\in N_t} \mathbb{1}_{y_t=v_k} \exp(\frac{-d_k}{T} + c_k), \tag{2}$$

$$T = \mathbf{W}_1(\tanh(\mathbf{W}_2[d_1, ..., d_K; r_1, ..., r_K])), \tag{3}$$

$$c_k = \mathbf{W}_3(\tanh(\mathbf{W}_4[p_{\text{NMT}}(v_k|\hat{h}_t); p_{\text{NMT}}(v_k|h_k)])), \tag{4}$$

where d_k is the L_2 distance between query $\hat{h}_t$ and key h_k, r_k is the number of non-duplicate values in top k neighbors, and $\mathbf{W}_*$ are parameter matrices.[1] Here, when calculating c_k, we mainly consider two kinds of information: 1) $p_{\text{NMT}}(v_k|\hat{h}_t)$, the predicted probability on v_k from the NMT model given the decoder representation $\hat{h}_t$, and 2) $p_{\text{NMT}}(v_k|h_k)$, the predicted probability on v_k given the key h_k.[2] In this way, the kNN distribution can be optimized by exploiting the knowledge of NMT, where the pairs with low confidence are expected to be assigned with lower probabilities.

However, this still can not make the model sufficiently robust to the noisy datastore. As mentioned previously in the introduction and preliminary study, when the retrieved pairs contain much noise, it is not appropriate to estimate λ only based on retrieved pairs (Zheng et al., 2021a). Therefore, we propose a lightweight WP network that simultaneously exploits the confidence of kNN distribution and NMT distribution to dynamically estimate λ_t:

$$\lambda_t = \frac{\exp(s_{k\text{NN}})}{\exp(s_{k\text{NN}}) + \exp(s_{\text{NMT}})}, \tag{5}$$

$$s_{k\text{NN}} = \mathbf{W}_5(\tanh(\mathbf{W}_2[d_1, ..., d_K; r_1, ..., r_K])), \tag{6}$$

$$\begin{aligned} s_{\text{NMT}} = \mathbf{W}_6[&p_{\text{NMT}}(v_1|\hat{h}_t), ..., p_{\text{NMT}}(v_K|\hat{h}_t);\\ &p_{\text{NMT}}(v_1|h_1), ..., p_{\text{NMT}}(v_K|h_K);\\ &p^{top1}_{\text{NMT}}, ..., p^{topK}_{\text{NMT}}], \end{aligned} \tag{7}$$

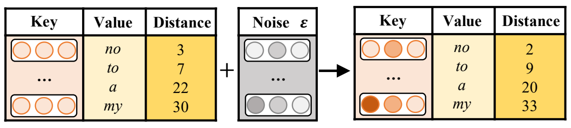
where p^{topk}_{NMT} is the k-th highest probability of the NMT distribution.

Lastly, the final distribution is computed as

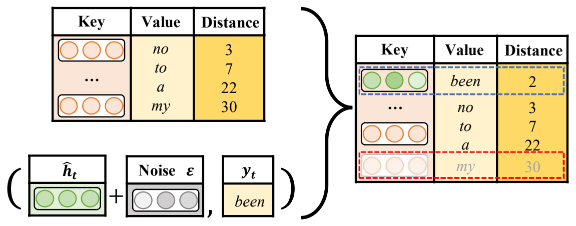
$$p(y_t|x, y_{<t}) = \lambda_t p_{k\text{NN}} + (1 - \lambda_t) p_{\text{NMT}}. \tag{8}$$

[1]In this paper, all $\mathbf{W}_*$ denote parameter matrices.

[2]We use the logarithm of probability as the feature, and we simplify the formula by omitting the log in this paper.



(a) Adding noise to the retrieved keys.



(b) Constructing a pseudo pair with the ground-truth token as value. The pseudo pair in the blue dotted box is inserted into the retrieved pairs and the pair in the red dotted box is removed.

Figure 4: Two different perturbations are used for robust training. In both cases, the ground-truth token "been" is not retrieved by the kNN-MT model.

By doing so, we expect that $p_{k\text{NN}}$ will be assigned with a small λ_t if the NMT model is highly confident on the predicted token.

4.2 Model Training

Although through the above modification, our model is able to generate a better kNN distribution and make a more accurate estimation of λ, it may be still not robust enough for two reasons. First, the datastore may be incompatible with the test set, resulting in the retrieved pairs cannot help the model. Second, the retrieved pairs do not always contain the ground-truth token. In that case, the probability of this token is zero in the kNN distribution. As a result, our DC network will not be optimized on this training sample. Especially, when the datastore size is limited, these two problems are more serious. To address them, we add two types of perturbations to the retrieved pairs at the training stage.

For the first problem, as shown in Figure 4(a), we add Gaussian noise to the keys of retrieved pairs with a certain ratio α. At each training timestep, we generate a random value between 0 and 1 and add noise only if it is less than α, so as to construct a noisy datastore:

$$h'_k = h_k + \epsilon, \;\; \epsilon \sim \mathbf{N}(\mathbf{0}, \sigma^2\mathbf{I}), \tag{9}$$

where the noise vector ϵ is sampled from a Gaussian distribution with variance σ^2, and σ is set to 0.01 as implemented in Cheng et al. (2018).

Model	IT	Medical	Koran	Law	Avg.
base NMT	38.35 / 0.391	40.06 / 0.468	16.26 / -0.018	45.48 / 0.574	35.04 / 0.354
vanilla kNN-MT	45.92 / 0.531	54.46 / 0.548	20.29 / -0.014	61.27 / 0.662	45.48 / 0.432
adaptive kNN-MT	47.88 / 0.567	56.10 / 0.572	20.43 / 0.037	63.20 / 0.692	46.90 / 0.467
our model	**48.90‡ / 0.585‡**	**57.28‡ / 0.578**	**20.71 / 0.047†**	**64.07‡ / 0.703‡**	**47.74 / 0.478**

Table 2: The BLEU (%) / Comet scores on test sets of different domains. † or ‡: significantly better than adaptive kNN-MT with t-test p<0.05 or p<0.01. Here we conducted 1,000 bootstrap tests (Koehn, 2004) to measure the significance in score differences.

For the second problem, as shown in Figure 4(b), we construct pseudo retrieved pairs with ground-truth tokens as values. Specifically, at the t-th timestep, if the ground-truth token y_t is not retrieved, we use the current decoder representation $\hat{h}_t$ and y_t to construct a pseudo pair $(\hat{h}_t + \epsilon, y_t)$. Then, we add this pair into the retrieved pairs N_t, where the pairs are sorted according to query-key distances, and the pair with the largest distance is removed to ensure the pair number is unchanged. Similarly, this perturbation vector ϵ is added with the same ratio α.

However, we find that using a fixed perturbation ratio results in performance degradation. We speculate that applying too large perturbations in the final training stage impairs the model's ability to handle real samples in the datastore. To avoid its negative impact, we dynamically adjust the perturbation ratio α according to the training step:

$$\alpha = \alpha_0 * \exp(-step/\beta), \tag{10}$$

where α_0 and β control the initial value and the declining speed of α, respectively. By doing so, we expect the perturbation ratio to be large at the beginning and gradually decrease during the subsequent stages.

5 Experiments

To investigate the effectiveness and robustness of our model, we carry out experiments on several commonly-used datasets.

5.1 Experimental Settings

5.1.1 Datasets and Evaluation

To ensure fair comparisons, we follow Zheng et al. (2021a) to conduct experiments on four commonly-used benchmarks, of which domains include IT, Koran, Medical and Law. The details of these datasets are given in Table 3. We use the Moses toolkit[3] to tokenize sentences and split words into subword units (Sennrich et al., 2016). As for the datastore, we adopt Faiss (Johnson et al., 2021) to conduct quantization and retrieval. Finally, all translation results are evaluated with case-sensitive detokenized BLEU by SacreBLEU (Post, 2018), we also adopt the Comet (Rei et al., 2020) as a complementary metric.

[3]https://github.com/moses-smt/mosesdecoder

Dataset	IT	Medical	Koran	Law
Train	223K	248K	18K	467K
Dev	2K	2K	2K	2K
Test	2K	2K	2K	2K
Size	3.71M	6.90M	524K	19.0M

Table 3: The statistics of datasets in different domains. We also list the size of the datastore, which is the number of stored tokens.

5.1.2 Baselines

We use the following models as our baselines:

1. **Base NMT**. We use the winner model (Ng et al., 2019) of WMT'19 German-English news translation task as the base NMT model, which is also used to initialize other kNN-MT models.
2. **Vanilla kNN-MT** (Khandelwal et al., 2021). It is our basic baseline. Note that it tunes hyper-parameters including λ on development sets.
3. **Adaptive kNN-MT** (Zheng et al., 2021a). It is our most important contrastive model that uses a light-weight network to dynamically estimate λ.

As for our model, we empirically set the hidden size of our WP and DC networks to 4 and 32, respectively, and the number of retrieved pairs (K) is set to 8 in all experiments. We empirically

Pruning Rate	Adaptive kNN-MT	Our model	Size
0%	47.88	48.90	3.6M
20%	46.56	47.30	2.9M
40%	44.44	44.95	2.2M
60%	42.24	42.69	1.4M
80%	39.87	40.22	0.7M

Table 4: The BLEU scores of the models equipped with the randomly reduced datastores on IT dataset.

Pruning Rate	Adaptive kNN-MT	Our model	Size
0%	47.88	48.90	3.6M
20%	45.76	47.68	2.9M
40%	42.43	46.89	2.2M
60%	41.05	45.52	1.4M
80%	38.79	41.02	0.7M

Table 5: The BLEU scores of the models equipped the reduced datastores on IT dataset, where the pairs having top x% largest NMT confidence are removed.

set α_0 to 1.0 and β to 1000, except for the Koran dataset where β is set to 10 due to its small data size. During the model training, we use the development sets to train our networks for about 5K steps following Zheng et al. (2021a). As for other hyper-parameters, we use the same experimental setup as adaptive kNN-MT, so as to ensure fair comparisons. We use Adam to optimize our networks, the batch size is set to 32, and the learning rate is set to 3e-4. All experiments are conducted on one NVIDIA V100 GPU.

5.2 Main Results

Table 2 shows the main results. Echoing previous studies (Khandelwal et al., 2021; Zheng et al., 2021a), vanilla kNN-MT significantly outperforms base NMT on all datasets. Moreover, due to the advantage of dynamic λ, adaptive kNN-MT exhibits significant improvement compared to vanilla kNN-MT on most datasets except for Koran, where only 18K training samples are available. Furthermore, our model achieves the best performance, obtaining the average +0.84 BLEU score over adaptive kNN-MT, which demonstrates the effectiveness of our model and training strategy. This conclusion remains valid when testing with the Comet score, where our model still outperforms adaptive kNN-MT.

Note that on the Koran dataset, adaptive kNN-MT only performs slightly better than vanilla kNN-MT. Likewise, our model achieves a slight improvement over adaptive kNN-MT. For these results, we speculate that the extremely small size of the datastore for Koran limits the potential of both adaptive kNN-MT and our model.

5.3 Robustness of Our Model

To verify the robustness of our model, we explore the performance of models with retrieved pairs of different qualities. Specifically, we decrease the quality of retrieved pairs by pruning the datastore in the following two ways and then test our model.

Firstly, we randomly remove the pairs of datastore and report the performance of our model and adaptive kNN-MT in Table 4. Overall, our model performs well in all situations and even surpasses adaptive kNN-MT by 0.35 BLEU when reducing the size of the datastore to 20%. It is reasonable to observe that the performances of two models get closer when the size of datastore becomes smaller.

Secondly, we conduct another experiment on datastore pruning from the perspective of NMT confidence. Intuitively, words with higher NMT confidence are less necessary to be saved as they are easier to be correctly predicted by the NMT model. Thus, we rank all datastore pairs according to their NMT confidence and remove those with the largest NMT confidence. Table 5 reports the experimental results. Compared to adaptive kNN-MT, our model exhibits much less performance decline. Particularly, when the datastore is compressed to 40%, our model still outperforms adaptive kNN-MT by a large margin (+ 4.47 BLEU). This result demonstrates the potential of our model on pruned datastores.

5.4 Analysis

We also study the effect of the important hyper-parameters: the number of retrieved pairs (K), to further validate the robustness of our model.

As shown in Figure 5, we find that performance of both vanilla kNN-MT and adaptive kNN-MT are not further improved when increasing K. This is because retrieving more neighbors may add noise to the kNN distribution. However, our model has a better performance when K=16. Overall, our model always exhibits better performance than adaptive kNN-MT especially when K is large, demonstrating its robustness.

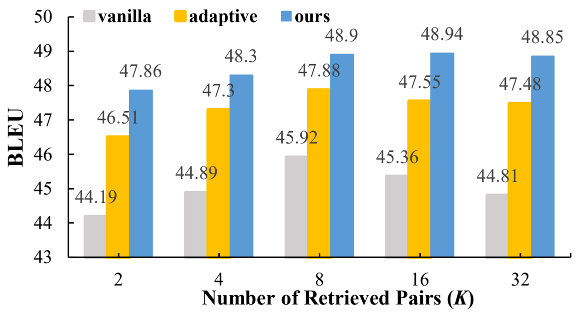


Figure 5: The BLEU scores of the models equipped with different K on IT dataset.

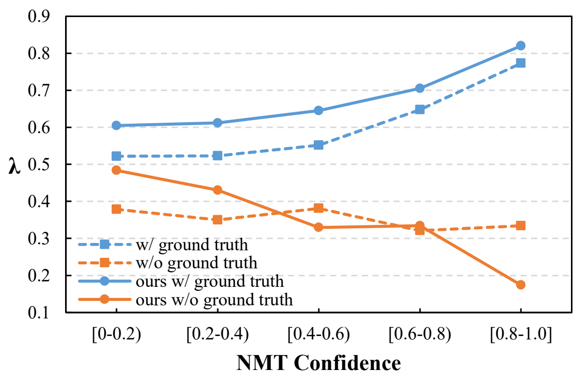


Figure 6: λ with respect to different NMT confidences on IT test set. We calculate the average generated λ in different confidence intervals. Solid and dotted lines represent that λ is generated by our model or adaptive kNN-MT, respectively.

Besides, to verify that our model is able to generate a better λ to improve the final translation, we study the predicted λ within different confidence intervals. Figure 6 reports the experimental results on the IT test set. The blue curve represents the situation when the ground-truth token is successfully retrieved. We can see that the λ generated by our model is larger than that generated by adaptive kNN-MT, especially when the NMT model fails to predict the ground truth (See [0-0.2) interval in Figure 6). Looking at the orange curve, we find that when models fail to retrieve the ground-truth token, the generated λ by our model has a stronger correlation with the NMT confidence. When model has confident NMT distribution (See [0.8-1.0] interval in Figure 6), our model can generate a lower weight (λ) of kNN distribution than adaptive kNN-MT.

Overall, it shows that our model can dynamically estimate the λ based on the NMT confidence. It also confirms our assumption mentioned in the preliminary study that the high-confidence prediction NMT distribution is expected to be assigned with a greater λ.

Model	BLEU
vanilla kNN-MT	45.92
adaptive kNN-MT	47.88
our model	**48.90**
w/o WP network	48.36
w/o DC network	48.45
w/o vector perturbation	48.77
w/o pseudo pair perturbation	48.75
w/o robust training	48.68
w/o perturbation rate's decline	48.38

Table 6: Ablation study of different networks and training strategies on IT test set. "w/o robust training" means removing both the "vector perturbation" and "pseudo pair perturbation" training strategy.

5.5 Ablation Study

To investigate the effects of our proposed networks and training strategies on our model, we also provide the performance of different variants of our model. As shown in Table 6, we find that removing any proposed network or not using any training strategy leads to a performance decline, demonstrating the effectiveness of all proposed networks and training strategies. Particularly, when discarding the WP network for prediction of λ, our model shows the most significant performance drop.

As for our training strategy, "w/o vector perturbation" represents removing the perturbation of the key vector (See Equation 9), "w/o pseudo pair perturbation" means removing the perturbation of constructing pseudo pair. It shows that constructing pseudo pair is more effective. It should be noted that if we do not decrease the perturbation rate, the model performance will degrade severely because of the overwhelming noise.

6 Conclusion

In this paper, via preliminary study, we first point out that existing kNN-MT models are very susceptible to the quality of retrieved pairs. Then, we explore robust kNN-MT, which improves kNN-MT models in the aspects of model architecture and training. Concretely, we incorporate the confidence of NMT prediction into modeling kNN distribution and dynamic estimation of λ. Besides, during the model training, we inject two types of perturbations into the retrieved pairs, which can effectively enhance the generalization of the model. Extensive results and in-depth analysis strongly demonstrate the effectiveness of our model.

To further verify the generality of our model, we will extend our model to other conditional text generation tasks, such as speech translation. Besides, we will try to combine kNN-MT with topic information, which has been successfully applied in previous studies (Su et al., 2012; Yu et al., 2013; Su et al., 2015; Ruan et al., 2018), to constraint retrieval in the future.

Limitations

In terms of efficiency, the storage cost of datastore and the time cost of retrieval are proportional to the size of training data and thus quite high for kNN-MT models. Besides, our model involves an additional small amount of parameters compared to vanilla kNN-MT (Khandelwal et al., 2021), requiring at least some in-domain samples for training. Although it can be applied in low-resource scenarios, it is not suitable for the scenario where in-domain samples are extremely few.

Acknowledgements

The project was supported by National Key Research and Development Program of China (No. 2020AAA0108004), National Natural Science Foundation of China (No. 62276219), Natural Science Foundation of Fujian Province of China (No. 2020J06001), Youth Innovation Fund of Xiamen (No. 3502Z20206059), and the Major Key Project of PCL. We also thank the reviewers for their insightful comments.

References

Ankur Bapna and Orhan Firat. 2019. Non-parametric adaptation for neural machine translation. In *Proceedings of NAACL*, pages 1921–1931.

Bram Bulté and Arda Tezcan. 2019. Neural fuzzy repair: Integrating fuzzy matches into neural machine translation. In *Proceedings of ACL*, pages 1800–1809.

Deng Cai, Yan Wang, Huayang Li, Wai Lam, and Lemao Liu. 2021. Neural machine translation with monolingual translation memory. In *Proceedings of ACL*, pages 7307–7318.

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. In *Proceedings of ACL*, pages 4324–4333.

Yong Cheng, Zhaopeng Tu, Fandong Meng, Junjie Zhai, and Yang Liu. 2018. Towards robust neural machine translation. In *Proceedings of ACL*, pages 1756–1766.

Qiuxiang He, Guoping Huang, Qu Cui, Li Li, and Lemao Liu. 2021. Fast and accurate neural machine translation with translation memory. In *Proceedings of ACL*, pages 3170–3180.

Qingnan Jiang, Mingxuan Wang, Jun Cao, Shanbo Cheng, Shujian Huang, and Lei Li. 2021. Learning kernel-smoothed machine translation with retrieved examples. In *Proceedings of EMNLP*, pages 7280–7290.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with gpus. *IEEE Trans. Big Data*, 7(3):535–547.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. In *Proceedings of ICLR*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*, pages 388–395.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39.

Kai Liu, Xin Liu, An Yang, Jing Liu, Jinsong Su, Sujian Li, and Qiaoqiao She. 2020. A robust adversarial training approach to machine reading comprehension. In *Proceedings of AAAI*, pages 8392–8400.

Yuxian Meng, Xiaoya Li, Xiayu Zheng, Fei Wu, Xiaofei Sun, Tianwei Zhang, and Jiwei Li. 2022. Fast nearest neighbor machine translation. In *Findings of ACL*, pages 555–565.

Zhongjian Miao, Xiang Li, Liyan Kang, Wen Zhang, Chulun Zhou, Yidong Chen, Bin Wang, Min Zhang, and Jinsong Su. 2022. Towards robust neural machine translation with iterative scheduled data-switch training. In *Proceedings of COLING*, pages 5266–5277.

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 WMT*, pages 314–319.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of WMT*, pages 186–191.

Ricardo Rei, Craig Stewart, Ana C. Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of EMNLP*, pages 2685–2702.

Zhiwei Ruan, Jinsong Su, Deyi Xiong, and Rongrong Ji. 2018. Context-aware phrase representation for statistical machine translation. In *Proceedings of PRICAI*, pages 137–149.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of ACL*, pages 1715–1725.

Jinsong Su, Hua Wu, Haifeng Wang, Yidong Chen, Xiaodong Shi, Huailin Dong, and Qun Liu. 2012. Translation model adaptation for statistical machine translation with monolingual topic information. In *Proceedings of ACL*, pages 459–468.

Jinsong Su, Deyi Xiong, Yang Liu, Xianpei Han, Hongyu Lin, Junfeng Yao, and Min Zhang. 2015. A context-aware topic model for statistical machine translation. In *Proceedings of ACL*, pages 229–238.

Zhaopeng Tu, Yang Liu, Shuming Shi, and Tong Zhang. 2018. Learning to remember translation history with a continuous cache. *Trans. Assoc. Comput. Linguistics*, 6:407–420.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NIPS*, pages 5998–6008.

Dexin Wang, Kai Fan, Boxing Chen, and Deyi Xiong. 2022. Efficient cluster-based k-nearest-neighbor machine translation. In *Proceedings of ACL*, pages 2175–2187.

Mengzhou Xia, Guoping Huang, Lemao Liu, and Shuming Shi. 2019. Graph based translation memory for neural machine translation. In *Proceedings of AAAI*, pages 7297–7304.

Jitao Xu, Josep Maria Crego, and Jean Senellart. 2020. Boosting neural machine translation with similar translations. In *Proceedings of ACL*, pages 1580–1590.

Heng Yu, Jinsong Su, Yajuan Lv, and Qun Liu. 2013. A topic-triggered language model for statistical machine translation. In *Proceedings of IJCNLP*, pages 447–454.

Jingyi Zhang, Masao Utiyama, Eiichiro Sumita, Graham Neubig, and Satoshi Nakamura. 2018. Guiding neural machine translation with retrieved translation pieces. In *Proceedings of NAACL*, pages 1325–1335.

Xin Zheng, Zhirui Zhang, Junliang Guo, Shujian Huang, Boxing Chen, Weihua Luo, and Jiajun Chen. 2021a. Adaptive nearest neighbor machine translation. In *Proceedings of ACL*, pages 368–374.

Xin Zheng, Zhirui Zhang, Shujian Huang, Boxing Chen, Jun Xie, Weihua Luo, and Jiajun Chen. 2021b. Non-parametric unsupervised domain adaptation for neural machine translation. In *Findings of EMNLP*, pages 4234–4241.