Learning Kernel-Smoothed Machine Translation with Retrieved Examples

Qingnan Jiang[1][*] Mingxuan Wang[2], Jun Cao[2], Shanbo Cheng[2], Shujian Huang[1][†] Lei Li[3][*]

[1]National Key Laboratory for Novel Software Technology, Nanjing University, China
[2]ByteDance AI Lab,   [3]University of California, Santa Barbara
[1]jiangqn@smail.nju.edu.cn, huangsj@nju.edu.cn
[2]{wangmingxuan.89, caojun.sh, chengshanbo}@bytedance.com
[3]lilei@cs.ucsb.edu

Abstract

How to effectively adapt neural machine translation (NMT) models according to emerging cases without retraining? Despite the great success of neural machine translation, updating the deployed models online remains a challenge. Existing non-parametric approaches that retrieve similar examples from a database to guide the translation process are promising but are prone to overfit the retrieved examples. However, non-parametric methods are prone to overfit the retrieved examples. In this work, we propose to learn Kernel-Smoothed Translation with Example Retrieval (KSTER), an effective approach to adapt neural machine translation models online. Experiments on domain adaptation and multi-domain machine translation datasets show that even without expensive retraining, KSTER is able to achieve improvement of 1.1 to 1.5 BLEU scores over the best existing online adaptation methods. The code and trained models are released at https://github.com/jiangqn/KSTER.

1 Introduction

Over the past years, end-to-end Neural Machine Translation (NMT) models have achieved great success (Bahdanau et al., 2015; Wu et al., 2016; Vaswani et al., 2017). How to effectively update a deployed NMT model and adapt to emerging cases? For example, after a generic NMT model trained on WMT data, a customer wants to use service to translate financial documents. The costomer may have a handful of translation pairs for the finance domain, but do not have the capacity to perform a full retraining.

Non-parametric adaptation methods enable incorporating individual examples on-the-fly, by retrieving similar source-target pairs from an external database to guide the translation process (Bapna and Firat, 2019; Gu et al., 2018; Zhang et al., 2018;

Figure 1: The domain-specific and general domain translation performance in EN-DE translation. Base is a Transformer model trained on general domain WMT data. kNN-MT and our proposed KSTER are adapted for domain-specific translation with in-domain database. Both kNN-MT and KSTER achieve improvements over Base in domain-specific translation performance. But kNN-MT overfits to in-domain data and performs bad in general domain translation, while the proposed KSTER achieves comparable general domain translation performance with Base.

Cao and Xiong, 2018). The external database can be easily updated online. Most of these methods rely on effective sentence-level retrieval. Different from sentence retrieval, k-nearest-neighbour machine translation introduces token level retrieval to improve translation (Khandelwal et al., 2021). It shows promising results for online domain adaptation.

There are still limitations for existing non-parametric methods for online adaptation. First, since it is not easy for sentence-level retrieval to find examples that are similar enough to the test example, this low overlap between test examples

and retrieved examples brings noise to translation (Bapna and Firat, 2019). Second, completely non-parametric methods are prone to overfit the retrieved examples. For example, although kNN-MT improves domain-specific translation, it overfits severely and can not generalize to the general domain, as is shown in Figure 1. An ideal online adaptation method should introduce less noise to the translation process and generalize to the changeful test examples with the incrementally changing database.

In this paper, we propose to learn Kernel-Smoothed Translation with Example Retrieval (KSTER), to effectively learn and adapt neural machine translation online. KSTER retains the online adaptation advantage of non-parametric methods and avoids the drawback of easy overfitting. More specifically, KSTER improves the generalization ability of non-parametric methods in three aspects. First, we introduce a learnable kernel to dynamically measure the relevance of the retrieved examples based on the current context. Then, the exampled-based distribution is combined with the model-based distribution computed by NMT with adaptive mixing weight for next word prediction. Further, to make the learning of KSTER stable, we introduce a retrieval dropout strategy. The intuition is that similar examples can constantly be retrieved during training, but not the same situation during inference. We therefore drop the most similar examples during training to reduce this discrepancy.

With above improvements, KSTER shows the following advantages:

- Extensive experiments show that, KSTER outperforms kNN-MT, a strong competitor, in specific domains for 1.1 to 1.5 BLEU scores while keeping the performance in general domain.
- KSTER outperforms kNN-MT for 1.8 BLEU scores on average in unseen domains. Therefore, there is no strong restriction of the input domain, which makes KSTER much more practical for industry applications.

2   Related Work

This work is mostly related to two research areas in machine translation (MT), i.e., domain adaptation for machine translation and online adaptation of MT models by non-parametric retrieval.

Domain Adaptation for MT   Domain adaptation for MT aims to adapt general domain MT mod-els for domain-specific language translation (Chu and Wang, 2018). The most popular method for this task is finetuning general domain MT models on in-domain data. However, finetuning suffers from the notorious catastrophic forgetting problem (McCloskey and Cohen, 1989; Santoro et al., 2016). There are also some sparse domain adaptation methods that only update part of the MT parameters (Bapna et al., 2019; Wuebker et al., 2018; Guo et al., 2021).

In real-world translation applications, the domain labels of test examples are often not available. This dilemma inspires a closely related research area — multi-domain machine translation (Pham et al., 2021; Farajian et al., 2017; Liang et al., 2020; Lin et al., 2021; Zhu et al., 2021), where one model translates sentences from all domains.

Online Adaptation of MT by Non-parametric Retrieval   Non-parametric methods enable online adaptation of deployed NMT models by updating the database from which similar examples are retrieved.

Traditional non-parametric methods search sentence-level examples to guide the translation process (Cao and Xiong, 2018; Gu et al., 2018; Zhang et al., 2018). Recently, n-gram level (Bapna and Firat, 2019) and token level (Khandelwal et al., 2021) retrieval are introduced and shows strong empirical results. Generally, similar examples are retrieved based on fuzzy matching (Bulte and Tezcan, 2019; Xu et al., 2020), embedding similarity, or a mixture of the two approaches (Bapna and Firat, 2019).

3   Methodology

In this section, we first formulate the kernel-smoothed machine translation (KSTER), which smooths neural machine translation (NMT) output with retrieved token level examples. Then we introduce the modeling and training of the learnable kernel and adaptive mixing weight. The overview of KSTER is shown in Figure 2.

3.1   Kernel-Smoothed Machine Translation

Base Model for Neural Machine Translation The state-of-the-art NMT models are based on the encoder-decoder architecture. The encoder encodes the source text x into a sequence of hidden states. The decoder takes the representations of the source text as input and generates target text autoregressively. In each decoding step, the decoder
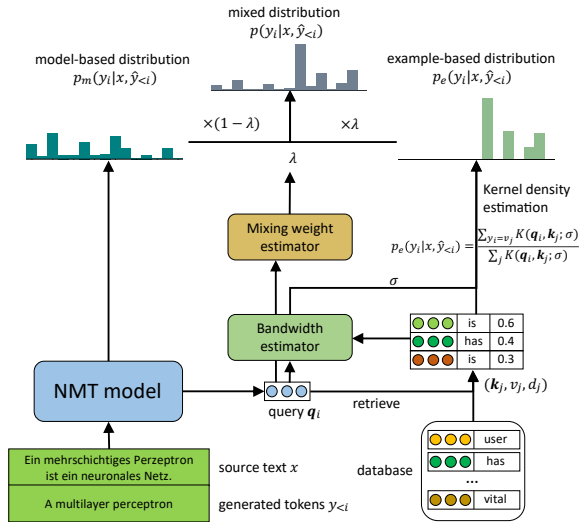
Figure 2: Overview of the KSTER. The left part is the NMT model denoted as base model. The right part shows the process of retrieving similar examples and estimating the example-based distribution.

predicts the probability distribution of next tokens $p_m(y_i|x, \hat{y}_{<i}; \theta)$ conditioned on the source text x and previously generated target tokens $\hat{y}_{<i}$. The NMT model parameters are denoted as θ.

Kernel Smoothing with Retrieved Examples
The model-based distribution $p_m(y_i|x, \hat{y}_{<i}; \theta)$ is then smoothed by an example-based distribution $p_e(y_i|x, \hat{y}_{<i})$. It is computed using kernel density estimation (KDE) on retrieved examples.

We build a database from which similar examples are retrieved. The database consists of all token level examples from the training set in the form of key-value pairs. In each key-value pair $(\mathbf{k}, v)$, the key $\mathbf{k} = f_{\text{NMT}}(x, y_{<i}; \theta)$ is the intermediate representation of a certain layer in the NMT decoder. The value is the corresponding ground truth target token y_i. The key $\mathbf{k}$ can be seen as a vector representation of the context of value v. We obtain the key-value pairs from (x, y) by running force-decoding with a trained NMT model.

In each decoding step, we compute the query $\mathbf{q}_i = f_{\text{NMT}}(x, \hat{y}_{<i}; \theta)$ and retrieve k similar examples based on the L^2 distance [1] query and keys. Each retrieved example forms a triple $(\mathbf{k}_j, v_j, d_j)$, where $\mathbf{k}_j$ is the key; v_j is the corresponding value token and d_j is the L^2 distance between query $\mathbf{q}_i$ and key $\mathbf{k}_j$. The example-based distribution $p_e(y_i|x, \hat{y}_{<i})$ is then computed with these retrieved

---

[1]For two d-dimension vectors $\mathbf{x}$ and $\mathbf{y}$, we compute the L^2 distance between $\mathbf{x}$ and $\mathbf{y}$ as $\sqrt{\sum_{i=1}^{d}(\mathbf{x}_i - \mathbf{y}_i)^2}$.
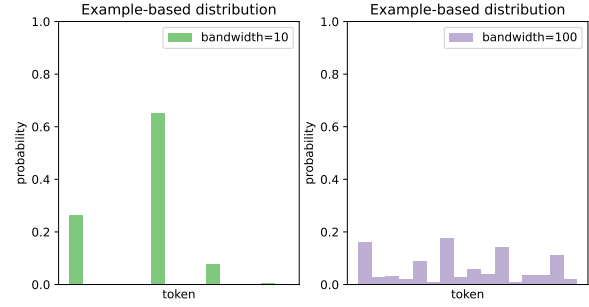

Figure 3: A visualization of example-based distribution estimated by Gaussian kernel KDE with different bandwidth. Bandwidth controls the smoothness of example-based distribution.

examples using the following equation.

$$p_e(y_i|x, \hat{y}_{<i}) = \frac{\sum_{y_i=v_j} K(\mathbf{q}_i, \mathbf{k}_j; \sigma)}{\sum_j K(\mathbf{q}_i, \mathbf{k}_j; \sigma)} \quad (1)$$

where $K(\mathbf{q}, \mathbf{k}; \sigma)$ is the kernel function and σ is the parameter of the kernel.

Finally, the NMT output is smoothed by combing the model-based distribution and the example-based distribution with a mixing weight λ.

$$p(y_i|x, \hat{y}_{<i}; \theta) = \lambda p_e(y_i|x, \hat{y}_{<i}) \quad (2)$$
$$+ (1 - \lambda)p_m(y_i|x, \hat{y}_{<i}; \theta) \quad (3)$$

3.2 Learnable Kernel Function

Although all kernel functions can be used in KDE, we choose two specific kernels in this study — Gaussian kernel and Laplacian kernel, since they are easy to parameterize.

The only parameter in Gaussian kernel $K_g(\mathbf{q}_i, \mathbf{k}_j; \sigma) = \exp(-\frac{\|\mathbf{q}_i - \mathbf{k}_j\|^2}{\sigma})$ is the bandwidth σ, it controls the smoothness of the example-based distribution, as is shown in Figure 3.

In a learnable Gaussian kernel, the bandwidth is not a fixed hyper-parameter. Instead, it is estimated in each decoding step by a learned affine network with exponential activation.

$$\sigma = \exp(\mathbf{W}_1[\mathbf{h}_i; \overline{\mathbf{k}}_i] + \mathbf{b}_1) \quad (4)$$

$\overline{\mathbf{k}}_i = \frac{1}{k}\sum_{j=1}^{k} \mathbf{k}_j$ is average-pooled keys and $[\mathbf{W}_1; \mathbf{b}_1]$ are trainable parameters.

The bandwidth of learnable Laplacian kernel $K_l(\mathbf{q}_i, \mathbf{k}_j; \sigma) = \exp(-\frac{\|\mathbf{q}_i - \mathbf{k}_j\|}{\sigma})$ is modeled in the same way as the bandwidth of learnable Gaussian kernel.

	Law	Medical	Koran	IT	Subtitles
Train	467k	248k	18k	223k	500k
Dev	2k	2k	2k	2k	2k
Test	2k	2k	2k	2k	2k

Table 1: The number of training, development and test examples of 5 domain-specific datasets. The training data of Subtitles domain is sampled from the full Subtitles training set by Aharoni and Goldberg (2020).

3.3 Adaptive Mixing of Base Prediction and Retrieved Examples

To mix the model-based distribution and example-based distribution adaptively, we model the mixing weight λ with a learnable neural network.

The mixing weight λ is computed by a multi-layer perceptron with query $\mathbf{q}_i$ and weighted sum of keys $\widetilde{\mathbf{k}}$ as inputs, where $[\mathbf{W}_2; \mathbf{b}_2; \mathbf{W}_3; \mathbf{b}_3]$ are trainable parameters.

$$\lambda = \text{sigmoid}(\mathbf{W}_3 \text{ReLU}(\mathbf{W}_2[\mathbf{q}_i; \widetilde{\mathbf{k}}_i] + \mathbf{b}_2) + \mathbf{b}_3) \tag{5}$$

$$\widetilde{\mathbf{k}}_i = \sum_{j=1}^{k} w_j \mathbf{k}_j \tag{6}$$

$$w_j \propto K(\mathbf{q}_i, \mathbf{k}_j; \theta) \tag{7}$$

In this way, kNN-MT (Khandelwal et al., 2021) could be seen as a specific case of KSTER, with fixed Gaussian kernel and mixing weight.

3.4 Training

We optimize the KSTER model by minimizing the cross entropy loss between the mixed distribution and ground truth target tokens:

$$\min_{\phi} -\sum_{i=1}^{n} \log p(y_i|x, y_{<i}; \theta, \phi) \tag{8}$$

where n is the length of a target sentence y.

We keep the NMT model parameters θ fixed. Only parameters of learnable kernel and mixing weight $\phi = [\mathbf{W}_1; \mathbf{b}_1; \mathbf{W}_2; \mathbf{b}_2; \mathbf{W}_3; \mathbf{b}_3]$ are trained.

3.5 Retrieval Dropout

Since the database is built from the training data and KSTER is trained on the training data, similar examples can constantly be retrieved from the database during training. However, in test time, there may be no example in the database that is similar to the query. This discrepancy between training and inference may lead to overfitting.

	Law	Medical	Koran	IT	Subtitles
EN-DE	16.1M	6.5M	0.5M	3.4M	6.3M
DE-EN	15.8M	6.2M	0.5M	3.0M	6.4M

Table 2: The database size — number of examples, of each domain in DAMT.

We propose a simple training strategy called retrieval dropout to alleviate this problem. During training, we search top $k + 1$ similar examples instead of top k examples. Then we drop the most similar example and use the remaining k examples for training. Retrieval dropout is not used in test time.

4 Experiments

We evaluate the proposed methods on two machine translation tasks: domain adaptation for machine translation (DAMT) and multi-domain machine translation (MDMT). In DAMT, in-domain translation model is built for each specific domain, since the domain labels of examples are available in test time. In MDMT, the domain labels of examples are not available in test time, so examples from all domains are translated with one model, which is a more practical setting.

4.1 Datasets and Implementation Details

Datasets We conduct experiments in EN-DE translation and DE-EN translation. We use WMT14 EN-DE dataset (Bojar et al., 2014) as general domain training data, which consists of 4.5M sentence pairs. *newstest2013* and *newstest2014* are used as the general domain development set and test set, respectively. 5 domain-specific datasets proposed by Koehn and Knowles (2017) and re-splitted by Aharoni and Goldberg (2020)[2] are used to evaluate the domain-specific translation performance. The detailed statistics of the 5 datasets are shown in Table 1.

Implementation Details We use joint Byte Pair Encoding (BPE) (Sennrich et al., 2016) with 30k merge operations for subword segmentation. The resulted vocabulary is shared between source and target languages. We employ Transformer Base (Vaswani et al., 2017) as the base model. Following Khandelwal et al. (2021), the normalized inputs of feed forward network in the last Transformer decoder block are used as keys to build the

---

[2]https://github.com/roeeaharoni/unsupervised-domain-clusters

Direction	Methods	Law	Medical	Koran	IT	Subtitles	Average-specific	Average-general (WMT14)
EN-DE	Base	33.36	30.54	10.16	22.99	20.65	23.54	**27.20**
	Finetuning	49.07	47.10	**25.98**	**36.28**	**26.00**	**36.89**	14.17
	*k*NN-MT	51.88	47.02	18.51	29.12	22.46	33.80	8.32
	KSTER	**53.63**	**49.18**	19.10	30.28	22.54	34.95	25.63
DE-EN	Base	36.80	33.36	11.24	29.21	23.13	26.75	**31.49**
	Finetuning	55.19	51.35	**22.87**	**41.88**	**28.33**	**39.92**	17.82
	*k*NN-MT	57.40	50.92	15.74	34.92	25.38	36.87	13.18
	KSTER	**59.41**	**53.40**	16.97	35.74	25.94	38.29	30.23

Table 3: Test set BLEU scores of DAMT. Laplacian kernel is used in KSTER. Average-specific and average-general domain represent the averaged performance of adapted models in domain-specific translation and general domain translation. KSTER outperforms *k*NN-MT for 1.2 and 1.4 BLEU scores on average in EN-DE and DE-EN directions. Significance test by paired bootstrap resampling shows that KSTER outperforms *k*NN-MT significantly in all domains except for Subtitles domain in EN-DE translation and IT domain in DE-EN translation.

databases and queries for retrieval. The translation performance is evaluated with detokenized BLEU scores (Papineni et al., 2002), computed by Sacre-BLEU (Post, 2018) [3].

We build a FAISS (Johnson et al., 2017) index for nearest neighbour search. We employ inverted file and product quantization to accelerate retrieval in large scale databases. The keys of examples are stored in the fp16 format to reduce the memory demand. We set $k = 16$ to keep a balance between translation quality and inference speed.

We train the base model for 200k steps. The best 5 checkpoints are averaged to obtain the final model. We train KSTER for 30k steps. For the training procedures of all models, each batch contains 32,768 tokens approximately. The models are optimized by Adam optimizer (Kingma and Ba, 2015) with learning rates set to 0.0002.

KSTER introduced 526k trainable parameters, which is 0.85% of the base model. We implement all the models based on JoeyNMT (Kreutzer et al., 2019) [4].

4.2 Domain Adaptation for Machine Translation

We build individual database for each specific domain with in-domain training data in DAMT. The sizes of databases are shown in Table 2.

Baselines   We compare the proposed method with the following baselines.
- **Base** base model trained on general-domain data.
- **Finetuning** base model trained on general domain dataset and then finetuned with in-domain data for each specific domain individually.

Method	EN-DE	DE-EN
*k*NN-MT	33.80	36.87
+ 10% source noise	31.26 (-2.54)	/
+ 10% target noise	/	33.43 (-3.44)
KSTER	34.78	38.17
+ 10% source noise	32.89 (-1.89)	/
+ 10% target noise	/	35.67 (-2.50)

Table 4: The averaged BLEU scores over 5 specific domains in DAMT with noisy database. KSTER is more robust than *k*NN-MT when the quality of database is not good.

- *k***NN-MT** *k*NN-MT with in-domain database individually, where the hyper-parameters are tuned on development set of each domain.

The KSTER model is trained for each specific domain individually for fair comparison .

Main results   The DAMT experiment results are shown in Table 3. For domain-specific translation, KSTER outperforms *k*NN-MT for 1.2 and 1.4 BLEU scores on average in EN-DE and DE-EN translation respectively. Finetuning achieves best domain-specific performance on average. However, the performance of finetuned models on general domain drops significantly due to the catastrophic forgetting problem. The even worse general domain performance of *k*NN-MT indicates that it overfits to the retrieved examples severely. KSTER performs far better than finetuning and *k*NN-MT in general domain, which shows strong generalization ability. We notice that KSTER with Laplacian kernel performs slightly better than Gaussian kernel, since KSTER with Gaussian kernel tends to ignore the long-tailed retrieved examples.

Robustness test   The performance of MT model with non-parametric retrieval is influenced by the size and quality of database. Khandelwal et al. (2021) have studied how translation performance

Direction	Methods	General (WMT14)	Law	Medical	Koran	IT	Subtitles	Average-specific
EN-DE	Base	27.20	33.36	30.54	10.16	22.99	20.65	23.54
	Joint-training	27.25	45.02	44.52	15.43	**34.48**	**25.16**	32.92
	kNN-MT	24.72	51.24	46.54	**16.29**	29.55	21.80	33.08
	KSTER	**27.69**	**53.04**	**49.23**	15.94	31.82	22.63	**34.53**
DE-EN	Base	31.49	36.80	33.36	11.24	29.21	23.13	26.75
	Joint-training	31.62	50.95	47.48	**18.13**	**39.57**	**27.73**	36.77
	kNN-MT	25.87	57.38	50.83	14.57	37.56	22.86	36.64
	KSTER	**31.94**	**58.64**	**52.79**	15.24	36.90	25.15	**37.74**

Table 5: Test set BLEU scores of multi-domain machine translation. Average-specific is the averaged performance in 5 specific domains. For general domain sentence translation, KSTER outperforms kNN-MT for 3 and 6 BLEU scores in EN-DE and DE-EN direction respectively. For domain-specific translation, KSTER outperforms kNN-MT for 1.5 and 1.1 BLEU scores in EN-DE and DE-EN direction. Significance test by paired bootstrap (Koehn, 2004) resampling shows that KSTER outperforms kNN-MT significantly in all domains except for Koran domain in EN-DE translation and IT domain in DE-EN translation.

of kNN-MT changes with the size of database. In this work, we study the performance change of kNN-MT and KSTER with low-quality database. Specifically, we test the robustness of these models in DAMT when the database is noisy.

We add token-level noise to the English sentences in parallel training data by EDA (Wei and Zou, 2019) [5]. For each word in a sentence, it is modified with a probability of 0.1. The candidate modifications contain synonym replacement, random insertion, random swap and random deletion with equal probability. Then we use the noisy training data to construct the noisy database.

We study the effects of source side noise and target side noise on translation performance. The experiment results are presented in Table 4. Target side noise has more negative effect to translation performance than source side noise. The BLEU scores of KSTER drop less apparently in all settings, which indicates that the proposed method is more robust with low-quality database.

4.3 Multi-Domain Machine Translation

In MDMT, since there is no domain label available in test time, examples from all domains are translated with one model. We build a mixed database with training data of general domain and 5 specific domains, which is used in all MDMT experiments. The mixed database for EN-DE translation and DE-EN translation contains 172M and 167M key-value pairs respectively.

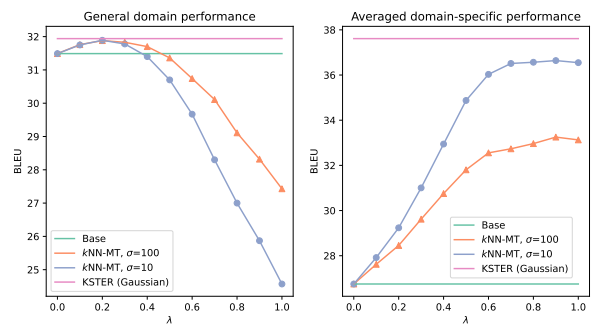**Baselines** We compare the proposed method with the following baselines.

Figure 4: General domain and averaged domain-specific performance of kNN-MT with different hyper-parameter selections in DE-EN direction, together with the performance of Base and KSTER with Gaussian kernel.

- **Base** base model trained on general domain dataset.
- **Joint-training** base model trained on the mixture of general domain dataset and 5 specific domain datasets.
- **kNN-MT** kNN-MT with mixed database. The hyper-parameters are selected that achieve highest averaged development set BLEU scores over general domain and 5 specific domains.

We sample 500k training examples from general domain training set, which are then mixed with all 5 specific domain training examples for KSTER training.

Main results The experiment results of MDMT are shown in Table 5. For general domain sentence translation, KSTER outperforms kNN-MT for 3 and 6 BLEU scores in EN-DE and DE-EN direction respectively. For domain-specific translation, KSTER outperforms kNN-MT for 1.5 and 1.1 BLEU scores in EN-DE and DE-EN direction.
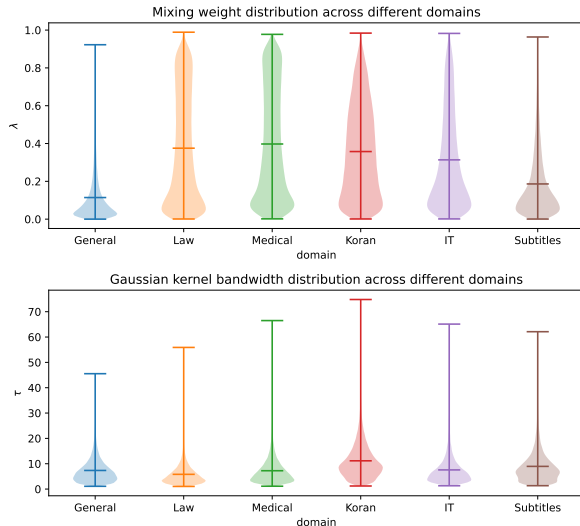
Figure 5: The bandwidth and mixing weight distribution of KSTER with Gaussian kernel.

Method	Bible	QED	Averaged
Base	12.51	25.21	18.86
Joint-training	12.69	25.90	19.30
kNN-MT	12.35	23.56	17.96
KSTER (Laplacian)	**13.32**	**26.16**	**19.74**

Table 6: The test set BLEU scores of MDMT models in unseen domains. KSTER generalizes better than all baselines in unseen domains.

	EN-DE		DE-EN	
	General	Specific	General	Specific
None	24.72	33.08	25.87	36.64
Kernel	26.06	33.40	27.89	37.37
Weight	**27.80**	34.02	31.88	37.19
Both	27.74	**34.38**	**31.94**	**37.61**

Table 7: Ablation study of learnable kernel and mixing weight in KSTER with Gaussian kernel in MDMT. Both learnable kernel and learnable mixing weight bring improvement. None represents that both kernel and mixing weight are fixed, in which case KSTER degenerates to kNN-MT.

Besides, KSTER also outperforms joint-training in both general domain performance and averaged domain-specific performance significantly. The proposed method achieves advantages over joint-training in both online adaptation and translation performance.

General-specific domain performance trade-off We plot the general domain performance and averaged domain-specific performance of kNN-MT with different hyper-parameter selections in Figure 4. We find that kNN-MT performs better in domain-specific translation when the system prediction relies more on the searched examples (low bandwidth and higher mixing weight). In contrast, better general domain translation performance is achieved when the system prediction relies more on NMT prediction (high bandwidth and low mixing weight).

There is a trade-off between general and specific domain performance in kNN-MT. Applying the identical kernel and mixing weight for all test examples can not achieve best performance in general domain and specific domains simultaneously.

KSTER with Gaussian kernel, which is a generalization of kNN-MT, achieves better performance in both general domain and domain-specific translation since it applies adaptive kernel and mixing weight for different test examples. Distributions in Figure 5 indicates that KSTER learns different kernels and different weights for different examples.

Generalization ability over unseen domains We test the generalization ability of baselines and

KSTER with Laplacian kernel in unseen domains, which is important in real-world MDMT applications. We take Bible and QED from OPUS (Tiedemann, 2012)[6] as unseen domains and randomly sample 2k examples from each domain for test. We directly use the MDMT models to translate sentences from unseen domains. The results of EN-DE translation are presented in Table 6. KSTER outperforms all baselines, which shows strong generalization ability.

4.4 Inference Speed

A common concern about non-parametric methods in MT is that searching similar examples may slow the inference speed. We test the inference speed KSTER in MDMT in EN-DE translation, which is the setting with the largest database. The averaged inference time in general domain and 5 specific domain test sets of kNN-MT is 1.15 times of the base model. The averaged inference time of KSTER is 1.19 times of the base model, which is only slightly slower than the baselines.

5 Analysis

In this section, we first conduct ablation studies to verify the effectiveness of each part of the proposed method. Then we conduct detailed analysis on how kernel-smoothing with retrieved examples helps translation.
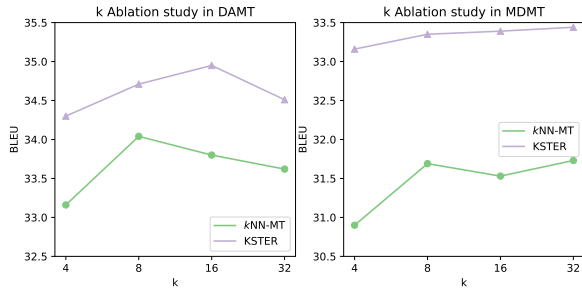
---

[6] https://opus.nlpl.eu/

Figure 6: The ablation study of k selections in DAMT and MDMT settings in EN-DE translation. In DAMT, we report the averaged performance on 5 specific domains. In MDMT, we report the averaged performance on general domain and 5 specific domains. With all k selections, KSTER outperforms kNN-MT consistently.

5.1 Ablation Studies of Proposed Methods

Both learnable kernel and learnable mixing weight bring improvement In KSTER, both the kernel and mixing weight are learnable. We study the effect of keeping only one of the two parts learnable in MDMT.

We take KSTER with Gaussian kernel for analysis. The ablation experiment results are presented in Table 7.

Both learnable kernel and learnable mixing weight bring improvement in both general domain and domain-specific translation. Keeping the two parts learnable simultaneously brings additional improvement. Overall, learnable mixing weight is more important than learnable kernel function.

KSTER outperforms kNN-MT with all k selections We conduct ablation study over different k selections in both DAMT and MDMT settings in EN-DE translation. We experiment with four k selections — $\{4, 8, 16, 32\}$, and plot the results in Figure 6. In DAMT, KSTER achieves best performance with $k = 16$. In MDMT, the performance of our method increases with k. With all k selections, KSTER outperforms kNN-MT consistently.

Retrieval dropout improves generalization We study the effect of retrieval dropout in MDMT and select the KSTER with Laplacian kernel for analysis.

We plot the general domain and averaged domain-specific translation performance of models trained with or without retrieval dropout in Figure 7. Without retrieval dropout, the performance of both general domain and domain-specific translation drops dramatically. The discrepancy between
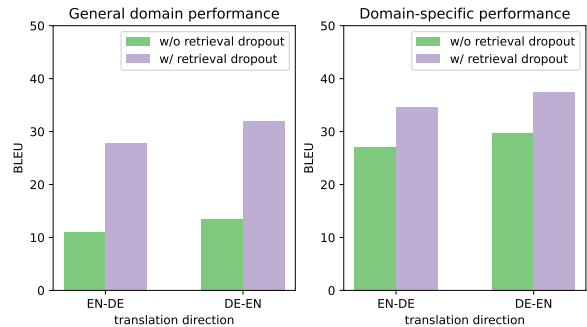

Figure 7: The ablation study of KSTER with Laplacian kernel in MDMT. Without retrieval dropout, KSTER overfits severely.

training and inference leads to severe overfitting. This problem is alleviated by the proposed retrieval dropout, which shows that this training strategy improves the generalization ability of KSTER.

5.2 Fine-grained Effects of Kernel-smoothing with Retrieved Examples on Translation

For better understanding the effects of kernel-smoothing with retrieved examples on translation, we study the following two research questions.

- RQ1. Which types of word kernel-smoothing influences most?
- RQ2. Does kernel-smoothing help word sense disambiguation?

Kernel-smoothing influences verbs, adverbs and nouns most To study the first research question, we categorize the predicted words with their Part-of-Speech tags (POS tags). In each decoding step, if the predicted word y_i obtains the highest probability of example-based distribution but it does not obtain the highest probability of model-based distribution, it is recognized as a prediction determined by kernel-smoothing with retrieved examples.

We compute the ratio of predictions determined by kernel-smoothing across different POS tags. We conduct this analysis on DAMT task in EN-DE direction and select Medical and Subtitles domains as representatives.

We report the results in Figure 8. Medical and Subtitles represent two opposite cases where non-parametric retrieval contributes more in the former and contributes less in the latter. We find that across the 2 different domains, kernel-smoothing contributes most to the predictions of verbs, adverbs and nouns, which are morphologically complex word types. Retrieving words in similar con-
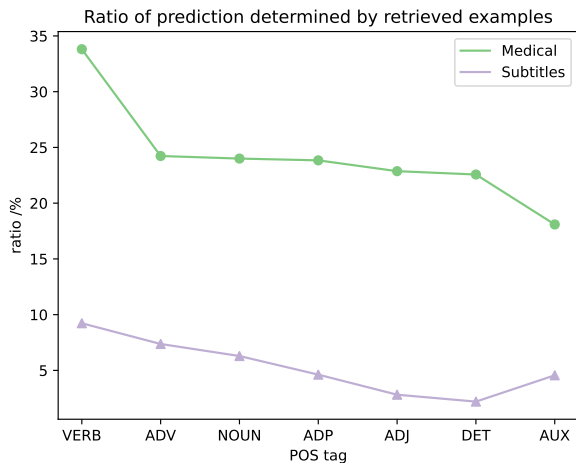
Figure 8: The ratio of word predictions determined by kernel-smoothing. Kernel-smoothing influences verbs, adverbs and nouns most.
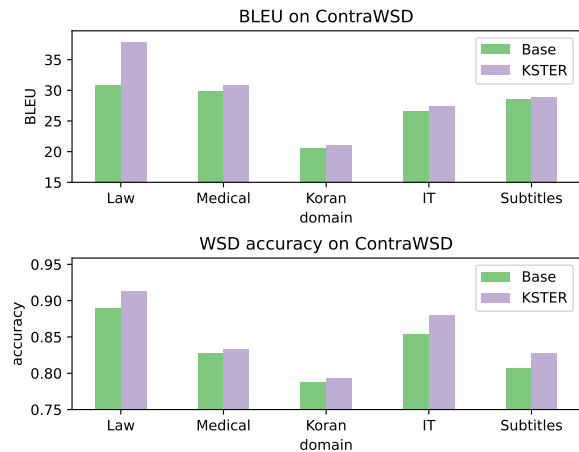

Figure 9: BLEU and word sense disambiguation accuracy of base model and KSTER with Gaussian kernel on ContraWSD dataset. Kernel-smoothing helps word sense disambiguation.

text may helps selecting the correct form of morphologically complex words.

Kernel-smoothing helps word sense disambiguation In kernel-smoothing, we search examples with similar keys — contextualized hidden states. We hypothesize that the retrieved examples contains useful context information which helps word sense disambiguation. We test this hypothesis with contrastive translation pairs.

A contrastive translation pair contains a source, a reference and one or more contrastive translations. Contrastive translations are constructed by replacing a word in reference with a word which is another translation of an ambiguous word in the source. NMT systems are used to score reference and contrastive translations. If an NMT system assign higher score to reference than all contrastive translations in an example, the NMT system is recognized as making correct prediction on this example.

We use ContraWSD (Gonzales et al., 2017) [7] as the test suite, which contains 7,359 contrastive translation pairs for DE-EN translation. We encode the source sentences from ContraWSD and training data of 5 specific domains by averaged BERT embeddings (Devlin et al., 2018). Then we whiten the sentence embeddings with BERT-whitening proposed by Huang et al. (2021); Li et al. (2020). For each domain, we select 300 examples from ContraWSD that most similar to the in-domain data based on the cosine similarity of sentence embeddings.

We evaluate the translation performance and word sense disambiguation ability of base model and KSTER for MDMT on selected examples for each domain. The results are shown in Figure 9. Experimental results show that KSTER consistently outperforms base model in both translation performance and word sense disambiguation accuracy, which indicates that kernel-smoothing helps word sense disambiguation in machine translation.

6 Conclusion

In this work, we propose kernel-smoothed machine translation with retrieved examples. It improves the generalization ability over existing non-parametric methods, while keeps the advantage of online adaptation.

7 Acknowledgements

---

[7] https://github.com/ZurichNLP/ContraWSD

References

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7747–7763.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly

learning to align and translate. In *ICLR 2015 : International Conference on Learning Representations 2015*.

Ankur Bapna, Naveen Arivazhagan, and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. 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 1538–1548.

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.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58.

Bram Bulte and Arda Tezcan. 2019. Neural fuzzy repair: Integrating fuzzy matches into neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1800–1809.

Qian Cao and Deyi Xiong. 2018. Encoding gated translation memory into neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3042–3047.

Chenhui Chu and Rui Wang. 2018. A survey of domain adaptation for neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1304–1319.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina N. Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186.

M. Amin Farajian, Marco Turchi, Matteo Negri, and Marcello Federico. 2017. Multi-domain neural machine translation through unsupervised adaptation. In *Proceedings of the Second Conference on Machine Translation*, pages 127–137.

Annette Rios Gonzales, Laura Mascarell, and Rico Sennrich. 2017. Improving word sense disambiguation in neural machine translation with sense embeddings. In *Rios, Annette; Mascarell, Laura; Sennrich, Rico (2017). Improving Word Sense Disambiguation in Neural Machine Translation with Sense Embeddings. In: Second Conference on Machine Translation, Kopenhagen, Denmark, 7 September 2017 - 8 September 2017, 11-19.*, pages 11–19.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor OK Li. 2018. Search engine guided neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Demi Guo, Alexander M Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. In *ACL 2021: 59th annual meeting of the Association for Computational Linguistics*, pages 4884–4896.

Junjie Huang, Duyu Tang, Wanjun Zhong, Shuai Lu, Linjun Shou, Ming Gong, Daxin Jiang, and Nan Duan. 2021. Whiteningbert: An easy unsupervised sentence embedding approach. *arXiv preprint arXiv:2104.01767*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*.

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

Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. In *ICLR 2015 : International Conference on Learning Representations 2015*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395.

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

Julia Kreutzer, Jasmijn Bastings, and Stefan Riezler. 2019. Joey nmt: A minimalist nmt toolkit for novices. 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): System Demonstrations*, pages 109–114.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130.

Jianze Liang, Chengqi Zhao, Mingxuan Wang, Xipeng Qiu, and Lei Li. 2020. Finding sparse structures for domain specific neural machine translation. In *AAAI*, pages 13333–13342.

Zehui Lin, Liwei Wu, Mingxuan Wang, and Lei Li. 2021. Learning language specific sub-network for multilingual machine translation. In *ACL 2021: 59th annual meeting of the Association for Computational Linguistics*, pages 293–305.

Michael McCloskey and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In Gordon H. Bower, editor, *Psychology of Learning and Motivation*, volume 24 of *Psychology of Learning and Motivation*, pages 109–165. Academic Press.

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

Minh Quang Pham, Josep Maria Crego, and François Yvon. 2021. Revisiting multi-domain machine translation. *Transactions of the Association for Computational Linguistics*, 9:17–35.

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

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy P. Lillicrap. 2016. One-shot learning with memory-augmented neural networks. *arXiv preprint arXiv:1605.06065*.

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)*, volume 1, pages 1715–1725.

Jorg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pages 2214–2218.

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 *Proceedings of the 31st International Conference on Neural Information Processing Systems*, volume 30, pages 5998–6008.

Jason W. Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. 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 6381–6387.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Joern Wuebker, Patrick Simianer, and John DeNero. 2018. Compact personalized models for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 881–886.

Jitao Xu, Josep Maria Crego, and Jean Senellart. 2020. Boosting neural machine translation with similar translations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1580–1590.

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.

Yaoming Zhu, Jiangtao Feng, Chengqi Zhao, Mingxuan Wang, and Lei Li. 2021. Serial or parallel? plug-able adapter for multilingual machine translation. *arXiv preprint arXiv:2104.08154*.