Interpretable Neural Machine Translation from Translation to Post-Editing

NAIST (currently affiliated with NTT)

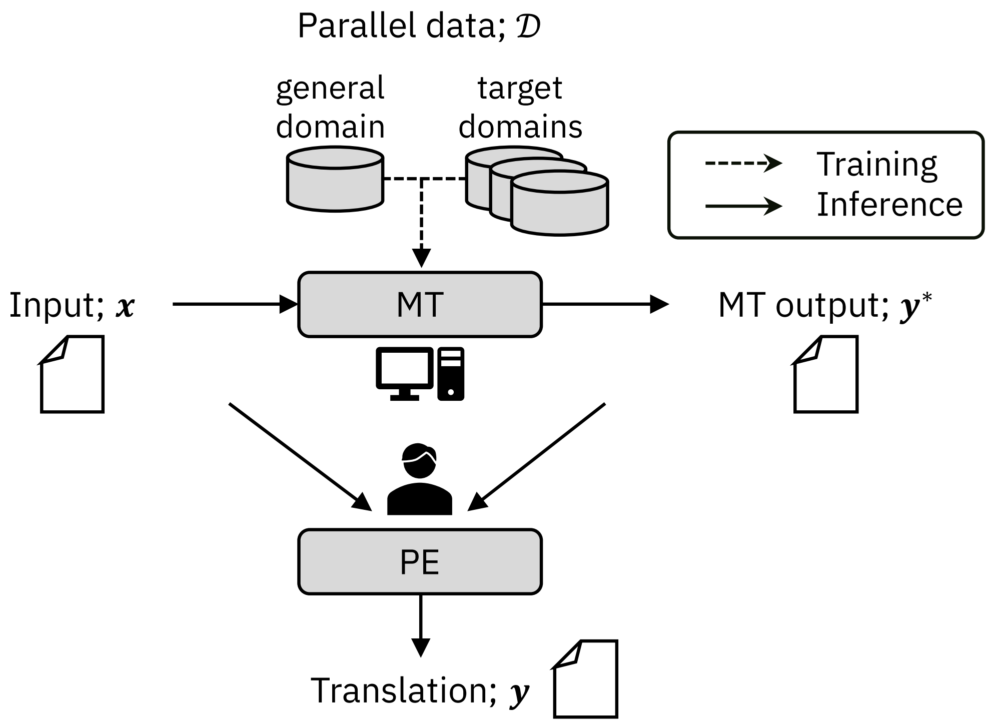
Hiroyuki Deguchi

2025/06/18: AAMT

hiroyuki.deguchi@ntt.com

Machine Translation (MT) and Post-Editing (PE)

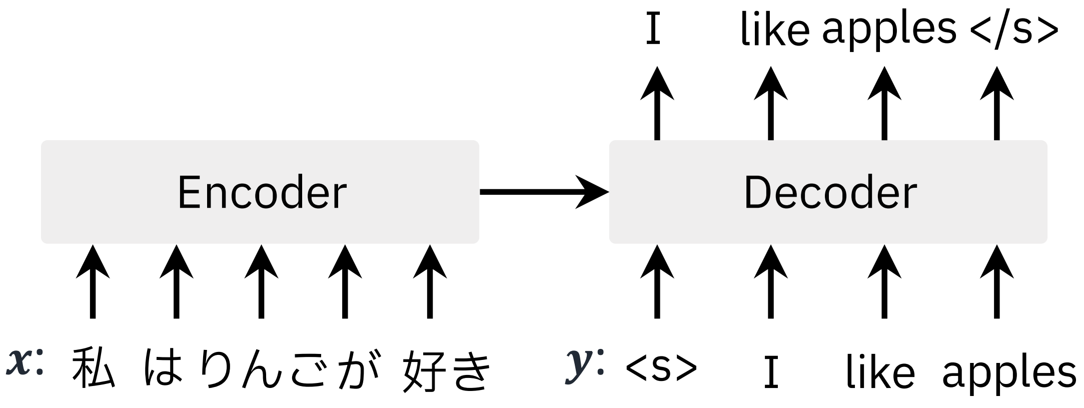
- Typical translation process
  - MT: generates translation drafts
  - PE: refines the translations by human translators
- Various approaches of MT
  - Example-based MT (EBMT): refers to translation examples at run time (Nagao, 1984)
  - Statistical MT (SMT): learns statistical information from parallel data (Brown+, CL1990)
  - Neural MT (NMT): learns converting a sentence to its translation using neural network (Sutskever+, NIPS2014)
    - NMT has been achieved high translation quality



A Framework of a Mechanical Translation Between Japanese and English by Analogy Principle (Nagao, 1984)
A Statistical Approach to Machine Translation (Brown+, CL1990)
Sequence to Sequence Learning with Neural Networks (Sutskever+, NIPS2014)

Overview of Neural Machine Translation (NMT)

- Typical NMT employs the encoder-decoder model
  - Encoder projects the input tokens $\boldsymbol{x} = (x_1, \ldots, x_{|\boldsymbol{x}|})$ into its hidden vectors
  - Decoder generates the target tokens $\boldsymbol{y} = (y_1, \ldots, y_{|\boldsymbol{y}|})$ from left to right, autoregressively
  - Each target token is generated according to its output probabilities: $p(y_t | \boldsymbol{y}_{<t}, \boldsymbol{x})$



Large language models (LLM) encodes the input tokens through the prefix of decoder inputs instead of using the encoder.

- Neural networks used for NMT models
  - Recurrent neural network (Sutskever+, NIPS2014)
  - Convolutional neural network (Gehring+, ICML2017)
  - Transformer (Vaswani+, NIPS2017)

Sequence to Sequence Learning with Neural Networks (Sutskever+, NIPS2014)
Convolutional Sequence to Sequence Learning (Gehring+, ICML2017)
Attention Is All You Need (Vaswani+, NIPS2017)

Challenges in Neural Machine Translation

- NMT generates fluent translations; however:
  - NMT sometimes make errors, especially in the out-of-domains.
    - e.g., train: web corpus, test: medical text
  - Post-editing (PE) is still crucial in fields where mistakes cannot be allowed like medical domain.
- Tasks
  1. Adapt NMT trained from general corpora to various domains efficiently
  2. Assist post-editing to reduce the workload of human post-editors

Subset Retrieval Nearest Neighbor Machine Translation

Accepted at ACL2023 (main)

Background | In-domain and Out-of-domain

- In-domain: Training data and test data are same domain
  - Various methods have improved translation performance

    e.g.,
    - Use syntactic information (Eriguchi+, ACL2017; Deguchi+, RANLP2019)
    - Rerank the translation candidates (Lee+, ACL2021; Fernandes+, NAACL2022)
    - Employ the curriculum learning approaches (Bengio+, NIPS2015)
- Out-of-domain: Training data and test data are different domain
  - Domain adaptation is a challenge in machine translation
    - —2021: The Workshop on Machine Translation (WMT), an international competition for machine translation, held the news translation task.
    - 2022—present: The task was replaced with the mixed-domain translation task.

Learning to Parse and Translate Improves Neural Machine Translation (Eriguchi+, ACL2017)
Dependency-Based Self-Attention for Transformer NMT (Deguchi+, RANLP2018)
Discriminative Reranking for Neural Machine Translation (Lee+, ACL2021)
Quality-aware Decoding for Neural Machine Translation (Fernandes+, NAACL2022)
Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks (Bengio+, NIPS2015)

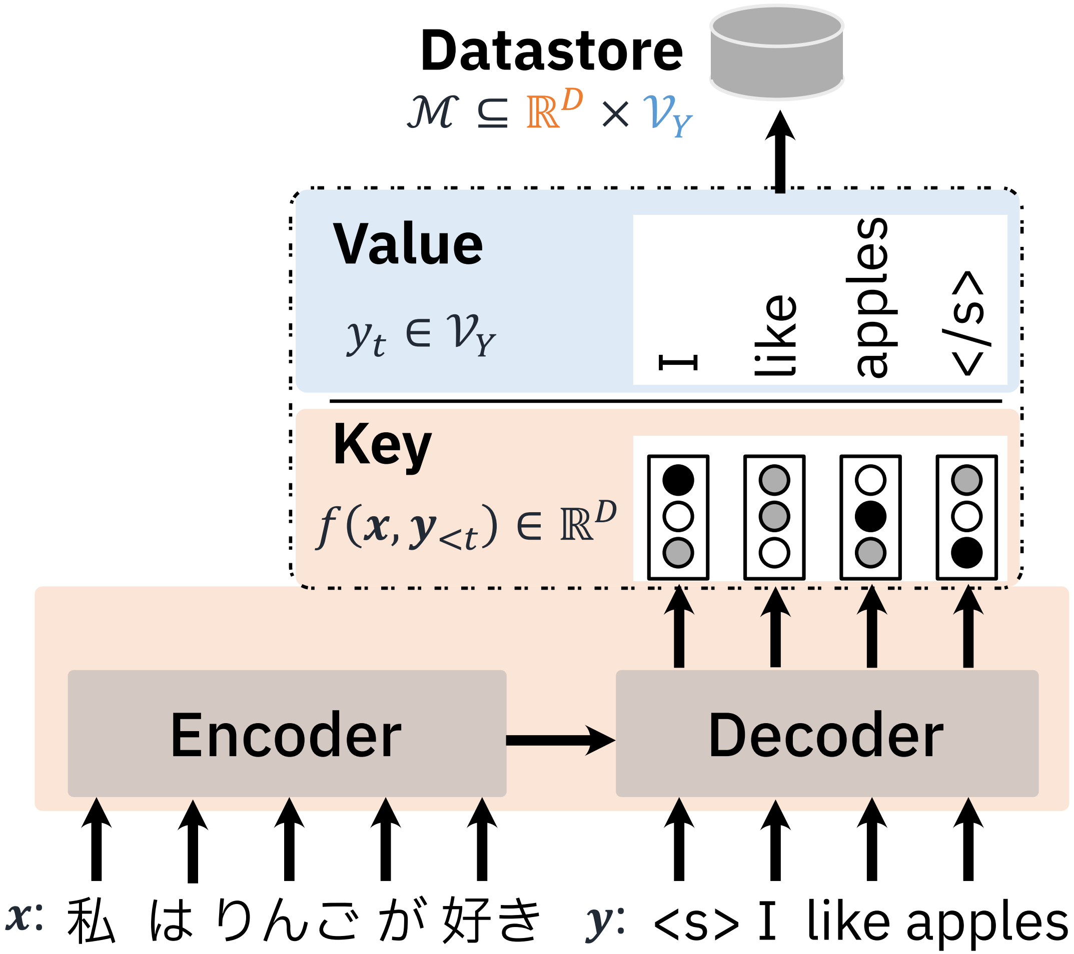
Previous Work | Domain Adaptation

- Train NMT using domain data
  - It needs additional training costs for each domain.
- Retrieve translation examples (Zhang+, NAACL2018; Gu+, AAAI2018; Khandelwal+, ICLR2021)
  - Incorporate the example-based approach into NMT
  - No need to update models for each domain.
    - kNN-MT (Khandelwal+, ICLR2021) achieved SOTA performance in the domain adaptation task.

Guiding Neural Machine Translation with Retrieved Translation Pieces (Zhang+, NAACL2018)
Search Engine Guided Neural Machine Translation (Gu+, AAAI2018)
Nearest Neighbor Machine Translation (Khandelwal+, ICLR2021)

kNN-MT (Khandelwal+, ICLR2021) : Datastore Construction

- Datastore; $\mathcal{M} \subseteq \mathbb{R}^D \times \mathcal{V}_Y$
  - **Key** $\in \mathbb{R}^D$: D-dimensional intermediate representation of a target token
    - ▶ Teacher-forcing a parallel sentence pair $(\boldsymbol{x}, \boldsymbol{y})$ to a trained NMT model
    - ▶ Intermediate representation of the final decoder layer
  - **Value** $\in \mathcal{V}_Y$: Ground truth target token
    - ▶ $\mathcal{V}_Y$: Vocabulary of the target language Y
- Datastore size; $|\mathcal{M}|$
  - The number of all target tokens in a parallel text
  - e.g., WMT'19 De-En: 29.5M sent., 862.6M tok.
    - ▶ 32bit x 1024-D x 1B tokens ≈ 3.7 TiB



Nearest Neighbor Machine Translation (Khandelwal+, ICLR2021)

kNN-MT (Khandelwal+, ICLR2021) : Generation

- First, the model retrieves k nearest neighbor tokens from the datastore in each timestep.
- Then, kNN probability is computed applying softmax to the distance between query and key vectors.

$$p_{k\mathrm{NN}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}) \propto \sum_{i=1}^{k} \mathbb{1}_{y_t=v_i} \exp \frac{-\|\boldsymbol{k}_i - f(\boldsymbol{x}, \boldsymbol{y}_{<t})\|_2^2}{\tau}$$

$k\mathrm{NN} \in \{(\boldsymbol{k}_i \in \mathbb{R}^D, v_i \in \mathcal{V}_Y)\}_{i=1}^{k}$

Applying softmax to the similarity

- Finally, kNN probability and NMT probability are linearly interpolated.

$$P(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}) = \lambda\, p_{k\mathrm{NN}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}) + (1-\lambda)\, p_{\mathrm{MT}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t})$$

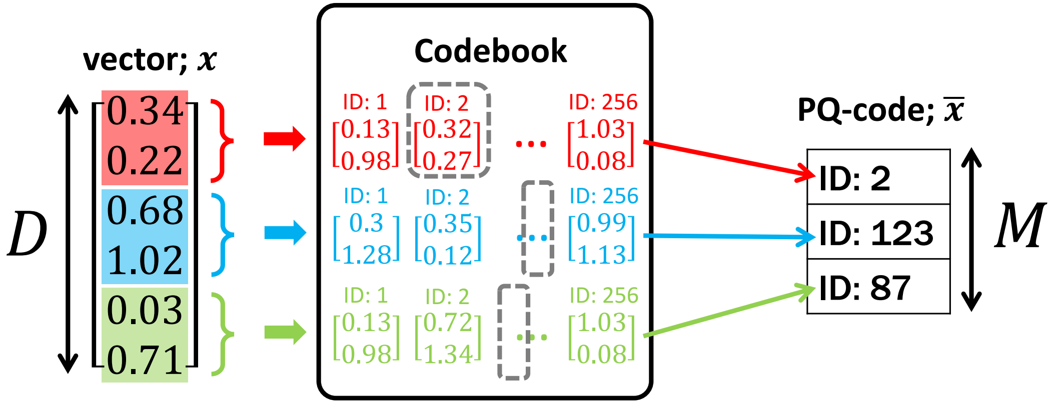
Nearest Neighbor Machine Translation (Khandelwal+, ICLR2021)

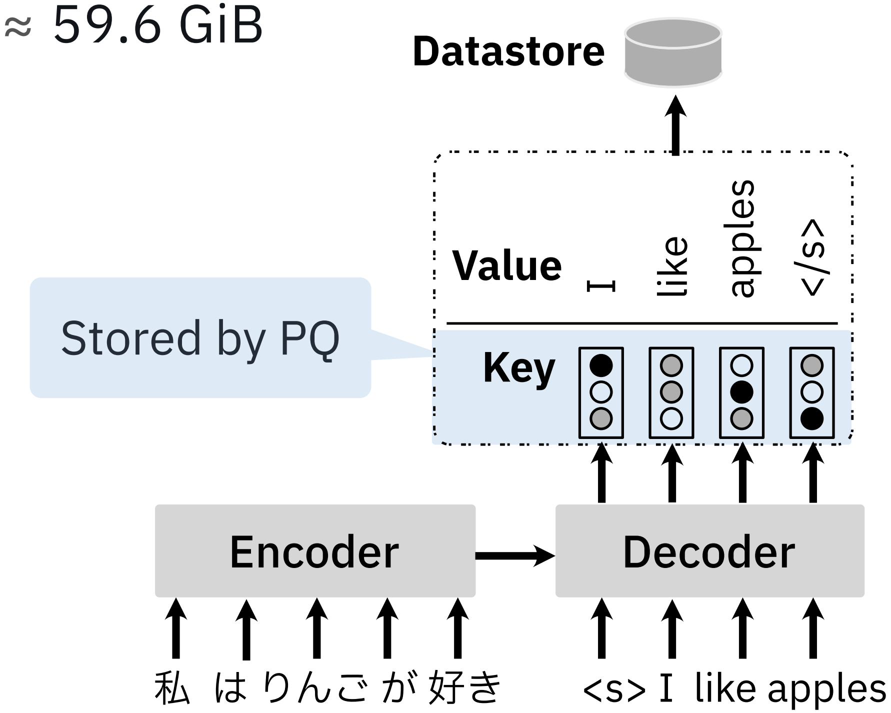
Product Quantization (PQ) (Jégou+, 2011)

Compress key vectors using a vector quantization approach

- Datastore size: 32-bit × 1024-dim × 1e+9 tokens ≈ 3.7 TiB
- PQ: Split a D-dim vector into M sub-vectors and quantize in each sub-space
  - It can achieve lower approximation error than direct D-dim VQ
  - 8-bit (uint8) × 64 (if $M = 64$) × 1e+9 tokens ≈ 59.6 GiB

Product Quantization: Memory efficient





Product Quantization for Nearest Neighbor Search (Jégou+, 2011)

Preliminary Experiments: kNN-MT on Medical Domain (De-En)

Model	↑BLEU	↑tok/s
Base MT	42.1	4392.1
kNN-MT	48.2 (+6.1)	19.8 (×1/222)

☺ improves 6.1 BLEU w/o additional training

☹ 222 times slower than Base MT

Prior work

- Group n-grams and retrieve them at a time (4x faster) (Martins+, EMNLP2022)
- Search for each source token and map to its corresponding target token using word alignment (10x faster) (Meng+, ACL Findings2022)
  - It is still 5% of speed of the base MT.

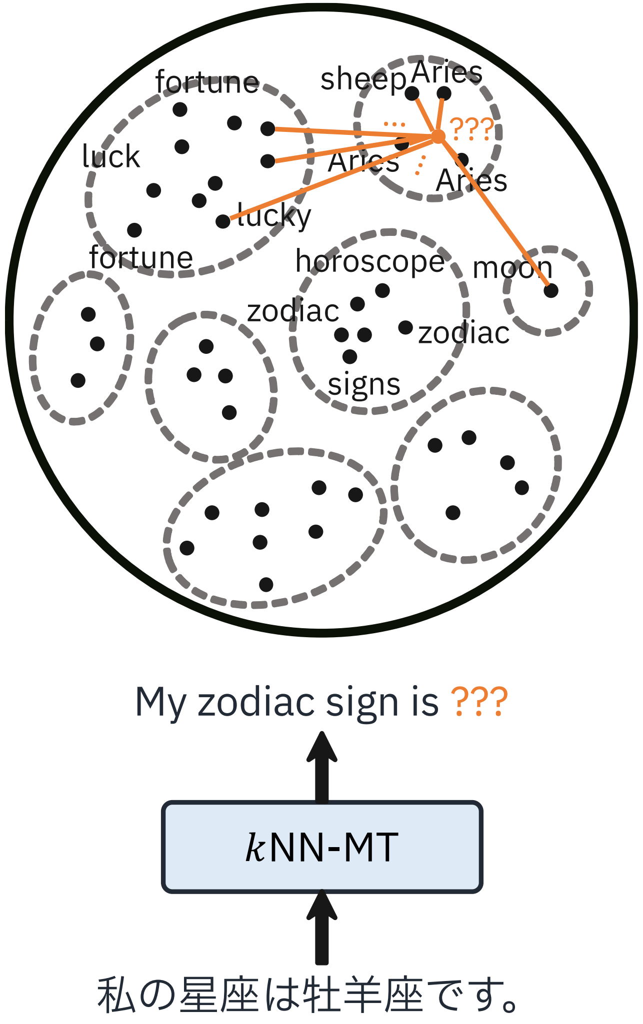
Parameter		Value
Data	Test set	2,000 sentences
	Datastore	Various domain corpora 31M sentence pairs 896M tokens
Model	Base MT	Transformer big trained on WMT'19 De-En
	Interpolation	$\lambda = 0.5$
	Top-k	$k = 16$
Evaluation	Quality	↑sacreBLEU (%)
	Speed	↑Tokens per second (tok/s)

Chunk-based Nearest Neighbor Machine Translation (Martins+, EMNLP2022)
Fast Nearest Neighbor Machine Translation (Meng+, ACL Findings2022)

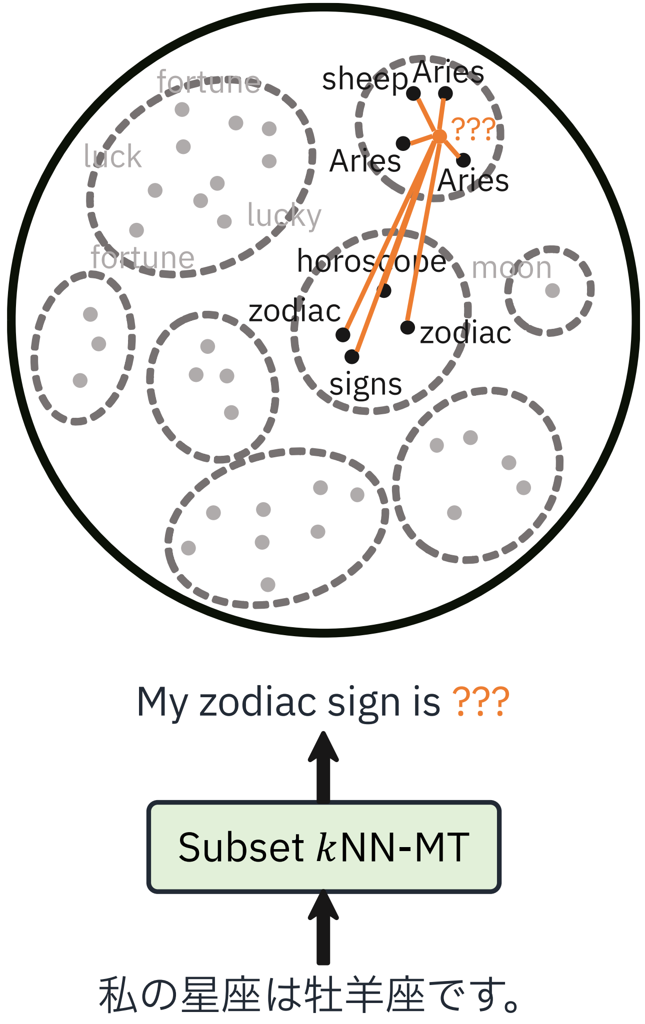
We aim to improve the decoding speed of kNN-MT

- Proposed model: Subset kNN-MT
  - Reduce the kNN search space by searching for the neighbor sentences of the input sentence
  - Use a distance look-up table for efficient distance computation
    - Existing billion-scale kNN search algorithms are designed for only full set search. (Matsui+, ACMMM2018)
    - Subset kNN-MT employs the distance computation method which can used for subset search.

Conventional model



Our model



Reconfigurable Inverted Index (Matsui+, ACMMM2018)

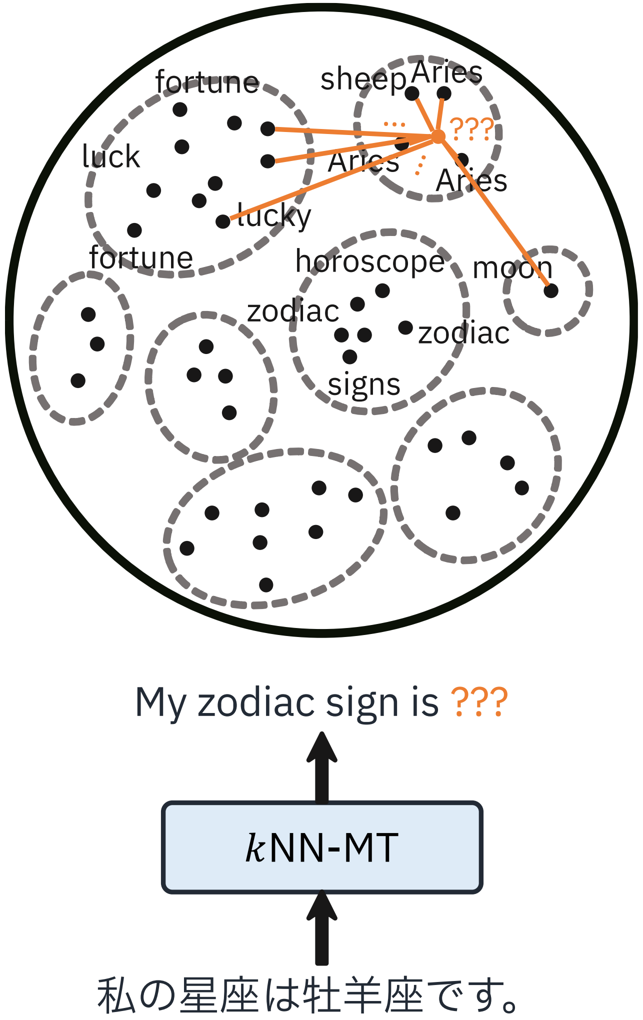
Research Objective

We aim to improve the decoding speed of kNN-MT

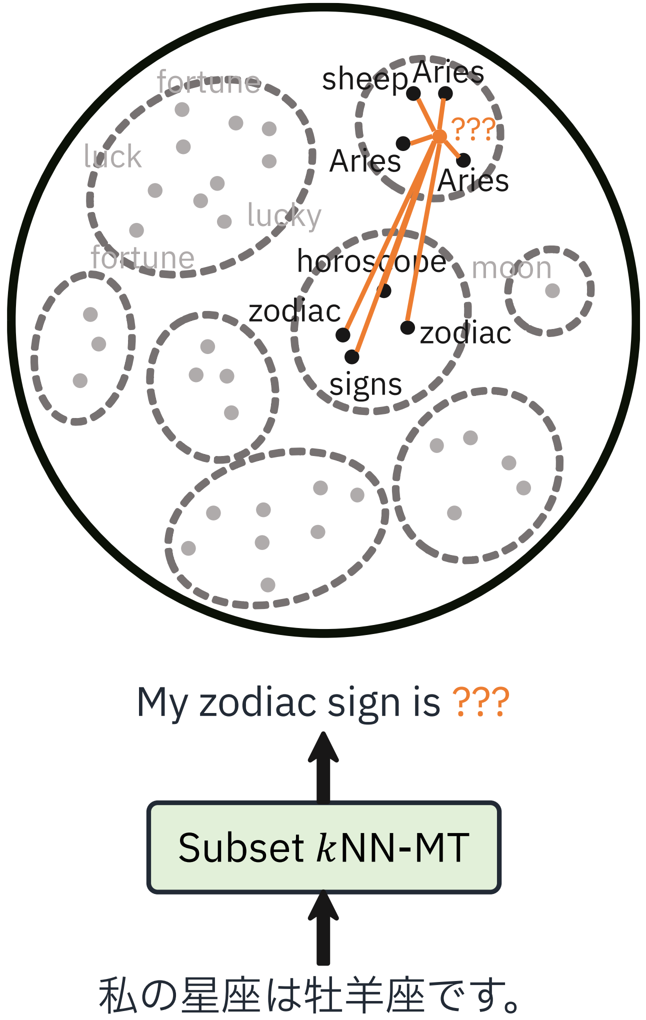
- Proposed model: Subset kNN-MT
  - Reduce the kNN search space by searching for the neighbor sentences of the input sentence
  - Use a distance look-up table for efficient distance computation
    - Existing billion-scale kNN search algorithms are designed for only full set search. (Matsui+, ACMMM2018)
    - Subset kNN-MT employs the distance computation method which can used for subset search.

Reconfigurable Inverted Index (Matsui+, ACMMM2018)

Conventional model

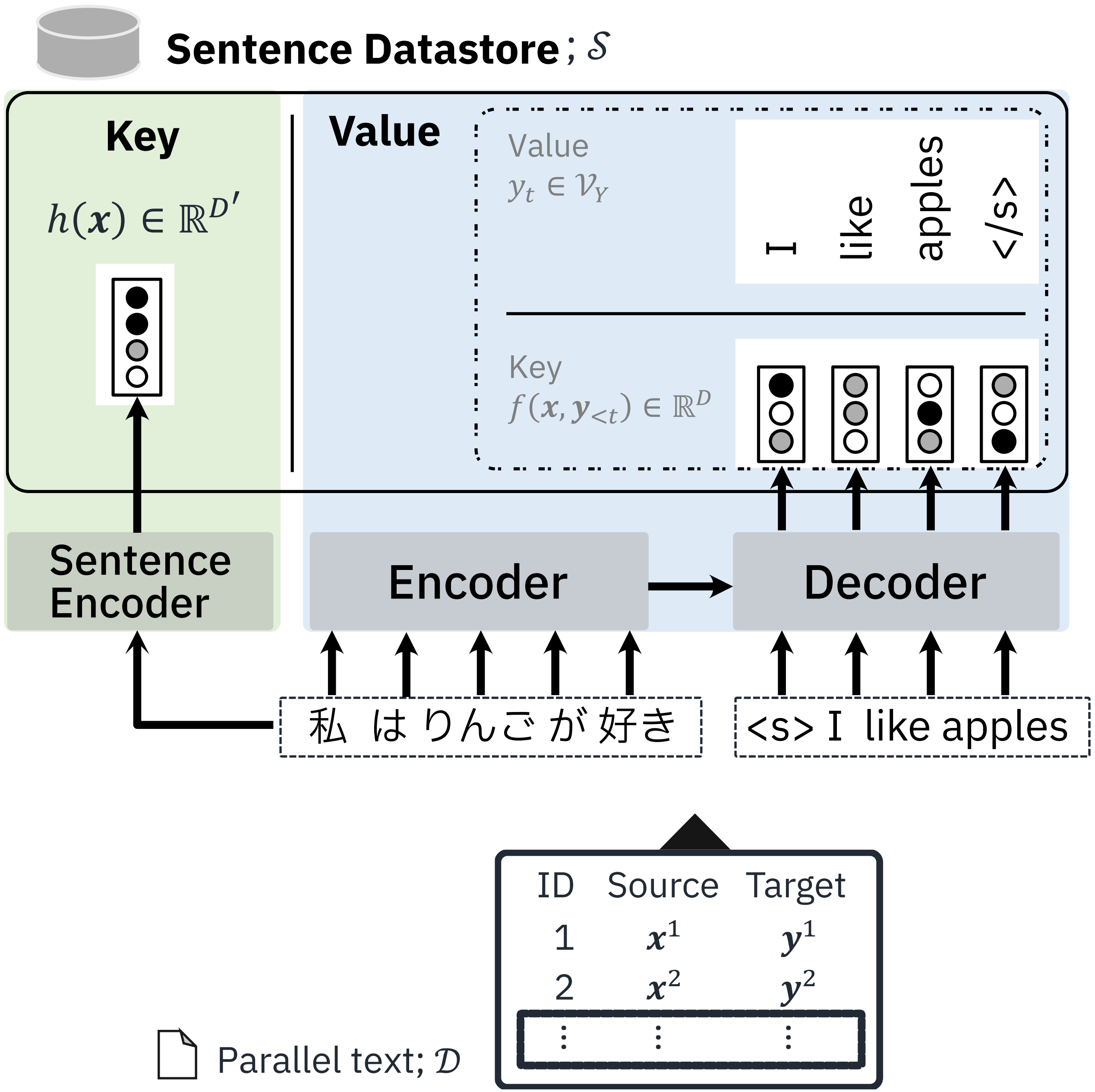


Our model



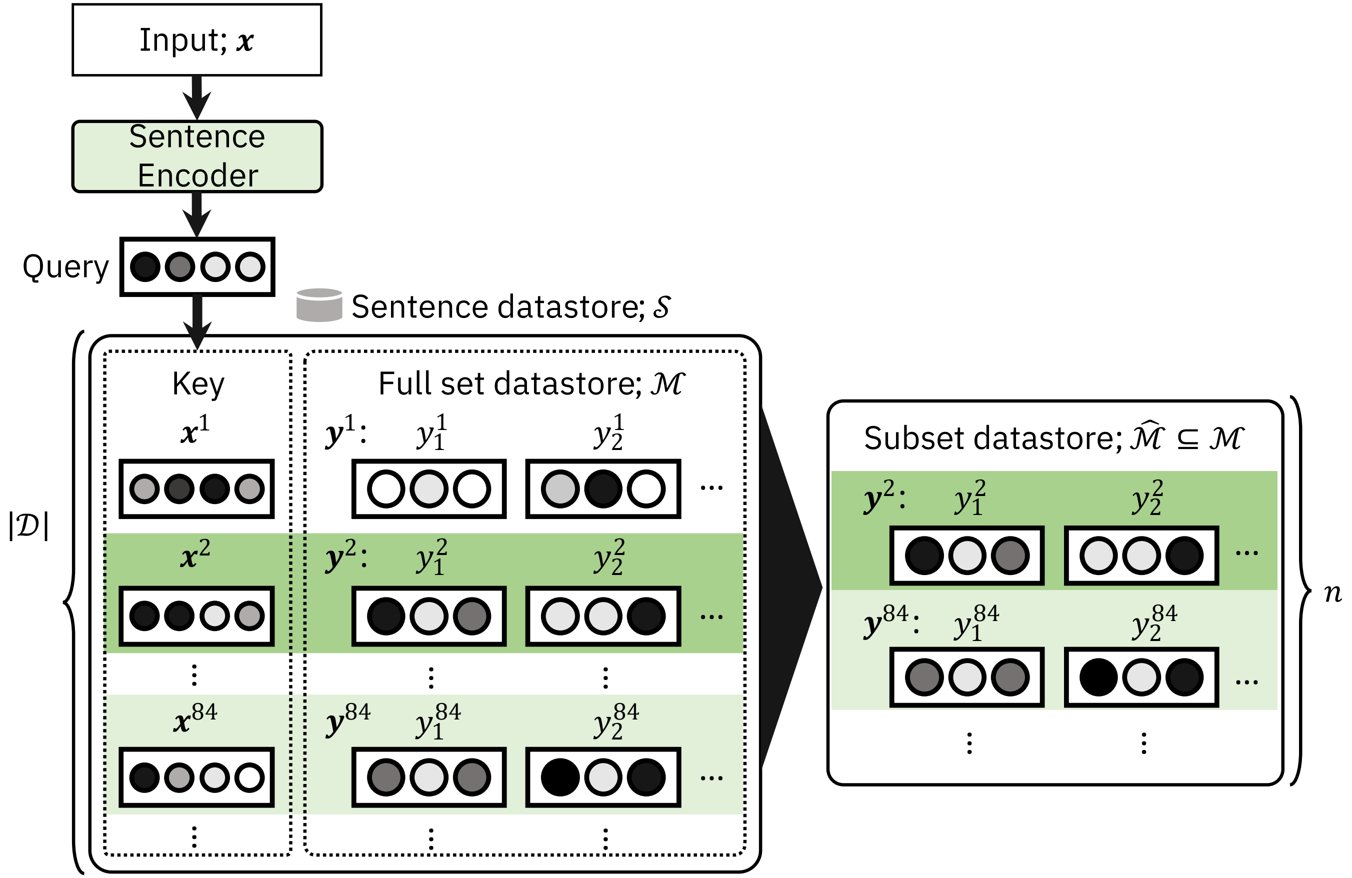
Sentence Datastore Construction

- The sentence datastore $\mathcal{S}$ links source sentence vectors to its corresponding target tokens from the kNN-MT datastore $\mathcal{M}$.
- Key $\in \mathbb{R}^{D'}$: D'-dimensional vector of the source sentence
- Value: target tokens and their key–value pairs from the datastore $\mathcal{M}$.



Generation

1. Retrieve the n-nearest-neighbor sentences of the input sentence $\boldsymbol{x}$ from the sentence datastore $\mathcal{S}$
   - The retrieved target token representations $\widehat{\mathcal{M}}$ are a subset of the datastore $\mathcal{M}$.
2. Use the subset datastore $\widehat{\mathcal{M}}$ at each timestep using kNN-MT

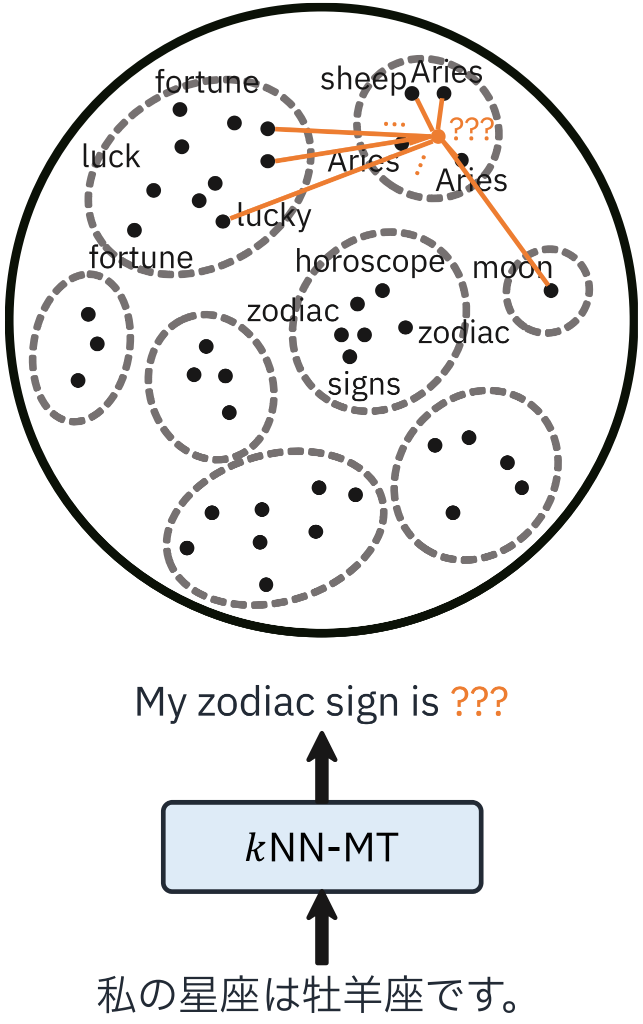


Research Objective

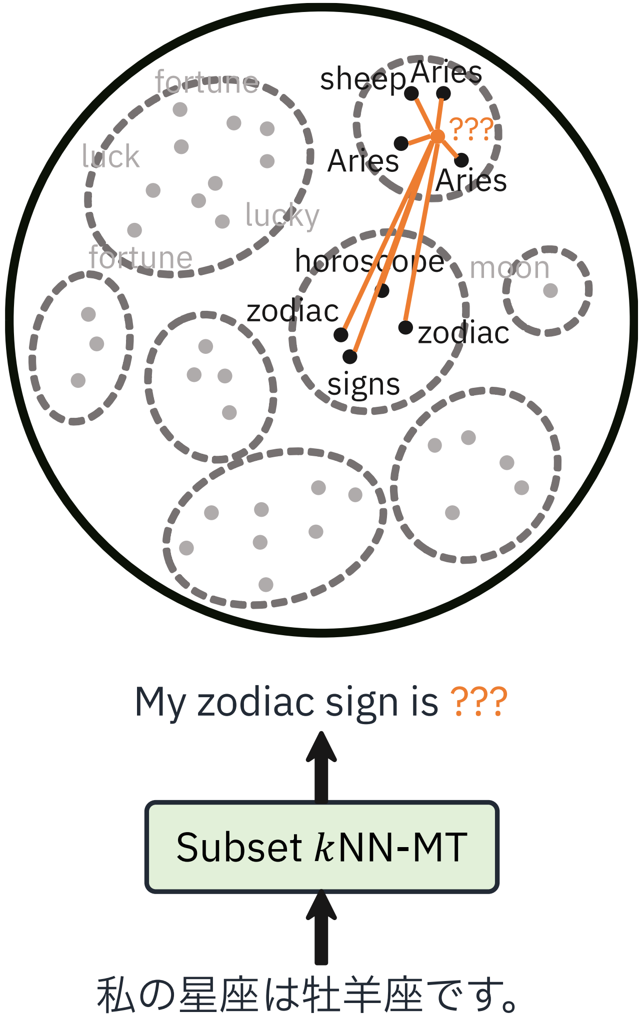
We aim to improve the decoding speed of *k*NN-MT

- Proposed model: Subset *k*NN-MT
  - Reduce the *k*NN search space by searching for the neighbor sentences of the input sentence
  - Use a distance look-up table for efficient distance computation
    - Existing billion-scale *k*NN search algorithms are designed for only full set search. (Matsui+, ACMMM2018)
    - Subset *k*NN-MT employs the distance computation method which can used for subset search.

Conventional model



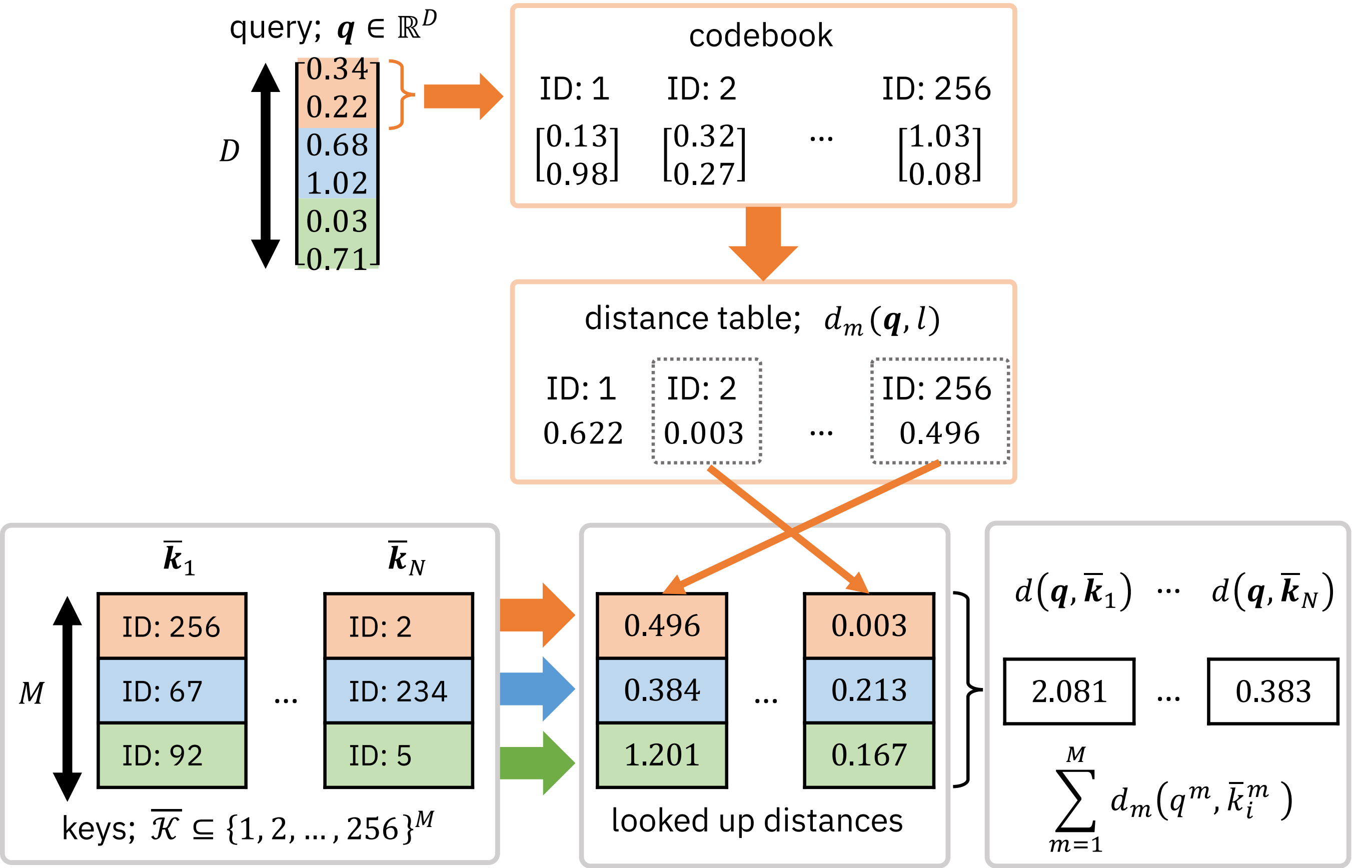
Our model



Reconfigurable Inverted Index (Matsui+, ACMMM2018)

Asymmetric Distance Computation (ADC) (Jégou+, 2011)

1. Compute distances between a query and each code vector in the codebook
2. Look up distances of each quantized key vectors from the pre-computed distance table
3. Calculate the sum of distances per subspace



Product Quantization for Nearest Neighbor Search (Jégou+, 2011)

Experiments: Domain Adaptation De-En

Parameter		Value
Data	General domain	WMT'19 De-En: 29M
	Target domains	• IT: 185K • Koran: 15K • Law: 451K • Medical: 210K • Subtitles: 443K
	Test set	2,000 sentences for each domain
	Datastore	31M sentence pairs 896M tokens
Model	Weight for $p_{k\mathrm{NN}}$	$\lambda = 0.5$
	Top-k	$k = 16$
	neighboring sentences	$n = 256$

Results: Domain Adaptation in De-En

	IT		Koran		Law		Medical		Subtitles	
Model	**BLEU**	**tok/s**	**BLEU**	**tok/s**	**BLEU**	**tok/s**	**BLEU**	**tok/s**	**BLEU**	**tok/s**
Base MT	38.7	4433.2	17.1	5295.0	46.1	4294.0	42.1	4392.1	29.4	6310.5
kNN-MT	41.0	22.3	19.5	19.3	52.6	18.6	48.2	19.8	29.6	30.3
Subset kNN-MT										
h: LaBSE	**41.9**	2362.2	**20.1**	2551.3	**53.6**	2258.0	**49.8**	2328.3	29.9	3058.4
h: AvgEnc	**41.9**	2197.8	19.9	2318.4	53.2	1878.8	49.2	2059.9	**30.0**	3113.0
h: TF-IDF	40.0	2289.0	19.3	2489.5	51.4	2264.3	47.5	2326.6	29.3	2574.4
h: BM25	40.0	1582.4	19.1	2089.5	50.8	1946.3	47.4	1835.6	29.4	1567.7

- Compared with kNN-MT,
  - Speed: Roughly 100 times faster (up to 132.2 times)
  - Quality: Improved about 1 BLEU% on all domains (up to 1.6%)
    - The noise was reduced by limiting the search space to the neighboring sentences.

Summary

Subset kNN-MT improved the decoding speed of kNN-MT

- Proposed methods
  - Online datastore reduction using similar sentence search
  - Efficient distance computation using a distance look-up table
- From the experiments, subset kNN-MT achieved
  - a speed-up of up to 132.2 times
  - an improvement in BLEU of up to 1.6%

  compared with kNN-MT.
- Future work
  - Apply our method to other generation tasks like text summarization.

Detector-Corrector: Edit-Based Automatic Post-Editing Model for Human Post-Editing

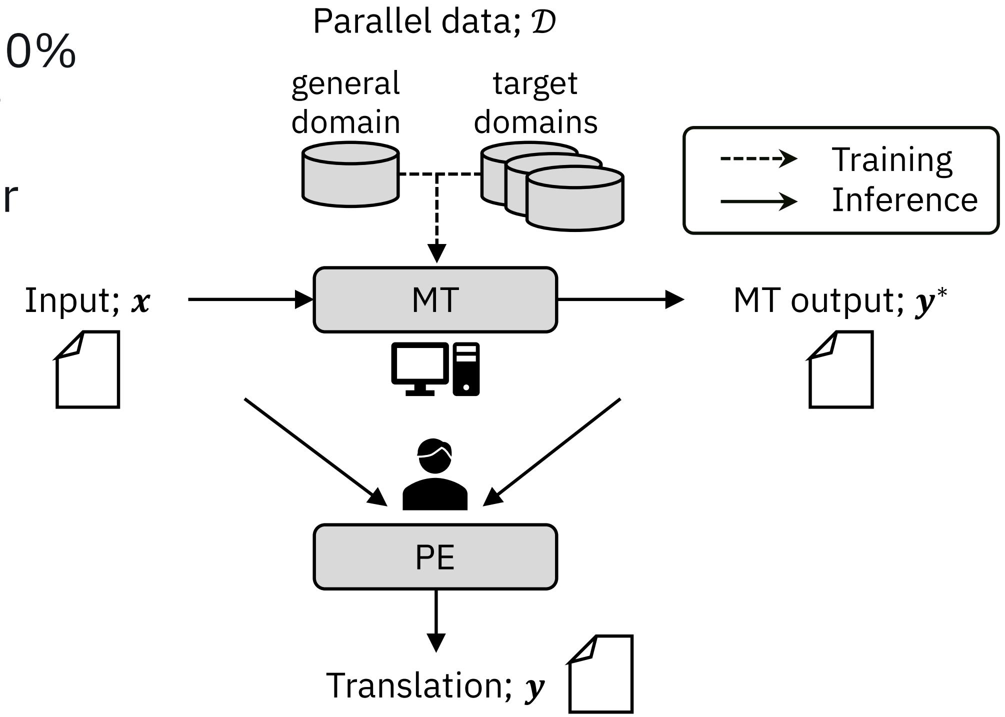
Accepted at EAMT2024

Challenges in Neural Machine Translation

- NMT generates fluent translations; however:
  - NMT sometimes make errors, especially in the out-of-domains.
    - e.g., train: web corpus, test: medical text
  - Post-editing (PE) is still crucial in fields where mistakes cannot be allowed like medical domain.
- Tasks
  1. Adapt NMT trained from general corpora to various domains efficiently
  2. Assist post-editing to reduce the workload of human post-editors

Task 2: Reduce the workload of human post-editors

- Professional translators:
  - ▶ "Even using the latest NMT, PE has saved only about 20—30% of the working time compared to translating from scratch."
  - ▶ They take time to read the source and MT texts and look for mistranslations and omissions.



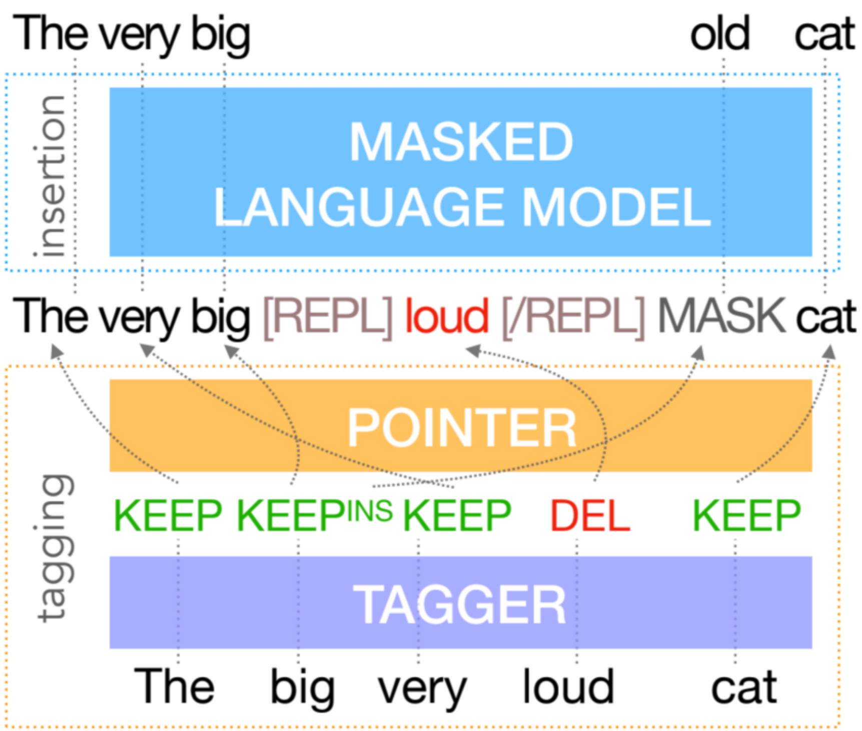
■ How can we reduce the working time of PE?

- Detect and present erroneous spans
- Detect and present omitted spans in a source sentence

etc.

Edit model for monolingual text generation tasks

- The model predicts edit operation tags instead of output words
  - This model improved human interpretability by showing the editing process.
- FELIX is not designed for post-editing
  - It cannot predict untranslated word spans.
  - It cannot insert long spans.

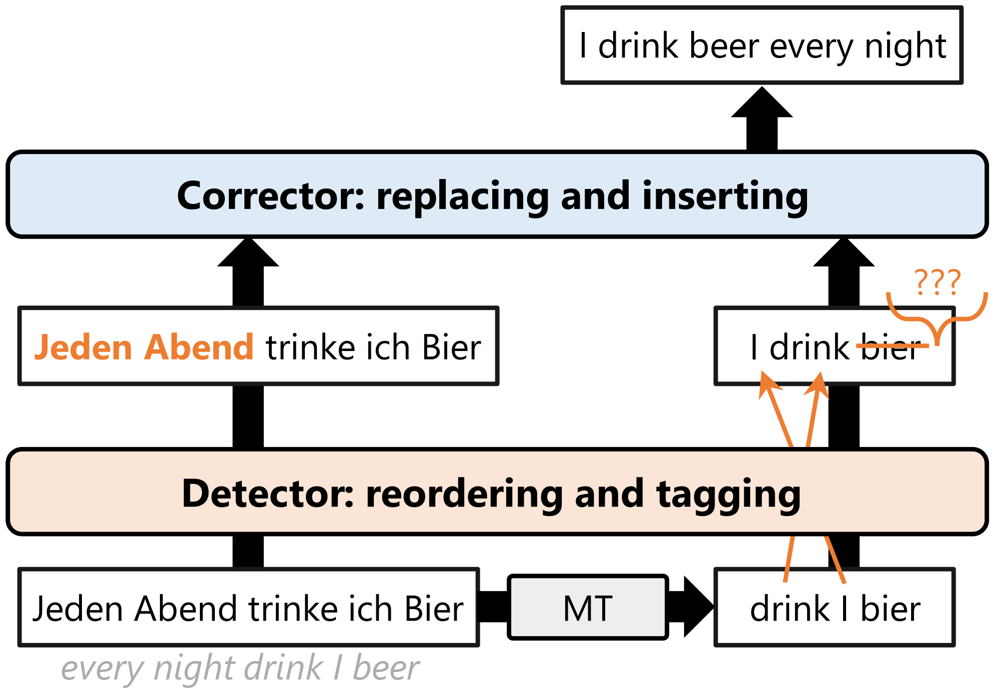


FELIX: Flexible Text Editing Through Tagging and Insertion (Mallinson+, EMNLP Findings 2020)

Improve post-editing efficiency using edit-based approach

Proposed model: Detector-Corrector

- Detector
  - Predict edit operations
    - annotate erroneous spans
    - reorder MT tokens
- Corrector
  - Correct words within erroneous spans



Design of Edit Operations for Post-Editing

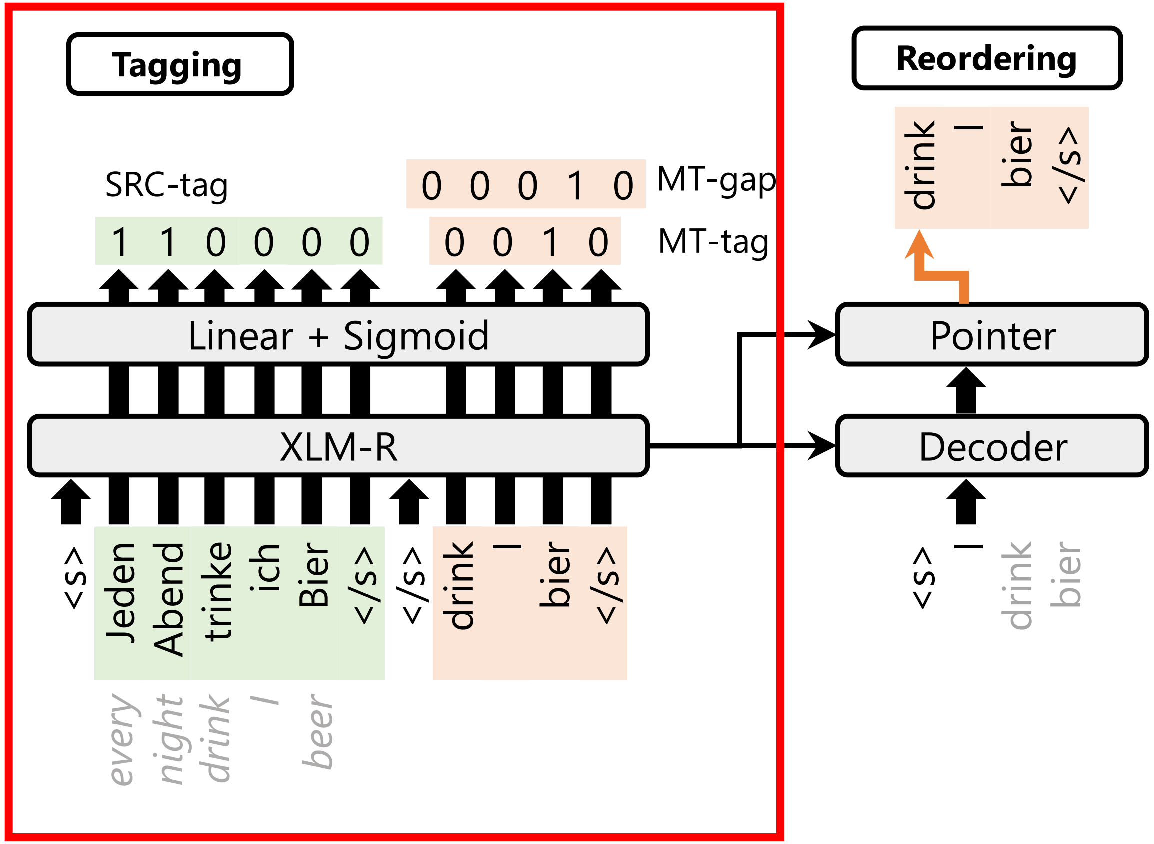
- Translation Edit Rate (TER)
  - Evaluation metric of translation quality
  - Number of edits required to transform an MT sentence to the reference translation
  - How to calculate TER
    1. Shift: Reorder the MT sentence to minimize the edit distance from the reference
    2. Edit: Compute the edit distance between the shifted MT sentence and reference

    ▶ This algorithm can be regarded as representing the edit operations of PE.

Detector: Tagging

Three types of tags are predicted by binary classification

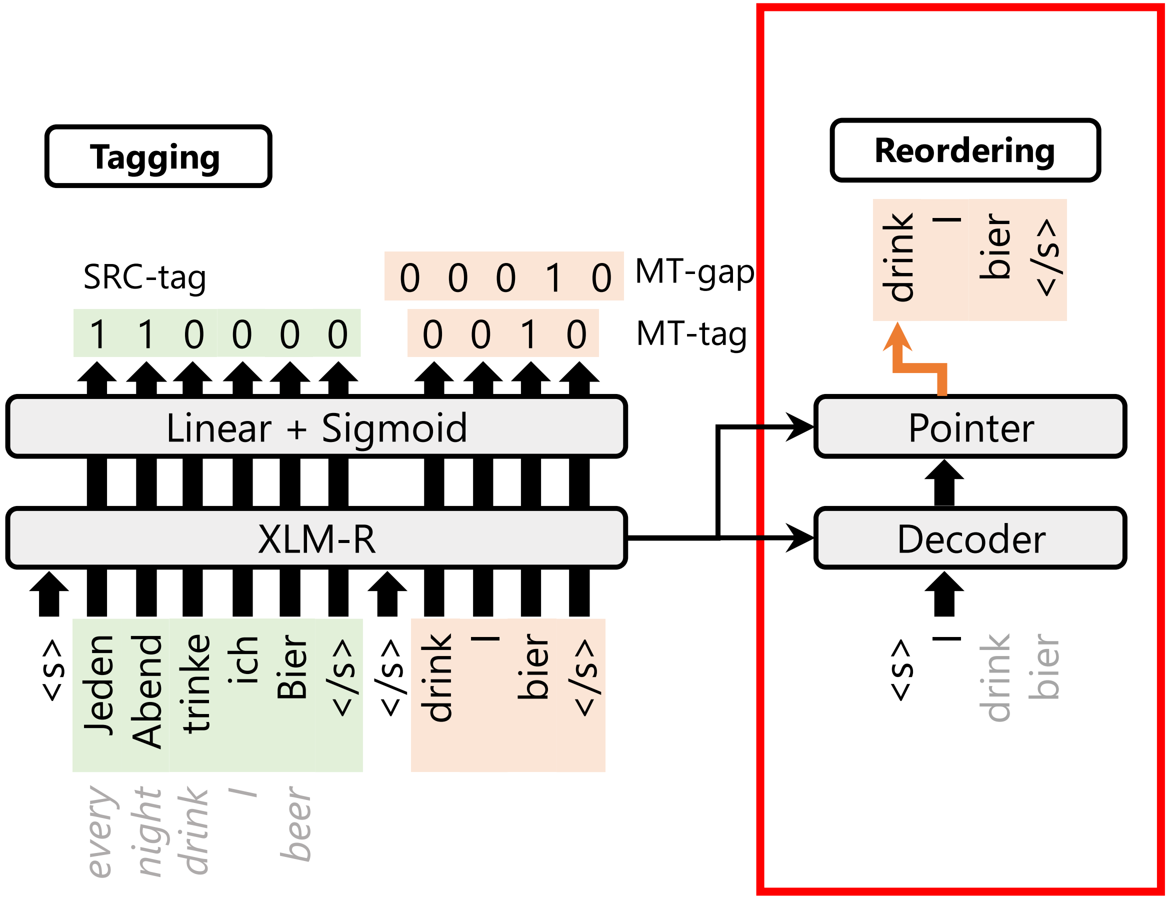
- MT-tag: which tokens are errors
  - The gold tags are created from TER edits: deletion and replacement
- MT-gap: the word boundaries where the words are inserted.
  - The gold tags are created from TER edits: insertion
- SRC-tag: which tokens are untranslated
  - The gold tags are created from word



Word Alignment by Fine-tuning Embeddings on Parallel Corpora (Dou and Neubig, EACL 2021)

Detector: Reordering

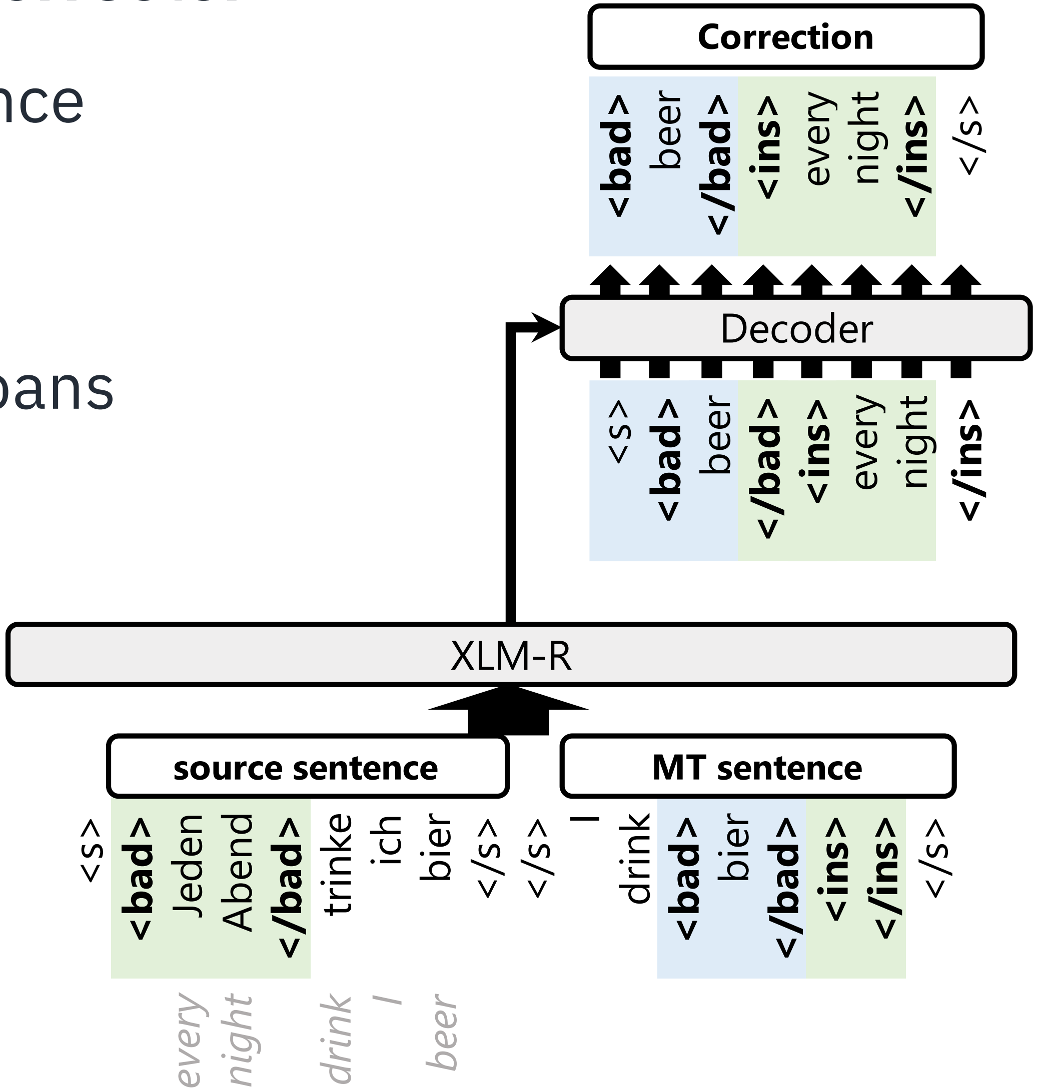
- The pointer network stacked on the decoder selects the next token from the MT sentence
- The gold order is obtained from TER shift alignment



Corrector: Predict Words within Erroneous Spans

Correct the erroneous spans by seq2seq corrector

- Encoder receives the annotated MT sentence
  - bad span: <bad> A B </bad>
  - insertion span: <ins> </ins>
- Decoder generates words within tagged spans



Experimental Setup

- Evaluation data: WMT’20 Automatic Post Editing in En-De and En-Zh
- Training data (2,140,000 sentences)
  - WMT’20 APE: 7K sentences (x 20 up-sampling)
  - Additional data: 2M sentences
    - Created from the training data of the WMT’20 news translation tasks
    - We created triplets from the parallel data by generating MT sentences using the NMT model which is used for creating official training data in the WMT’20 APE tasks
- Baseline models
  - Do nothing (MT): The outputs of the MT model
  - Seq2seq: Black-box Transformer model
  - LevT (Gu+, NeurIPS 2019) : Baseline model for the edit-based model

Levenshtein Transformer (Gu+, NeurIPS 2019)

Experimental Setup: Model

Setting	Seq2Seq	Detector	Corrector
Architecture	XLM-R (large) + 6L Transformer Decoder	XLM-R (large) + 4L Transformer Decoder	XLM-R (large) + 6L Transformer Decoder
Learning rate	1e-4	3e-5	1e-4
Dropout	0.1	0.1	0.1
Optimizer	Adam ($\beta_1 = 0.9, \beta_2 = 0.98$)	Adam ($\beta_1 = 0.9, \beta_2 = 0.98$)	Adam ($\beta_1 = 0.9, \beta_2 = 0.98$)
Batch size	24,000 tokens	6,000 tokens	24,000 tokens
LR scheduler	inverse square root	inverse square root	inverse square root
Warmup steps	4,000	4,000	4,000
Training steps	60,000	40,000	60,000

Main Results: WMT’20 APE task

	En-De			En-Zh		
Model	**↓TER**	**↑BLEU**	**↑COMET**	**↓TER**	**↑BLEU**	**↑COMET**
do nothing (MT)	31.3	50.2	77.1	58.3	24.3	86.3
Seq2Seq	28.4	53.3	77.7	56.7	26.0	**89.4**
LevT (Gu+, NeurIPS2019)	31.9	49.4	75.6	59.3	23.6	86.0
Detector-Corrector	**27.7**	**53.6**	**79.6**	**56.0**	**26.1**	89.2

- Detector-Corrector achieved the best TER scores in both En-De and En-Zh.

Levenshtein Transformer (Gu+, NeurIPS 2019)

Case Study

Source		Georgia Lee , 89 , Australian jazz and blues singer .
MT		89 岁 的 佐治亚州 李 , 澳大利亚 爵士乐 和 布鲁斯 歌手 .
Seq2Seq PE		佐治亚 · 李 (George Lee) , 89 岁 , 澳大利亚 爵士乐 和 布鲁斯 歌手 。
Reference		乔治亚 · 李 (Georgia Lee) , 89 岁 , 澳大利亚 爵士 和 蓝调 歌手 。
PE (1st)	Reordered	的 佐治亚州 李 89 岁 , 澳大利亚 爵士乐 和 布鲁斯 歌手 .
	Detector	的 佐治亚州 李 [INS] 89 岁 , 澳大利亚 爵士乐 和 布鲁斯 歌手 .
	Corrector	∅ · , ∅ 蓝调 。
	Sysout	佐治亚 · 李 , 89 岁 , 澳大利亚 爵士 和 蓝调 歌手 。
PE (2nd)	Detector	佐治亚 · 李 [INS] , 89 岁 , 澳大利亚 爵士 和 蓝调 歌手 。
	Corrector	(George Lee)
	Sysout	佐治亚 · 李 (George Lee) , 89 岁 , 澳大利亚 爵士 和 蓝调 歌手 。

- Our detector-corrector presents the edit process
- The first PE corrected the translation a lot, while the second PE made minor corrections

Summary

Detector-corrector provides the editing process in post-editing

- For human post-editors, detector-corrector explains:
  - mistranslation spans
  - omitted spans

  etc.
- Future work
  - Human evaluation: Which is easier to post-edit, the MT outputs or our model outputs?

Conclusion

Summary

1. Adapt NMT trained from general corpora to various domains efficiently
   - Subset kNN-MT improved translation quality for domain adaptation tasks with faster translation speed compared to the original kNN-MT.
2. Reduce the workload of human post-editors
   - Detector-corrector presented erroneous spans and untranslated spans, which are needed by post-editors, without degradation of translation quality compared to the black-box seq2seq model.

- Performance of error detection of detector--corrector is not enough
  - Especially, the MCC and F1-BAD scores in the target side tagging are about 50%
  - We would like to investigate more effective methods of pseudo-data creation
    - In this dissertation, we found that the data augmentation significantly improves the detection performance.
- The error correction might be improved by other approaches
  - To make the model more robust, we should try to train an end-to-end detector-corrector model, where the detector and corrector are connected as a single model.

Future Directions

- Introduce proposed methods to actual translation scene
  - Evaluate how much the workload of human translators is reduced
- Apply proposed methods to large language models
  - Subset kNN-MT: It is necessary to create a sentence datastore from monolingual data.
  - Detector-Corrector: It is necessary to represent tagging and reordering using generation models.
    - These could be realized by using constrained decoding.

Appendices

Case Study: Medical Domain

Source	Eine gemeinsame Anwendung von Nifedipin und Rifampicin ist daher kontraindiziert.
Reference	Co-administration of nifedipine with rifampicin is therefore contra-indicated.
Base MT	A joint use of nifedipine and rifampicin is therefore contraindicated.
kNN-MT	A joint use of nifedipine and rifampicin is therefore contraindicated.
Subset kNN-MT (s: LaBSE)	Co-administration of nifedipine and rifampicin is therefore contraindicated.

- Subset kNN-MT generated the medical terminology "Co-administration".

Case Study: Retrieved Subset

Input	Eine gemeinsame Anwendung von Nifedipin und Rifampicin ist daher kontraindiziert.
Src-1	Die gemeinsame Anwendung von Ciprofloxacin und Tizanidin ist kontraindiziert.
Src-2	Rifampicin und Nilotinib sollten nicht gleichzeitig angewendet werden.
Src-3	Die gleichzeitige Anwendung von Ribavirin und Didanosin wird nicht empfohlen.
Tgt-1	**Co-administration** of ciprofloxacin and tizanidine is contra-indicated.
Tgt-2	Rifampicin and nilotinib should not be used concomitantly.
Tgt-3	**Co-administration** of ribavirin and didanosine is not recommended.

- *“Co-administration”* is included in the subset.
  - The noise was reduced by limiting the search space to the neighboring sentences.

Experiments: WMT'19 De-En

Setup

- Subset size: $n = 512$
- Batch size: 1 sentence (B1) / 12,000 tokens (B∞)
- Sentence encoder; s
  - LaBSE (Feng+, ACL2022) : Pretrained multilingual sentence encoder model
  - AvgEnc: Average pooled NMT encoder hidden vectors
  - TF-IDF/BM25 weighted vectors

Results

- Speed: More than 100 times faster than kNN-MT
- Quality: Only -0.2 to 0.0 BLEU% degradation

Language-agnostic BERT Sentence Embedding (Feng+, ACL2022)
Chunk-Based Nearest Neighbor Machine Translation (Martins+, EMNLP2022)
Fast Nearest Neighbor Machine Translation (Meng+, Findings of ACL2022)

		↑tok/s	
Model	**↑BLEU**	**B1**	**B∞**
Base MT	39.2	129.14	6375.2
kNN-MT	40.1	2.5	19.6
Chunk kNN-MT (Martins+, 2022)	39.5	22.3	74.6
Fast kNN-MT (Meng+, 2022)	40.3	28.1	286.9
Subset kNN-MT (ours)			
s: LaBSE	40.1	118.4	2191.4
s: AvgEnc	39.9	97.3	1816.8
s: TF-IDF	40.0	113.0	2199.1
s: BM25	40.0	108.4	1903.9

Data Augmentation for Detector

Motivation: Improving the tagging accuracy will lead to improved translation quality because the detector-corrector is trained to correct only erroneous spans detected by the detector.

- Create synthetic data from target sentences of the parallel data

Example:

Source 私は本がとても好きです **Target** I really like books

Data Augmentation for Detector

Motivation: Improving the tagging accuracy will lead to improved translation quality because the detector-corrector is trained to correct only erroneous spans detected by the detector.

- Create synthetic data from target sentences of the parallel data

Example:

Source 私は本がとても好きです

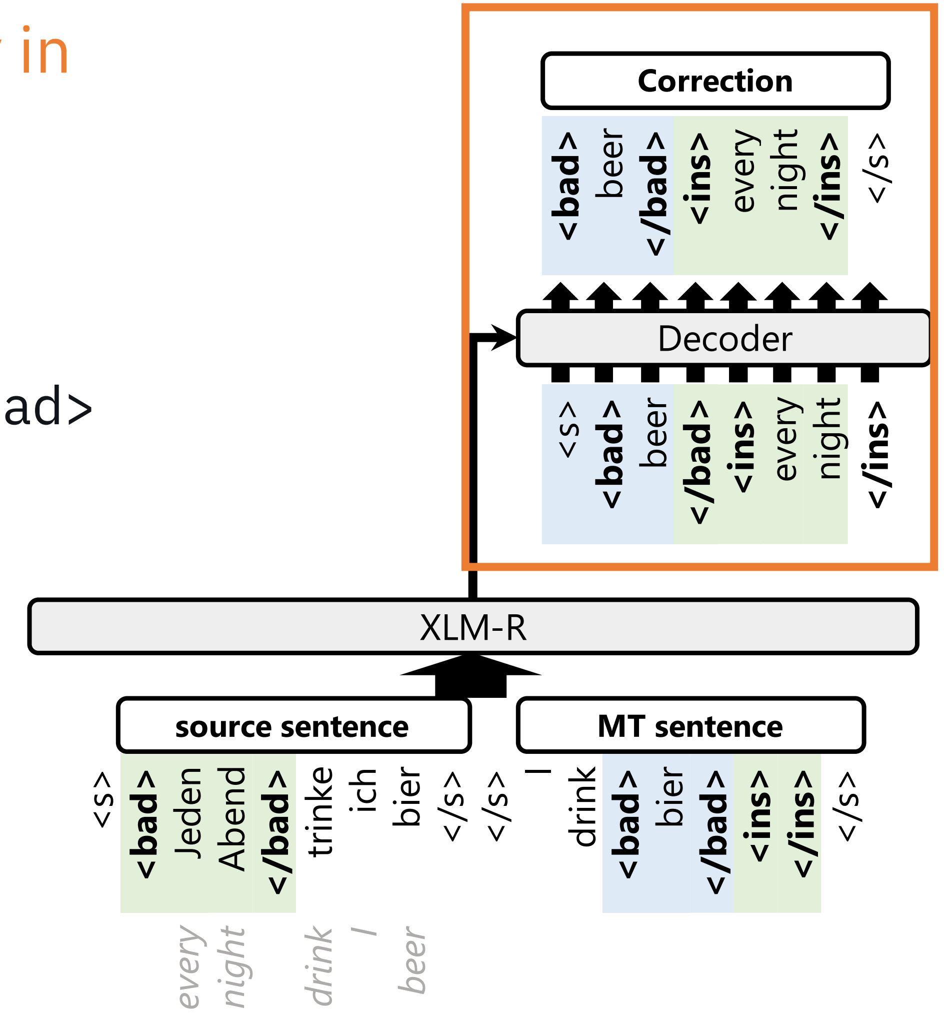
Target I ~~really~~ like books

Pseudo MT I like an book

Data Augmentation for Corrector

Motivation: The performance of the corrector might suffer from the limited coverage of the vocabulary in the training data when compared with a seq2seq model.

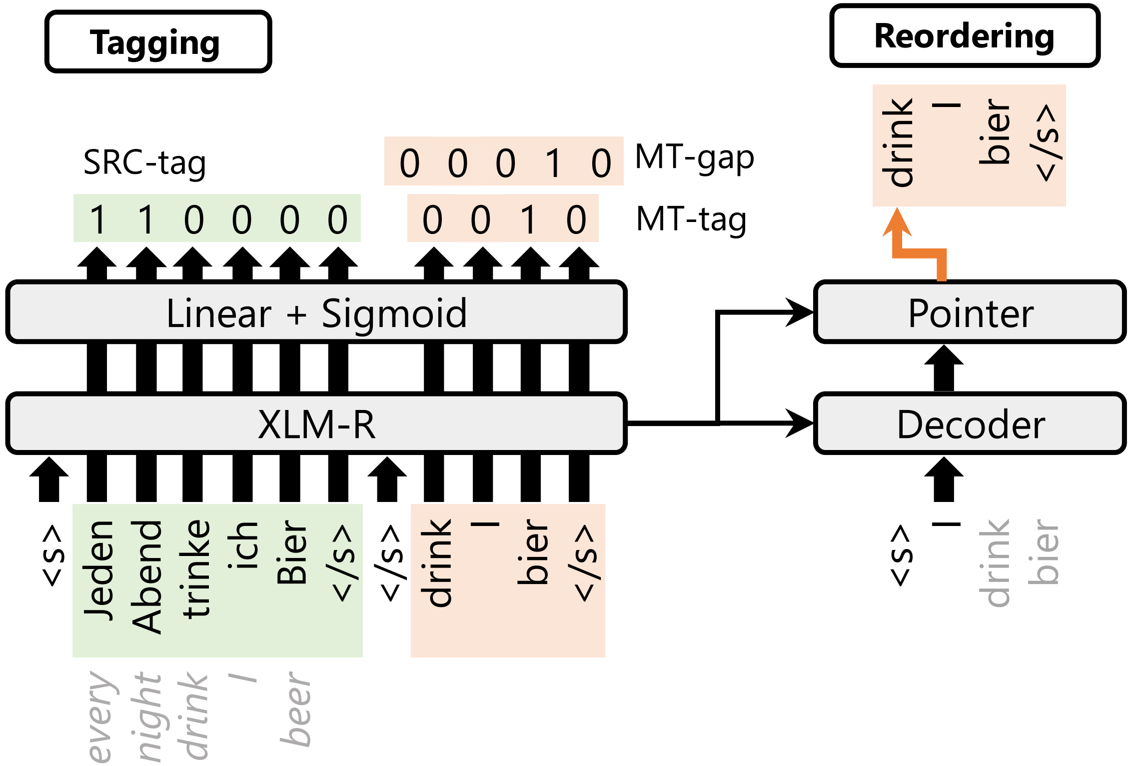
- MT training: SRC + <ins> </ins> → <ins> TGT </ins>
- PE training: SRC + <bad> MT </bad> → <bad> TGT </bad>



Lightweight Iterative Refinement

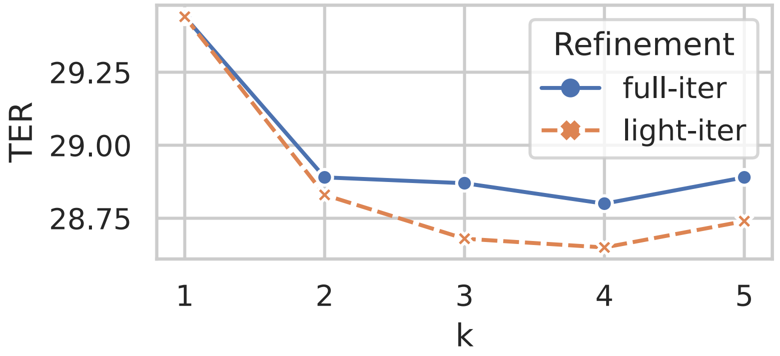
- Iterative refinement
  - It further corrects the corrected sentence, iteratively.
- Lightweight iterative refinement

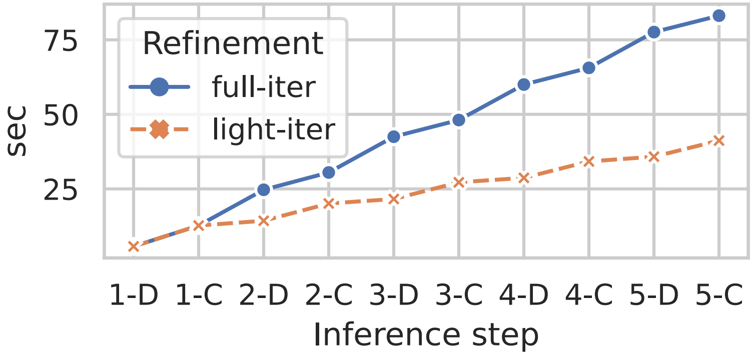
  Motivation: Detector performs tagging non-autoregressively, so a single inference may not generate a consistent correction.
  - full-iter: Tagging + Reordering → Correcting
  - light-iter: Tagging → Correcting
    - Reordering is only performed in the first iteration.



Results of Iterative Refinement

- TER scores of iterative refinement
  - The second inference (k=2) significantly improved TER scores from the first inference (k=1).
- Inference times of full-iter and light-iter
  - Light-iter infers faster than full-iter without performance degradation.





Word-level Quality Estimation Performance of Detector

Model	Target			Source		
	MCC	F1-OK	F1-BAD	MCC	F1-OK	F1-BAD
Detector (w/o synthetic data)	0.453	0.935	0.510	0.781	0.985	0.793
Detector (w/ synthetic data)	0.470	0.938	0.522	0.789	0.985	0.802

- Word-level QE performance of the detector can be improved by using the synthetic data
- The main results and this results show that using a detector with more accurate QE performance improves the correction performance.

Main Results: WMT’20 APE task

Model	En-De			En-Zh		
	↓TER	↑BLEU	↑COMET	↓TER	↑BLEU	↑COMET
do nothing (MT)	31.3	50.2	77.1	58.3	24.3	86.3
Seq2Seq	28.4	53.3	77.7	56.7	26.0	**89.4**
LevT (Gu+, NeurIPS2019)	31.9	49.4	75.6	59.3	23.6	86.0
Detector-Corrector	**27.7**	**53.6**	**79.6**	**56.0**	**26.1**	89.2
- light-iter	28.9	52.1	77.7	56.6	25.5	88.0
-- DAug for corrector	30.2	50.1	77.6	57.0	24.9	88.6
--- DAug for detector	31.2	49.0	77.1	61.2	22.7	86.7

- Detector-Corrector achieved the best TER scores in both En-De and En-Zh.
- Lightweight iterative refinement and two data augmentation approaches (DAug) are effective.

Levenshtein Transformer (Gu+, NeurIPS 2019)

Correction Performance of Oracle Tagged Sentences

Experiment

- Evaluate the correction performance of the corrector when given oracle edit tags
  - Upper bound of the corrector performance
- The oracle edit calculated from TER between the MT sentence and reference

Result

- Given the oracle tags, the correction performance improved by -17.89% for TER and by +26.01% for BLEU.
- The corrector has been successfully trained.
- A further improvement in post-editing performance can be achieved by improving the detector model.

Model	↓TER	↑BLEU
Baseline (MT)	31.33	50.21
Detector-Corrector	31.75	48.68
+ Oracle tagging	13.86 (-17.89)	74.49 (+26.01)