kNN-BOX: A Unified Framework for Nearest Neighbor Generation

Wenhao Zhu*, **Qianfeng Zhao***, **Yunzhe Lv***,
Shujian Huang, **Siheng Zhao**, **Sizhe Liu**, **Jiajun Chen**
National Key Laboratory for Novel Software Technology, Nanjing University, China
`{zhuwh,qianfeng,lvyz,zhaosh,liusz}@smail.nju.edu.cn`, `{huangsj,chenjj}@nju.edu.cn`

Abstract

Augmenting the base neural model with a token-level symbolic datastore is a novel generation paradigm and has achieved promising results in machine translation (MT). In this paper, we introduce a unified framework kNN-BOX, which enables quick development and visualization for this novel paradigm. kNN-BOX decomposes the datastore-augmentation approach into three modules: datastore, retriever and combiner, thus putting diverse kNN generation methods into a unified way. Currently, kNN-BOX has provided implementation of seven popular kNN-MT variants, covering research from performance enhancement to efficiency optimization. It is easy for users to reproduce these existing work or customize their own models. Besides, users can interact with their kNN generation systems with kNN-BOX to better understand the underlying inference process in a visualized way. In experiment section, we apply kNN-BOX for machine translation and three other seq2seq generation tasks (text simplification, paraphrase generation and question generation). Experiment results show that augmenting the base neural model with kNN-BOX can bring large performance improvement in all these tasks. The code and document of kNN-BOX is available at `https://github.com/NJUNLP/knn-box`. The demo can be accessed at `http://nlp.nju.edu.cn/demo/knn-box/`. The introduction video is available at `https://www.youtube.com/watch?v=m0eJ1dHVR3w`.

1 Introduction

Equipping the base neural model with a symbolic datastore is a novel paradigm for enhancing generation quality. Khandelwal et al. (2021) apply this paradigm in machine translation, known as kNN-MT, and achieves promising results, especially in MT domain adaptation and multilingual MT. Afterwards, the following work keep optimizing this approach, making it a more mature methodology, e.g., dynamically deciding the usage of retrieval results (Zheng et al., 2021), building a light and explainable datastore (Zhu et al., 2023a), injecting kNN knowledge into the neural model (Zhu et al., 2023b).

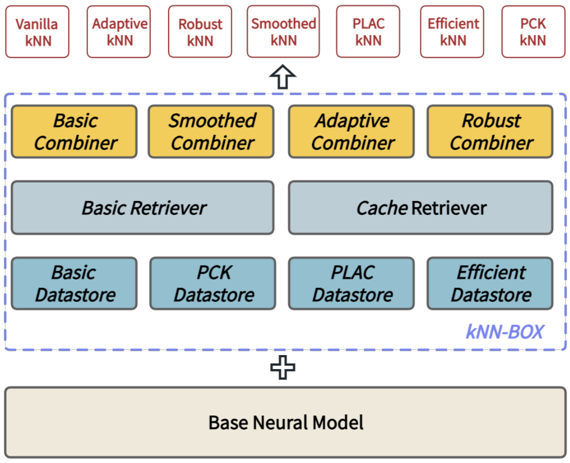

Figure 1: kNN-BOX decomposes the datastore-augmentation approach into three modules, namely, DATASTORE, RETRIEVER and COMBINER, putting diverse kNN generation methods into an unified way.

However, we notice that these kNN generation methods are implemented with diverse codebases, e.g., *Fairseq*[1], *Transformers*[2] and *JoeyNMT*[3], which hinders comparison between these methods and potential fusion of latest research advances. Interpretability is another interesting point in kNN research, as the community is curious why kNN generation works and whether it is reliable.

In this paper, we introduce a unified framework kNN-BOX for nearest neighbor generation, which supports quick development and visualization anal-

*Equal Contributions.

[1] `https://github.com/facebookresearch/fairseq`

[2] `https://github.com/huggingface/transformers`

[3] `https://github.com/joeynmt/joeynmt`

Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics System Demonstrations, pages 10–17
March 17-22, 2024

ysis. Our framework decomposes the datastore-augmentation approach into three modules: DATASTORE, RETRIEVER and COMBINER, thus putting diverse kNN generation methods into a unified way (Figure 1). Up till now, kNN-BOX has released implementation of seven popular kNN-MT models, covering research from performance enhancement (Khandelwal et al., 2021; Jiang et al., 2021; Zheng et al., 2021; Jiang et al., 2022) to efficiency optimization (Martins et al., 2022; Wang et al., 2022; Zhu et al., 2023a), which can help users to quickly reproduce existing works. Moreover, users can easily fuse advanced models with kNN-BOX, for example, jointly using a better combiner and a lighter datastore, to achieve the best of both worlds.

Another useful feature of kNN-BOX is supporting visualized interactive analysis. Via our provided web service, users can interact with their kNN model and observe its inference process, e.g. the content and distribution of its retrieval results (Figure 3). We hope kNN-BOX can help the community to better understand the interpretability of kNN generation.

Experiment results on machine translation datasets show that kNN-BOX is a reliable platform for model reproduction and development. In addition, we apply kNN-BOX for three other seq2seq tasks, i.e., text simplification, paraphrase generation and question generation. Experiment results show that augmenting the base neural model with kNN-BOX is also beneficial in these tasks, showing the great potential of nearest neighbor generation and the wide usage of our kNN-BOX toolkit. At the time of writing, we are happy to see that kNN-BOX has been used as the backbone of this year's ACL paper (Liu et al., 2023) and EMNLP papers (Li et al., 2023; Zhang et al., 2023), and we hope this toolkit to support more valuable research in the future.

2 Background: kNN-MT

Before introducing kNN-BOX, we recap kNN-MT approach in this section. Generally, kNN-MT framework aims at memorizing translation knowledge in parallel corpus $\mathcal{C}$ into a datastore $\mathcal{D}$ and use it to augment the NMT model $\mathcal{M}$ during inference.

Memorizing Knowledge into Datastore To extract translation knowledge, translation pair $(\mathcal{X}, \mathcal{Y})$ is fed into $\mathcal{M}$ for teacher-forcing decoding. At time step t, the continuous representation of the translation context $(\mathcal{X}, \mathcal{Y}_{<t})$, i.e. the hidden state h_t from the last decoder layer, is taken as *key*:

$$h_t = \mathcal{M}(\mathcal{X}, \mathcal{Y}_{<t})$$

and the target token y_t is taken as *value*. Each *key-value* pair explicitly memorizes the translation knowledge: generating the *value* token at the decoder hidden state *key*. With a single forward pass over the entire corpus, the full datastore $\mathcal{D}$ can be constructed:

$$\mathcal{D} = \{(h_t, y_t) \mid \forall y_t \in \mathcal{Y}, (\mathcal{X}, \mathcal{Y}) \in \mathcal{C}\}, \tag{1}$$

Generating with Memorized Knowledge The constructed datastore is then combined with the base NMT model as an augmentation memory. During inference, the NMT model retrieves related knowledge from the datastore to adjust its own translation prediction.

Specifically, the NMT model uses the contextualized representation of the test translation context $(\mathcal{X}, \mathcal{Y}_{<t})$ to query the datastore for nearest neighbor representations and the corresponding target tokens $\mathcal{N}_k = \{(h^j, y^j)\}_{j=1}^{k}$. The retrieved entries are then converted to a distribution over the vocabulary:

$$p_{\text{knn}}(y|\mathcal{X}, \mathcal{Y}_{<t}) \propto \sum_{(h^j, y^j) \in \mathcal{N}_k} \mathbb{1}(y = y^j) \cdot s(h_t, h^j) \tag{2}$$

where s measures the similarity between h_t and h^j:

$$s(h_t, h^j) = \exp[\frac{-d(h_t, h^j)}{T}]$$

Here, d denotes L_2-square distance and T is the temperature. In the end, the output distribution of the NMT model and symbolic datastore are interpolated with the weight λ:

$$\begin{aligned} p(y|\mathcal{X}, \mathcal{Y}_{<t}) = \lambda \cdot\ & p_{\text{knn}}(y|\mathcal{X}, \mathcal{Y}_{<t}) \\ + (1-\lambda) \cdot\ & p_{\text{nmt}}(y|\mathcal{X}, \mathcal{Y}_{<t}) \end{aligned} \tag{3}$$

Recent Advances in kNN-MT To make kNN-MT more effective, efficient and explainable, various methods have been devised. Zheng et al. (2021) and Jiang et al. (2022) propose to dynamically decide the usage of retrieval results to exclude potential noise in nearest neighbors. Jiang et al. (2021) explore the setting of multi-domain adaptation and remedy the catastrophic forgetting problem. Inspired by He et al. (2021), Martins et al. (2022)

introduce three ways to improve the efficiency of kNN-MT, i.e. dimension reduction, datastore pruning and adaptive retrieval. Later, Wang et al. (2022) propose to reduce dimension and prune datastore with a learnable network. Recently, Zhu et al. (2023a) explore the interpretability issue in kNN-MT and builds a light and more explainable datastore according to the capability of the NMT model.

3 Unified Framework: kNN-BOX

This section describes how we design and implement kNN-BOX, and introduce how users run kNN-BOX for developing kNN generation models and interacting with the deployed model visually.

3.1 Design and Implementation

We develop kNN-BOX based on the widely-used generation framework *Fairseq*, making it easy to apply kNN-BOX for other generation tasks. The overall workflow of kNN-BOX is illustrated in Figure 2. For better compatibility and extensibility, we decompose the datastore-augmentation approach into three modules: DATASTORE, RETRIEVER and COMBINER, where each module has its own function:

- DATASTORE: saving generation knowledge as *key-values* pairs (Equation 1).
- RETRIEVER: retrieving nearest neighbors from the datastore during inference.
- COMBINER: converting retrieval results to a distribution (Equation 2) and interpolating the output distribution of the neueal model and symbolic datastore (Equation 3).

With this design, diverse kNN models can be implemented in a unified way. For a specific kNN variant, it usually makes a modification on one of the three modules, compared to vanilla kNN generation model. Therefore, users can customize the corresponding module and quickly develop a kNN generation model.

Supporting visual interactive analysis is another useful feature of kNN-MT. By saving intermediate computation results, we enable kNN-BOX to visualize the inference process. We hope this feature will help users to better understand their own model.

3.2 Usage

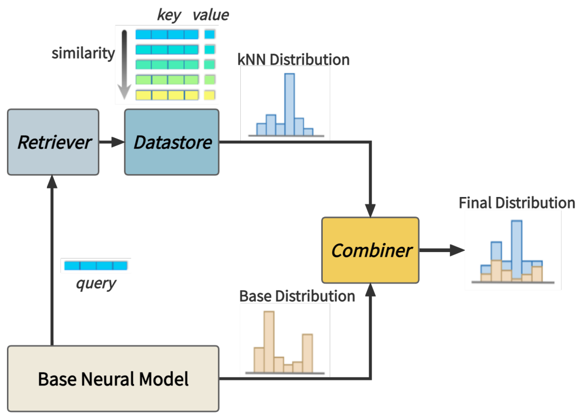

Figure 2: Overall workflow of augmenting the base neural model with kNN-BOX.

Reproducing Existing Work Until now, kNN-BOX has released implementation of seven popular kNN-MT models [4], covering research from performance enhancement to efficiency optimization. Besides, kNN-BOX has also provided the corresponding shell scripts to run them, enabling users to quickly reproduce existing work. Detailed guidance can be found in `README.md`[5].

Developing New Models kNN-BOX is designed not only for reproducing existing work, but also for developing new models on new tasks. For each module, users can pick one of its implementation from kNN-BOX or customize their own version, and combine three modules together to build a new kNN generation model. In this process, only few lines of codes needs to be added, which can save users a lot of time. More importantly, this implementation fashion enables users to easily build a fused model, e.g., combining the most explainable datastore (PLACDATSTORE) with the strongest combiner (ROBUSTCOMBINER). To perform generation tasks other than machine translation, users only need to switch the training corpus to build a task-specific datastore.

Visualizing Generalization Process By running our provided script to launch a web page (shown in Figure 3), users can interact with their kNN general model visually. Users can type in text in the upper

[4] They are vanilla kNN-MT (Khandelwal et al., 2021), Adaptive kNN-MT (Zheng et al., 2021), Smoothed kNN-MT (Jiang et al., 2021), Robust kNN-MT (Jiang et al., 2022), PCK kNN-MT (Wang et al., 2022), Efficient kNN-MT (Martins et al., 2022), PLAC kNN-MT (Zhu et al., 2023a).

[5] https://github.com/NJUNLP/knn-box/blob/master/README.md

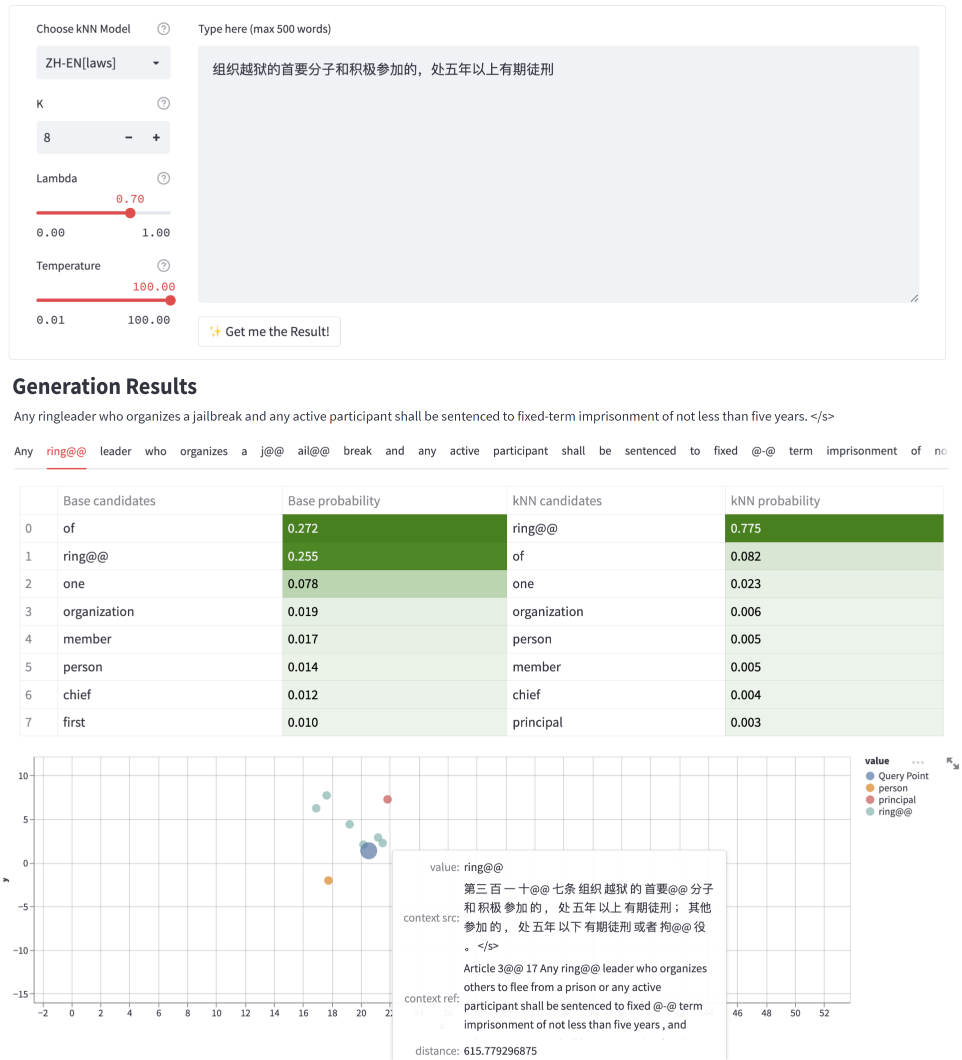

Figure 3: A screenshot of visualization web page provided by kNN-BOX, where users can interact with their own kNN model and analyze its inference process visually. The upper panel allows users to type in text and tune hyperparameters. The middle panel displays the generation result (words with "@@" means that they are generated subwords) and prediction distribution of each decoding step. The bottom panel shows the relative distribution of query and retrieval results, and more detailed information of each nearest neighbor. For example, in this figure, the user moves mouse to one of the nearest entries and check its detailed information.

input window and tune generation hyperparameters in the upper-left panel. The generated results, both detokenized and tokenized, will then be displayed. Taking kNN-MT as an example, after clicking a word in the translation, users can see the translation probability given by both NMT model and kNN-MT model. Moreover, detailed information of the retrieved datastore entries will be displayed in the bottom panel. By selecting on a certain nearest neighbor point, users can see the corresponding value token, translation context and *query-key* distance. Overall, the visualization page can help user to interact with their kNN generation model and explore its inner working process.

4 Experiments

To evaluate the effectiveness of kNN-BOX, we conduct experiments on machine translation and three other seq2seq tasks.

4.1 Experimental Settings

Dataset For machine translation, we adopt four German-English OPUS datasets [6] (Medical, Law, IT and Koran) (Tiedemann, 2012), which are used in almost all kNN-MT work. We use TED dataset [7] (Qi et al., 2018) to evaluate kNN-BOX on multi-

[6] https://opus.nlpl.eu/
[7] https://github.com/neulab/word-embeddings-for-nmt

Model	Reference	Law		Medical		IT		Koran	
		Scale↓	BLEU↑	Scale↓	BLEU↑	Scale↓	BLEU↑	Scale↓	BLEU↑
Base Neural Model	Ng et al., 2019	-	45.5	100%	40.0	-	38.4	-	16.3
Vanilla kNN-MT	Khandelwal et al., 2021	100%	61.3	100%	54.1	100%	45.6	100%	20.4
Adaptive kNN-MT	Zheng et al., 2021	100%	62.9	100%	56.1	100%	47.2	100%	20.3
Smoothed kNN-MT	Jiang et al., 2021	100%	63.3	100%	56.8	100%	47.7	100%	19.9
Robust kNN-MT	Jiang et al., 2022	100%	63.6	100%	57.1	100%	48.6	100%	20.5
PCK kNN-MT	Wang et al., 2022	90%	62.8	90%	56.4	90%	47.4	90%	19.4
Efficient kNN-MT	Martins et al., 2022	57%	59.9	58%	52.3	63%	44.9	66%	19.9
PLAC kNN-MT	Zhu et al., 2023a	55%	62.8	55%	56.2	60%	47.0	75%	19.9

Table 1: Some works implemented by kNN-BOX. Scale refers to the relative datastore size compared to a full datastore that covers all target language token occurrences in the parallel corpus. Smaller scale means a lighter datastore and higher BLEU means better translation quality.

Directions	Model	**Avg.**	**Cs**	**Da**	**De**	**Es**	**Fr**	**It**	**Nl**	**Pl**	**Pt**	**Sv**
En → X	M2M-100	29.1	20.7	36.2	26.7	35.1	33.7	29.8	27.7	15.6	31.9	33.7
	+ kNN-BOX	**32.6**	**22.3**	**40.2**	**29.5**	**39.2**	**38.7**	**33.5**	**31.9**	**17.9**	**37.1**	**36.0**
X → En	M2M-100	33.4	27.5	40.0	31.8	36.6	35.1	33.4	31.9	21.1	38.9	37.3
	+ kNN-BOX	**37.7**	**31.3**	**44.5**	**37.1**	**42.0**	**40.4**	**38.4**	**36.2**	**24.9**	**41.8**	**41.0**

Table 2: Effect of augmenting M2M100 with kNN-BOX (Robust kNN-MT) on multilingual TED dataset. For brevity, we only show the effect of applying Robust kNN with kNN-BOX. "En → X" and "X → En" denotes translating English into other languages and translating other languages into English respectively. Bold text indicates the higher score across two models

lingual machine translation [8]. Moreover, we conduct experiments on two text simplification dataset: NEWSELA-AUTO [9] and WIKI-AUTO [10] (Jiang et al., 2020), a paraphrase generation dataset QQP [11], and a question generation dataset QUASAR-T [12] (Dhingra et al., 2017) to demonstrate effectiveness of kNN-BOX on these generation tasks.

Base Neural Model On OPUS dataset, we follow previous kNN-MT work and use the winner model of WMT'19 De-En news translation task (Ng et al., 2019) as the base model. On multilingual TED dataset, we use M2M100 (Fan et al., 2021) as the base model, which is a many-to-many multilingual translation model. On the rest of dataset, Transformer (Vaswani et al., 2017) is used as the base model.

Metric We use BLEU score calculated by *sacrebleu*[13] to evaluate the generation quality for all tasks except text simplification, where we use SARI score (Xu et al., 2016) calculated by *easse*[14] to evaluate simplification quality.

4.2 Main Results

kNN-BOX can help user to quickly augment the base NMT model with kNN methods. By running our provided shell scripts, users can quickly reproduce existing kNN-MT models. Table 1 show the translation performance of these models on OPUS dataset. We see that augmenting the base neural machine translation model with a datastore brings significant performance enhancement. Among these methods, Robust kNN-MT achieves the highest BLEU scores, and PLAC kNN-MT builds a lightest datastore while maintaining translation performance. Table 2 reports experiment results on TED dataset. We can see that applying kNN-BOX brings large performance improvement on all translation directions.

kNN-BOX is reliable platform for model reproduction We carefully compare the reproduced results with the results produced by the original implementation. We find that two groups of results

[8] We evaluate English-centric translation performance on ten languages: Cs, Da, De, Es, Fr, It, Nl, Pl, Pt and Sv.

[9] https://newsela.com/data/

[10] https://github.com/chaojiang06/wiki-auto/tree/master/wiki-auto/ACL2020/

[11] https://www.kaggle.com/c/quora-question-pairs

[12] https://github.com/bdhingra/quasar

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

[14] https://github.com/feralvam/easse

Task	Dataset	Metric	Base Model	kNN-BOX
Text Simplification	Wiki-Auto	SARI	38.6	**39.4**
	Newsela-Auto	SARI	35.8	**38.2**
Paraphrase Generation	QQP	BLEU	28.4	**29.5**
Question Generation	Quasar-T	BLEU	9.6	**15.7**

Table 3: The performance of applying kNN-BOX (vanilla kNN-MT) on three other seq2seq tasks: text simplification, paraphrase generation and question generation. Here, we apply the vanilla kNN generation method for augmentation. Bold text indicates the higher score across two models. Augmenting base neural models in these tasks with kNN-BOX also bring large performance improvement.

Datastore	Retriever	Combiner	Scale↓	BLEU↑
BasicDatastore	BasicRetriever	BasicCombiner	100%	61.3
PCKDatastore	BasicRetriever	AdaptiveCombiner	90%	62.8
EfficientDatastore	BasicRetriever	AdaptiveCombiner	57%	61.5
EfficientDatastore	BasicRetriever	RobustCombiner	57%	61.8
PLACDatastore	BasicRetriever	AdaptiveCombiner	55%	62.8
PLACDatastore	BasicRetriever	RobustCombiner	55%	63.7

Table 4: Experiment results of fusing advanced datastore and combiner. Smaller scale means a lighter datastore and higher BLEU means better translation quality.

are well-aligned (shown in Appendix A), demonstrating that kNN-BOX is reliable platform for reproducing kNN-MT models.

kNN-BOX shows great potential in other seq2seq generation tasks as well Apart from machine translation task, we further evaluate kNN-BOX on three other seq2seq tasks: text simplification, paraphrase generation and question generation. Experiment results are shown in Table 3. Augmenting the base neural model with kNN-BOX brings performance enhancement in all three tasks. The performance improvement on three tasks is up to 2.4 SARI, 1.1 BLEU and 6.1 BLEU respectively, which shows the great potential of studying datastore-augmentation in generation tasks and the wide usage of our toolkit.

kNN-BOX accelerates the fusion of lasted research advances A potential drawback of implementing kNN-MT with diverse codebases is hindering the fusion of lasted research advances. With kNN-BOX, research advances on Datastore, Combiner and Retriever can be fused conveniently. Table 4 shows the performance of some mixed models on OPUS-Law dataset, where we jointly use different Datastore and Combiner. We can see that jointly using PLACDatastore and RobustCombiner achieve strong translation performance with a much smaller datastore.

5 Conclusion and Future Work

This paper introduces kNN-BOX, an open-sourced toolkit for nearest neighbor generation. kNN-BOX decomposes datastore-augmented approach into three decoupled modules: Datastore, Retriever and Combiner, thus putting diverse kNN generation methods into a unified way. kNN-BOX provides implementation of several kNN-MT models, covering research from performance enhancement and efficiency optimization, which can help users to quickly reproduce existing work. kNN-BOX also enjoys great extensibility, which can be used to develop new models and be applied for new generation tasks. More importantly, kNN-BOX supports users to interact with their deployed model in a visualized way, which enables in-depth analysis on the inner working process of the model. In experiment section, we show that kNN-BOX can not only be applied for enhancing neural machine translation model, but also for enhancing neural generation model in other seq2seq tasks.

In the future, we will keep update this toolkit to provide implementation of more retrieve-and-generate methods and optimize the framework to make it more user-friendly, and explore the possibility to apply kNN-BOX for more generation tasks.

Limitation

We discuss two potential limitations of our kNN-BOX toolkit below:

- Inference Latency: The nearest neighbor retrieval system queries the datastore at each timestep, which introduces inference latency.
- Datastore reusability: The datastore is constructed using a specific model, which limits its reusability. This means that the datastore cannot be seamlessly integrated or utilized with other models.

Acknowledgement

Shujian Huang is the corresponding author. This work is supported by National Science Foundation of China (No. 62376116, 62176120), the Liaoning Provincial Research Foundation for Basic Research (No. 2022-KF-26-02).

References

Bhuwan Dhingra, Kathryn Mazaitis, and William W Cohen. 2017. Quasar: Datasets for question answering by search and reading. *arXiv preprint arXiv:1707.03904*.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *The Journal of Machine Learning Research*, 22(1):4839–4886.

Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. 2021. Efficient nearest neighbor language models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Chao Jiang, Mounica Maddela, Wuwei Lan, Yang Zhong, and Wei Xu. 2020. Neural CRF model for sentence alignment in text simplification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7943–7960, Online. Association for Computational Linguistics.

Hui Jiang, Ziyao Lu, Fandong Meng, Chulun Zhou, Jie Zhou, Degen Huang, and Jinsong Su. 2022. Towards robust k-nearest-neighbor machine translation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5468–5477, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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 the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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

Xuanhong Li, Peng Li, and Po Hu. 2023. Revisiting source context in nearest neighbor machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Shudong Liu, Xuebo Liu, Derek F. Wong, Zhaocong Li, Wenxiang Jiao, Lidia S. Chao, and Min Zhang. 2023. kNN-TL: k-nearest-neighbor transfer learning for low-resource neural machine translation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*.

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

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 Conference on Machine Translation (WMT)*.

Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. 2018. When and why are pre-trained word embeddings useful for neural machine translation? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC)*.

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 (NeurIPS)*.

Dexin Wang, Kai Fan, Boxing Chen, and Deyi Xiong. 2022. Efficient cluster-based k-nearest-neighbor machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Xiangyu Zhang, Yu Zhou, Guang Yang, and Taolue Chen. 2023. Syntax-aware retrieval augmented code generation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*.

Xin Zheng, Zhirui Zhang, Junliang Guo, Shujian Huang, Boxing Chen, Weihua Luo, and Jiajun Chen. 2021. Adaptive nearest neighbor machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Wenhao Zhu, Shujian Huang, Yunzhe Lv, Xin Zheng, and Jiajun Chen. 2023a. What knowledge is needed? towards explainable memory for kNN-MT domain adaptation. In *Findings of the Association for Computational Linguistics: ACL 2023*.

Wenhao Zhu, Jingjing Xu, Shujian Huang, Lingpeng Kong, and Jiajun Chen. 2023b. INK: Injecting kNN knowledge in nearest neighbor machine translation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*.

A Performance Alignment between kNN-BOX's implementation and original implementation

Table 5 compares the reproduced results with kNN-BOX and the results produced by the original implementation, where the same base neural model and the same dataset is used. Comparison results show that there is only a minor gap between two groups of results, demonstrating that the reliability of kNN-BOX.

Model	Law	Medical	IT	Koran
Base NMT[15]	45.5	40.0	38.4	16.3
$\hookrightarrow$ kNN-BOX	45.5	40.0	38.4	16.3
Vanilla kNN-MT [16]	61.3	54.1	45.6	20.4
$\hookrightarrow$ kNN-BOX	61.3	54.1	45.6	20.4
Adaptive kNN-MT [17]	62.9	56.6	47.6	20.6
$\hookrightarrow$ kNN-BOX	62.9	56.1	47.2	20.3
PCK kNN-MT [18]	63.1	56.5	47.9	19.7
$\hookrightarrow$ kNN-BOX	62.8	56.4	47.4	19.4
Robust kNN-MT [19]	63.8	57.0	48.7	20.8
$\hookrightarrow$ kNN-BOX	63.6	57.1	48.6	20.5

Table 5: BLEU scores of original implementation and kNN-BOX's implementation. "$\hookrightarrow$ kNN-BOX' denotes the reults reproduced using our framework.

[15] https://github.com/facebookresearch/fairseq

[16] https://github.com/urvashik/knnmt

[17] https://github.com/zhengxxn/adaptive-knn-mt

[18] https://github.com/tjunlp-lab/PCKMT

[19] https://github.com/DeepLearnXMU/Robust-knn-mt