Adaptive Nearest Neighbor Machine Translation

Xin Zheng[1], **Zhirui Zhang**[2], **Junliang Guo**[3], **Shujian Huang**[1]*, **Boxing Chen**[2], **Weihua Luo**[2] **and Jiajun Chen**[1]
[1]National Key Laboratory for Novel Software Technology, Nanjing University, China
[2]Machine Intelligence Technology Lab, Alibaba DAMO Academy
[3]University of Science and Technology of China
[1]zhengxin@smail.nju.edu.cn, {huangsj,chenjj}@nju.edu.cn
[2]{zhirui.zzr, boxing.cbx, weihua.luowh}@alibaba-inc.com
[3]guojunll@mail.ustc.edu.cn

Abstract

kNN-MT, recently proposed by Khandelwal et al. (2020a), successfully combines pre-trained neural machine translation (NMT) model with token-level k-nearest-neighbor (kNN) retrieval to improve the translation accuracy. However, the traditional kNN algorithm used in kNN-MT simply retrieves a same number of nearest neighbors for each target token, which may cause prediction errors when the retrieved neighbors include noises. In this paper, we propose Adaptive kNN-MT to dynamically determine the number of k for each target token. We achieve this by introducing a light-weight *Meta-k Network*, which can be efficiently trained with only a few training samples. On four benchmark machine translation datasets, we demonstrate that the proposed method is able to effectively filter out the noises in retrieval results and significantly outperforms the vanilla kNN-MT model. Even more noteworthy is that the Meta-k Network learned on one domain could be directly applied to other domains and obtain consistent improvements, illustrating the generality of our method. Our implementation is open-sourced at https://github.com/zhengxxn/adaptive-knn-mt.

1 Introduction

Retrieval-based methods (Gu et al., 2018; Zhang et al., 2018; Bapna and Firat, 2019; Khandelwal et al., 2020a) are increasingly receiving attentions from the machine translation (MT) community recently. These approaches complement advanced neural machine translation (NMT) models (Sutskever et al., 2014; Bahdanau et al., 2015; Vaswani et al., 2017; Hassan et al., 2018) to alleviate the performance degradation when translating out-of-domain sentences (Dou et al., 2019; Wei et al., 2020), rare words (Koehn and Knowles, 2017), etc. The ability of accessing any provided datastore during translation makes them scalable, adaptable and interpretable.

kNN-MT, recently proposed in (Khandelwal et al., 2020a), equips a pre-trained NMT model with a kNN classifier over a datastore of cached context representations and corresponding target tokens, providing a simple yet effective strategy to utilize cached contextual information in inference. However, the hyper-parameter k is fixed for all cases, which raises some potential problems. Intuitively, the retrieved neighbors may include noises when the target token is relatively hard to determine (e.g., relevant context is not enough in the datastore). And empirically, we find that the translation quality is very sensitive to the choice of k, results in the poor robustness and generalization performance.

To tackle this problem, we propose Adaptive kNN-MT that determines the choice of k regarding each target token adaptively. Specifically, instead of utilizing a fixed k, we consider a set of possible k that are smaller than an upper bound K. Then, given the retrieval results of the current target token, we propose a light-weight *Meta-k Network* to estimate the importance of all possible k-Nearest Neighbor results, based on which they are aggregated to obtain the final decision of the model. In this way, our method dynamically evaluate and utilize the neighbor information conditioned on different target tokens, therefore improve the translation performance of the model.

We conduct experiments on multi-domain machine translation datasets. Across four domains, our approach can achieve 1.44$\sim$2.97 BLEU score improvements over the vanilla kNN-MT on average when $K \geq 4$. The introduced light-weight Meta-k Network only requires thousands of parameters and can be easily trained with a few training samples. In addition, we find that the Meta-k Net-

* Corresponding author.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Short Papers), pages 368–374
August 1–6, 2021.

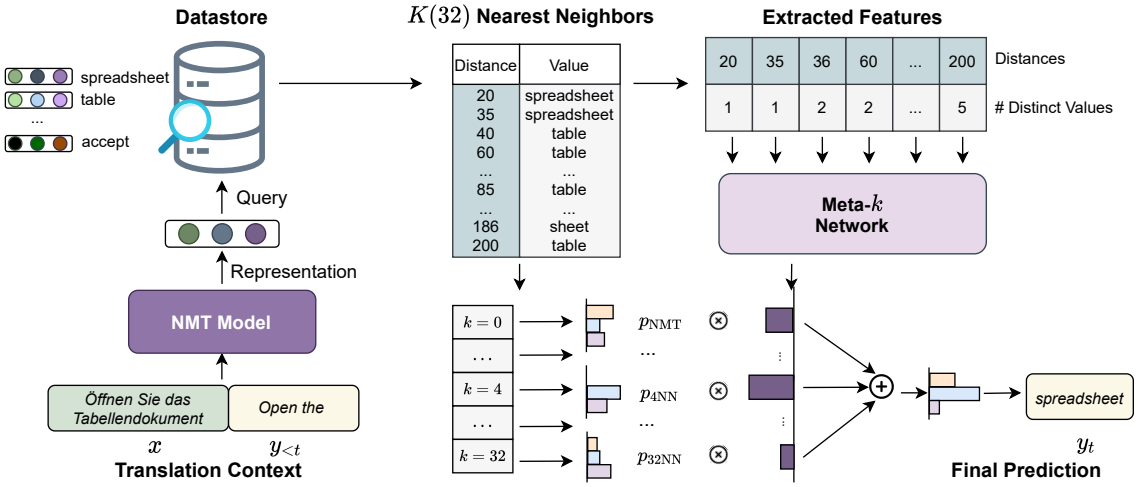

Figure 1: An overview of the proposed Adaptive kNN-MT, which could dynamically evaluate and aggregate a set of kNN predictions based on the distances as well as count of distinct values of retrieved neighbors.

work trained on one domain can be directly applied to other domains and obtain strong performance, showing the generality and robustness of the proposed method.

2 Background: kNN-MT

In this section, we will briefly introduce the background of kNN-MT, which includes two steps: creating a datastore and making predictions depends on it.

Datastore Creation. The datastore consists of a set of key-value pairs. Formally, given a bilingual sentence pair in the training set $(x, y) \in (\mathcal{X}, \mathcal{Y})$, a pre-trained autoregressive NMT decoder translates the t-th target token y_t based on the translation context $(x, y_{<t})$. Denote the hidden representations of translation contexts as $f(x, y_{<t})$, then the datastore is constructed by taking $f(x, y_{<t})$ as keys and y_t as values,

$$(\mathcal{K}, \mathcal{V}) = \bigcup_{(x,y)\in(\mathcal{X},\mathcal{Y})} \{(f(x, y_{<t}), y_t), \forall y_t \in y\}.$$

Therefore, the datastore can be created through a single forward pass over the training set $(\mathcal{X}, \mathcal{Y})$.

Prediction. While inference, at each decoding step t, the kNN-MT model aims to predict $\hat{y}_t$ given the already generated tokens $\hat{y}_{<t}$ as well as the context representation $f(x, \hat{y}_{<t})$, which is utilized to query the datastore for k nearest neighbors w.r.t the l_2 distance. Denote the retrieved neighbors as $N^t = \{(h_i, v_i), i \in \{1, 2, ..., k\}\}$, their distribution over the vocabulary is computed as:

$$p_{\text{kNN}}(y_t|x, \hat{y}_{<t}) \propto \sum_{(h_i,v_i)} \mathbb{1}_{y_t=v_i} \exp\left(\frac{-d(h_i, f(x, \hat{y}_{<t}))}{T}\right), \quad (1)$$

where T is the temperature and $d(\cdot, \cdot)$ indicates the l_2 distance. The final probability when predicting y_t is calculated as the interpolation of two distributions with a hyper-parameter λ:

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

where p_{NMT} indicates the vanilla NMT prediction.

3 Adaptive kNN-MT

The vanilla kNN-MT method utilizes a fixed number of translation contexts for every target token, which fails to exclude noises contained in retrieved neighbors when there are not enough relevant items in the datastore. We show an example with $k = 32$ in Figure 1. The correct prediction *spreadsheet* has been retrieved as top candidates. However, the model will finally predict *table* instead because it appears more frequently in the datastore than the correct prediction. A naive way to filter the noises is to use a small k, but this will also cause over-fitting problems for other cases. In fact, the optimal choice of k varies when utilizing different datastores in vanilla kNN-MT, leading to poor robustness and generalizability of the method, which is empirically discussed in Section 4.2.

To tackle this problem, we propose a dynamic method that allows each untranslated token to utilize different numbers of neighbors. Specifically,

we consider a set of possible ks that are smaller than an upper bound K, and introduce a light-weight *Meta-k Network* to estimate the importance of utilizing different ks. Practically, we consider the powers of 2 as the choices of k for simplicity, as well as $k = 0$ which indicates ignoring kNN and only utilizing the NMT model, i.e., $k \in \mathcal{S}$ where $\mathcal{S} = \{0\} \cup \{k_i \in \mathbb{N} \mid \log_2 k_i \in \mathbb{N}, k_i \leq K\}$. Then the Meta-$k$ Network evaluates the probability of different kNN results by taking retrieved neighbors as inputs.

Concretely, at the t-th decoding step, we first retrieve K neighbors N^t from the datastore, and for each neighbor (h_i, v_i), we calculate its distance from the current context representation $d_i = d(h_i, f(x, \hat{y}_{<t}))$, as well as the count of distinct values in top i neighbors c_i. Denote $d = (d_1, ..., d_K)$ as distances and $c = (c_1, ..., c_K)$ as counts of values for all retrieved neighbors, we then concatenate them as the input features to the Meta-k Network. The reasons of doing so are two-fold. Intuitively, the distance of each neighbor is the most direct evidence when evaluating their importance. In addition, the value distribution of retrieved results is also crucial for making the decision, i.e., if the values of each retrieved results are distinct, then the kNN predictions are less credible and we should depend more on NMT predictions.

We construct the Meta-k Network $f_{\text{Meta}}(\cdot)$ as two feed-forward Networks with non-linearity between them. Given $[d; c]$ as input, the probability of applying each kNN results is computed as:

$$p_{\text{Meta}}(k) = \text{softmax}(f_{\text{Meta}}([d; c])). \quad (3)$$

Prediction. Instead of introducing the hyper-parameter λ as Equation (2), we aggregate the NMT model and different kNN predictions with the output of the Meta-k Network to obtain the final prediction:

$$p(y_t|x, \hat{y}_{<t}) = \sum_{k_i \in \mathcal{S}} p_{\text{Meta}}(k_i) \cdot p_{k_i\text{NN}}(y_t|x, \hat{y}_{<t}), \quad (4)$$

where $p_{k_i\text{NN}}$ indicates the k_i Nearest Neighbor prediction results calculated as Equation (1).

Training. We fix the pre-trained NMT model and only optimize the Meta-k Network by minimizing the cross entropy loss following Equation (4), which could be very efficient by only utilizing hundreds of training samples.

4 Experiments

4.1 Experimental Setup

We evaluate the proposed model in domain adaptation machine translation tasks, in which a pre-trained general-domain NMT model is used to translate domain-specific sentences with kNN searching over an in-domain datastore. This is the most appealing application of kNN-MT as it could achieve comparable results with an in-domain NMT model but without training on any in-domain data. We denote the proposed model as Adaptive kNN-MT (A) and compare it with two baselines. One of that is vanilla kNN-MT (V) and the other is uniform kNN-MT (U) where we set equal confidence for each kNN prediction.

Datasets and Evaluation Metric. We use the same multi-domain dataset as the baseline (Khandelwal et al., 2020a), and consider domains including **IT**, **Medical**, **Koran**, and **Law** in our experiments. The sentence statistics of datasets are illustrated in Table 1. The Moses toolkit[1] is used to tokenize the sentences and split the words into subword units (Sennrich et al., 2016) with the bpe-codes provided by Ng et al. (2019). We use SacreBLEU[2] to measure all results with case-sensitive detokenized BLEU (Papineni et al., 2002).

Dataset	IT	Medical	Koran	Laws
Train	222,927	248,009	17,982	467,309
Dev	2000	2000	2000	2000
Test	2000	2000	2000	2000

Table 1: Statistics of dataset in different domains.

Implementation Details. We adopt the fairseq toolkit[3](Ott et al., 2019) and faiss[4](Johnson et al., 2017) to replicate kNN-MT and implement our model. We apply the WMT'19 German-English news translation task winner model (Ng et al., 2019) as the pre-trained NMT model which is also used by Khandelwal et al. (2020a). For kNN-MT, we carefully tune the hyper-parameter λ in Equation (2) and report the best scores for each domain. More details are included in the supplementary materials. For our method, the hidden size of the two-layer FFN in Meta-k Network is set to 32. We

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

[2] https://github.com/mjpost/sacrebleu

[3] https://github.com/pytorch/fairseq

[4] https://github.com/facebookresearch/faiss

	Domain	IT (Base NMT: 38.35)			Med (Base NMT: 39.99)			Koran (Base NMT: 16.26)			Law (Base NMT: 45.48)			Avg (Base NMT: 35.02)		
	Model	V	U	A	V	U	A	V	U	A	V	U	A	V	U	A
K	1	42.19	41.21	42.52	51.41	50.32	51.82	18.12	17.15	18.10	58.76	58.05	58.81	42.62	41.68	42.81
	2	44.20	41.43	46.18	53.65	52.44	55.20	19.37	17.36	19.12	60.80	59.81	61.76	44.50	42.76	45.56
	4	44.89	42.31	47.23	54.16	53.01	55.84	19.50	17.88	19.69	61.31	60.75	62.89	44.97	43.49	46.41
	8	45.96	42.46	**48.04**	54.06	53.46	56.31	20.12	18.59	20.57	61.12	61.37	**63.21**	45.32	43.97	47.03
	16	45.36	43.05	47.71	53.54	54.08	**56.41**	20.30	19.45	**21.09**	60.21	61.52	63.07	44.85	44.53	**47.07**
	32	44.81	43.78	47.68	52.52	53.95	56.21	19.66	19.99	20.96	59.04	61.53	63.03	44.00	44.81	46.97
	$\sigma^2{}_{(K \geq 4)}$	0.21	0.33	**0.08**	0.42	0.18	**0.05**	**0.10**	0.65	0.30	0.81	0.10	**0.01**	0.24	0.26	**0.07**

Table 2: The BLEU scores of the vanilla kNN-MT (V) and uniform kNN-MT (U) baselines and the proposed Adaptive kNN-MT model (A). Underline results indicate the best results of baselines, and our best results are marked bold. σ^2 indicates the variance of results among different Ks.

Adaptive kNN-MT	IT	Medical	Koran	Law	Avg
In-domain	47.68	56.21	20.96	63.03	46.97
IT domain	47.68	56.20	20.52	62.33	46.68

Table 3: Generality Evaluation. We train the model on the IT domain and directly apply to other test sets.

Model	IT ⇒ Medical	Medical ⇒ IT
Base NMT	39.99	38.35
kNN-MT	25.82	15.79
Adaptive kNN-MT	37.78	30.09

Table 4: Robustness Evaluation, where the test sets are from Medical/IT domains and the datastore are from IT/Medical domains respectively.

directly use the dev set (about 2k sents) to train the Meta-k Network for about 5k steps. We use Adam (Kingma and Ba, 2015) to optimize our model, the learning rate is set to 3e-4 and batch size is set to 32 sentences.

4.2 Main Results

The experimental results are listed in Table 2. We can observe that the proposed Adaptive kNN-MT significantly outperforms the vanilla kNN-MT on all domains, illustrating the benefits of dynamically determining and utilizing the neighbor information for each target token. In addition, the performance of the vanilla model is sensitive to the choice of K, while our proposed model is more robust with smaller variance. More specifically, our model achieves better results when choosing larger number of neighbors, while the vanilla model suffers from the performance degradation when $K = 32$, indicating that the proposed Meta-k Network is able to effectively evaluate and filter the noise in retrieved neighbors, while a fixed K cannot. We also compare our proposed method with another naive baseline, uniform kNN-MT, where we set equal confidence for each kNN prediction and make it close to the vanilla kNN-MT with small k. It further demonstrates that our method could really learn something useful but not bias smaller k.

Generality. To demonstrate the generality of our method, we directly utilize the Meta-k Network trained on the IT domain to evaluate other domains. For example, we use the Meta-k Network trained on IT domain and medical datastore to evaluate the performance on medical test set. For comparison, we collect the in-domain results from Table 2. We set $K = 32$ for both settings. As shown in Table 3, the Meta-k Network trained on the IT domain achieves comparable performance on all other domains which re-train the Meta-k Network with in-domain dataset. These results also indicate that the mapping from our designed feature to the confidence of retrieved neighbors is common across different domains.

Robustness. We also evaluate the robustness of our method in the domain-mismatch setting, where we consider a scenario that the user inputs an out-of-domain sentence (e.g. IT domain) to a domain-specific translation system (e.g. medical domain) to evaluate the robustness of different methods. Specifically, in IT ⇒ Medical setting, we firstly use medical dev set and datastore to tune hyper-parameter for vanilla kNN-MT or train the Meta-k Network for Adaptive kNN-MT, and then use IT test set to test the model with medical datastore. We set $K = 32$ in this experiment. As shown in Table 4, the retrieved results are highly noisy so that the vanilla kNN-MT encounters drastic performance degradation. In contrast, our method could effectively filter out noises and therefore prevent performance degradation as much as possible.

Case Study. Table 5 shows a translation example selected from the test set in **Medical** domain with

Source	Wenn eine gleichzeitige Behandlung mit Vitamin K Antagonisten erforderlich ist, müssen die Angaben in Abschnitt 4.5 beachtet werden.
Reference	therapy with vitamin K antagonist should be administered in accordance with the information of Section 4.5.
Base NMT	If a simultaneous treatment with vitamin K antagonists is required, the information in section 4.5 must be observed.
kNN-MT	If concomitant treatment with vitamin K antagonists is required, please refer to section 4.5.
Adaptive kNN-MT	When required, concomitant ***therapy with vitamin K antagonist should be administered in accordance with the information of Section 4.5.***

Table 5: Translation examples of different systems in Medical domain.

Adaptive kNN-MT ($K = 8$)	48.04
- value count feature	46.76
- distance feature	45.60

Table 6: Effect of different features in Meta-k Network.

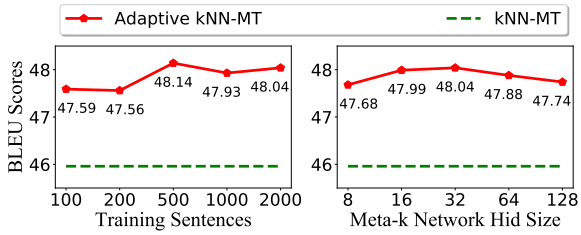

Figure 2: Effect of different number of training sentences and the hidden size of Meta-k Network.

$K = 32$. We can observe that the Meta-k Network could determine the choice of k for each target token respectively, based on which Adaptive kNN-MT leverages in-domain datastore better to achieve proper word selection and language style.

Analysis. Finally, we study the effect of two designed features, number of training sentences and the hidden size of the proposed Meta-k Network. We conduct these ablation study on IT domain with $K = 8$. All experimental results are summarized in Table 6 and Figure 2. It's obvious that both of the two features contribute significantly to the excellent performance of our model, in which the distance feature is more important. And surprisingly, our model could outperforms the vanilla kNN-MT with only 100 training sentences, or with a hidden size of 8 that only contains around 0.6k parameters, showing the efficiency of our model.

5 Conclusion and Future Works

In this paper, we propose Adaptive kNN-MT model to dynamically determine the utilization of retrieved neighbors for each target token, by introducing a light-weight *Meta-k Network*. In the experiments, on the domain adaptation machine translation tasks, we demonstrate that our model is able to effectively filter the noises in retrieved neighbors and significantly outperform the vanilla kNN-MT baseline. In addition, the superiority of our method on generality and robustness is also verified. In the future, we plan to extend our method to other tasks like Language Modeling, Question Answering, etc, which can also benefit from utilizing kNN searching (Khandelwal et al., 2020b; Kassner and Schütze, 2020).

6 Acknowledgments

We would like to thank the anonymous reviewers for the helpful comments. This work was supported by the National Key R&D Program of China (No. 2019QY1806), National Science Foundation of China (No. 61772261, U1836221) and Alibaba Group through Alibaba Innovative Research Program. We appreciate Weizhi Wang, Hao-Ran Wei and Jun Xie for the fruitful discussions. The work was done when the first author was an intern at Alibaba Group.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Ankur Bapna and Orhan Firat. 2019. Non-parametric adaptation for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1921–1931, Minneapolis, Minnesota. Association for Computational Linguistics.

Zi-Yi Dou, Junjie Hu, Antonios Anastasopoulos, and Graham Neubig. 2019. Unsupervised domain adaptation for neural machine translation with domain-

aware feature embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1417–1422, Hong Kong, China. Association for Computational Linguistics.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O. K. Li. 2018. Search engine guided neural machine translation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5133–5140. AAAI Press.

Hany Hassan, Anthony Aue, C. Chen, Vishal Chowdhary, J. Clark, C. Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, W. Lewis, M. Li, Shujie Liu, T. Liu, Renqian Luo, Arul Menezes, Tao Qin, F. Seide, Xu Tan, Fei Tian, Lijun Wu, Shuangzhi Wu, Yingce Xia, Dongdong Zhang, Zhirui Zhang, and M. Zhou. 2018. Achieving human parity on automatic chinese to english news translation. *ArXiv*, abs/1803.05567.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *CoRR*, abs/1702.08734.

Nora Kassner and Hinrich Schütze. 2020. BERT-kNN: Adding a kNN search component to pretrained language models for better QA. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3424–3430, Online. Association for Computational Linguistics.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020a. Nearest neighbor machine translation. *CoRR*, abs/2010.00710.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020b. Generalization through memorization: Nearest neighbor language models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation, NMT@ACL 2017, Vancouver, Canada, August 4, 2017*, pages 28–39. Association for Computational Linguistics.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook FAIR's WMT19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 314–319, Florence, Italy. 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.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

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 *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Hao-Ran Wei, Zhirui Zhang, Boxing Chen, and Weihua Luo. 2020. Iterative domain-repaired back-translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5884–5893, Online. Association for Computational Linguistics.

Jingyi Zhang, Masao Utiyama, Eiichro Sumita, Graham Neubig, and Satoshi Nakamura. 2018. Guiding neural machine translation with retrieved translation pieces. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1325–1335, New Orleans, Louisiana. Association for Computational Linguistics.

Datastore	IT	Medical	Koran	Laws
Size	3,613,350	6,903,320	524,400	19,070,000
Hard Disk Space (Datastore)	6.9 Gb	15 Gb	1.1 Gb	37 Gb
Hard Disk Space (faiss index)	266 Mb	492 Mb	54 Mb	1.3 Gb

Table 7: Statistics of datastore in different domains.

ms / sent	K	Batch=1	Batch=16	Batch=32	Batch=64
NMT	0	165	16.3	10.5	7.9
kNN-MT	8	291.0(×1.76)	51.0(×3.1)	43.6(×4.2)	38.0(×4.8)
	16	311.4(×1.89)	81.1(×5.0)	70.4(×6.7)	64.4(×8.2)
	32	385.5(×2.34)	136.5(×8.4)	123.8(×11.8)	114.9(×14.5)
Adaptive kNN-MT	8	299.1(×1.81)	51.1(×3.1)	42.8(×4.1)	38.1(×4.8)
	16	315.0(×1.91)	80.2(×4.9)	70.2(×6.7)	63.7(×8.1)
	32	394.5(×2.40)	147.5(×9.0)	128.0(×12.2)	116.8(×14.8)

Table 8: Decoding time of different models. All results are tested on 20 cores Intel(R) Xeon(R) Gold 6248 CPU @ 2.50GHz with a V100-32GB GPU.

A Appendix

A.1 Datastore Creation

We first use numpy array to save the key-value pairs over training sets as datastore. Then, faiss is used to build index for each datastore to carry out fast nearest neighbor search. We utilize faiss to learn $4k$ cluster centroids for each domain, and search 32 clusters for each target token in decoding. The size of datastore (count of target tokens), and hard disk space of datastore as well as faiss index are shown in Table 7.

A.2 Hyper-Parameter Tuning for kNN-MT

The performance of vanilla kNN-MT is highly related to the choice of hyper-parameter, i.e. k, T and λ. We fix T as 10 for IT, Medical, Law, and 100 for Koran in all experiments. Then, we tuned k and λ for each domain when using kNN-MT and the optimal choice for each domain are shown in Table 9. The performance of kNN-MT is unstable with different hyper-parameters while our Adaptive kNN-MT avoids this problem.

Dataset	IT	Medical	Koran	Laws
k	8	4	16	4
T	10	10	10	100
λ	0.7	0.8	0.8	0.8

Table 9: Optimal choice of hyper-parameters for each domain in vanilla kNN-MT.

A.3 Decoding Time

We compare the decoding time on IT test set of NMT, kNN-MT (our replicated) and Adaptive kNN-MT condition on different batch size. In decoding, the beam size is set to 4 with length penalty 0.6. The results are summarized in Table 8.