Bridging the Domain Gaps in Context Representations for k-Nearest Neighbor Neural Machine Translation

Zhiwei Cao[1,3]*, **Baosong Yang**[2], **Huan Lin**[2], **Suhang Wu**[1], **Xiangpeng Wei**[2]
Dayiheng Liu[2], **Jun Xie**[2], **Min Zhang**[4] and **Jinsong Su**[1,3]†
[1]School of Informatics, Xiamen University, China
[2]Language Technology Lab, Alibaba DAMO Academy
[3]Institute of Artificial Intelligence, Xiamen University, China
[4]Institute of Computer Science and Technology, Soochow University, China
lines1@stu.xmu.edu.cn, jssu@xmu.edu.cn

Abstract

k-Nearest neighbor machine translation (kNN-MT) has attracted increasing attention due to its ability to non-parametrically adapt to new translation domains. By using an upstream NMT model to traverse the downstream training corpus, it is equipped with a datastore containing vectorized key-value pairs, which are retrieved during inference to benefit translation. However, there often exists a significant gap between upstream and downstream domains, which hurts the retrieval accuracy and the final translation quality. To deal with this issue, we propose a novel approach to boost the datastore retrieval of kNN-MT by reconstructing the original datastore. Concretely, we design a reviser to revise the key representations, making them better fit for the downstream domain. The reviser is trained using the collected semantically-related key-queries pairs, and optimized by two proposed losses: one is the key-queries semantic distance ensuring each revised key representation is semantically related to its corresponding queries, and the other is an L2-norm loss encouraging revised key representations to effectively retain the knowledge learned by the upstream NMT model. Extensive experiments on domain adaptation tasks demonstrate that our method can effectively boost the datastore retrieval and translation quality of kNN-MT.[1]

1 Introduction

The recently proposed k-Nearest Neighbors Machine Translation (kNN-MT) (Khandelwal et al., 2021) is increasingly receiving attention from the community of machine translation due to its advantage on non-parametric domain adaptation (Zheng et al., 2021a; Wang et al., 2022a; Meng et al., 2022). Given an *upstream NMT model*, kNN-MT first uses

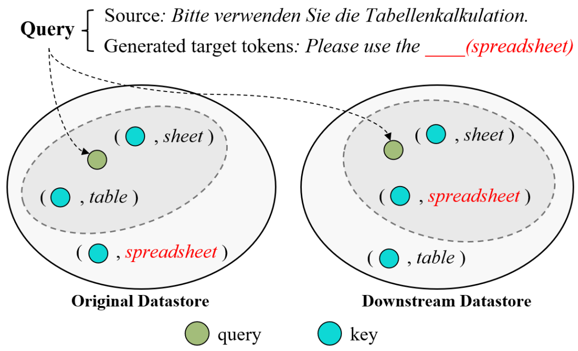

Figure 1: An example of datastore retrieval, where News and IT are the upstream and downstream domains, respectively. We first build a downstream NMT model by fine-tuning the upstream NMT model on the downstream training corpus. Then, we use the downstream NMT model to re-traverse the downstream training corpus, constructing a downstream datastore. Finally, we reuse the upstream and downstream NMT model to conduct retrieval on the original and downstream datastores, respectively. The result shows that the nearest neighbors retrieved by the same query are quite different, and only the retrieved nearest neighbors from the downstream datastore contain the ground-truth token "*spreadsheet*".

the downstream training corpus to establish a datastore containing key-value pairs, where each key is the representation of the NMT decoder and its value is the corresponding target token. During inference, it uses the current decoder representation as a query to retrieve N_k nearest key-value pairs from the datastore. Afterwards, the retrieved values are transformed into a probability distribution based on the query-key distances, denoted as kNN distribution. Finally, this distribution is interpolated with the prediction distribution of the NMT model to adjust the prediction translation. By doing so, the upstream NMT model can be easily adapted to diverse domains by equipping domain-specific datastores without additional parameters. To avoid confusion in subsequent descriptions, we name the datastore in conventional kNN-MT as the *original datastore*.

However, there often exists a significant domain

* This work was done when Zhiwei Cao was interning at DAMO Academy, Alibaba Group.

† Corresponding author.

[1]Our code is available at https://github.com/DeepLearnXMU/RevisedKey-knn-mt.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 5841–5853
July 9-14, 2023 ©2023 Association for Computational Linguistics

gap between the upstream NMT model and the downstream training corpus (Koehn and Knowles, 2017; Hu et al., 2019). The learned key representations of the original datastore deviate from the ideal distribution of downstream-domain key representation. As shown in Figure 1, in the original datastore built by the News domain NMT model, the nearest neighbors of the query contain the out-domain token "*table*" rather than the target token "*spreadsheet*" from the IT domain. This hurts the datastore retrieval of kNN-MT. To alleviate the negative impact of the retrieval error, previous studies resort to dynamically estimating the weight of kNN distribution for the final prediction (Zheng et al., 2021a; Jiang et al., 2021, 2022). However, these studies ignore the key representation learning, which is the basis of constructing datastore, and low-quality key representations tend to result in retrieval errors.

To bridge the domain gap, a natural choice is to fine-tune the NMT model on the downstream training corpus to obtain the *downstream NMT model* and then use it to build a *downstream datastore*. However, this method has two serious defects: 1) it is required to deploy multiple domain-specific NMT models when dealing with multi-domain translations, involving huge system deployment overhead. For example, in the commonly-used kNN-MT datasets (Aharoni and Goldberg, 2020) involving four downstream domains, this method has to construct four NMT models with datastores, consuming 37.2G GPU memory with 1,028M parameters. By contrast, kNN-MT involves only one NMT model and four datastores, consuming 11.3G GPU memory with 257M parameters; 2) it tends to be affected by the notorious catastrophic forgetting problem, weakening the adaptability of kNN-MT. This may result from the fine-tuned NMT model tending to forget previous upstream-domain knowledge and are therefore challenging to adapt to other domains. Thus, how to make more effective domain adaptation using kNN-MT remains a problem worth exploring.

In this paper, we propose a novel approach to boost the datastore retrieval of kNN-MT by reconstructing the original datastore. Concretely, we design a *Key Representation Reviser* that revises the key representations in an offline manner, so that they can better adapt to the retrieval from the downstream domain. This reviser is a two-layer feed-forward (FFN) with a ReLU function, which is fed with the information about a key representation k, and outputs an inductive bias Δk to revise k as $\hat{k} = k + \Delta k$. To train the reviser, we first use the downstream NMT model to extract semantically-related key-queries pairs from the downstream datastore, and then use their counterparts in the upstream NMT model and original datastore as supervision signals of the reviser. For each key-queries pair, we introduce two training losses to jointly optimize the reviser: 1) the *semantic distance loss*, which encourages each revised key representation to be adjacent to its semantically-related queries; 2) the *semantic consistency loss*, which avoids the revised key representation to be far from the original one, and thus preserving the knowledge learned by the upstream NMT model.

To summarize, our contributions are as follows:

- Through in-depth analysis, we reveal that the issue of the domain gap in kNN-MT hurts the effectiveness of the datastore retrieval.
- We propose a novel method to boost the datastore retrieval of kNN-MT by revising the key representations. To the best of our knowledge, our work is the first attempt to revise key representations of the kNN-MT datastore in an offline manner.
- Extensive experiments on a series of translation domains show that our method can strengthen the domain adaptation of kNN-MT without additional parameters during inference.

2 Preliminary Study

In this section, we first briefly introduce kNN-MT (Khandelwal et al., 2021), and then conduct a group of experiments to study the domain gap in kNN-MT.

2.1 kNN-MT

The construction of a kNN-MT model involves two key steps: using the downstream training corpus to create a datastore, and conducting translation with the help of the datastore.

Datastore Creation The common practice is to first use the upstream NMT model to traverse a downstream training corpus, where the decoder autoregressively extracts the contextual representations and corresponding target tokens to build a datastore. Specifically, for each bilingual sentence (x, y) from the downstream training corpus $\mathcal{C}_{prime}$, the NMT model generates the contextual representation $f(x, y_{<t})$ of the t-th target token y_t condition

on both source sentence x and preceding target tokens $y_{<t}$. Then, the key-value pair $(f(x, y_{<t}), y_t)$ will be added to the original datastore $(\mathcal{K}, \mathcal{V})$.

Translation with kNN Distribution During translation, the decoder outputs a probability distribution $p_{\text{NMT}}(\hat{y}_t|x, \hat{y}_{<t})$ at each timestep t, where $\hat{y}_{<t}$ represents the previously-generated target tokens. Then, the decoder outputs the contextual representation $f(x, \hat{y}_{<t})$ as the query to retrieve the datastore $(\mathcal{K}, \mathcal{V})$, obtaining N_k nearest key-value pairs according to the query-key l_2 distance. Denote the retrieved pairs as $\mathcal{R}$, the kNN distribution is computed as follows:

$$p_{\text{kNN}}(\hat{y}_t|x, \hat{y}_{<t}) \propto \sum_{\mathcal{R}} \mathbb{1}_{\hat{y}_t=v_i} \exp(\frac{-d(k_i, f(x, \hat{y}_{<t}))}{T}), \quad (1)$$

where T is the softmax temperature and $d(\cdot, \cdot)$ is the L_2 distance function. Finally, the predictive probability of $\hat{y}_t$ is defined as the interpolation of the decoder predictive probability and the kNN distribution probability:

$$p(\hat{y}_t|x, \hat{y}_{<t}) = \lambda \cdot p_{\text{kNN}}(\hat{y}_t|x, \hat{y}_{<t}) + (1 - \lambda) \cdot p_{\text{NMT}}(\hat{y}_t|x, \hat{y}_{<t}), \quad (2)$$

where $\lambda \in [0, 1]$ is a fixed interpolation weight.

2.2 The Domain Gap in kNN-MT

As mentioned previously, the performance of kNN-MT depends heavily on the quality of its datastore, which directly affects the datastore retrieval of the NMT model. However, the datastore key representations are provided by the upstream NMT model without considering the downstream information. Therefore, it is difficult for the upstream NMT model to effectively retrieve the key-value pairs related to the downstream domain, and thus negatively affect the subsequent translation prediction.

To verify this conjecture, we conduct a group of experiments on the development sets of four downstream domains, of which details are provided in Section 4.1. Concretely, we first construct two kNN-MT models: 1) ***k*NN-MT**. It is a vanilla kNN-MT model, which uses the upstream NMT model to traverse the downstream training corpus, forming an original datastore; 2) ***k*NN-MT(F)**. We first fine-tune the upstream NMT model on the downstream training corpus to obtain a downstream NMT model, and then use it to build a downstream

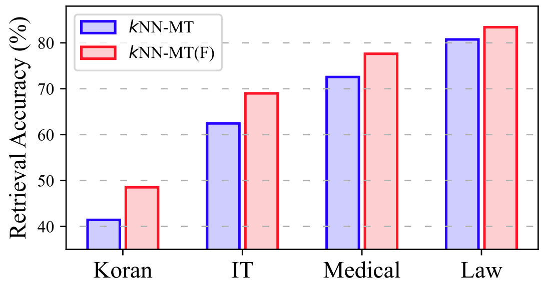

Figure 2: The retrieval accuracy of the conventional kNN-MT model on the original datastore, and kNN-MT(F) on the downstream datastore.

datastore on the training corpus above. Apparently, compared with the conventional kNN-MT model, kNN-MT(F) is less affected by the domain gap and its key representations are more in line with the ideal distribution of downstream-domain key representation. Afterwards, we adopt the above two models to traverse the development sets of four downstream domains, where the decoder contextual representations are used to retrieve the corresponding datastores[2], respectively.

To measure the retrieval quality of an NMT model on a datastore, we focus on those words retrieved with the maximal probability and define the proportion of ground-truth words in them as *retrieval accuracy*. Figure 2 illustrates the retrieval accuracy of the above kNN-MT models. We have two important findings. First, kNN-MT(F) achieves higher retrieval accuracy than the conventional kNN-MT model in all domains. These results demonstrate that alleviating the domain gap can improve the datastore retrieval of kNN-MT; Second, although kNN-MT(F) is more suitable for the downstream domain, it is not perfect and there are still some retrieval errors.

Although kNN-MT(F) can achieve higher retrieval accuracy, it still suffers from huge system deployment overhead for multi-domain translation and catastrophic forgetting, as mentioned previously. To avoid these issues, we explore a trade-off solution that directly revises the key representations of the original datastore, so as to enhance the retrieval effectiveness for the conventional kNN-MT model.

3 Our Method

To alleviate the influence of the domain gap on the datastore retrieval of kNN-MT, we propose a

[2]During this process, we skip some meaningless tokens, like stopwords.

simple yet effective approach to directly revise the original datastore, of which revised key representations are required to satisfy two properties: 1) they are more in line with the ideal distribution of downstream-domain key representation; 2) they can effectively retain the translation knowledge learned by the upstream NMT model.

To this end, we design a *Key Representation Reviser* to revise the key representations of the original datastore. To train this reviser, we first identify some key-queries pairs from the original datastore and upstream NMT model as the training data, where each key is expected to be semantically related to its corresponding queries. Then, we propose two training losses to jointly train the reviser. Using the reviser to reconstruct the original datastore, the original datastore can also effectively capture the semantically related key-queries pairs contained in the downstream datastore and NMT model, and thus is more suitable for the downstream translation task.

3.1 Key Representation Reviser

Our reviser is a two-layer FFN with a ReLU function. It is not embedded into the kNN-MT model, but can be used to modify key representations in an offline manner. For each key-value pair (k, v) in the original datastore, we obtain its corresponding counterpart (k', v) from the downstream datastore[3], and feed them into the reviser to generate an *inductive bias vector* Δk for revising k:

$$\Delta k = \text{FFN}([k; k'; \text{Emb}(v); \text{Emb}'(v)]), \quad (3)$$

$$\hat{k} = k + \Delta k, \quad (4)$$

where $\hat{k}$ denotes the revised key representation, $\text{Emb}(\cdot)$ and $\text{Emb}'(\cdot)$ are the token embeddings of the upstream and the downstream NMT models, respectively.

3.2 Training Data Construction

To train the key representation reviser, we adopt three steps to construct training data. Specifically, we first use the downstream NMT model to extract semantically-related key-queries pairs from the downstream datastore. Then, we filter some extracted low-quality key-queries pairs. Finally, from the original datastore and the upstream NMT model, we determine the corresponding counterparts of the above-mentioned key-queries pairs as the training data. Next, we introduce these three steps in detail.

[3]Given the same source sentence x and preceding target tokens $y_{<t}$, the key representation k and k' generated by upstream and downstream NMT models correspond to each other.

Step 1. As implemented in the previous preliminary study, we first construct a downstream NMT model θ' and its corresponding downstream datastore $\mathcal{D}'$. Then, we use the model θ' to re-traverse the downstream training corpus $\mathcal{C}_{prime}$, where the decoder representation is used as the query q' to retrieve N_k nearest key-value pairs $\{(k', v)\}$ from $\mathcal{D}'$. In this process, we collect these queries and their corresponding key-value pairs from the datastore. By doing so, we can easily determine a subset $\{q'\}$ corresponding to each k' from all queries, and further obtain a set of semantically-related key-queries pairs.

Step 2. As mentioned in Section 2.2, the downstream datastore is not perfect. Thus, the above key-queries pairs may contain noise.

To alleviate this issue, we learn from the related studies (Tomasev et al., 2013; He et al., 2021), and filter the low-quality key-queries pairs according to the retrieval numbers of keys. As analyzed in (He et al., 2021), in high-dimensional data, a data point is considered more reliable if it belongs to the nearest neighbors of many other data points. Inspired by this, we count the retrieved numbers $\text{Count}(k')$ of each key k' to measure its reliability. However, the keys with high-frequency values are originally retrieved more frequently. Only considering $\text{Count}(k')$ may result in some unreliable keys with high-frequent values being retained while some reliable pairs with low-frequent values being excluded. Therefore, we normalize $\text{Count}(k')$ with the token frequency $\text{Freq}(v)$ of its corresponding value v, and finally select the top $r\%$ key-queries pairs sorted by $\text{Count}(k')/\text{Freq}(v)$.

Step 3. As mentioned previously, we hope that the original datastore $\mathcal{D}$ and the upstream NMT mode θ can also effectively model the above extracted semantically-related key-queries pairs via key representation revision, so as to make $\mathcal{D}$ more applicable to the downstream translation task. To this end, we traverse each extracted pair $(k', \{q'\})$ and determine their counterparts $(k, \{q\})$ using the datastore $\mathcal{D}$ and the model θ. Note that k and k' are actually the hidden states at the same timestep, which are respectively generated by the models θ and θ' when traversing the same parallel sentence. Similarly, we determine the counterpart q for q'.

By doing so, we can obtain a set of key-queries pairs, denoted as $\mathcal{S}_r = \{(k, \{q\})\}$, as the training data of the reviser, where the key k of each pair is expected to be semantically related to its corresponding queries in the semantic space of the original datastore.

3.3 Training Objective

With the above extracted key-queries pair set $\mathcal{S}_r$, we propose a training objective with two training losses to train the reviser:

$$\mathcal{L} = \sum_{(k,\{q\})\in\mathcal{S}_r} (\mathcal{L}_{sd} + \alpha\mathcal{L}_{sc}), \tag{5}$$

where α is a hyper-parameter that is used to control the effects of these two losses.

The first loss is the **semantic distance loss** $\mathcal{L}_{sd}$. Formally, given an extracted key-queries pair $(k, \{q\}) \in \mathcal{S}_r$, we define $\mathcal{L}_{sd}$ as follows:

$$\mathcal{L}_{sd} = d(k + \Delta k, \text{Avg}(\{q\})), \tag{6}$$

where Δk is the inductive bias vector produced by our reviser, and $\text{Avg}(\{q\})$ is the fixed average representation of extracted queries $\{q\}$. Note that $\mathcal{L}_{sd}$ constrains the direction of Δk. By minimizing this loss, the revised key representation is encouraged to approach the average representation of queries. In this way, the original datastore and upstream NMT model are also able to capture the key-queries semantic relevance revealed by the downstream datastore and NMT model.

However, it is widely known that a fine-tuned model often suffers from catastrophic forgetting (McCloskey and Cohen, 1989; Ratcliff, 1990). Likewise, if the key representations of the original datastore are significantly changed, they will forget a lot of translation knowledge learned by the upstream NMT model.

In order to avoid catastrophic forgetting, previous studies attempt to incorporate regularization relative to the original domain during fine-tuning (Miceli Barone et al., 2017; Kirkpatrick et al., 2017). Inspired by these studies, we propose the second loss, called **semantic consistency loss** $\mathcal{L}_{sc}$, to constrain the modulus of Δk:

$$\mathcal{L}_{sc} = ||\Delta k||^2. \tag{7}$$

Apparently, $\mathcal{L}_{sc}$ is essentially also a regularization term, which is used to retain the knowledge of the upstream NMT model by limiting the change of key representations.

4 Experiments

To investigate the effectiveness of our method, we conduct experiments in the task of NMT domain adaptation.

4.1 Settings

Datasets and Evaluation We conduct experiments using the multi-domain datasets released by Aharoni and Goldberg (2020). The details of these datasets are shown in Table 6 of the Appendix. Unlike the previous studies (Khandelwal et al., 2021; Zheng et al., 2021a; Jiang et al., 2022) only using News as the upstream domain, we additionally use other available domains as upstream ones, which include Koran, IT, Medical, and Law. We first use the Moses toolkit[4] to tokenize the sentences and split the tokens into subwords units (Sennrich et al., 2016). Finally, we use two metrics: case-sensitive detokenized BLEU (Post, 2018) and COMET (Rei et al., 2020), to evaluate the quality of translation.

Baselines We select the following models as our baselines.

- **NMT**. When using News as the upstream domain, we directly use WMT'19 German-English news translation task winner (Ng et al., 2019) as the basic model. In the experiments with other upstream domains, we fine-tune this winner model on the corresponding upstream training corpus.

- k**NN-MT**. It is a vanilla kNN-MT model, which is our most important baseline. It equips the conventional NMT model with a downstream datastore, where hyper-parameters are tuned on the corresponding development set.

Implementation Details Following Khandelwal et al. (2021), we adopt *Faiss* (Johnson et al.) to conduct quantization and retrieval. As for the hyper-parameters of kNN-MT models including the weight λ and temperature T, we directly use the setting of (Zheng et al., 2021a). Besides, we set the number of retrieved pairs N_k as 8 with Koran or IT as the downstream domain, and 4 otherwise. When filter pairs for the reviser training, we only retain 30% extracted semantically-related key-queries pairs from the original datastore. The

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

	News		Koran		IT		Medical		Law	
	kNN-MT	Ours	kNN-MT	Ours	kNN-MT	Ours	kNN-MT	Ours	kNN-MT	Ours
Koran	20.31	21.28‡	-	-	12.64	14.69‡	9.51	10.79‡	11.25	12.32‡
IT	45.99	46.57†	39.89	41.40‡	-	-	29.06	30.82‡	30.37	31.73‡
Medical	54.12	55.77†	50.66	52.55‡	45.92	47.71‡	-	-	46.96	49.14‡
Law	61.27	61.77‡	59.05	59.49‡	44.82	46.22‡	48.18	49.61‡	-	-
Avg.	45.42	**46.35**	49.87	**51.15**	34.46	**36.21**	28.92	**30.41**	29.53	**31.06**

	News		Koran		IT		Medical		Law	
	kNN-MT	Ours	kNN-MT	Ours	kNN-MT	Ours	kNN-MT	Ours	kNN-MT	Ours
Koran	-0.183	-0.163‡	-	-	-0.482	-0.368‡	-0.717	-0.639‡	-0.623	-0.541†
IT	0.524	0.526	0.394	0.455‡	-	-	-0.011	0.066‡	0.054	0.100‡
Medical	0.539	0.539	0.472	0.507‡	0.304	0.348‡	-	-	0.346	0.413‡
Law	0.529	0.533†	0.611	0.626‡	0.184	0.232‡	0.296	0.353‡	-	-
Avg.	0.352	**0.359**	0.492	**0.529**	0.002	**0.071**	-0.144	**-0.073**	-0.074	**-0.009**

Table 1: The ScareBLEU and COMET scores of the conventional kNN-MT model and ours on test sets, where all models are individually trained with an upstream training corpus and a downstream one, and then evaluated on multiple downstream test sets. The involved upstream and downstream domains are listed in the first row and the first column, respectively. **Bold** indicates the best result. † or ‡: significantly better than kNN-MT with t-test p<0.05 or p<0.01. Here we conduct 1,000 bootstrap tests (Koehn, 2004) to measure the significant difference between scores.

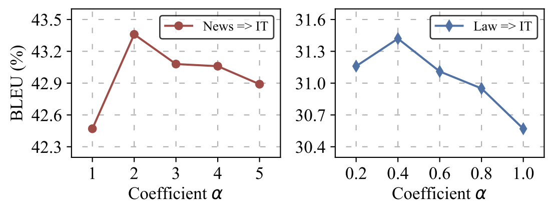

Figure 3: The ScareBLEU scores of our method with different coefficient α on News ⇒ IT and Law ⇒ IT (News ⇒ IT indicates News is the upstream domain and IT is the downstream domain.).

hidden size of the reviser is set as 8,192. When training this reviser, we empirically set the hyper-parameter α of the training objective (See Equation 5) to 2.0 in the experiments with upstream News domain, 0.4 for other experiments, and the number of training epoch as 100. During this process, we optimize the parameters using the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 5e-5.

4.2 Effects of Hyper-parameter α

From Equation 5, we clearly know that the coefficient α is an important hyper-parameter controlling the effects of two losses. Hence, we first investigate its effects on our model.

Concretely, in the experiments with upstream News domain, we select IT as the downstream domain following previous studies (Zheng et al., 2021a; Wang et al., 2022a). Then, we explore the model performances with different α on the development set. The left subfigure of Figure 3 illustrates the model performances with α varying from 1 to 5. We can find that our model achieves the best performance when α is 2.0. Therefore, we set α as 2.0 for all subsequent experiments using News as the upstream domain. In other groups of experiments, we still uniformly choose IT as the downstream domain, and Law as the upstream domain, where exists the largest amount of available data. We gradually vary α from 0.2 to 1.0 with an increment of 0.2, and also analyze the model performances on the corresponding development set. According to the experimental results reported in the right subfigure of Figure 3, we set α as 0.4 for all subsequent experiments with other upstream domains.

Notice that when setting News as the upstream domain, the optimal α is much larger than those of other upstream domains. As for this phenomenon, we speculate that the pre-trained NMT model of News domain involves large-scale training data and thus has learned more translation knowledge. Therefore, when applying our approach to experiments with upstream News domain, we set a relatively large α to effectively retain the translation

Upstream	News		Law	
Downstream	IT	Medical	IT	Medical
Our method	**46.57**	**55.77**	**31.73**	**49.14**
w/o data filtering	45.24	54.70	31.56	48.82
w/o $\mathcal{L}_{sc}$	45.06	53.83	30.98	48.42

Table 2: The ScareBLEU scores of ablation study.

knowledge of the pre-trained NMT model.

4.3 Main Results

Table 1 reports the performance of models on different domains. Overall, our model performs better than kNN-MT without introducing additional parameters in terms of two metrics. These results prove that our method is indeed able to effectively refine the kNN-MT datastore.

Specifically, in the experiments with upstream News domain, our model achieves only an average of +0.93 BLEU score on all domains, since the pre-trained NMT model for the upstream News domain is a competitive one and it involves the training data of other domains. Nevertheless, please note that this improvement is still significant at p<0.05. By contrast, in the experiments with other upstream domains, ours obtains more significant improvements.

Ablation Study To explore the effects of the data filtering strategy (See Section 3.2) and $\mathcal{L}_{sc}$ (See Equation 7) on our model, we provide the performance of two variants of our model: 1) w/o data filtering. During the process of training the reviser, we do not filter any key-queries pairs extracted from the downstream datastore by the downstream NMT model. 2) w/o $\mathcal{L}_{sc}$. We only use the semantic distance loss to train the reviser for this variant. Following previous studies (Zheng et al., 2021a; Wang et al., 2022a), we consider News and Law as upstream domains and select IT and Medical as downstream domains. In Figure 5 of the Appendix, we find that these two domains are least related to News and Law. As shown in Table 2, the removal of the data filtering strategy or $\mathcal{L}_{sc}$ leads to a performance decline, proving the effectiveness of our model.

4.4 Analysis

Performance Improvement vs. Domain Difference To further verify the rationality of our method, we explore the correlation between the performance improvements brought by our method and domain differences. To this end, following Aharoni and Goldberg (2020), we first represent each domain with the average TF-IDF representation of its sentences on the development set, and then measure the domain difference according to the cosine similarity based on domain representations: $\text{Diff}(d_1, d_2) = 1 - \text{Cosine}(d_1, d_2)$. In Figure 5, we plot the domain difference value and performance improvement for each domain pair. Here, we can observe a general trend that the greater the domain difference is, the more significant the performance improvement can be achieved by our method. Moreover, we measure Pearson's correlation coefficient between domain differences and performance improvements, resulting in a strong correlation value of 0.66[5]. These results prove the rationality of our method, and may also guide the evaluation of performance improvements of our approach in unseen domain pairs.

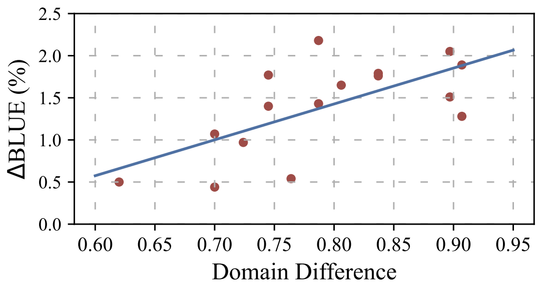

Figure 4: The domain differences for domain pairs and their corresponding performance improvements (ΔBLEU).

Upstream	News		Law	
Downstream	IT	Medical	IT	Medical
kNN-MT	45.99	54.12	30.37	46.96
Ours	46.57	55.77	31.73	49.14
Adaptive kNN-MT	47.51	55.87	31.52	48.43
+ Ours	**47.99**	**56.27**	**32.64**	**49.67**
Robust kNN-MT	48.69	56.89	32.12	49.97
+ Ours	**49.12**	**57.25**	**34.05**	**50.81**

Table 3: The ScareBLEU scores of Adaptive kNN-MT (Zheng et al., 2021a) and Robust kNN-MT (Jiang et al., 2022) with the datastore revised by our method.

Compatibility of Our Method with Adaptive kNN-MT As one of the most commonly-used kNN-MT variants, Adaptive kNN-MT (Zheng et al., 2021a) dynamically estimates the weight

[5]Given the significance level of 0.01 and the sample size of 16, the corresponding critical Pearson's correlation value is 0.59.

Upstream	News			
Downstream	Koran	IT	Medical	Law
kNN-MT	41.43	62.45	72.56	79.85
Ours	**44.87**	**63.92**	**74.18**	**81.45**

Table 4: The retrieval accuracy of kNN-MT and our model on the experiment with upstream News domain.

λ for kNN-MT to filter noises. Along this line, Robust kNN-MT (Jiang et al., 2022) incorporates the confidence of NMT prediction into the dynamic estimation of λ, achieving further improvement. Noteworthy, Adaptive kNN-MT, Robust kNN-MT, and our approach are able to alleviate the negative effects of the domain gap on kNN-MT from different perspectives. Furthermore, we explore whether our method is compatible with Adaptive kNN-MT and Robust kNN-MT. To ensure a fair comparison, we use the same retrieval number for Adaptive kNN-MT. From Table 3, we can observe that the performance of Adaptive kNN-MT and Robust kNN-MT can be further improved with our approach.

Retrieval Accuracy To verify the effectiveness of our method on datastore retrieval, we analyze the retrieval accuracy of the kNN-MT model with or without our strategy. As shown in 4, our method always achieves higher retrieval accuracy than the conventional kNN-MT. It indicates that the performance improvement of our method comes from the improvement of datastore quality.

Effects of Hyper-parameter r To demonstrate the effectiveness of our method, we also explore the effect of the hyper-parameter: the selected percentage $r\%$ of collected semantically-related key-queries pairs when constructing training data. As shown in Table 5, we find that our method outperforms kNN-MT with various $r\%$. Besides, with the percentage $r\%$ increasing, the performance of our method can be further improved. In practice, we set $r\%$ as 30% to balance the training resource overhead and performance improvement.

4.5 Discussion

Our Method vs. Fine-tuning As mentioned in Section 3.2, our method use the downstream NMT model to construct training data, where the downstream NMT model is obtained by fine-tuning the upstream NMT model on the downstream training corpus. Despite the requirement for more training resources, our method has a significant advantage in deploying resource overhead (see Section 1). Besides, our method still retains the following advantages of conventional kNN-MT: 1) Interpretable. This is because the retrieval process of kNN-MT is inspectable, the retrieved highly-relevant examples can be directly traced back to the specific sentence in the training corpus; 2) Flexible. We can use arbitrary amounts of data to build the datastore, and thus we can increase or decrease the amount of data in the datastore at will as needed immediately.

Upstream	News			
Downstream	Koran	IT	Medical	Law
kNN-MT	20.31	45.99	54.12	61.27
Ours (r = 20)	21.12	46.34	55.42	61.48
Ours (r = 30)	21.28	46.57	**55.77**	61.77
Ours (r = 40)	**21.30**	**46.90**	55.51	**61.82**

Table 5: The ScareBLEU scores of our method with different percentages $r\%$ of data retention on test sets.

5 Related Work

Our related work mainly includes two aspects: domain adaptation for NMT, and non-parametric retrieval-augmented approaches for NMT.

Domain Adaptation for NMT As summarized in Chu and Wang (2018), dominant methods in this aspect can be roughly divided into two categories: 1) model-centric approaches that focus on carefully designing NMT model architecture to learn target-domain translation knowledge (Wang et al., 2017; Zeng et al., 2018; Bapna and Firat, 2019a; Guo et al., 2021), or refining the training procedures to better exploit context (Wuebker et al., 2018; Bapna and Firat, 2019b; Lin et al., 2021; Liang et al., 2021); 2) data-centric methods resorting to leveraging the target-domain monolingual corpus (Zhang and Zong, 2016; Zhang et al., 2018b), synthetic corpus (Hoang et al., 2018; Hu et al., 2019; Wei et al., 2020) or parallel corpus (Chu et al., 2017) to improve the NMT model via fine-tuning.

Non-parametric Retrieval-augmented Approaches for NMT Generally, these methods retrieve sentence-level examples to enhance the robustness and expressiveness of NMT models (Zhang et al., 2018a; Bulte and Tezcan, 2019; Xu et al., 2020). For example, Zhang et al. (2018a) retrieves similar source sentences with target tokens from a translation memory, which are used to increase the probabilities of the collected tokens. Both Bulte and Tezcan (2019) and Xu et al. (2020) use the parallel sentence pairs retrieved via fuzzy matching as the auxiliary information of the current source sentence.

(Khandelwal et al., 2021) is the first attempt to explore kNN-MT, showing its effectiveness on non-parametric domain adaptation for NMT. Following this work, researchers have proposed kNN-MT variants, which mainly include two research lines: 1) the first line is mainly concerned with accelerating model inference by adaptive retrieval (He et al., 2021), datastore compression (He et al., 2021; Wang et al., 2022a; Martins et al., 2022), or limiting the search space by source tokens (Meng et al., 2022); 2) the second line focuses on reducing noises in retrieval results, through dynamically estimating the hyper-parameter N_k or the interpolation weight λ (Jiang et al., 2021; Zheng et al., 2021a; Wang et al., 2022b; Jiang et al., 2022). In addition, Zheng et al. (2021b) present a framework that uses downstream-domain monolingual target sentences to construct datastores for unsupervised domain adaptation.

Unlike the above studies caring more about filtering noise in retrieval results, inspired by representation learning (Su et al., 2015, 2016; Zhang et al.), we are mainly concerned with enhancing kNN-MT by revising the key presentations of the datastore. Note that very recently, Wang et al. (2022c) use an adapter to generate better retrieval representations in an online manner. However, unlike this work, we revise the key representation of the kNN-MT datastore in an offline manner. Besides, our method does not introduce additional parameters during inference, and thus maintains resource overhead.

6 Conclusion

In this paper, we first conduct a preliminary study to investigate the impact of the domain gap on the datastore retrieval of kNN-MT. Furthermore, we propose a reviser to refine the key representations of the original kNN-MT datastore in an offline manner, making them more suitable for the downstream domain. This reviser is trained on the collection of key-queries pairs, where the key of each pair is expected to be semantically related to its corresponding queries. Particularly, we introduce two losses to train the reviser, ensuring that the revised key representations conform to the downstream domain while effectively retaining their original knowledge. Through extensive experiments, we demonstrate the effectiveness of our method. Besides, in-depth analyses reveal that: 1) the performance improvement achieved by our method is positively correlated with the degree of the domain gap; 2) this improvement is primarily attributed to the enhancement of the datastore quality; 3) our method is able to compatible with existing Adaptive kNN-MT.

To further verify the generalization of our method, we will extend our method to kNN-LM or other text generation tasks, such as controllable generation.

Limitations

When using our method, we have to fine-tune the upstream NMT model to construct the downstream NMT model and then datastore for the reviser training. Hence, compared with the current commonly-used kNN-MT variant (Zheng et al., 2021a), our method requires more time for training. Nevertheless, it does not introduce additional parameters during inference.

Acknowledgements

The project was supported by National Natural Science Foundation of China (No. 62036004, No. 62276219), Natural Science Foundation of Fujian Province of China (No. 2020J06001), Youth Innovation Fund of Xiamen (No. 3502Z20206059), and Alibaba Group through Alibaba Innovative Research Program. We also thank the reviewers for their insightful comments.

References

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. In *ACL 2020*.

Ankur Bapna and Orhan Firat. 2019a. Non-parametric adaptation for neural machine translation. In *NAACL 2019*.

Ankur Bapna and Orhan Firat. 2019b. Simple, scalable adaptation for neural machine translation. In *EMNLP 2019*.

Bram Bulte and Arda Tezcan. 2019. Neural fuzzy repair: Integrating fuzzy matches into neural machine translation. In *ACL 2019*.

Chenhui Chu, Raj Dabre, and Sadao Kurohashi. 2017. An empirical comparison of domain adaptation methods for neural machine translation. In *ACL 2017*.

Chenhui Chu and Rui Wang. 2018. A survey of domain adaptation for neural machine translation. In *COLING 2018*.

Demi Guo, Alexander Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. In *ACL 2021*.

Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. 2021. Efficient nearest neighbor language models. In *EMNLP 2021*.

Vu Cong Duy Hoang, Philipp Koehn, Gholamreza Haffari, and Trevor Cohn. 2018. Iterative back-translation for neural machine translation. In *Proceedings of the 2nd workshop on neural machine translation and generation*.

Junjie Hu, Mengzhou Xia, Graham Neubig, and Jaime Carbonell. 2019. Domain adaptation of neural machine translation by lexicon induction. In *ACL 2019*.

Hui Jiang, Ziyao Lu, Fandong Meng, Chulun Zhou, Jie Zhou, Degen Huang, and Jinsong Su. 2022. Towards robust k-nearest-neighbor machine translation. In *EMNLP 2022*.

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

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*.

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

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *EMNLP 2004*.

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

Jianze Liang, Chengqi Zhao, Mingxuan Wang, Xipeng Qiu, and Lei Li. 2021. Finding sparse structures for domain specific neural machine translation. In *AAAI 2021*.

Zehui Lin, Liwei Wu, Mingxuan Wang, and Lei Li. 2021. Learning language specific sub-network for multilingual machine translation. In *ACL 2021*.

Pedro Martins, Zita Marinho, and Andre Martins. 2022. Efficient machine translation domain adaptation. In *Proceedings of the 1st Workshop on Semiparametric Methods in NLP: Decoupling Logic from Knowledge*.

Michael McCloskey and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. Psychology of Learning and Motivation.

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 2022*.

Antonio Valerio Miceli Barone, Barry Haddow, Ulrich Germann, and Rico Sennrich. 2017. Regularization techniques for fine-tuning in neural machine translation. In *EMNLP 2017*.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook fair's wmt19 news translation task submission. In *Proc. of WMT19*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*.

Roger Ratcliff. 1990. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*.

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

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

Jinsong Su, Deyi Xiong, Biao Zhang, Yang Liu, Junfeng Yao, and Min Zhang. 2015. Bilingual correspondence recursive autoencoder for statistical machine translation. In *EMNLP 2015*.

Jinsong Su, Biao Zhang, Deyi Xiong, Ruochen Li, and Jianmin Yin. 2016. Convolution-enhanced bilingual recursive neural network for bilingual semantic modeling. In *COLING 2016*.

Nenad Tomasev, Milos Radovanovic, Dunja Mladenic, and Mirjana Ivanovic. 2013. The role of hubness in clustering high-dimensional data. *IEEE transactions on knowledge and data engineering*.

Dexin Wang, Kai Fan, Boxing Chen, and Deyi Xiong. 2022a. Efficient cluster-based k-nearest-neighbor machine translation. In *ACL 2022*.

Dongqi Wang, Haoran Wei, Zhirui Zhang, Shujian Huang, Jun Xie, and Jiajun Chen. 2022b. Non-parametric online learning from human feedback for neural machine translation. In *AAAI 2022*.

Qiang Wang, Rongxiang Weng, and Ming Chen. 2022c. Learning decoupled retrieval representation for nearest neighbour neural machine translation. In *COLING 2022*.

Rui Wang, Masao Utiyama, Lemao Liu, Kehai Chen, and Eiichiro Sumita. 2017. Instance weighting for neural machine translation domain adaptation. In *EMNLP 2017*.

Hao-Ran Wei, Zhirui Zhang, Boxing Chen, and Weihua Luo. 2020. Iterative domain-repaired back-translation. In *EMNLP 2020*.

Joern Wuebker, Patrick Simianer, and John DeNero. 2018. Compact personalized models for neural machine translation. *arXiv preprint arXiv:1811.01990*.

Jitao Xu, Josep Crego, and Jean Senellart. 2020. Boosting neural machine translation with similar translations. In *ACL 2020*.

Jiali Zeng, Jinsong Su, Huating Wen, Yang Liu, Jun Xie, Yongjing Yin, and Jianqiang Zhao. 2018. Multi-domain neural machine translation with word-level domain context discrimination. In *EMNLP 2018*.

Biao Zhang, Deyi Xiong, and Jinsong Su. Battrae: Bidimensional attention-based recursive autoencoders for learning bilingual phrase embeddings. In *AAAI 2017*.

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. In *EMNLP 2016*.

Jingyi Zhang, Masao Utiyama, Eiichro Sumita, Graham Neubig, and Satoshi Nakamura. 2018a. Guiding neural machine translation with retrieved translation pieces. In *NAACL 2018*.

Zhirui Zhang, Shujie Liu, Mu Li, Ming Zhou, and Enhong Chen. 2018b. Joint training for neural machine translation models with monolingual data. In *AAAI 2018*.

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

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 2021*.

A Dataset Statistics

	Koran	IT	Medical	Law
Train	18K	223K	248K	467K
Dev	2K	2K	2K	2K
Test	2K	2K	2K	2K

Table 6: The example numbers of training, development, and test sets in four domains.

B Domain Difference

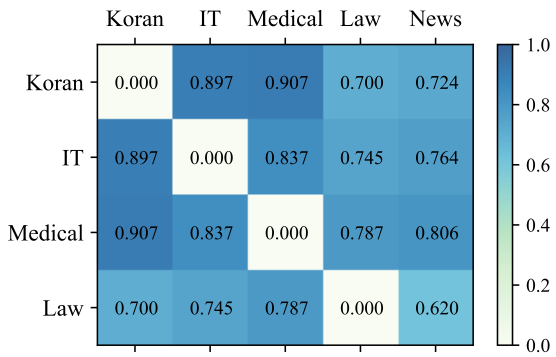

Figure 5: Domain Difference for each domain pair. The darker color denotes the greater difference.

C The Effect of Hyper-Parameter N_k

News ⇒ IT	$N_k = 4$	$N_k = 8$	$N_k = 12$	$N_k = 16$
kNN-MT	44.77	45.99	45.34	45.25
Ours	**45.40**	**46.57**	**45.88**	**45.63**

Table 7: The ScareBLEU scores of our method with different retrieve pairs N_k on News ⇒ IT.

To demonstrate the reliability of our method, we also explore our method with different hyper-parameter N_k. As shown in Table 7, our method enjoys consistent performance under different N_k.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 8

☐ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☐ Did you use or create scientific artifacts?

Not applicable. Left blank.

☐ B1. Did you cite the creators of artifacts you used?
Not applicable. Left blank.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Not applicable. Left blank.

C ☑ Did you run computational experiments?

Section 5

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 5

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 5 & Section 6

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 5

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 5

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.