# Understanding Knowledge Distillation in Neural Sequence Generation

Jiatao Gu

January 22, 2020

facebook
Artificial Intelligence Research

**Jiatao Gu**

Ph.D.

New York, US

Facebook AI Research

## About Me

I am currently a research scientist at the Facebook AI Research in New York City. My general research interests lie in applying deep learning approaches to natual language processing (NLP) problems. In particular, I am interested in building an efficient, effective and reliable neural machine translation (NMT) system for human languages.

I obtained my Ph.D. degree at the department of Electrical and Electronic Engineering, University of Hong Kong in 2018 and I was supervised by Prof. Victor O.K. Li. I spent a wonderful time visiting the CILVR Lab, New York University working with Prof. Kyunghyun Cho. Before that, I obtained my Bachelor's degree at the Electronic Engineering Department, Tsinghua University in 2014 with the guidance of Prof. Ji Wu.

# My Research Focus @ FAIR

**Towards Better Understanding & Interoperability of NMT**

**Low-Resource and Multilingual NMT**

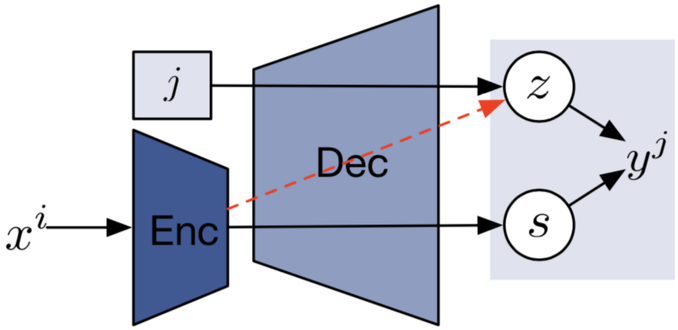

**Flexible Representation and Efficient Decoding for NMT**

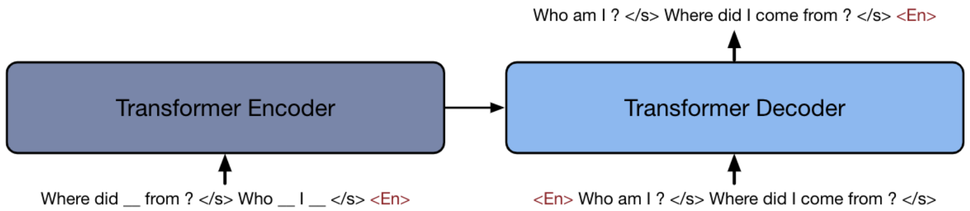
**End-to-end Simultaneous Speech Translation**

# My Research Focus @ FAIR

**Low-Resource and Multilingual Neural Machine Translation**

- Improved Zero-shot NMT (Gu et al. 2019, ACL 2019)
- Multilingual NMT with Byte-level subwords (Wang et al. 2019, AAAI 2020)
- The Source-Target Domain Mismatch Problem in NMT (Shen et al. 2019, submitted to TACL 2020)
- Multilingual Denoising Pre-training for NMT (arxiv today)



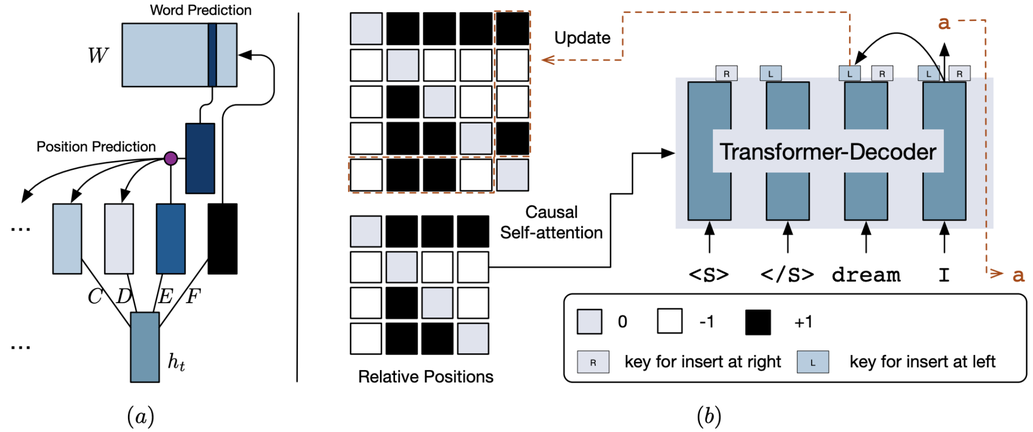



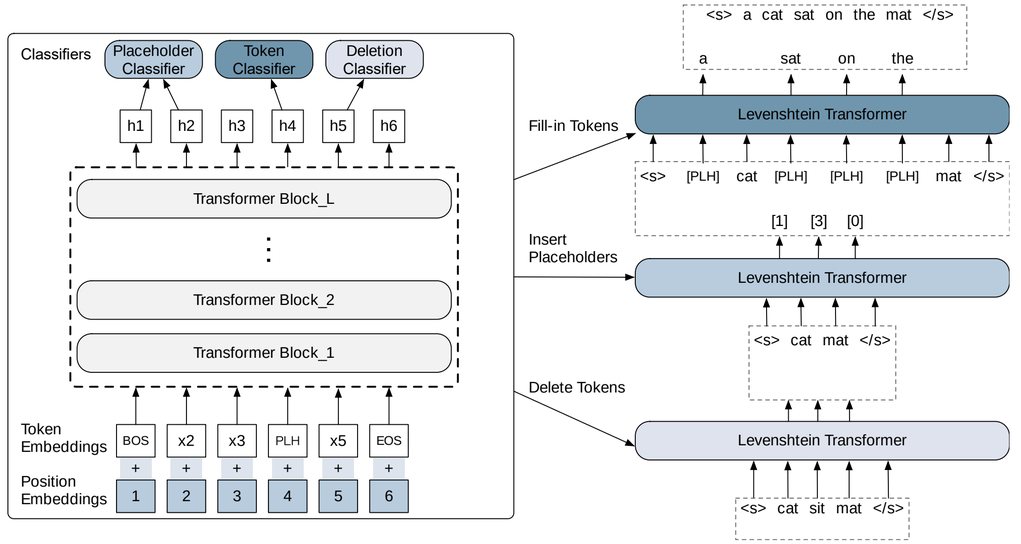

| | | | |
|---|---|---|---|
| Original | | 質問して__証明と証拠を求めましょう | Ask__questions,__demand__proof,__demand__evidence. |
| Byte | | E8 B3 AA E5 95 8F E3 81 97 E3 81 A6 E2 96 81 E8 A8 BC E6 98 8E E3 81 A8 E8 A8 BC E6 8B A0 E3 82 92 E6 B1 82 E3 82 81 E3 81 BE E3 81 97 E3 82 87 E3 81 86 | 41 73 6B E2 96 81 71 75 65 73 74 69 6F 6E 73 2C E2 96 81 64 65 6D 61 6E 64 E2 96 81 70 72 6F 6F 66 2C E2 96 81 64 65 6D 61 6E 64 E2 96 81 65 76 69 64 65 6E 63 65 2E |
| BBPE | 1K | E8 B3 AA E595 8F しE381 A6 __E8 A8 BC 明 E381 A8 E8 A8 BC E6 8B A0 をE6 B1 82 めE381 BE しょう | As k __quest ions , __dem and __pro of , __dem and __ev idence . |
| | 2K | E8 B3 AA 問 しE381 A6 __E8 A8BC 明 E381 A8 E8 A8BC E68B A0 をE6 B1 82 めE381 BE しょう | A s k __qu est ion s , __d em and __pro o f , __d em and __e v id ence . |
| | 4K | E8 B3 AA 問 しE381 A6 __E8 A8BC 明E381 A8 E8 A8BC 拠 をE6 B1 82 めE381 BE しょう | A s k __quest ions , __d em and __pro of , __d em and __ev id ence . |
| | 8K | E8 B3 AA問 しE381 A6 __E8 A8BC 明E381 A8 E8 A8BC 拠 をE6 B1 82めE381 BE しょう | As k __questions , __demand __pro of , __demand __evidence . |
| | 16K | E8 B3 AA問 しE381 A6 __E8 A8BC 明E381 A8 E8 A8BC 拠 をE6 B1 82めE381 BE しょう | As k __questions , __demand __proof , __demand __evidence . |
| | 32K | E8 B3 AA問しE381 A6 __E8 A8BC 明E381 A8 E8 A8BC 拠 をE6 B1 82 めE381 BE しょう | As k __questions , __demand __proof , __demand __evidence . |
| CHAR | | 質 問 し て __ 証 明 と 証 拠 を 求 め ま し ょ う | A s k __ q u e s t i o n s , __ d e m a n d __ p r o o f , __ d e m a n d __ e v i d e n c e . |
| BPE | 16K | 質問 して __ 証明 と 証拠 を求 め ましょう | As k __questions , __demand __pro of , __demand __evidence . |
| | 32K | 質問 して __証明 と 証拠 を求め ましょう | As k __questions , __demand __proof , __demand __evidence . |

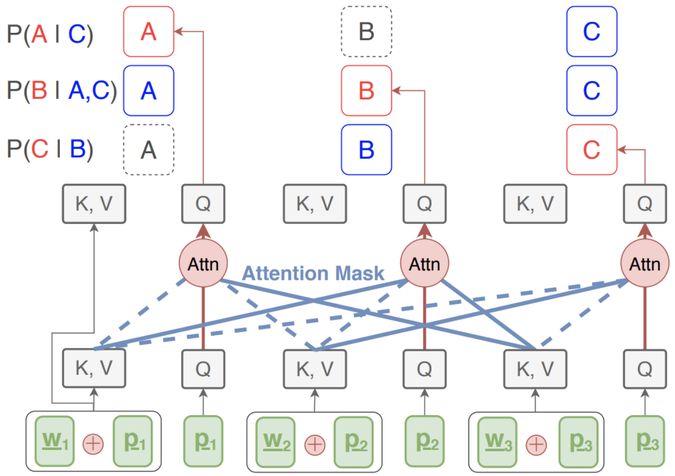
# My Research Focus @ FAIR

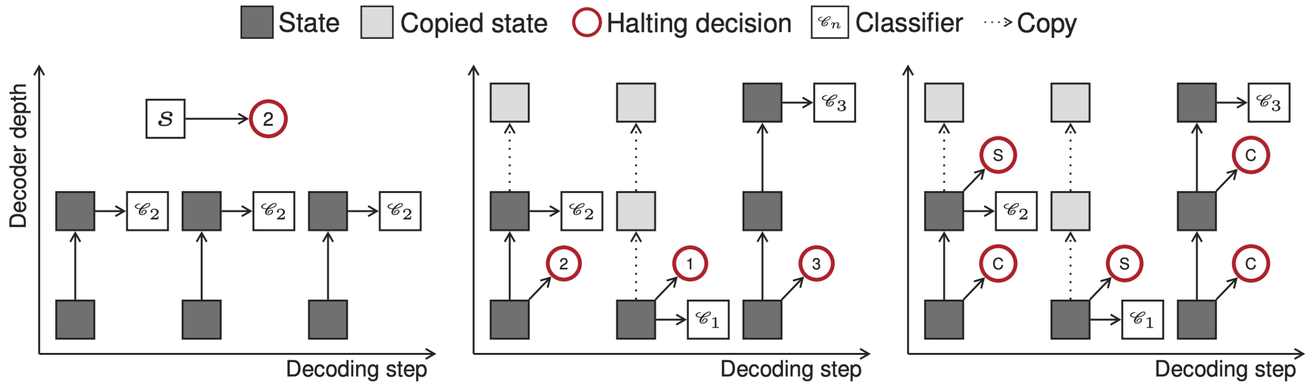
**Flexible Representation and Efficient Decoding for NMT**

- Insertion-based Generation (Gu et al. 2019, TACL 2019)
- Levenshtein Transformer (Gu et al. 2019, NeurIPS 2019)
- Generation with Adaptive Computational Time (Elbayad et al. 2019, ICLR 2020)
- Parallel Machine Translation with Disentangled Context Transformer (Kasai et al. 2020, to ICML2020)



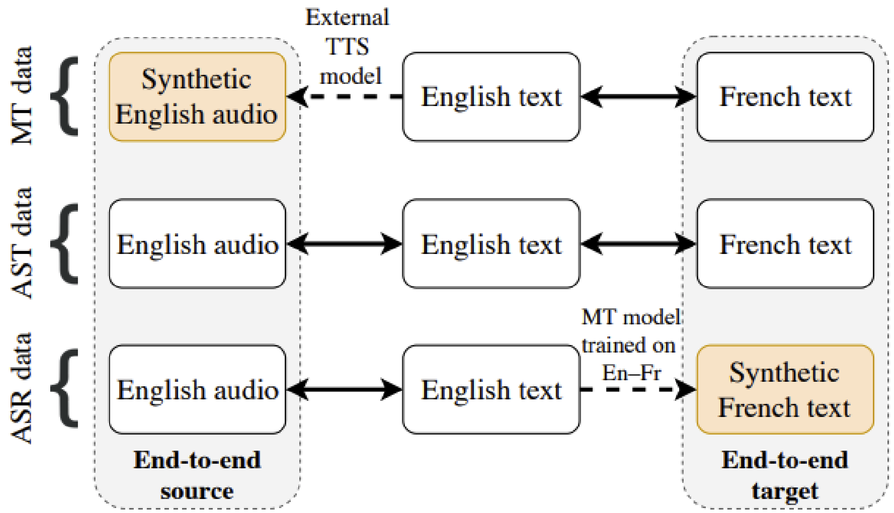



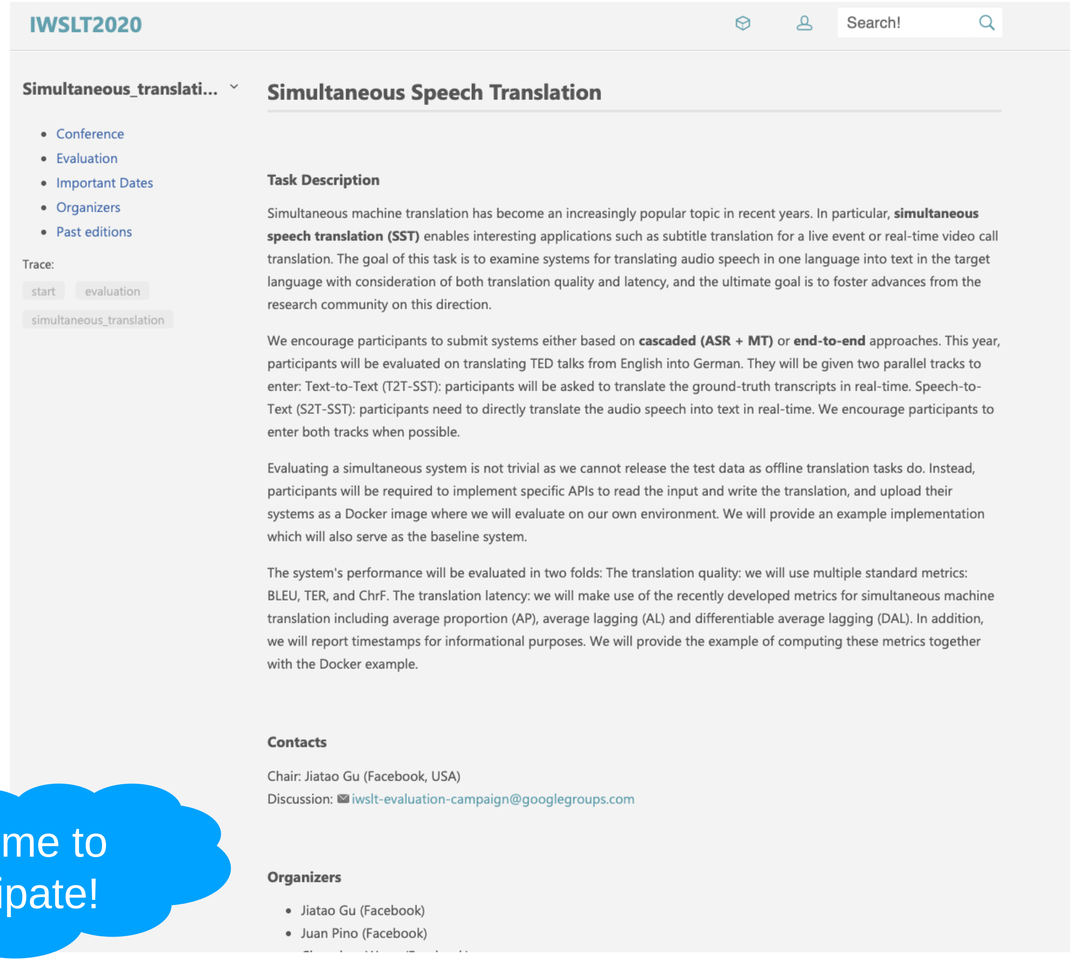

facebook
Artificial Intelligence Research

# My Research Focus @ FAIR

**End-to-end Simultaneous Speech Translation**

- End-to-End AST with Indirect Training Data (Pino et al. 2019, IWSLT 2019)
- Simultaneous Speech Translation (Ma et al. 2019, submitted ICLR2020)
- Multilingual Speech Translation (submitted to LREC2020)
- …




Welcome to participate!

# My Research Focus @ FAIR

**Towards Better Understanding & Interoperability of NMT**

**Low-Resource and Multilingual NMT**


**Flexible Representation and Efficient Decoding for NMT**

**End-to-end Simultaneous Speech Translation**

# Sequence-Level Knowledge Distillation

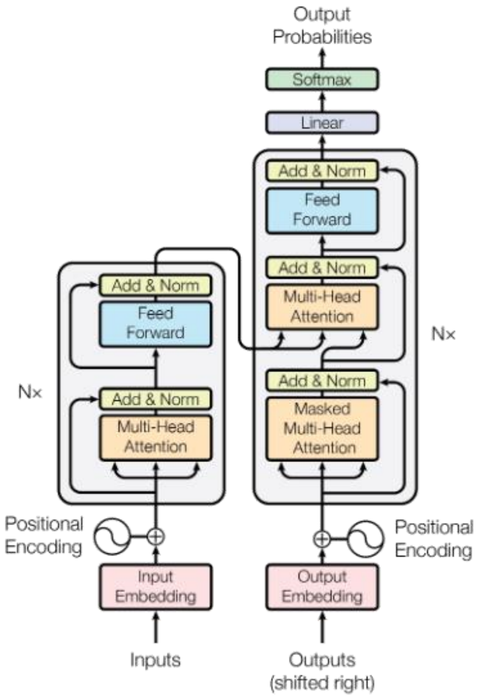
facebook
Artificial Intelligence Research

# Neural Machine Translation

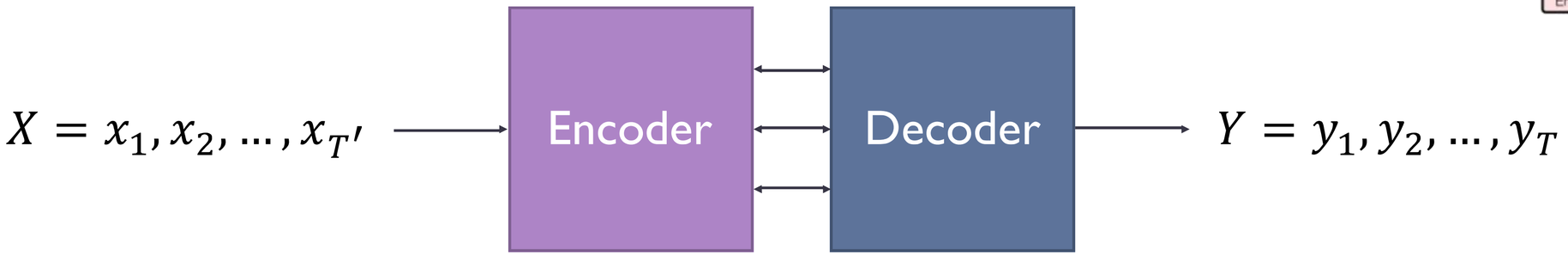
Model:

- Encoder/Decoder → RNNs (before 2018) / Transformers (now)
- We model the conditional probability in ***autoregressive*** factorization.

Autoregressive ≠ Recurrent/Recursive




Attention Mechanism

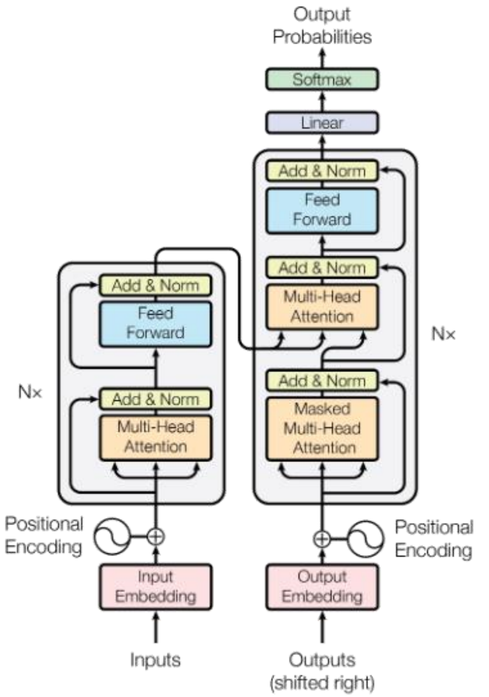



$$P(Y|X) = \prod_{t=1}^{T} P(y_t|y_{1\ldots t-1}, x_{1\ldots T'}; \theta)$$

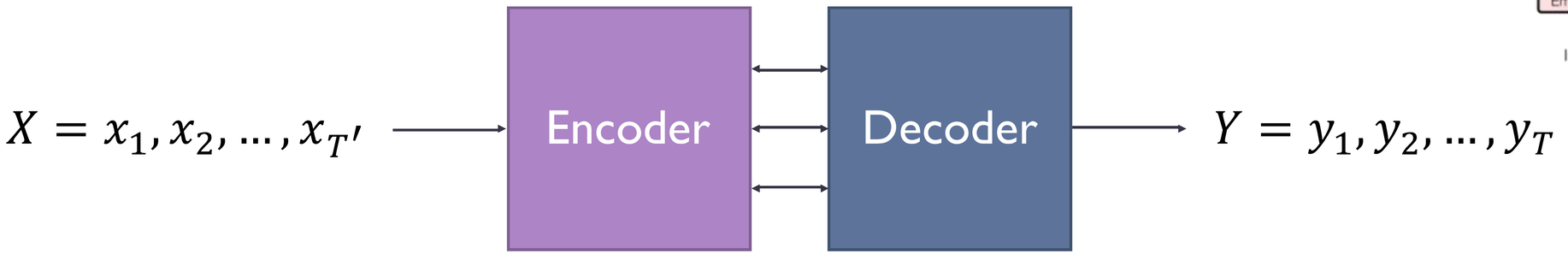
# Neural Machine Translation

Data:

- Parallel corpus (sentence-level aligned)
- Weakly supervised corpus
- Monolingual corpus → semi-supervised/unsupervised learning




Attention Mechanism




$$P(Y|X) = \prod_{t=1}^{T} P(y_t|y_{1\ldots t-1}, x_{1\ldots T'}; \theta)$$

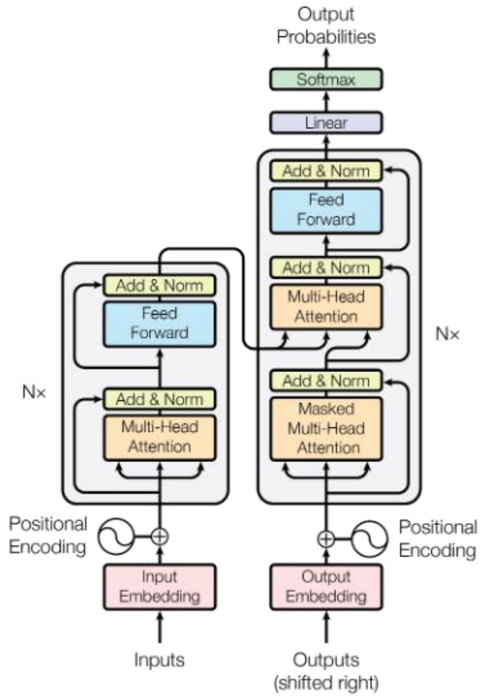
# Neural Machine Translation

- Training: maximum likelihood training

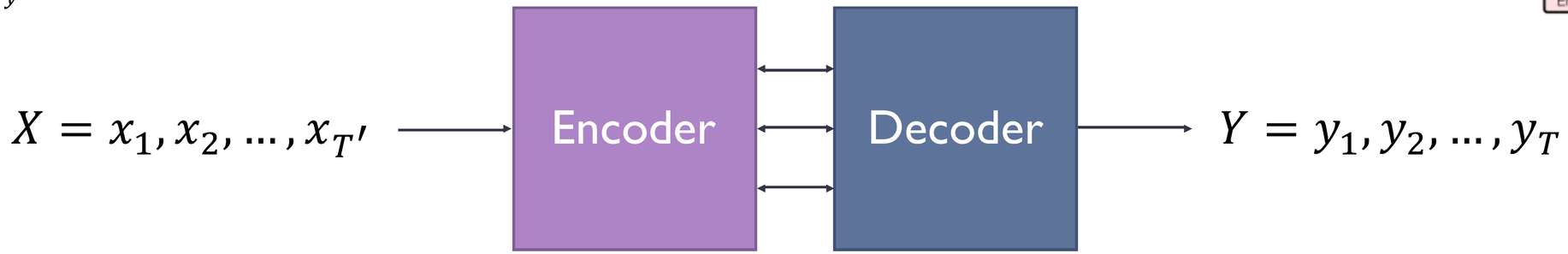
$$J^{ML}(\theta) = \sum_{t=1}^{T} \log P(y_t | y_{1\ldots t-1}, x_{1\ldots T'}; \theta)$$

- Decoding: greedy / beam-search

$$\hat{y}_t = \underset{y}{\operatorname{argmax}} \log P(y_t | \hat{y}_{t-1,\ldots,1}, x_{1\ldots T'}; \theta)$$




Attention Mechanism




$$P(Y|X) = \prod_{t=1}^{T} P(y_t | y_{1\ldots t-1}, x_{1\ldots T'}; \theta)$$

# Knowledge Distillation

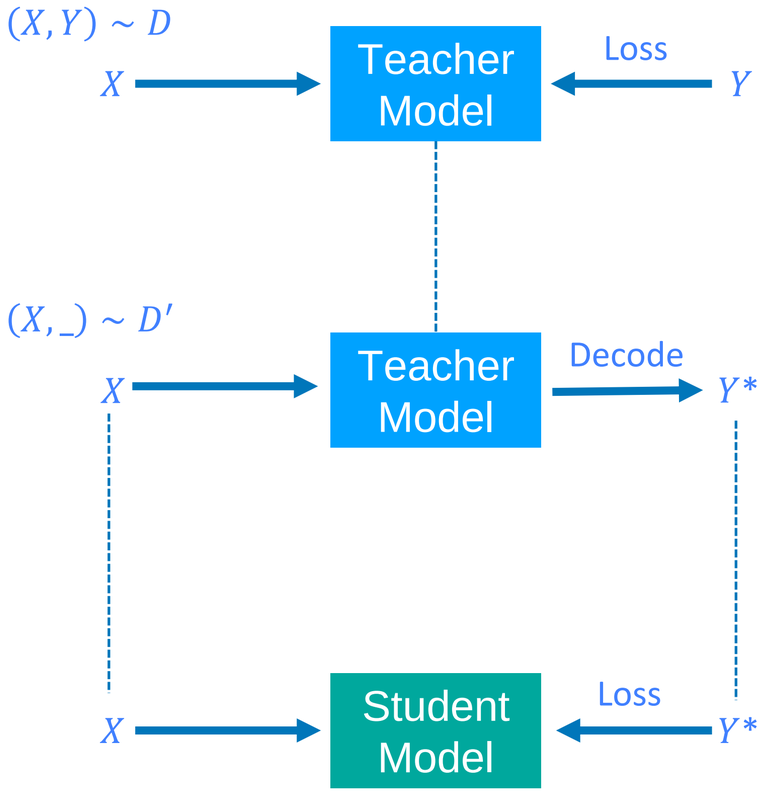
- Knowledge distillation (Liang et al., 2008; Hinton et al., 2015) was originally proposed for training a weaker student classifier on the targets predicted from a stronger teacher model.
- A typical approach is using the label probabilities produced by the teacher as "**soft targets**" (dark knowledge)

$$q_i = \frac{\exp(z_i/\tau)}{\sum_j \exp(z_j/\tau)}$$

- In the context of sequence generation, Kim & Rush (2016) extend this idea using "hard targets" from a teacher generation model. More precisely, $q(t|x) \approx \mathbb{I}\{t = \text{argmax}_{t \in \mathrm{T}} q(t|x)\}$:

$$\begin{aligned}\mathcal{L}_{\text{seq-KD}} &= -\mathbb{E}_{\boldsymbol{x}\sim\text{data}} \sum_{\boldsymbol{t}\in\mathcal{T}} q(\boldsymbol{t}|\boldsymbol{x}) \log p(\boldsymbol{t}|\boldsymbol{x}) \\ &\approx -\mathbb{E}_{\boldsymbol{x}\sim\text{data}, \hat{\boldsymbol{y}}=\arg\max_{\boldsymbol{t}\in\mathcal{T}} q(\boldsymbol{t}|\boldsymbol{x})} \left[\log p(\boldsymbol{t} = \hat{\boldsymbol{y}}|\boldsymbol{x})\right]\end{aligned}$$

# Sequence-level Knowledge Distillation



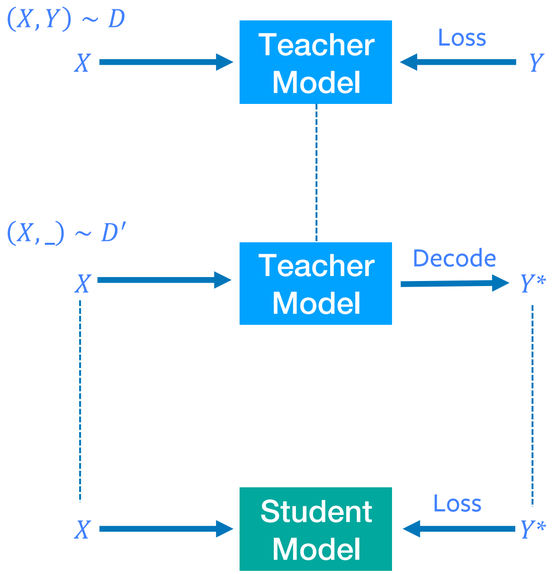

A Teacher-Student Framework in Three Steps:

(1) Train a teacher model with golden targets.

(2) Generate new targets with the pretrained teacher.

(3) Train the student model with the generated targets.

# Sequence-level Knowledge Distillation




## Questions:

(1) How to choose the teacher/student models?

(2) What kind of data can we use for distillation?

(3) In fact, why and how does distillation work in generation?

# Understanding Knowledge Distillation in Non-autoregressive Machine Translation

w/ Chunting Zhou and Graham Neubig

ICLR2020

facebook
Artificial Intelligence Research

# Non-autoregressive Neural Machine Translation

Standard NMT systems are *autoregressive (AT model)*:

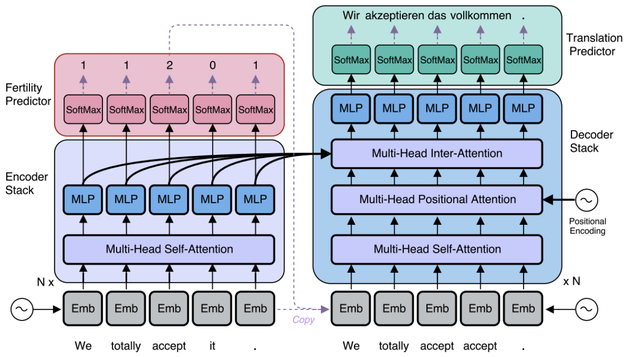
$$P(Y|X) = \prod_{t=1}^{T} P(y_t|y_{1:t-1}, x_{1:T'})$$

- **Strong:** Autoregressive model (e.g. Transformers) can in theory model any arbitrary distribution of sequences.
- **Slow:** we need to predict one word and a time during inference.

Non-autoregressive Translation (NAT model) predicts sequence generation in parallel:

- **Fast:** An alternative solution where we predict all the target tokens in parallel which is favorable for parallelism.
- **Weak:** It is harmful to assume all the output tokens are completely independent.

$$P(Y|X) = \prod_{t=1}^{T} P(y_t|\, x_{1:T'})$$

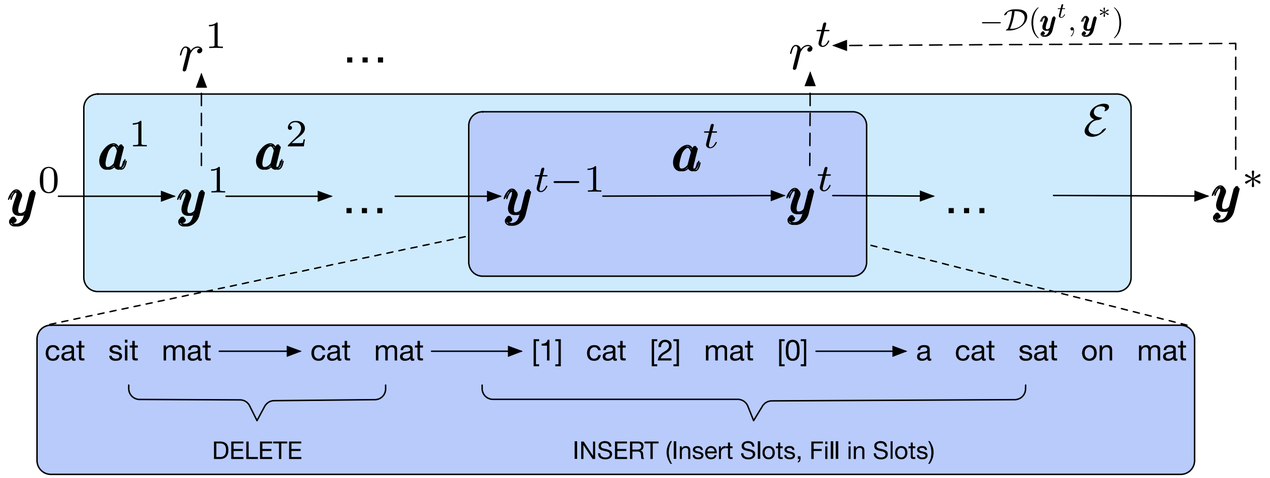



(Figure from Gu et.al, 2017)

# Levenshtein Transformer

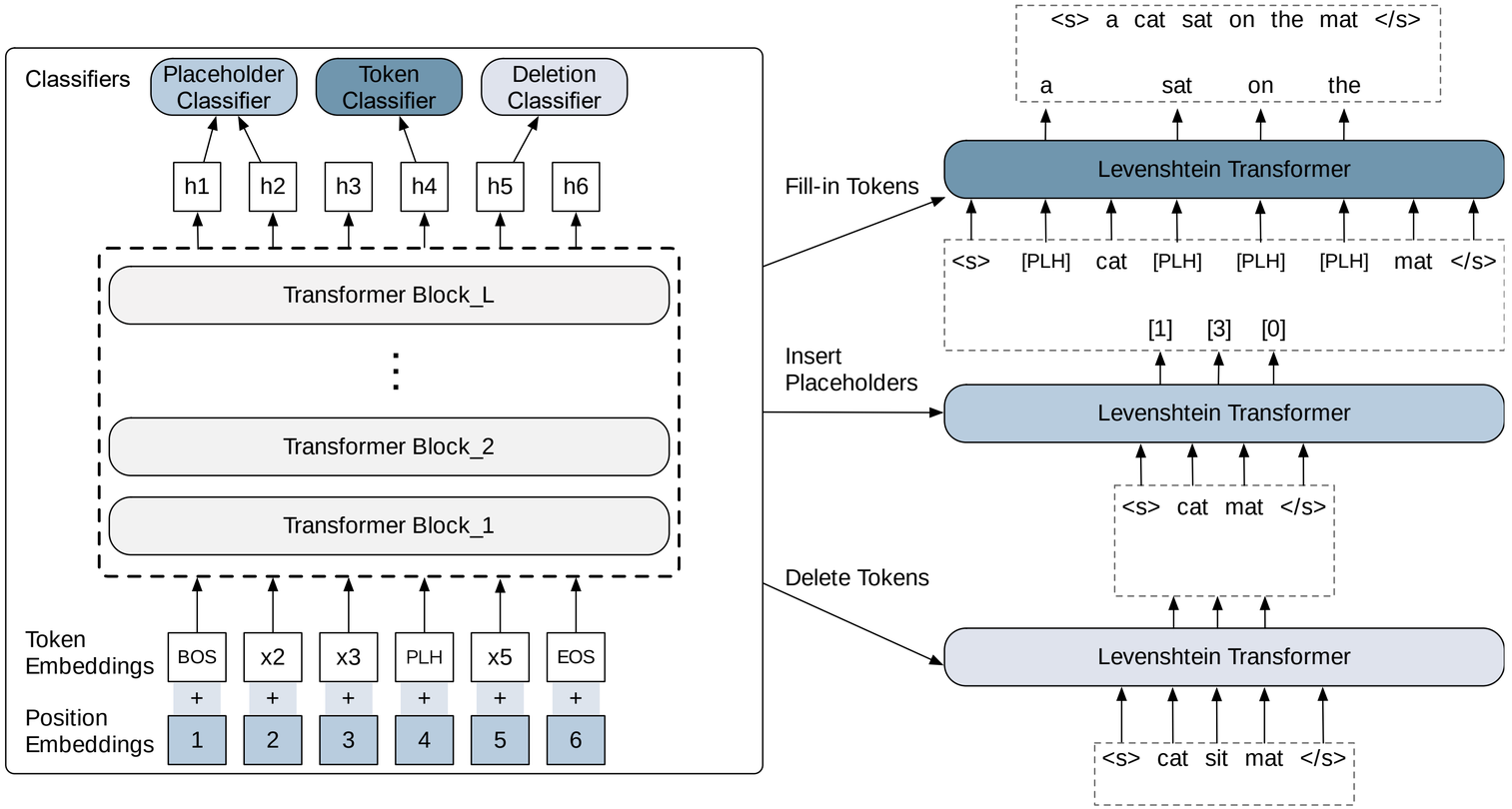
Our major contribution is still at the "modeling side":

- Iterative-based Parallel Refinement (a Markov Decision Process)
- For each iteration, we extend the model by considering "deletion" and "insertion" as the basic operations.
- Both insertion and deletion operations are "non-autoregressive"!




$$\pi(\boldsymbol{a}|\boldsymbol{y}) = \prod_{d_i \in \boldsymbol{d}} \pi^{\text{del}}(d_i|i, \boldsymbol{y}) \cdot \prod_{p_i \in \boldsymbol{p}} \pi^{\text{plh}}(p_i|i, \boldsymbol{y}') \cdot \prod_{t_i \in \boldsymbol{t}} \pi^{\text{tok}}(t_i|i, \boldsymbol{y}'')$$

# Model



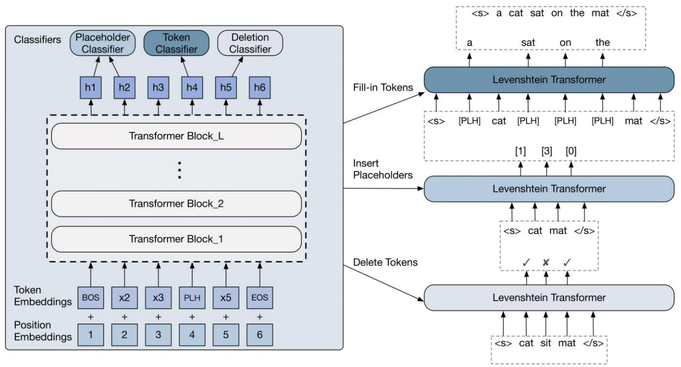

- Encoder-decoder attention was omitted.
- The parameters of three passes can be shared.
- We also propose to "**Early Exit**" which attaches the classifiers to an intermediate block instead of the last to save computation.

# Non-autoregressive Neural Machine Translation

In practice, it is always helpful to obtain some forms of **intermedia representation** Z to capture the ignored dependency between output tokens in NAT.

For instance,

$$P(Y|X) = \sum_{Z} P(Z|x_{1:T'}) \cdot \prod_{t=1}^{T} P(y_t|Z, x_{1:T'})$$

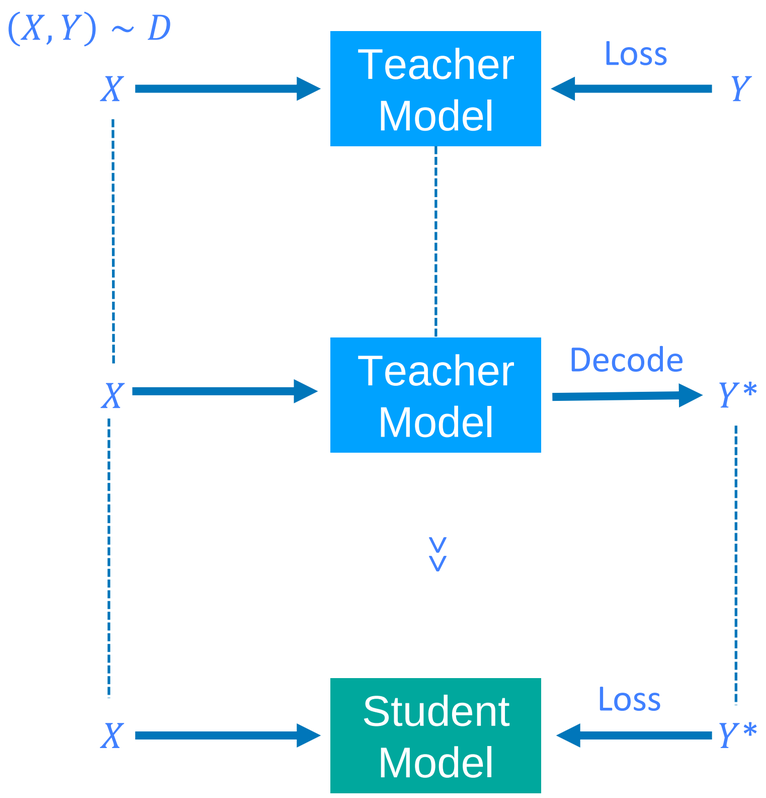



(Figure from Gu et.al, 2019)

Two types of NAT-based models are often considered:

- $Z$ as standard discrete/continuous latent variables (VAE-based NAT)
  - -- https://arxiv.org/abs/1803.03382
  - -- https://arxiv.org/abs/1909.02480
  - …
- $Z$ as intermedia partial generation (Refinement-based NAT)
  - -- https://www.aclweb.org/anthology/D18-1149/
  - -- https://papers.nips.cc/paper/9297-levenshtein-transformer.pdf
  - …

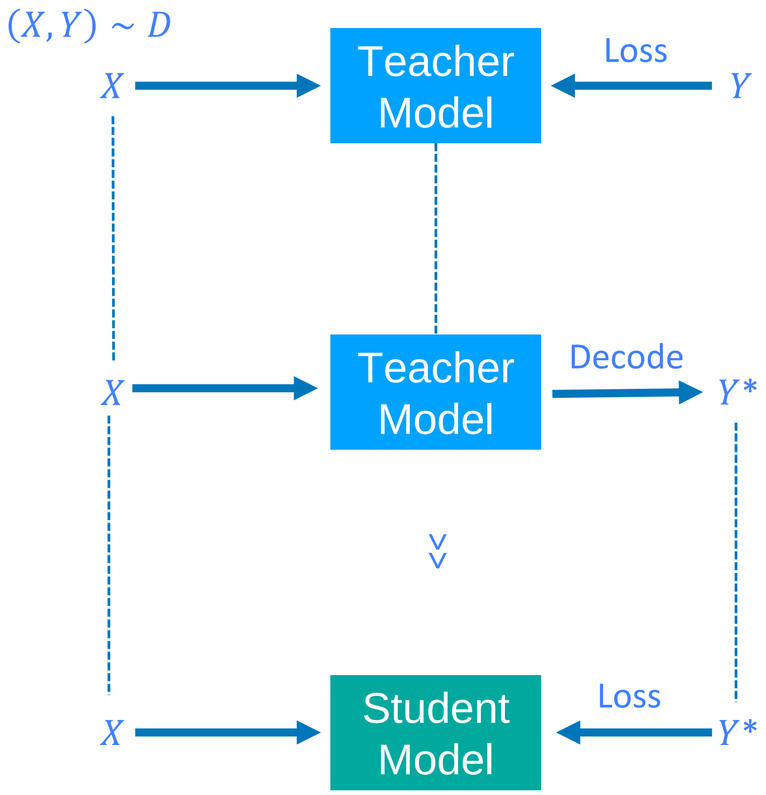
# Knowledge Distillation for NAT




As one of the most successful tricks, KD has been used in *almost* all existing NAT models.

- Typically, the student is our targeted NAT model, while we choose the teacher an autoregressive model (AT).
- As discussed earlier, we can assume “teacher” is much stronger than the student to model the data.
- Both teacher and student models are trained on the same source sentences.

# Knowledge Distillation for NAT




Here is the example performance w/ and w/o distillation for NAT models.

- Test set BLEU on WMT14 English-German (En-De)
- All three models distilled from the same AT Transformer with BLEU score of 27.13 on WMT En-De.

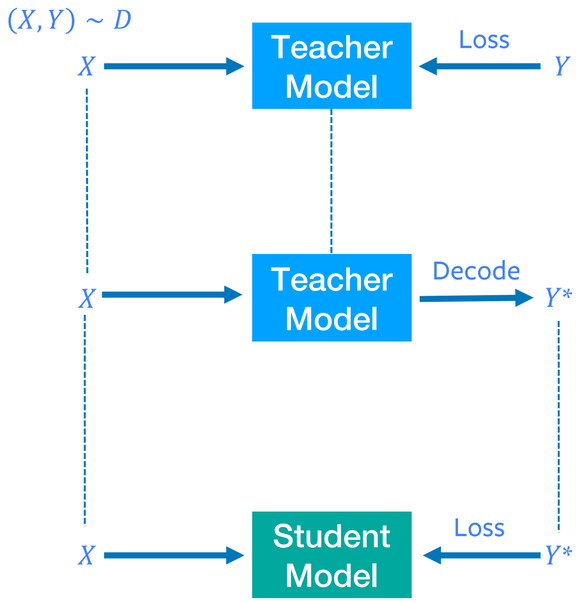
| | w/o distillation | w/ distillation |
|---|---|---|
| Vanilla NAT (Gu et al, 2017) | 11.4 | 19.5 (+8.1) |
| FlowSeq (Ma et al, 2019) | 18.6 | 21.7 (+3.1) |
| LevT (Gu et al, 2019) | 25.2 | 26.9 (+1.7) |

How does knowledge distillation improve NAT models so much?

# Multi-modality Problem

The original NAT paper (Gu et al, 2017) argues the fundamental issue for non-autoregressive models as the multi-modality problem in the data:

For example:




Thank you

Vielen Dank ✓

Danke schön ✓

Danke ✓

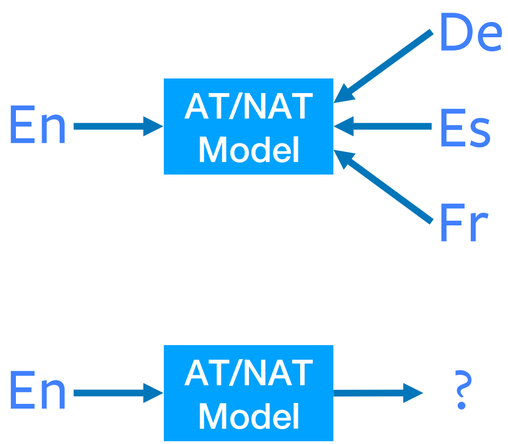
Danke Dank ✗

Vielen schön ✗

Our assumption is that distillation helps to reduce the multimodality in the data.

# Case study on Toy Data

When things are unclear and too difficult to explain in sequence generation (e.g. machine translation tasks), it is always a good idea to look at some toy cases.

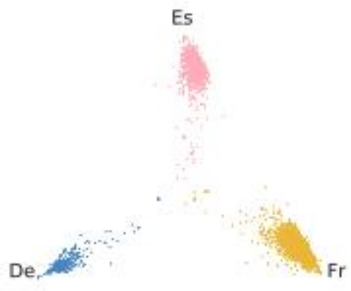



- We create a synthetic dataset compared with three language pairs -- English-German (En-De), English-French (En-Fr) and English-Spanish (En-Es) – from the Europarl corpus. We make sure every English sentence  will be aligned to ALL three languages, and no language ID was specified.

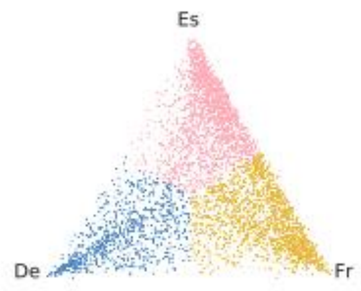
- We train both AT and NAT models directly on this synthetic dataset. During inference time, we input the English sentence without telling the model which language to be output.

We manually created the multi-modality (language id) in the data.

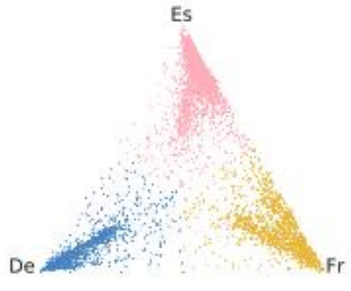
# Case study on Toy Data



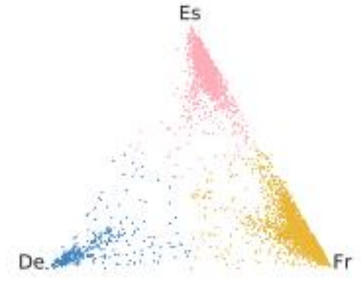

(a) AT baseline.



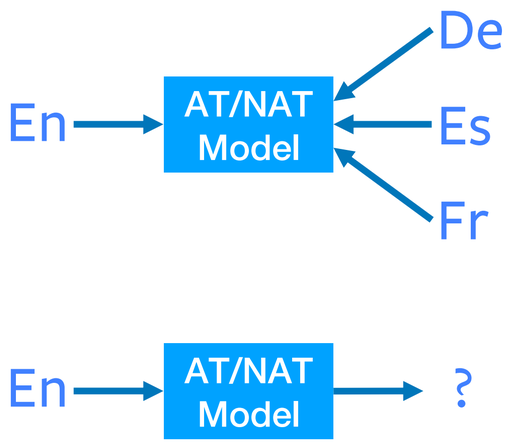

(b) NAT baseline.




(c) NAT trained on reduced mode by random selection.




(d) NAT trained on distilled data set.

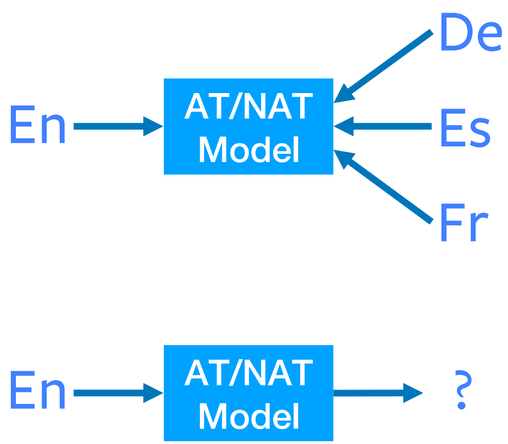



We visualize the mode of "language ID" from the decoded outputs by a simple approximation:

$$p(l_i|\boldsymbol{y}) \approx \frac{1}{T}\sum_{t=1}^{T} p(l_i|y_t) = \frac{1}{T}\sum_{t=1}^{T} \frac{p(y_t|l_i)p(l_i)}{\sum_k p(y_t|l_k)p(l_k)}$$

- Decoding from autoregressive model prefers to select ``modes" over data.
- Non-autoregressive translation fails to capture the mode of language types.
- Training on mode-reduced data set, NAT starts to select one mode in the output, but distillation is a more systematic way of mode selection.

# Case study on Toy Data




Inspired from the visualization on toy data, we propose to use “data uncertainty” to measure the multi-modality (complexity) for **general purpose**.

For simplicity, the data uncertainty is calculated by fitting an alignment model (we use fast-align) and compute the average of **token-level conditional entropy**.

$$\mathcal{H}(\mathbf{Y}|\mathbf{X}=\boldsymbol{x}) = \sum_{\boldsymbol{y}\in\mathcal{Y}} p(\boldsymbol{y}|\boldsymbol{x}) \log p(\boldsymbol{y}|\boldsymbol{x})$$

$$\approx \sum_{\boldsymbol{y}\in\mathcal{Y}} \prod_{t=1}^{T_y} p(y_t|\boldsymbol{x}))(\sum_{t=1}^{T_y} \log p(y_t|\boldsymbol{x}))$$

$$\approx \sum_{t=1}^{T_y} \sum_{y_t\in\mathcal{A}(\boldsymbol{x})} p(y_t|\mathrm{Align}(y_t)) \log p(y_t|\mathrm{Align}(\mathrm{y_t}))$$

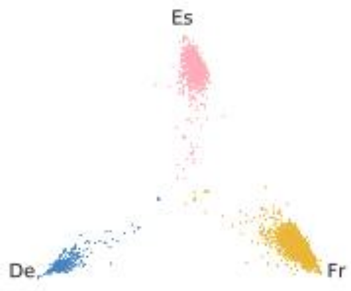
$$= \sum_{t=1}^{T_x} \mathcal{H}(y|x=x_t)$$

Align table obtained from the alignment model

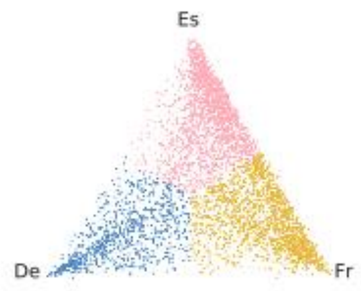
The corpus level complexity is a simple average of the token-level conditional entropy over the vocabulary.

$$C(d) = \frac{1}{|\mathcal{V}_x|} \sum_{x\in\mathcal{V}_x} \mathcal{H}(y|x)$$

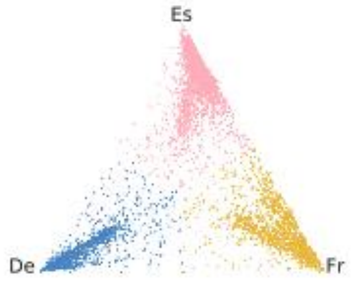
# Case study on Toy Data



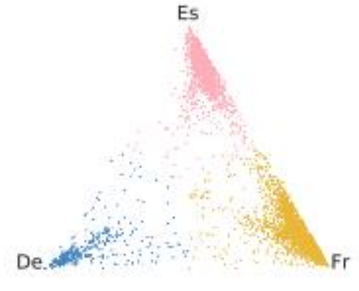

(a) AT baseline.




(b) NAT baseline.




(c) NAT trained on reduced mode by random selection.

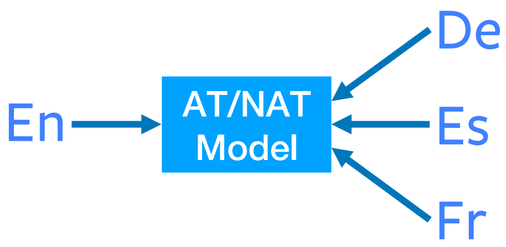


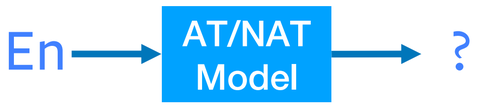

(d) NAT trained on distilled data set.

Complexity ($C(d)$): 3.67

Complexity ($C(d)$): 3.30

Complexity ($C(d)$): **2.64**




In practice, only measuring the complexity of the dataset is not enough for distillation data.

For distilled dataset, we also propose to measure the “faithfulness” which reflects to which extend, the distilled data is representative to the original parallel dataset.

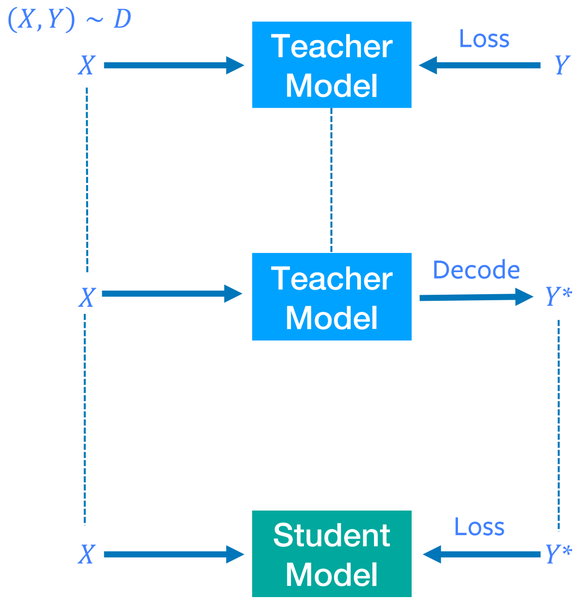
- We compute the **KL-divergence** of the alignment models between the real (r) and the distilled dataset (d)

$$F(d) = \frac{1}{|\mathcal{V}_x|} \sum_{x \in \mathcal{V}_x} \sum_{y \in \mathcal{V}_y} p_r(y|x) \log \frac{p_r(y|x)}{p_d(y|x)}$$

# Experiments

We perform an extensive study over a variety of NAT and AT models with the proposed tools to analyze the **complexity** and **faithfulness** of the distilled dataset.

- Dataset: WMT14 English-German (En-De)
- Models and baseline scores (w/o distillation):




weak

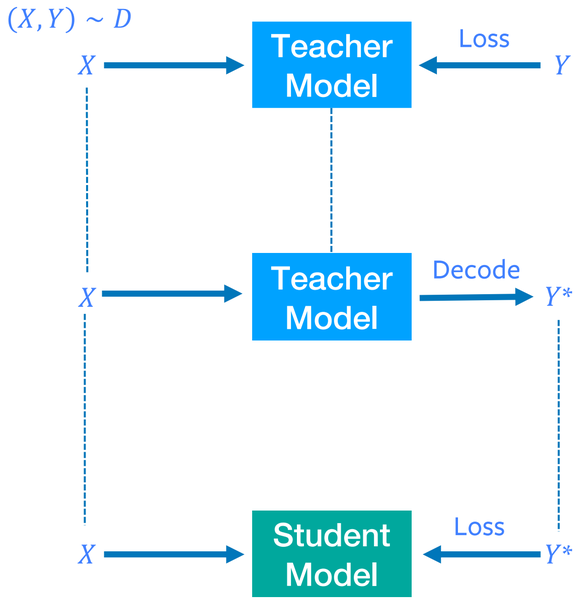
strong

weak

strong

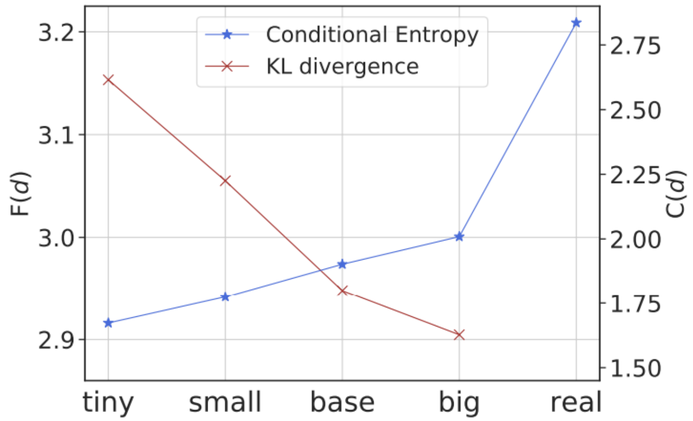
| Models | Params | BLEU | Pass | Iters |
|---|---|---|---|---|
| AT models | | | | |
| AT-tiny | 16M | 23.3 | – | $n$ |
| AT-small | 37M | 25.6 | – | $n$ |
| AT-base | 65M | 27.1 | – | $n$ |
| AT-big | 218M | 28.2 | – | $n$ |
| NAT models | | | | |
| vanilla | 71M | 11.4 | 1 | 1 |
| FlowSeq | 73M | 18.6 | 13 | 1 |
| iNAT | 66M | 19.3 | 1 | $k \ll n$ |
| InsT | 66M | 20.9 | 1 | $\approx \log_2 n$ |
| MaskT | 66M | 23.5 | 1 | 10 |
| LevT | 66M | 25.2 | 1 | $3k \ll n$ |
| LevT-big | 220M | 26.5 | $\approx$3 | $3k \ll n$ |

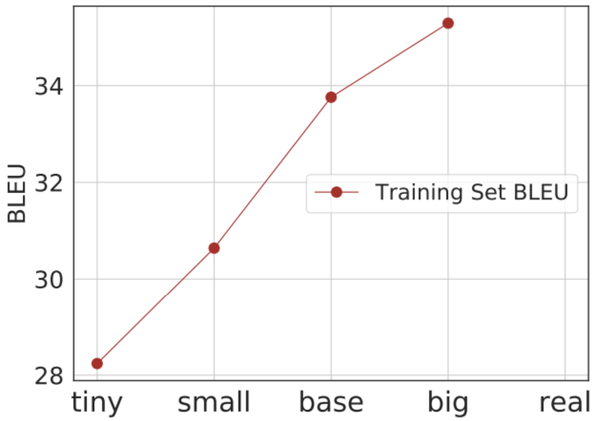
# Experiments




## Analysis of the distilled dataset

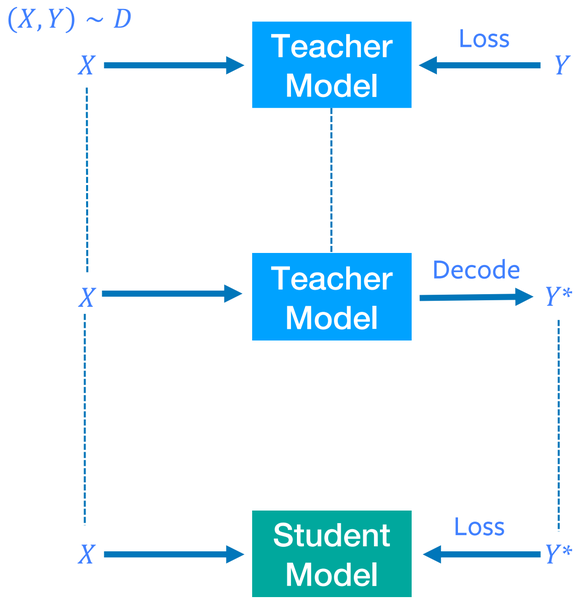
- We visualize the complexity and faithfulness of our all 4 AT models (tiny, small, base, big) as well as the real data.



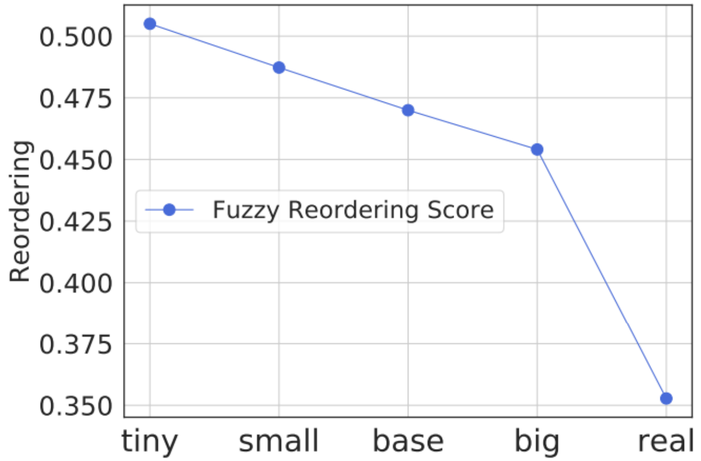

- As additional supporting metrics, we also plot the BLEU score (compared to the real data), showing it also correlates the data quality well.

# Experiments




## Analysis of the distilled dataset

- As additional supporting metrics, we also plot the fuzzing reordering score for each dataset (Talbolt et al. 2011). A larger fuzzy reordering score indicates the more monotonic alignments.




The distilled data looks much more monotonic to the English word order!

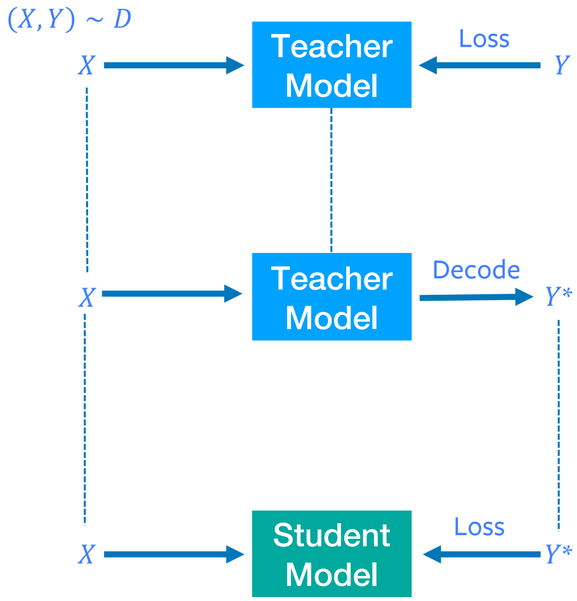
| | |
|---|---|
| Source | For more than 30 years , Josef Winkler has been writing from the heart , telling of the hardships of his childhood and youth . |
| Distilled Target | Seit mehr als 30 Jahren schreibt Josef Winkler aus dem Herzen und erzählt von der Not seiner Kindheit und Jugend . |
| Real Target | Josef Winkler schreibt sich seit mehr als 30 Jahren die Nöte seiner Kindheit und Jugend von der Seele . |

# Experiments

## Analysis of the distillation strategies

- In default, we take the beam-search output from the teacher model to create the distilled dataset. Will different decoding approaches affect the quality of distillation?

YES. We must use beam-search (or at least greedy decoding).

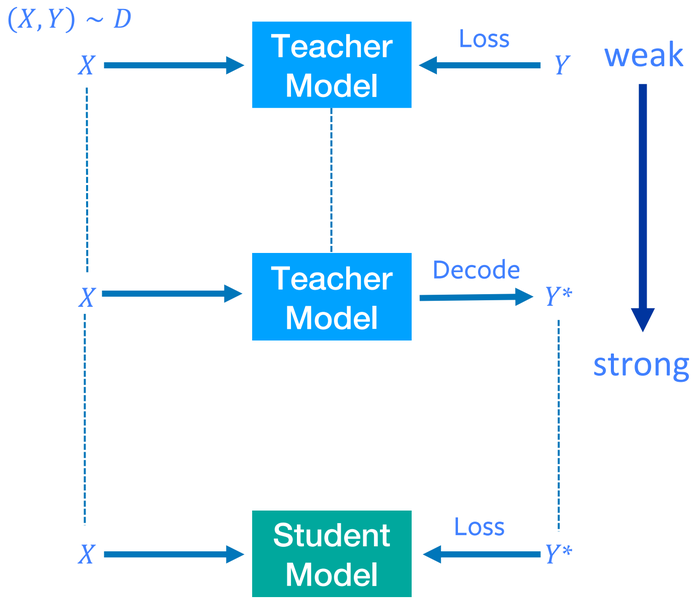


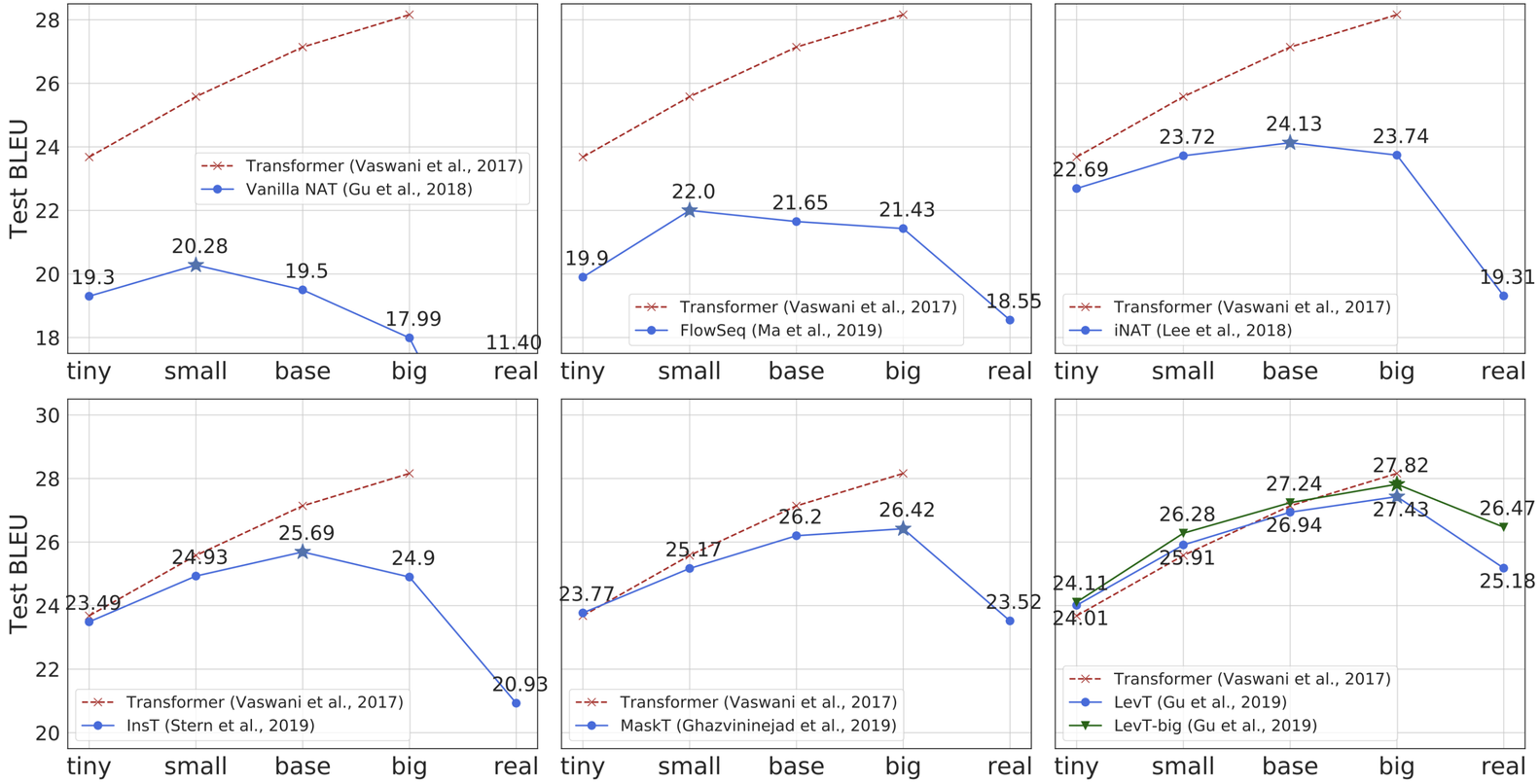

| Decoding Method | $C(d)$ | $F(d)$ | BLEU |
|---|---|---|---|
| sampling | 3.623 | 3.354 | 6.6 |
| sampling (Top 10) | 2.411 | 2.932 | 14.6 |
| greedy | 1.960 | 2.959 | 18.9 |
| beam search | 1.902 | 2.948 | **19.5** |

# Experiments

## Analysis of the NAT models

- Next, we show more results with different NAT models v.s. AT teachers are shown below. We always put the AT teacher scores (in red) for reference.

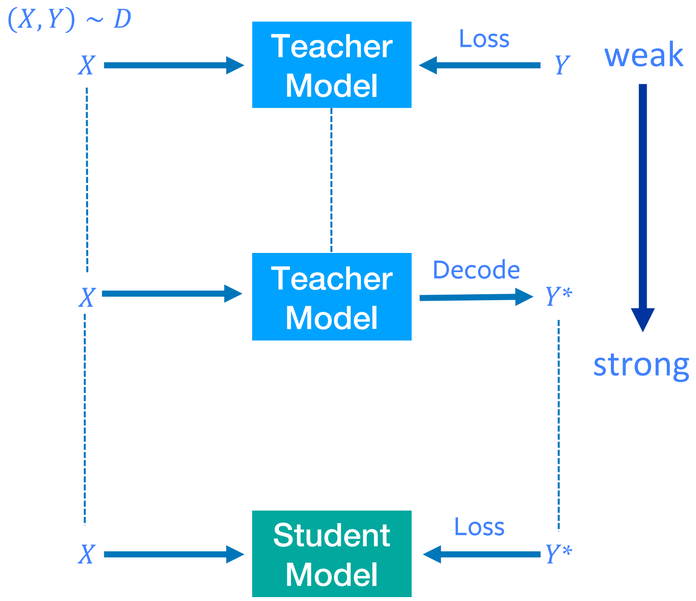


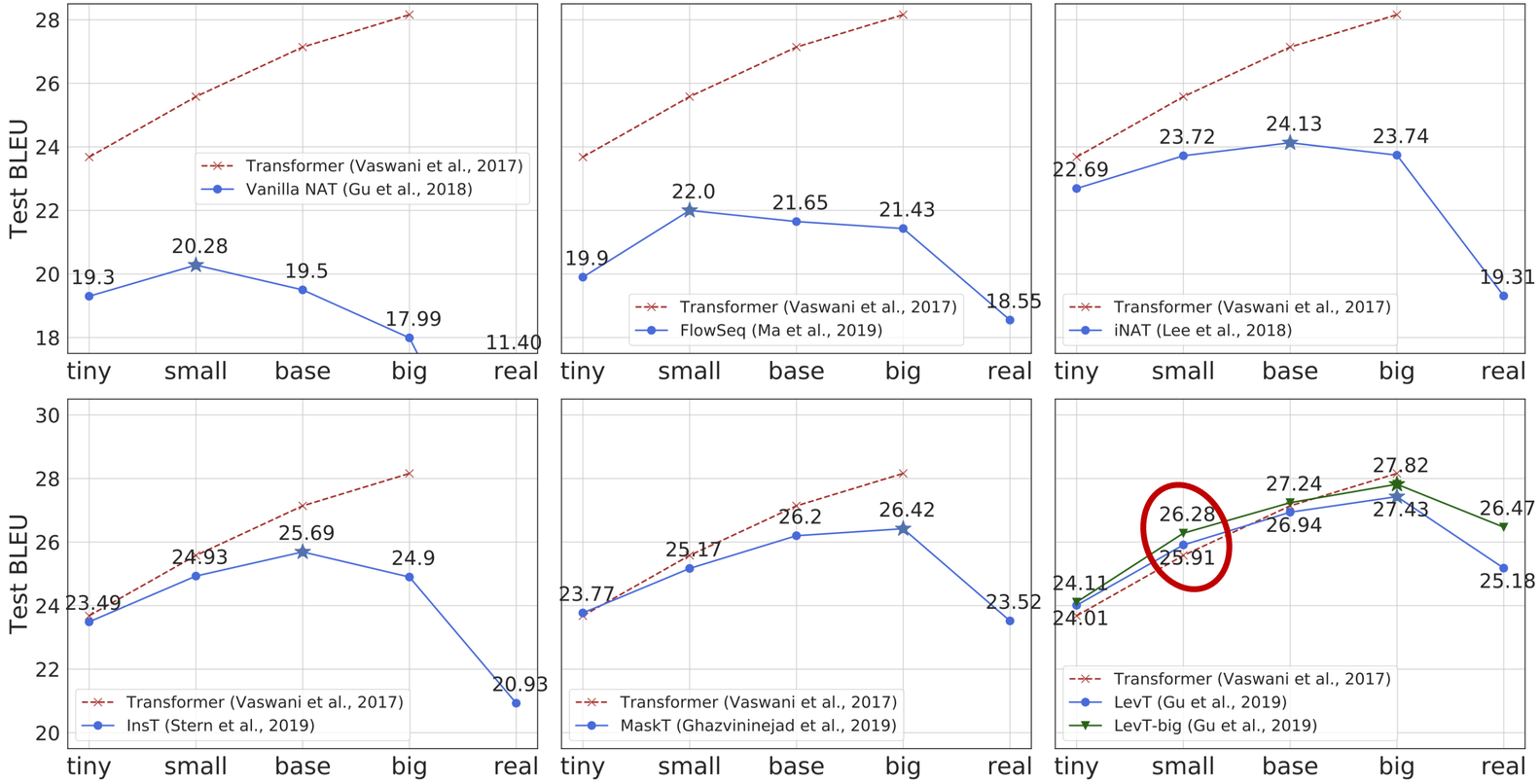



# Experiments

## Analysis of the NAT models

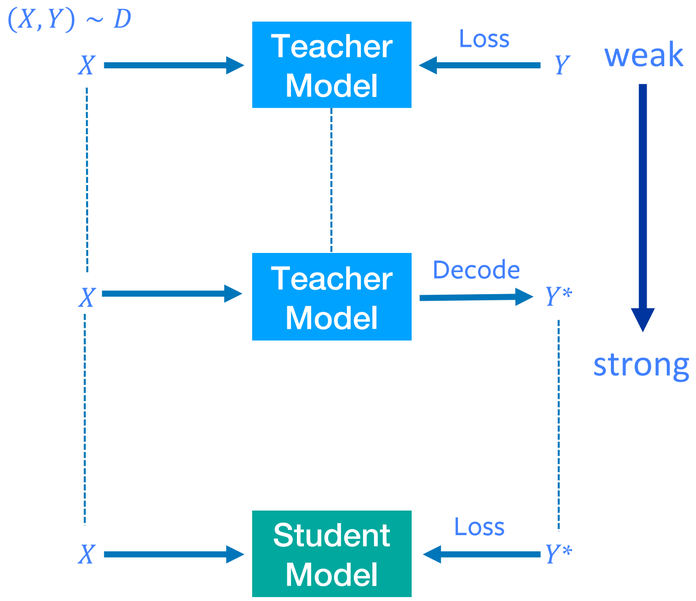
- The stronger the NAT model is, the closer it is to the AT teacher;
- The teacher model does not have to be the upper-bound of the student (we will also come to this question later)



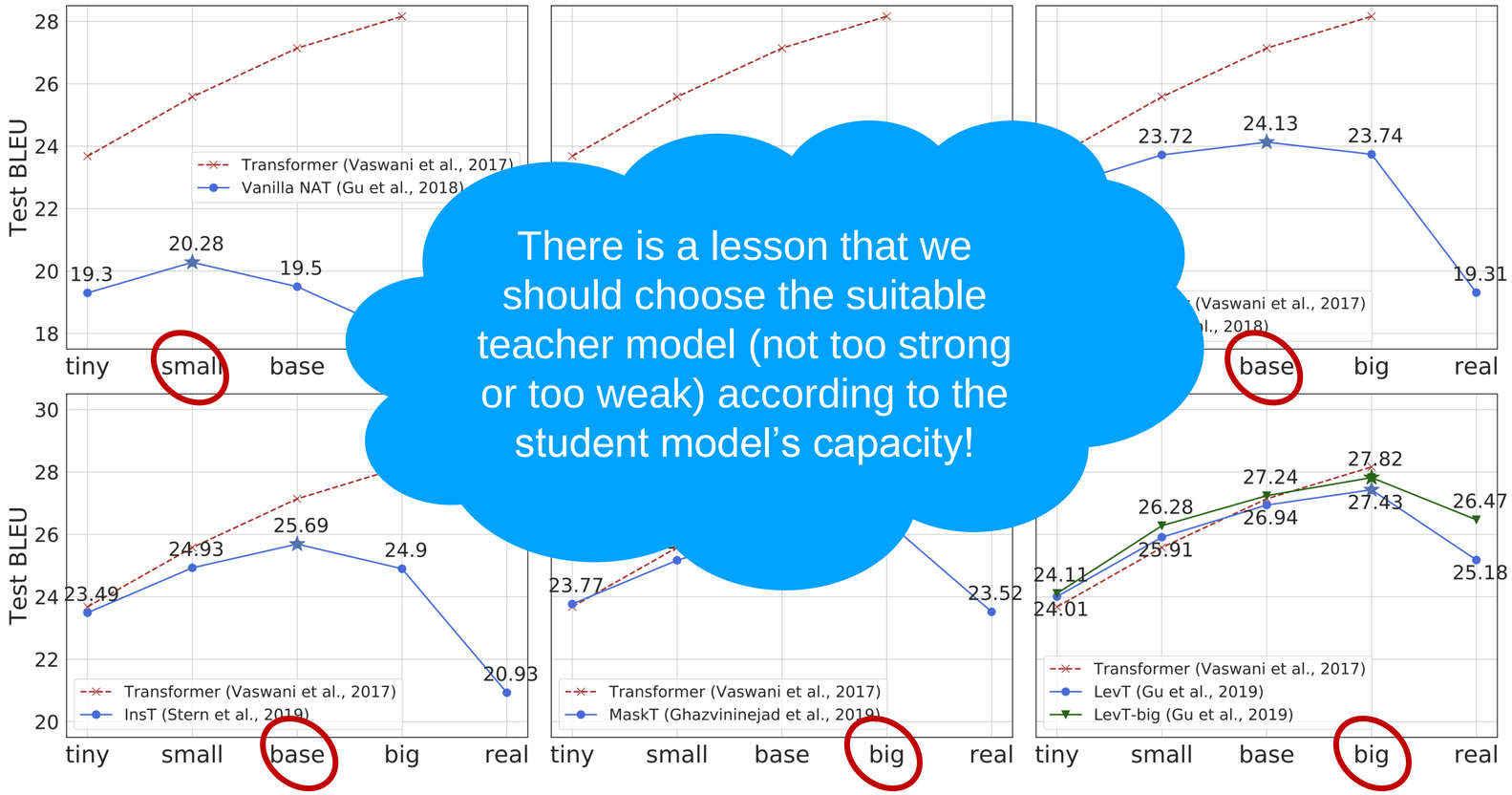

weak → strong

# Experiments




## Analysis of the NAT models

- All NAT performance curves give the same pattern: when increasing the capacity of the teacher model, distillation results first improve and then drop.
- The best performance of NAT models – from lower capacity ones to higher capacity ones – is achieved with distilled data of lower complexity to higher complexity



# Experiments

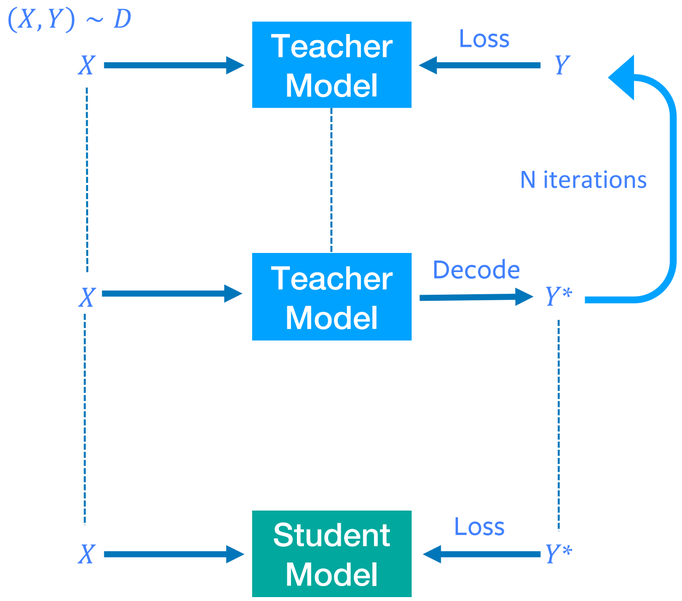
## Improvements for WEAK student models

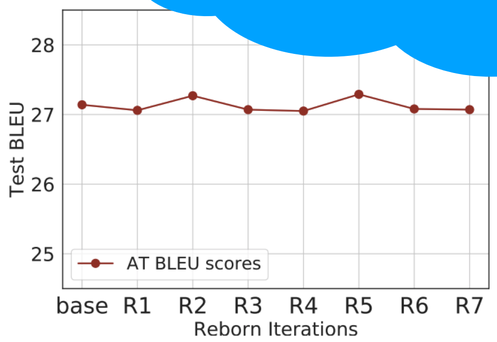
- Take the vanilla NAT model as an example.

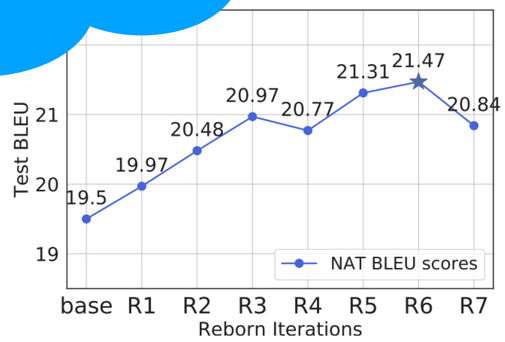
Born-Again Networks (BAN):

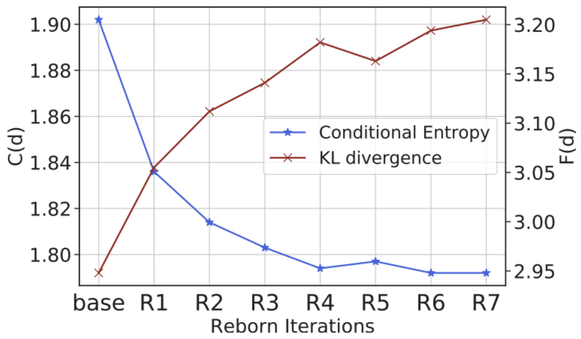
- Based on previous discussion, weak models require to be trained on simpler data. However, decreasing the size of the teacher model (e.g. base -> small) will hurt the faithfulness of the distilled data;
- BAN instead is a sim… …y distill the teacher model by its own out… …al output to train the stud…

Distilled from the same model will not affect the BLEU score.



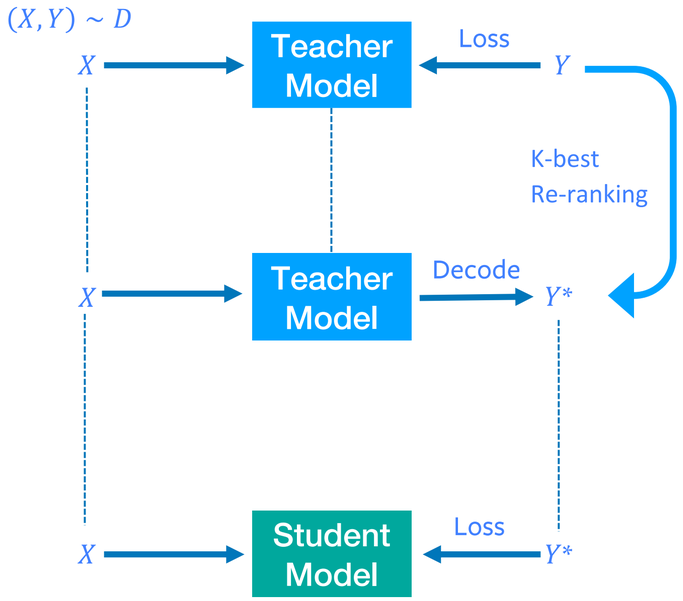
# Experiments

## Improvements for STRONG student models

- Take the Levenshtein Transformer model as an example.

Sequence-level Interpolation (Seq-Inter):

- Based on previous discussion, strong models can be trained on more difficult data with high faithfulness. However, it requires training much stronger autoregressive teacher models (which is not easy) ;
- Kim & Rush, 2016 in fact also proposed improved version of distillation named sequence-level interpolation, where we choose the K-best beam search results and re-rank to select the sentences with the highest sentence-BLEU score from the ground-truth.




| $d$ | $C(d)$ | $F(d)$ | BLEU |
|---|---|---|---|
| base | 1.902 | 2.948 | 26.94 |
| base-inter | 1.908 | 2.916 | **27.32** |

However, in practice this approach is very sensitive to the beam-size.

# Implementation

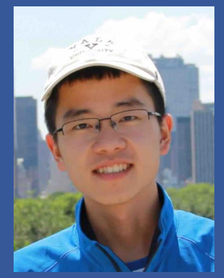
Code for most of the NAT models can be found in Fairseq-py
https://github.com/pytorch/fairseq/tree/master/examples/nonautoregressive_translation

# Revisiting Self-Training for Neural Sequence Generation

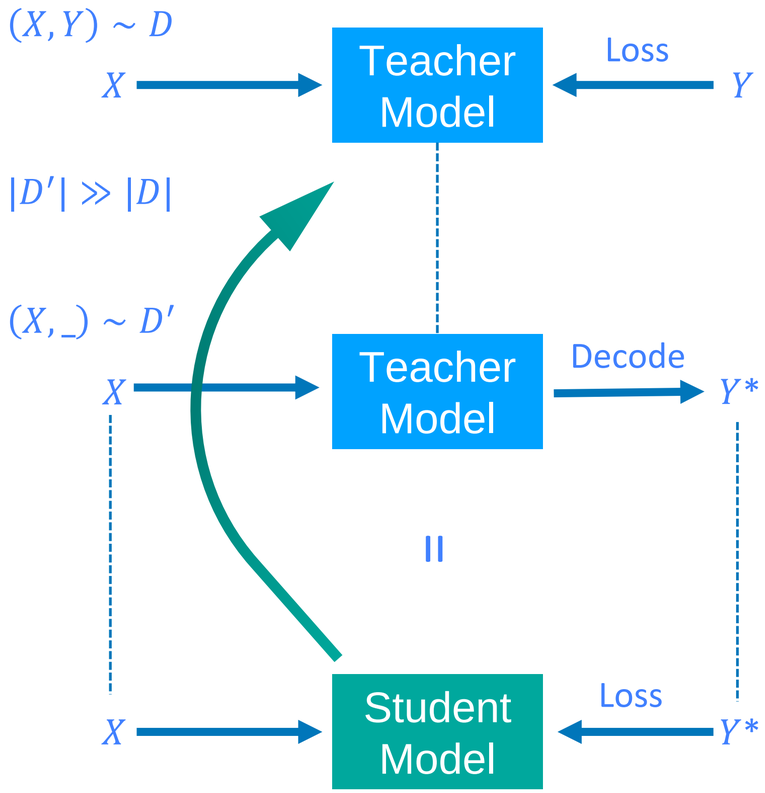
w/ Junxian He, Jiajun Shen and Marc'Aurelio Ranzato

ICLR2020

facebook
Artificial Intelligence Research

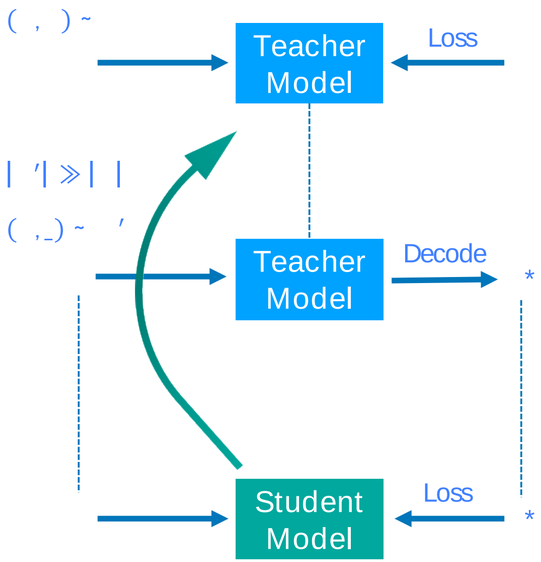


# Self-Training




- To answer the second question, we analyze how distillation works when introducing more data. We keep teacher and student the same architecture.
- In literature, such special setting of knowledge distillation is also called "**self-training**".
- Different from the previous part, we usually need to "fine-tune" the student model on the real data ($D$) again (green arrow).
- Furthermore, the fine-tuned student model can be treated as a new teacher, and we can repeat this loop multiple times, resulting in Iterative Self-Training.

# Self-Training




How does self-training works in practice?

- Test set BLEU on a subset of 100K parallel sentences from WMT14 English-German (En-De).

| | Baseline | Iteration-1 | Iteration-2 | Iteration-3 |
|---|---|---|---|---|
| Pseudo-train* | - | 16.5 | 18.2 | 18.7 |
| Train/Fine-tune | 15.6 | 17.9 | 18.6 | 18.7 |

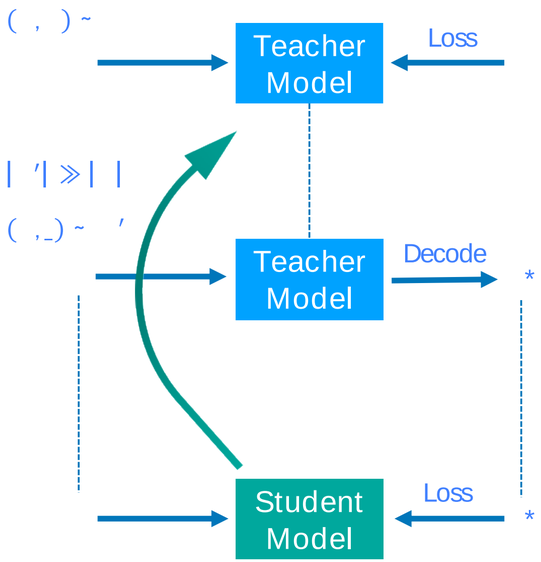
- Even with the equal size teacher/student, the performance of the student is still improving by many iterations!
- The student trained only with distillation data, can usually outperform its teacher!
- Fine-tuning on real data further boosts the translation quality, providing a better teacher model for the next iteration.

*Pseudo-train: training on the distilled dataset.

# The Secrets Behind Self-Training

We examine two possible hypotheses:

- **Decoding Strategy**




The first possibility is that the gain comes from the “better” target.

- Typically, we always use “beam-search” instead of “sampling” from the teacher model’s own distribution.
- The beam-searched targets serve as a “stronger” teacher model than the student.

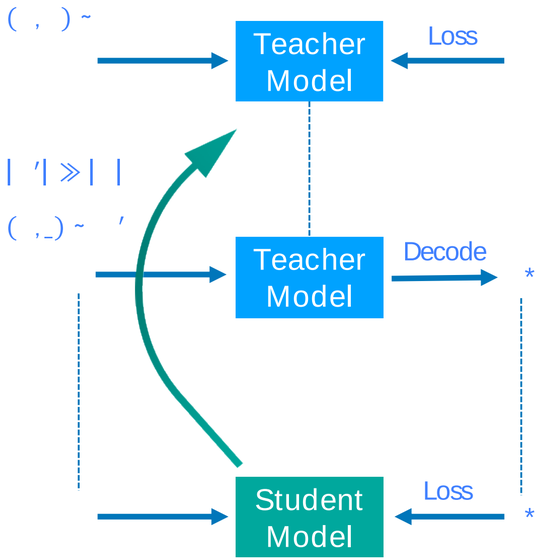
| | Baseline | Beam-search | Sampling |
|---|---|---|---|
| Pseudo-train | - | **16.5** | **16.1** |
| Train/Fine-tune | 15.6 | 17.9 | 17.0 |

The decoding strategy do affect the performance, however, is not the only secrets behind the improvement.

# The Secrets Behind Self-Training

We examine two possible hypotheses:

- Decoding Strategy
- **Noise during Training (Dropout)**



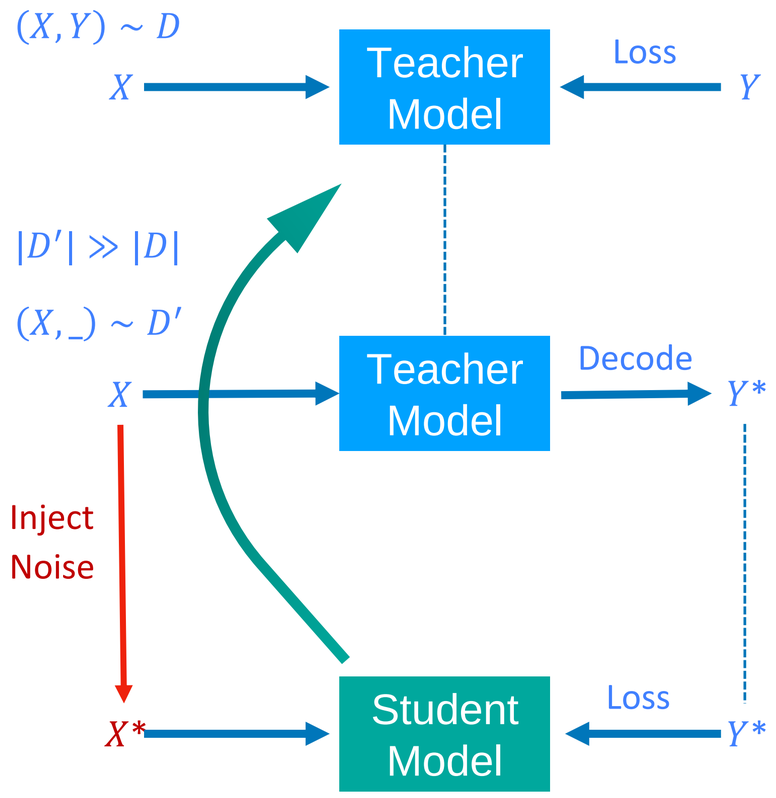

The second assumption comes from the mismatched behaviors of "training" and "inference":

- Dropouts are usually turned-off in the inference time, while open during training ➔ self-training is not really "self".

| | Baseline | Beam-search w/o Dropout | Sampling w/o Dropout | Beam-search | Sampling |
|---|---|---|---|---|---|
| Pseudo-train | - | **15.8** | 15.5 | **16.5** | **16.1** |
| Train/Fine-tune | 15.6 | 16.3 | 16.0 | 17.9 | 17.0 |

Improvements disappeared on the pseudo-training phase!!

# Noisy Self-Training



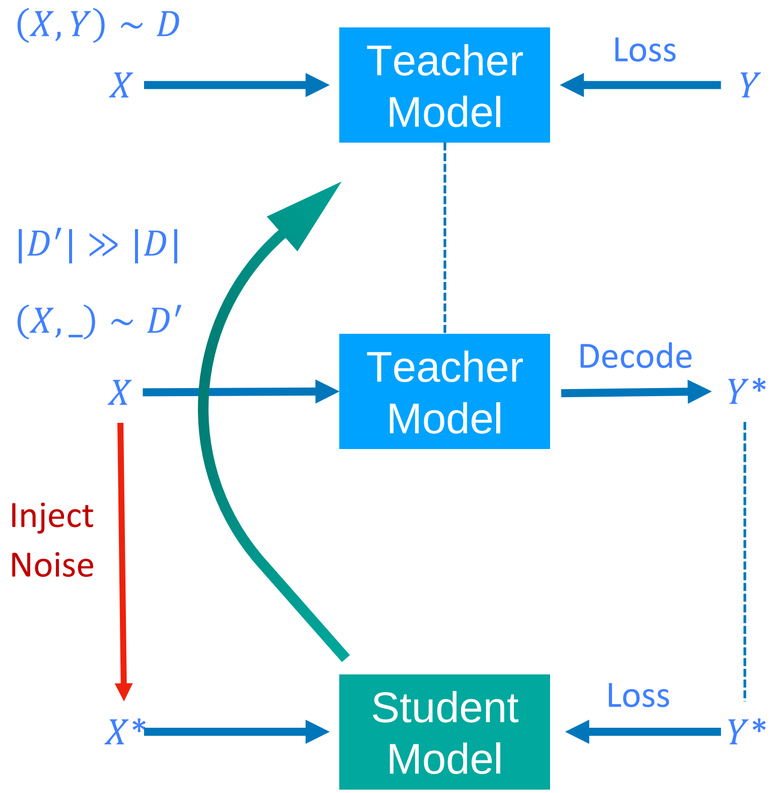

Since we found "noise" during training useful, what if we add more?

- Injecting synthetic noise in the input words, e.g. word swap, word deletion and word blanking (Lample et al., 2018).

| | Baseline | Beam-search | Noisy Input + Beam-search |
|---|---|---|---|
| Pseudo-train | - | **16.5** | **16.6** |
| Train/Fine-tune | 15.6 | 17.9 | **19.3** |

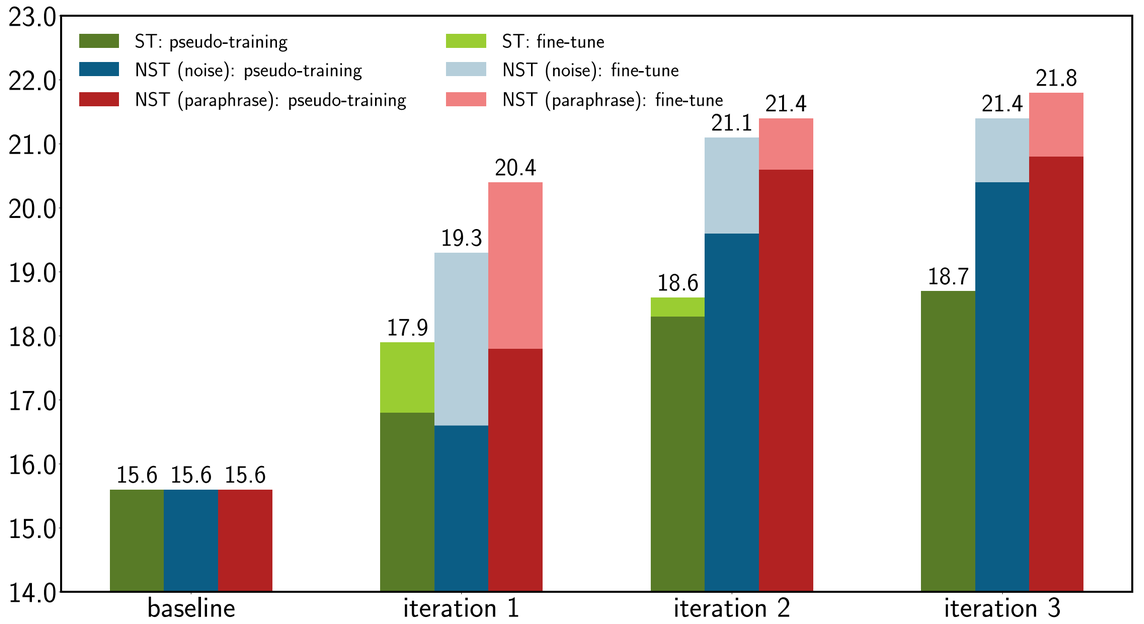
- Injecting noise will not improve the pseudo-train results (should be expected as neither the source or the target are "REAL" sentences.
- However, injecting noise largely improve the performance on fine-tuning!

# Noisy Self-Training




Since we found "noise" during training useful, what if we add more?

- Injecting synthetic noise in the input words, e.g. word swap, word deletion and word blanking (Lample et al., 2018).
- We also try using "round-trip" paraphrase instead of synthetic noise, however, the improvements are similar.



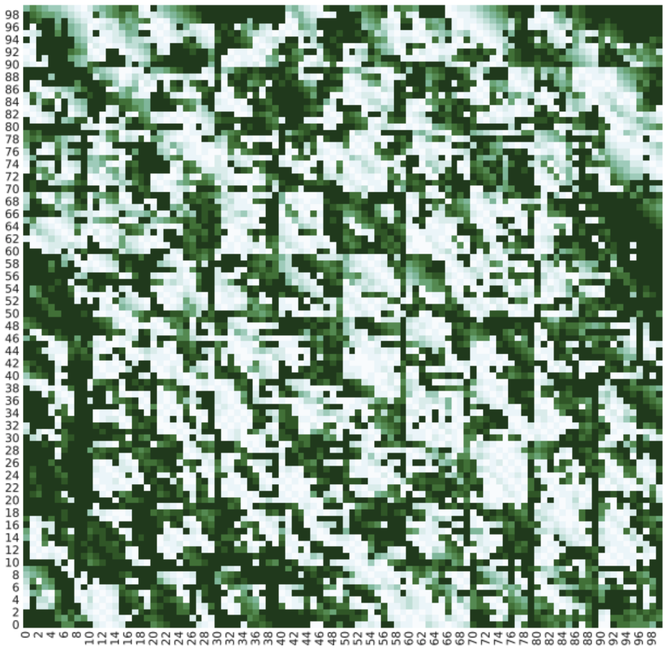

What is the **role** of "noise" in Self-training?

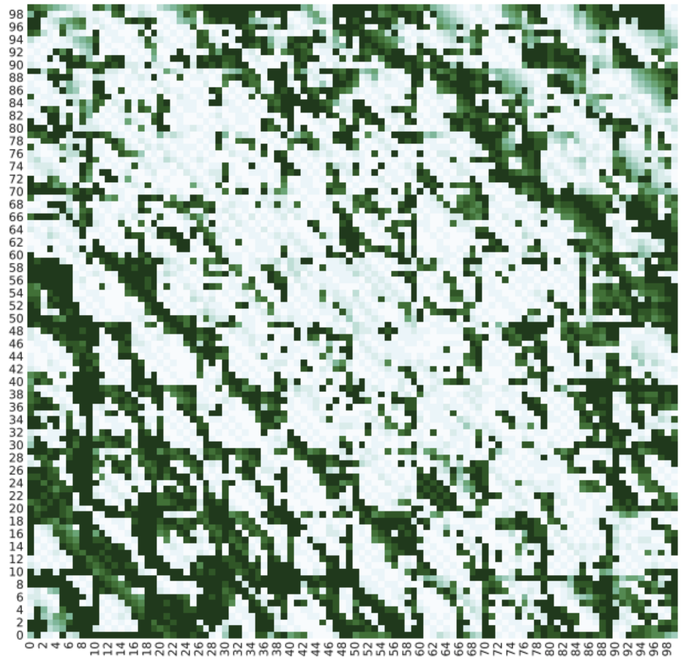
# Case study on Toy Data

When things are unclear and too difficult to explain in sequence generation, it is always a good idea to look at some toy cases.

- Summing two integers in 0~99 as a sequence generation task;
- Model works in the character level.
- We use only 250 pairs to training this task.


One good feature of this summing task is that we can easily visualize the results in a 2D space. For example:





# Case study on Toy Data

1 1 2 3 → Model → 3 4

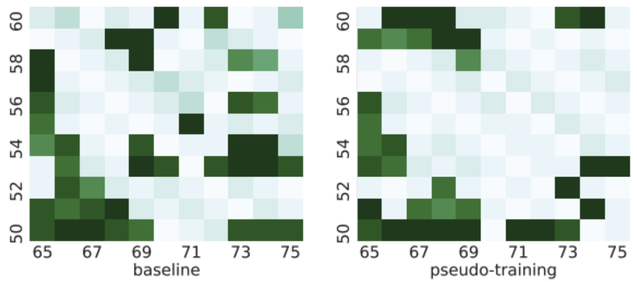
## Quantitative Analysis for Noisy Self-training*

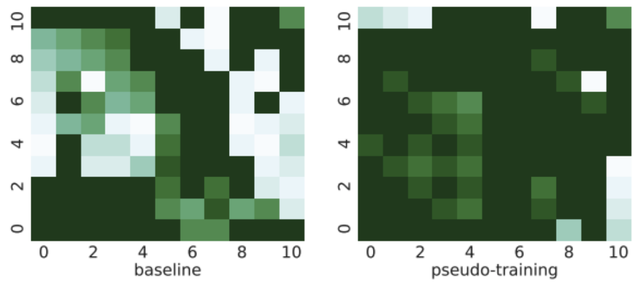
| Methods | smoothness | symmetric | error |
|---|---|---|---|
| baseline | 9.1 | 9.8 | 7.6 |
| ST | 8.2 | 9.0 | 6.2 |
| noisy ST | **7.3** | **8.2** | **4.5** |

Table 3: Results on the toy sum dataset. For ST and noisy ST, smoothness (↓) and symmetric (↓) results are from the pseudo-training step, while test errors (↓) are from fine-tuning, all at the first iteration.

The injected noise will smooth the output space!

## Qualitative Analysis for Noisy Self-training

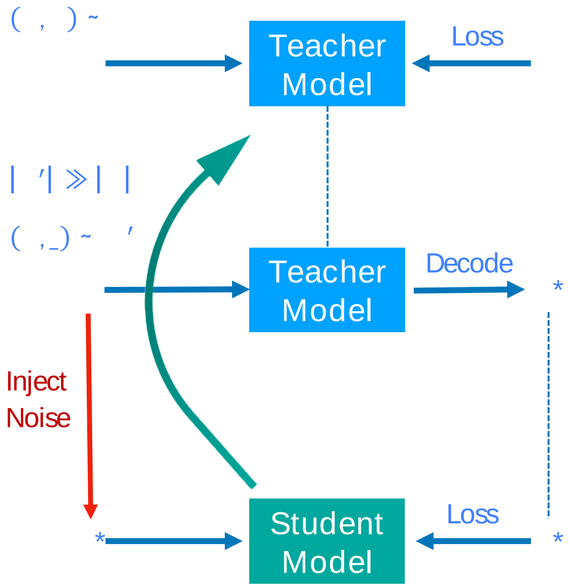



*Detailed definition of these metrics can be found in the paper.

# Experiments

We validate the proposed noisy self-training methods on both machine translation (MT) and text summarization (TS) tasks.




Machine Translation task:

- WMT14 English-German (En-De):
  simulated low-resource MT (100K) + 3.8M English (from the remaining)
  full parallel data (3.9M) + 20M English (sampled from News Crawl)

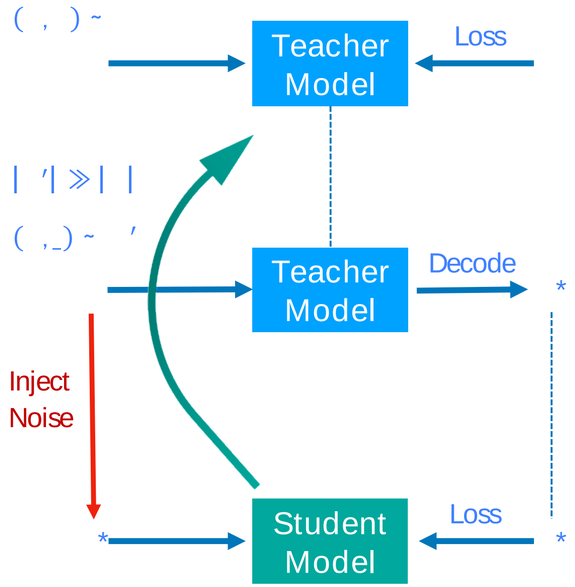
- FloRes English-Nepali (En-Ne)
  real low-resource MT (560K) + 5M English (sampled from Wikipedia)

- All noisy ST are performed 3 iterations. We also build up back-translation baselines for comparison with target side monolingual data.

| Methods | WMT English-German | | FloRes English-Nepali | | |
|---|---|---|---|---|---|
| | 100K (+3.8M mono) | 3.9M (+20M mono) | En-Origin | Ne-Origin | Overall |
| baseline | 15.6 | 28.3 | 6.7 | 2.3 | 4.8 |
| BT | 20.5 | – | 8.2 | **4.5** | **6.5** |
| noisy ST | **21.4** | **29.3** | **8.9** | 3.5 | **6.5** |

# Experiments

We validate the proposed noisy self-training methods on both machine translation (MT) and text summarization (TS) tasks.




Text Summarization:

- English Gigaword dataset
  simulated low-resource TS (100K, 640K);
  full data (3.8M) + 4M monolingual documents (from the filtered Gigaword dataset)

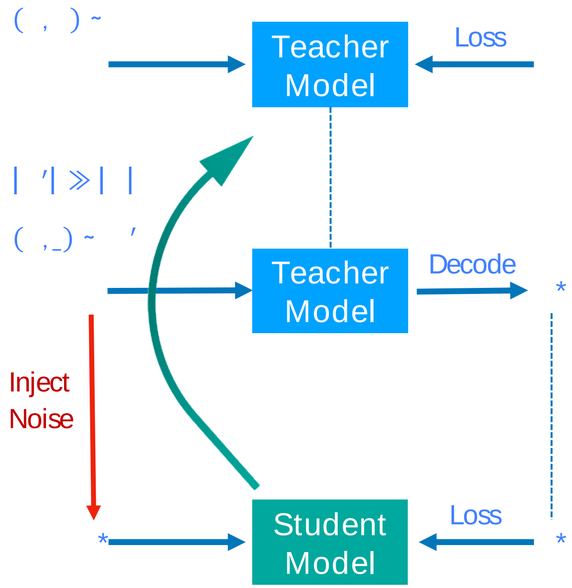
- All noisy ST are performed 3 iterations. We also build up back-translation baselines for comparison with target side summarizations.

| Methods | 100K (+3.7M mono) | | | 640K (+3.2M mono) | | | 3.8M (+4M mono) | | |
|---|---|---|---|---|---|---|---|---|---|
| | R1 | R2 | RL | R1 | R2 | RL | R1 | R2 | RL |
| MASS (Song et al., 2019)* | – | – | – | – | – | – | 38.7 | 19.7 | 36.0 |
| baseline | 30.4 | 12.4 | 27.8 | 35.8 | 17.0 | 33.2 | 37.9 | 19.0 | 35.2 |
| BT | 32.2 | 13.8 | 29.6 | **37.3** | **18.4** | **34.6** | – | – | – |
| noisy ST | **34.1** | **15.6** | **31.4** | 36.6 | 18.2 | 33.9 | **38.6** | **19.5** | **35.9** |

# Experiments

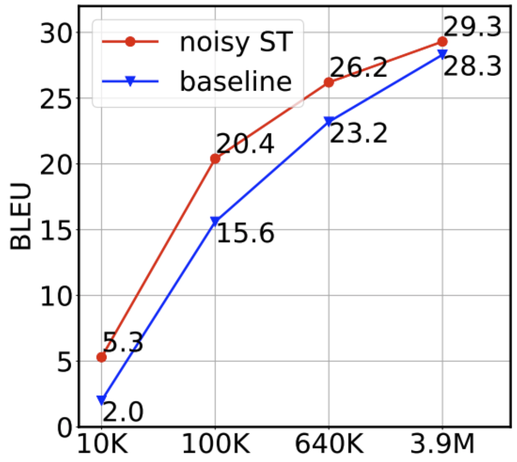
## Analysis of Dataset Size for Noisy Self-Training

- Take the simulated WMT14 En-De data as an example:



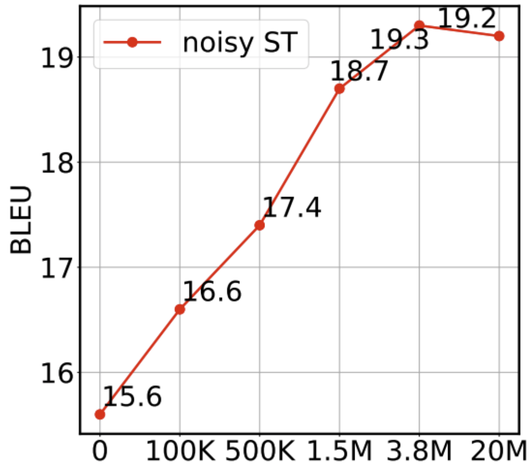

v.s. parallel data size

(Fixed 20M News Crawl monolingual)




v.s. monolingual data size

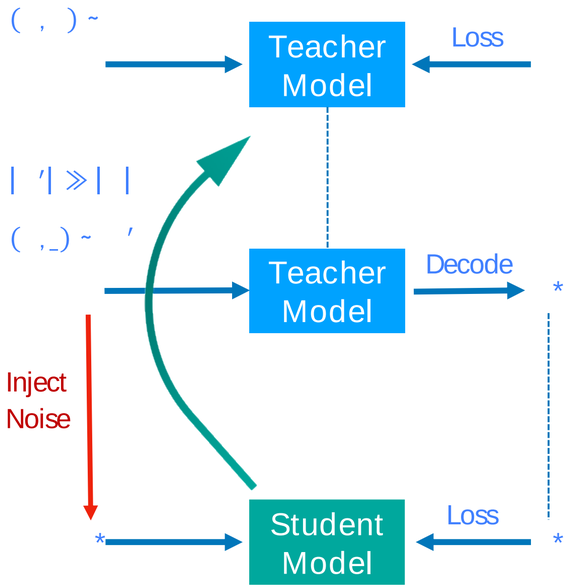
(Fixed 100K parallel)



# Experiments

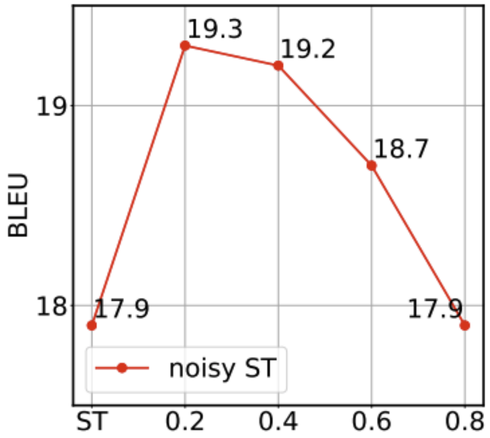
## Analysis of Noise-level injected in Noisy Self-Training

- Take the simulated WMT14 En-De data as an example:




We vary the ratio of “word blanking” when injecting the noise.

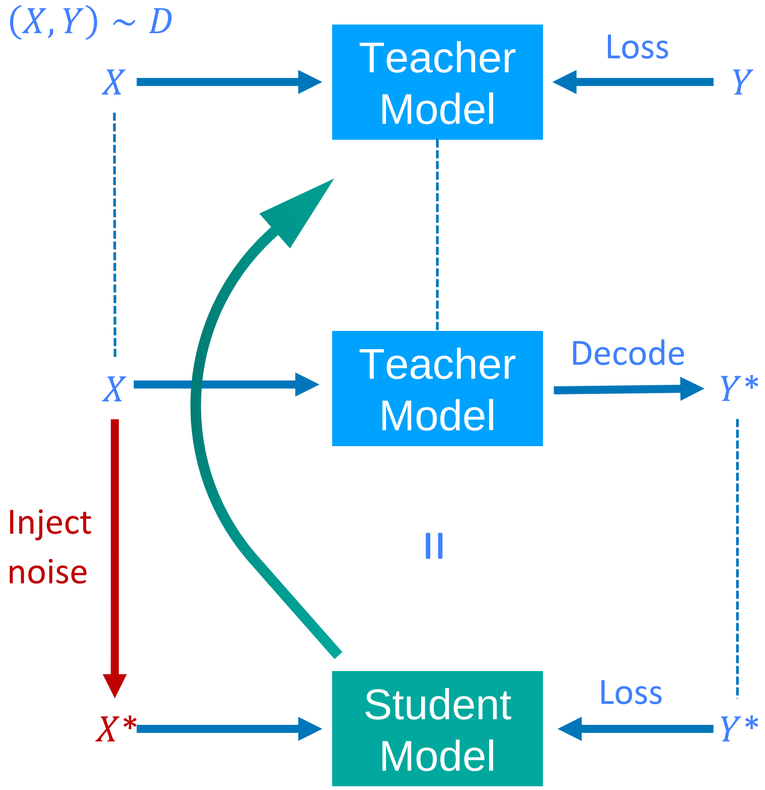
Not surprisingly, the performance of self-training drops a lot if the noise is too large.



# Experiments

## WAIT… one step back? What if we do not have new data, but inject noise onto parallel data?

- Take the simulated WMT14 En-De data as an example:




- If following the same process as noisy self-training, only with parallel data still improves the performance (not as much as with monolingual data)
- However, if we only inject noise onto the source side, with real sentence as the targets. The model will get much worse performance.

| Methods | PT | FT |
|---|---|---|
| parallel baseline | – | 15.6 |
| noisy ST, 100K mono + fake target | 10.2 | 16.6 |
| noisy ST, 3.8M mono + fake target | 16.6 | 19.3 |
| noisy ST, 100K parallel + real target | 6.7 | 11.3 |
| noisy ST, 100K parallel + fake target | 10.4 | 16.0 |

# Future works

Can we combine these two work?

- For instance, training a teacher AT model on limited parallel data;
- Distilled the model on much more monolingual data to train an NAT model

How can we get rid of distillation?

- For instance, GAN-style training for NAT models to handle multimodality

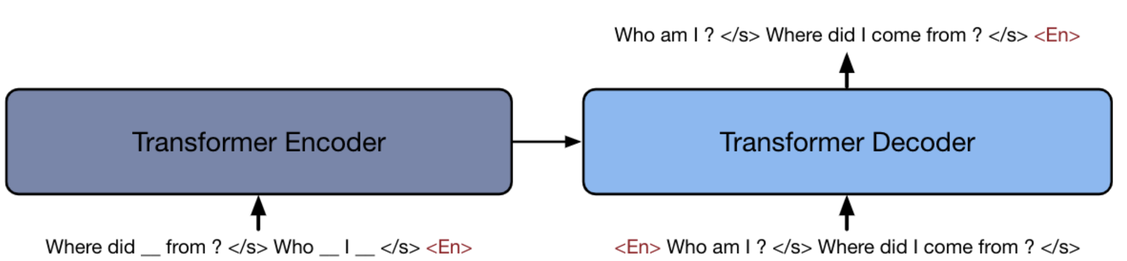
What is the best way to find the noise level for self-training?

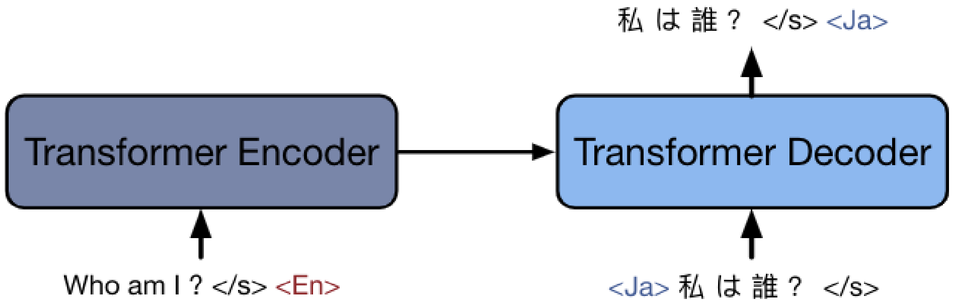
- For instance, can we use meta-learning to learn to inject noise?

# Some unrelated recent works...

## Multilingual Denoising Pre-training for Neural Machine Translation

**Yinhan Liu*, Jiatao Gu*, Naman Goyal*, Xian Li, Sergey Edunov**
**Marjan Ghazvininejad, Mike Lewis, Luke Zettlemoyer**
Facebook AI Research
{yinhanliu,jgu,naman,xianl,edunov
ghazvini,mikelewis,lsz}@fb.com

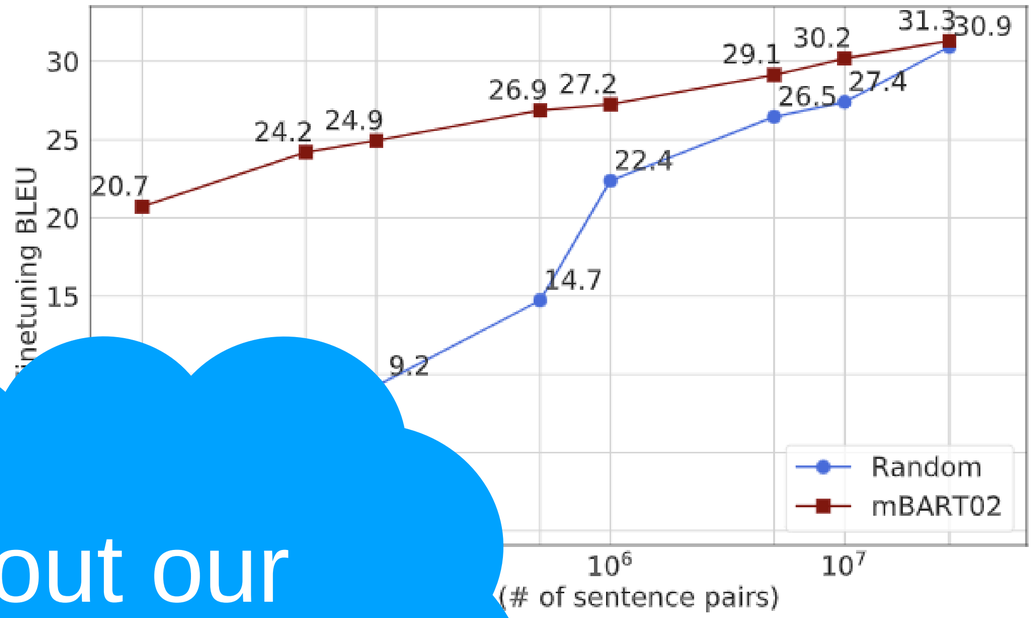


| Languages | En-Gu | | En-Kk | | En-Vi | | En-Tr | | En-Ja | | En-Ko | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Data Source | WMT19 | | WMT19 | | IWSLT15 | | WMT17 | | IWSLT17 | | IWSLT17 | |
| Size | 10K | | 91K | | 133K | | 207K | | 223K | | 230K | |
| Direction | ← | → | ← | → | ← | → | ← | → | ← | → | ← | → |
| Random | 0.0 | 0.0 | 0.8 | 0.2 | 23.6 | 24.8 | 12.2 | 9.5 | 10.4 | 12.3 | 15.3 | 16.3 |
| mBART25 | **0.3** | **0.1** | **7.4** | **2.5** | **36.1** | **35.4** | **22.5** | **17.8** | **19.1** | **19.4** | **24.6** | **22.6** |

| Languages | En-Nl | | En-Ar | | En-It | | En-My | | En-Ne | | En-Ro | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Data Source | IWSLT17 | | IWSLT17 | | IWSLT17 | | WAT19 | | FLoRes | | WMT16 | |
| Size | 237K | | 250K | | 250K | | 259K | | 564K | | 608K | |
| Direction | ← | → | ← | → | ← | → | ← | → | ← | → | ← | → |
| Random | 34.6 | 29.3 | 27.5 | 16.9 | 31.7 | 28.0 | 23.3 | 34.9 | 7.6 | 4.3 | 34.0 | 34.3 |
| mBART25 | **43.3** | **34.8** | **37.6** | **21.6** | **39.8** | **34.0** | **28.3** | **36.9** | **14.5** | **7.4** | | |

| Languages | En-Si | | En-Hi | | En-Et | | En-Lt | |
|---|---|---|---|---|---|---|---|---|
| Data Source | FLoRes | | ITTB | | WMT18 | | WMT19 | |
| Size | 647K | | 1.56M | | 1.94M | | 2.11M | |
| Direction | ← | → | ← | → | ← | → | ← | → |
| Random | 7.2 | 1.2 | 10.9 | 14.2 | 22.6 | 17.9 | 18.1 | |
| mBART25 | **13.7** | **3.3** | **23.5** | **20.8** | **27.8** | **21.4** | **22.4** | |




Check out our paper tomorrow morning!!!

| Pre-training | | Fine-tuning | | |
|---|---|---|---|---|
| Model | Data | En→Ro | Ro→En | +BT |
| **Random** | None | 34.3 | 34.0 | 36.8 |
| **XLM (2019)** | En Ro | - | 35.6 | 38.5 |
| **MASS (2019)** | En Ro | - | - | 39.1 |
| **BART (2019)** | En | - | - | 38.0 |
| **XLM-R (2019)** | CC100 | 35.6 | 35.8 | - |
| **BART-En** | En | 36.0 | 35.8 | 37.4 |
| **BART-Ro** | Ro | 37.6 | 36.8 | 38.1 |
| **mBART02** | En Ro | **38.5** | **38.5** | **39.9** |
| **mBART25** | CC25 | 37.7 | 37.8 | 38.8 |

| Model | Similar Pairs | | | | Dissimilar Pairs | | | |
|---|---|---|---|---|---|---|---|---|
| | En-De | | En-Ro | | En-Ne | | En-Si | |
| | ← | → | ← | → | ← | → | ← | → |
| **Random** | 21.0 | 17.2 | 19.4 | 21.2 | 0.0 | 0.0 | 0.0 | 0.0 |
| **XLM (2019)** | 34.3 | 26.4 | 31.8 | 33.3 | 0.5 | 0.1 | 0.1 | 0.1 |
| **MASS (2019)** | **35.2** | 28.3 | **33.1** | **35.2** | - | - | - | - |
| **mBART** | 34.0 | **29.8** | 30.5 | 35.0 | **10.0** | **4.4** | **8.2** | **3.9** |

| | ng Languages | | | | | | |
|---|---|---|---|---|---|---|---|
| | It | Ar | Hi | Ne | Si | Gu | |
| | TED | TED | News | Wiki | Wiki | Wiki | |
| | 7.0 | 6.8 | 6.2 | 7.2 | 4.2 | 5.9 | 0.0 |
| | 6.4 | 5.1 | 5.6 | 4.7 | 4.2 | 6.5 | 0.0 |
| | 9.5 | 9.1 | 8.7 | 9.6 | 8.8 | 11.1 | 0.0 |
| | 17.0 | 16.7 | 16.9 | 13.2 | 15.1 | 16.4 | 0.0 |
| | 22.3 | 21.6 | 22.6 | 16.4 | 18.5 | 22.1 | 0.0 |
| | 43.3 | **34.1** | 31.0 | 24.6 | 23.3 | 27.3 | 0.0 |
| | 39.8 | **30.6** | 20.1 | 18.5 | 23.2 | 0.0 | |
| | 14.7 | 37.6 | 11.6 | 13.0 | 16.7 | 0.0 | |
| | 6.1 | 5.0 | 7.6 | 23.5 | **14.5** | 13.0 | 0.0 |
| | 3.0 | 2.2 | 5.2 | **17.9** | 14.5 | 10.8 | 0.0 |
| | 0.9 | 0.5 | 3.5 | 8.1 | **8.9** | 13.7 | 0.0 |
| | 2.1 | 0.0 | 6.2 | **13.8** | 13.5 | 12.8 | 0.3 |

facebook
Artificial Intelligence Research

# Thank you!

We are also hiring Research Interns, AI Residents and Full-time Researchers at FAIR!
Let me know if you are interested!
jgu@fb.com

facebook
Artificial Intelligence Research