Knowledge Distillation in NLP

Valentin Malykh

A few words about me

- 9 years experience in NLP
- Worked in Yandex, VK
- Defended PhD
- Now I work in Huawei Noah’s Ark lab

Agenda

- Introduction to Knowledge Distillation
- KD in Non-Autoregressive Machine Translation
- TinyBERT
- Coding Session

Sequence-Level Knowledge Distillation

facebook
Artificial Intelligence Research

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)

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

Knowledge Distillation

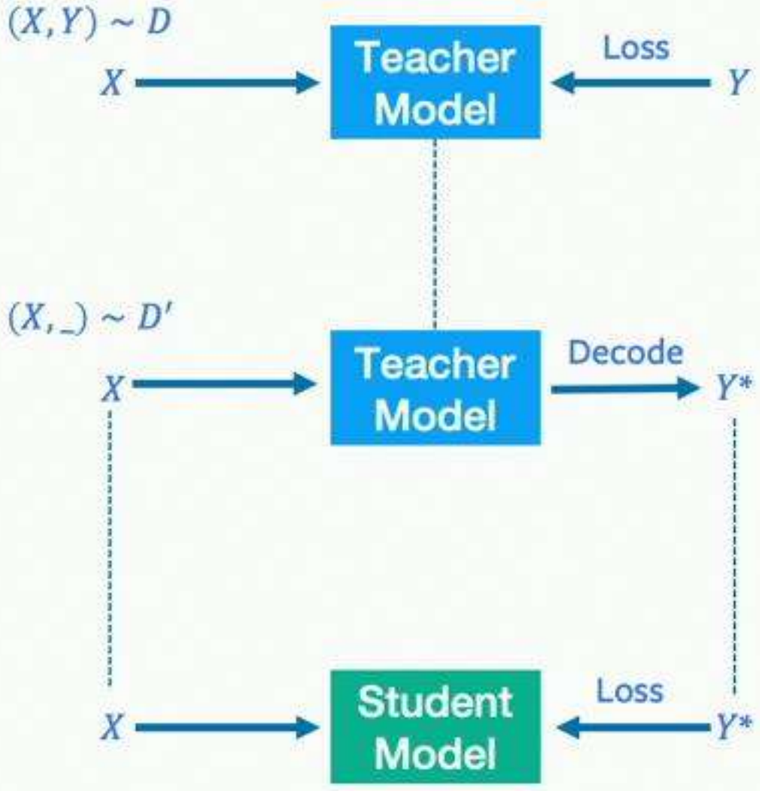
- Knowledge distillation (Liang et al., 2008; Hinton et al., 2015) was originally proposed for training a weaker student classifier on the targets predicted from a stronger teacher model.
- A typical approach is using the label probabilities produced by the teacher as "**soft targets**" (dark knowledge)

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

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

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

Sequence-level Knowledge Distillation



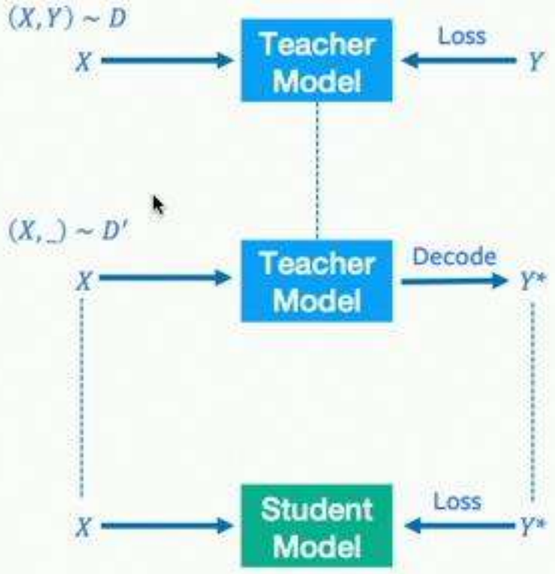
A Teacher-Student Framework in Three Steps:

(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

Submitted to ICLR2020

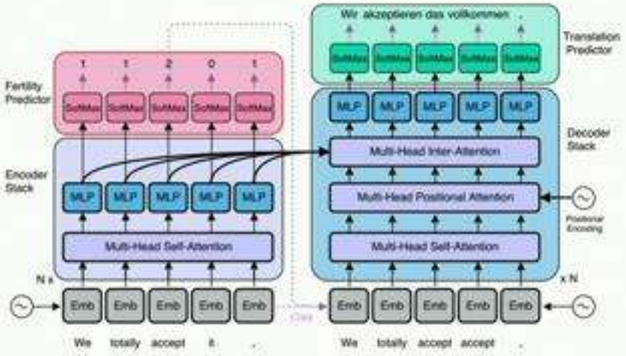
facebook
Artificial Intelligence Research

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.



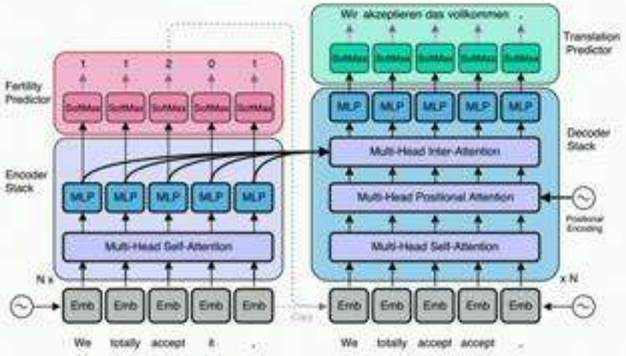
(Figure from Gu et.al, 2017)

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.



(Figure from Gu et.al, 2017)

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.

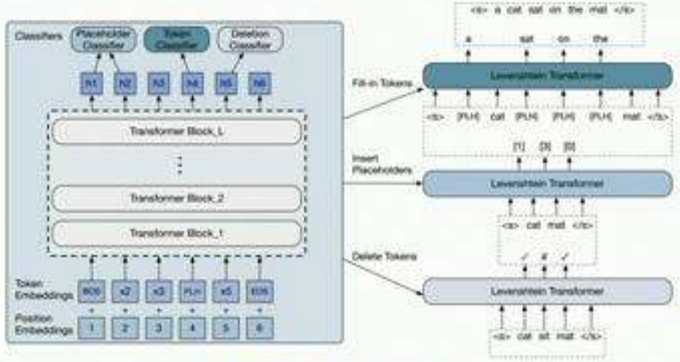
$$P(Y|X) = \prod_{t=1}^{T} P(y_t|\, x_{1:T'})$$

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,

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

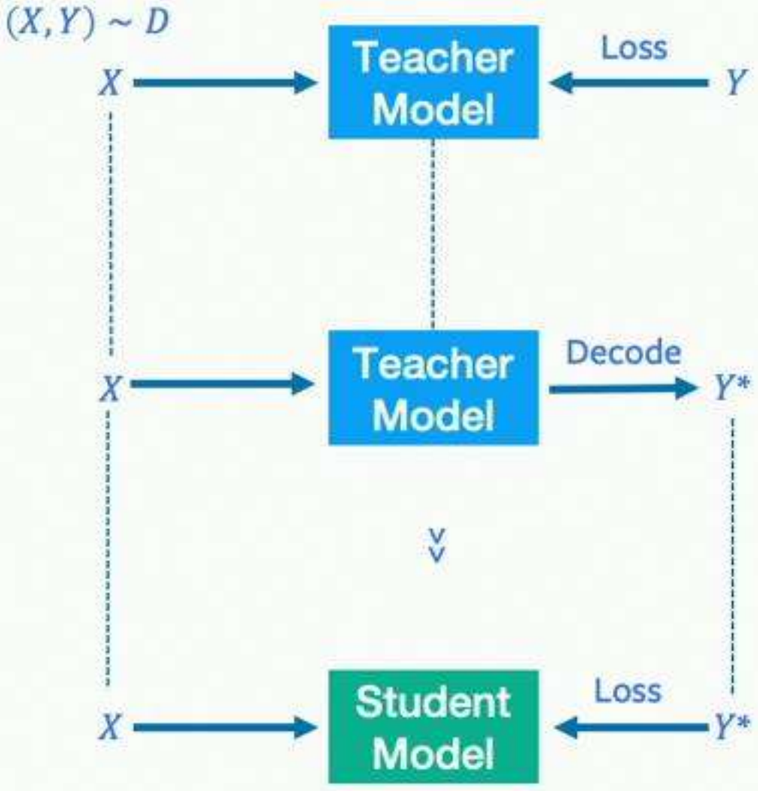


(Figure from Gu et.al, 2019)

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
  - ...

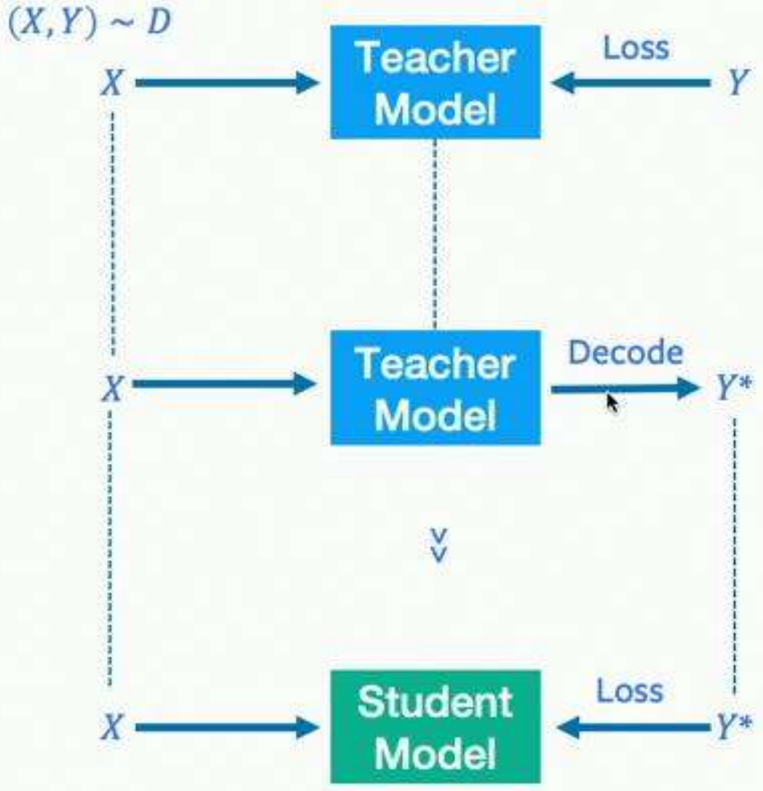
Knowledge Distillation for NAT



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.

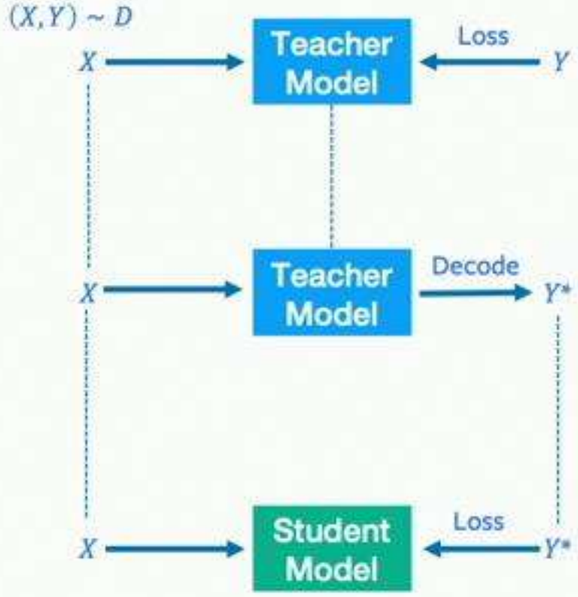
	w/o distillation	w/ distillation
Vanilla NAT (Gu et al, 2017)	11.4	19.5 (+8.1)
FlowSeq (Ma et al, 2019)	18.6	21.7 (+3.1)
LevT (Gu et al, 2019)	25.2	26.9 (+1.7)

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 ✗

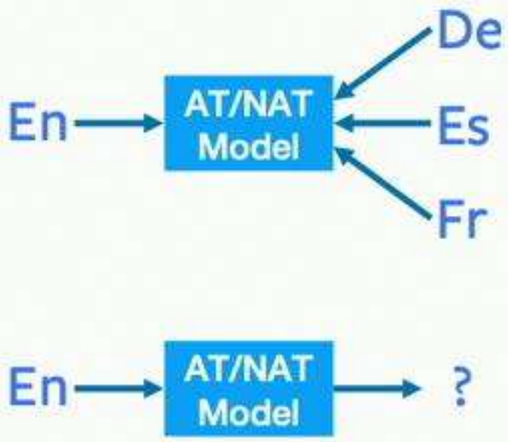
Our assumption is that distillation helps to reduce the multimodality in the data.

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.

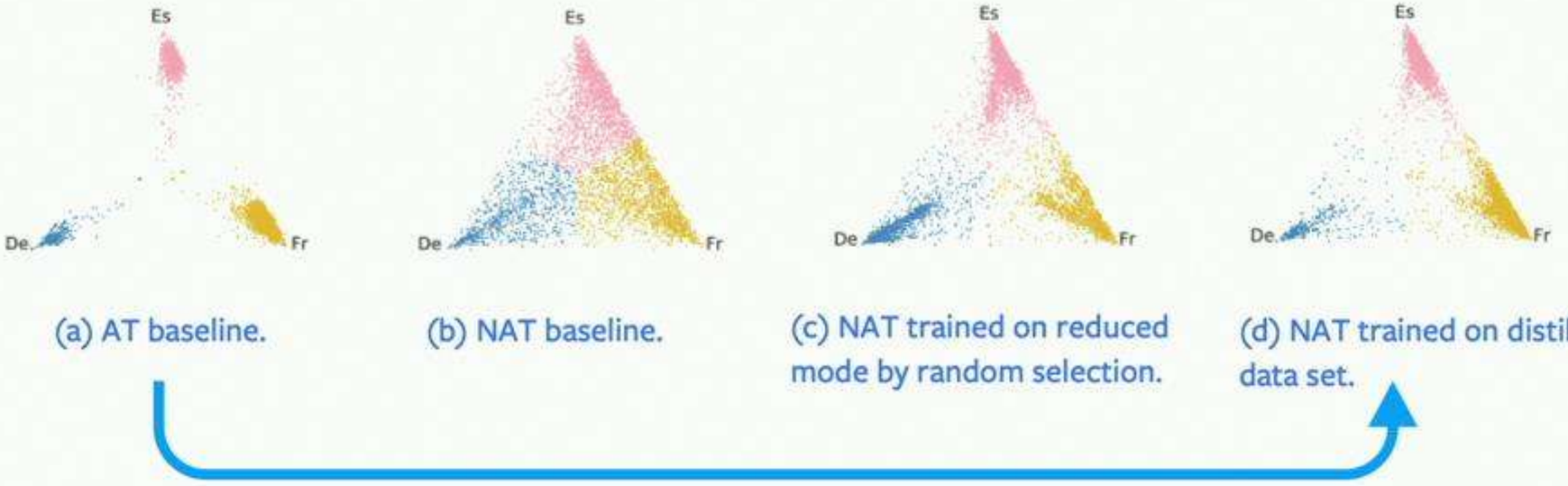
- We create a synthetic dataset compared with three language pairs -- English-German (En-De), English-French (En-Fr) and English-Spanish (En-Es) – from the Europarl corpus. We make sure every English sentence will be aligned to ALL three languages, and no language ID was specified.

- 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.

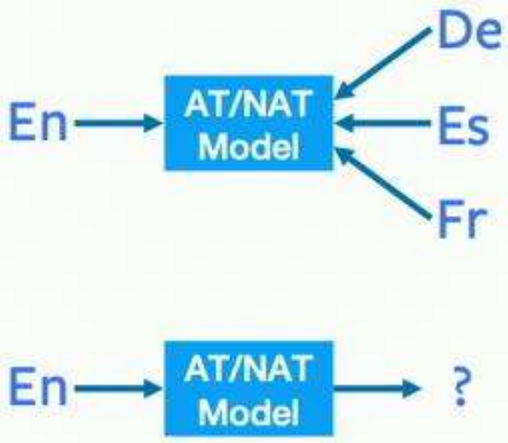


We manually created the multi-modality (language id) in the data.

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:

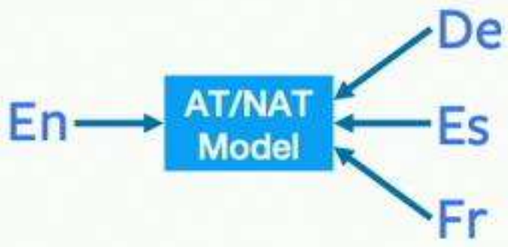
$$p(l_i|\boldsymbol{y}) \approx \frac{1}{T}\sum_{t=1}^{T} p(l_i|y_t) = \frac{1}{T}\sum_{t=1}^{T} \frac{p(y_t|l_i)p(l_i)}{\sum_k p(y_t|l_k)p(l_k)}$$

- 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

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.

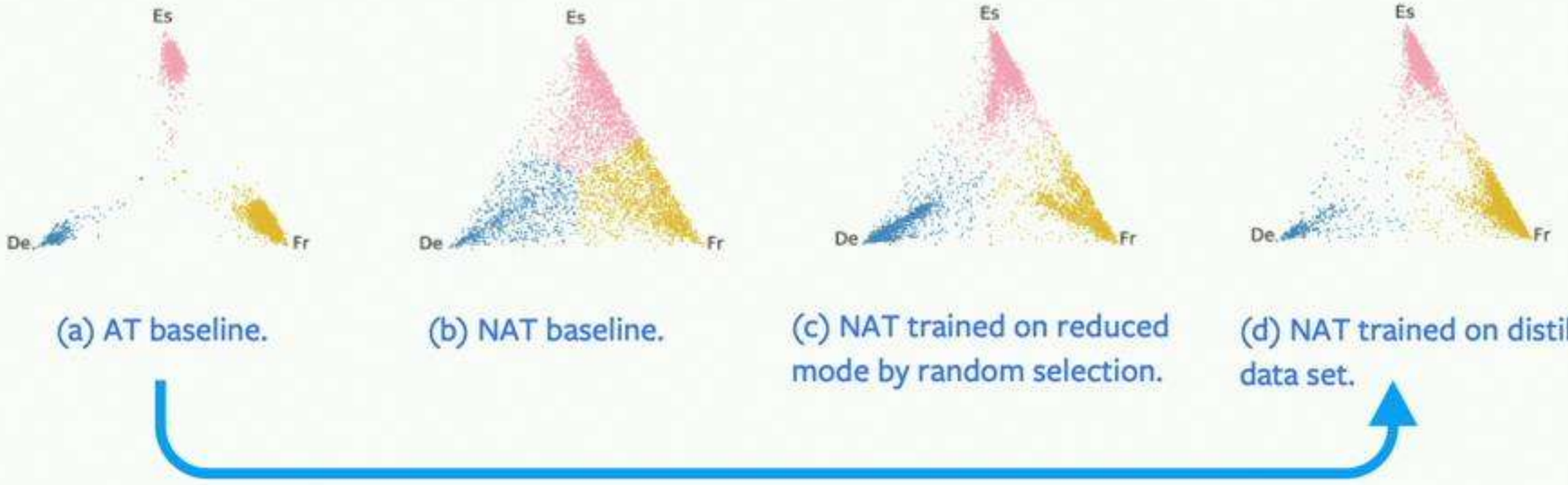
- We create a synthetic dataset compared with three language pairs -- English-German (En-De), English-French (En-Fr) and English-Spanish (En-Es) – from the Europarl corpus. We make sure every English sentence will be aligned to ALL three languages, and no language ID was specified.



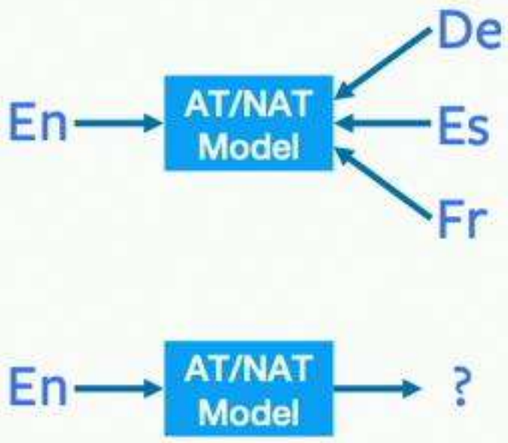
- We train both AT and NAT models directly on this synthetic dataset. During inference time, we input the English sentence without telling the model which language to be output.

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.

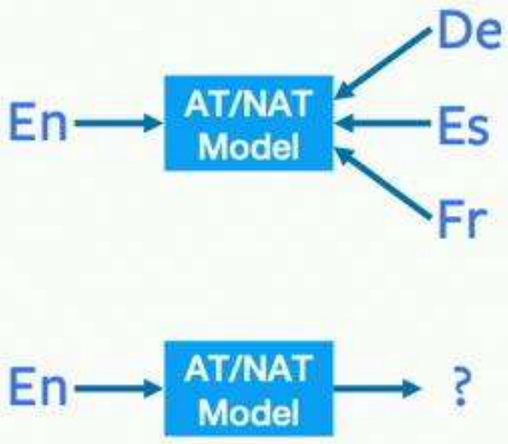


We visualize the mode of "language ID" from the decoded outputs by a simple approximation:

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

- 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**.

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

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

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

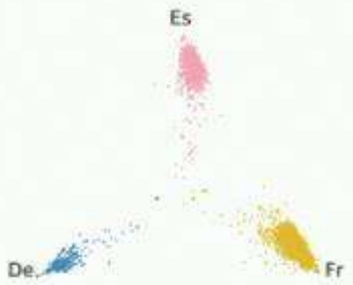
$$= \sum_{t=1}^{T_x} \mathcal{H}(y|x=x_t)$$

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



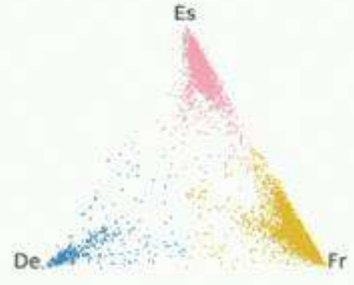
(a) AT baseline.

Complexity ($C(d)$): 3.67

(b) NAT baseline.

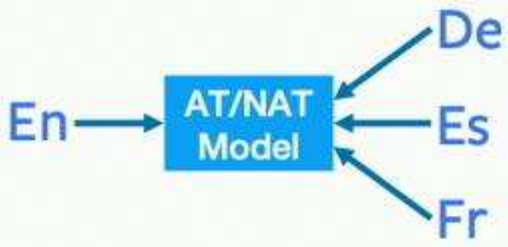
(c) NAT trained on reduced mode by random selection.

Complexity ($C(d)$): 3.30



(d) NAT trained on distilled data set.

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.

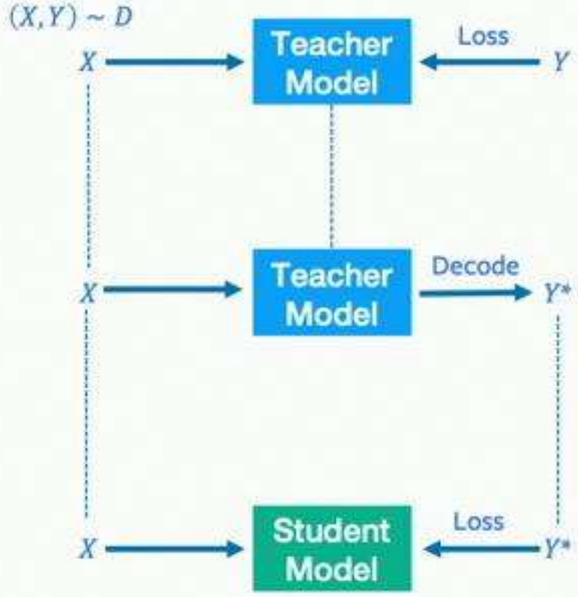
- We compute the **KL-divergence** of the alignment models between the real (r) and the distilled dataset (d)

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

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):

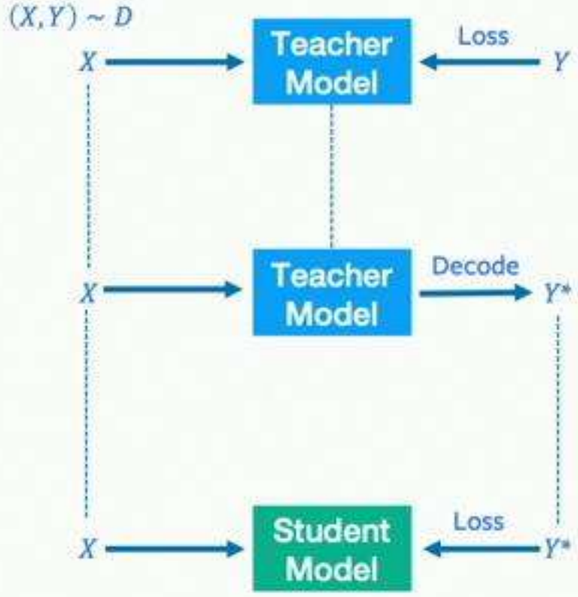


weak ↓ strong

weak ↓ strong

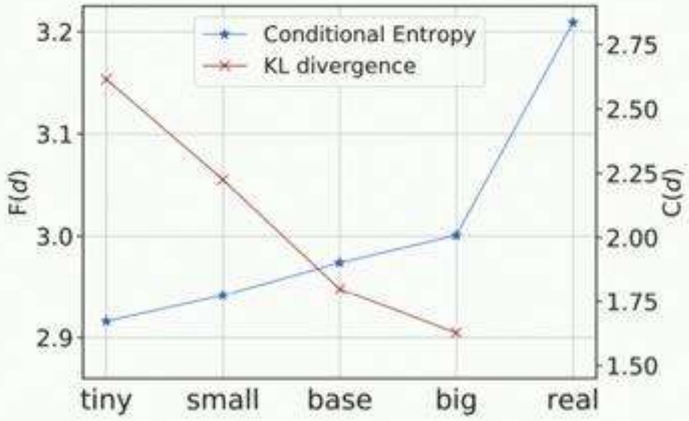
Models	Params	BLEU	Pass	Iters
AT models				
AT-tiny	16M	23.3	–	n
AT-small	37M	25.6	–	n
AT-base	65M	27.1	–	n
AT-big	218M	28.2	–	n
NAT models				
vanilla	71M	11.4	1	1
FlowSeq	73M	18.6	13	1
iNAT	66M	19.3	1	$k \ll n$
InsT	66M	20.9	1	$\approx \log_2 n$
MaskT	66M	23.5	1	10
LevT	66M	25.2	1	$3k \ll n$
LevT-big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

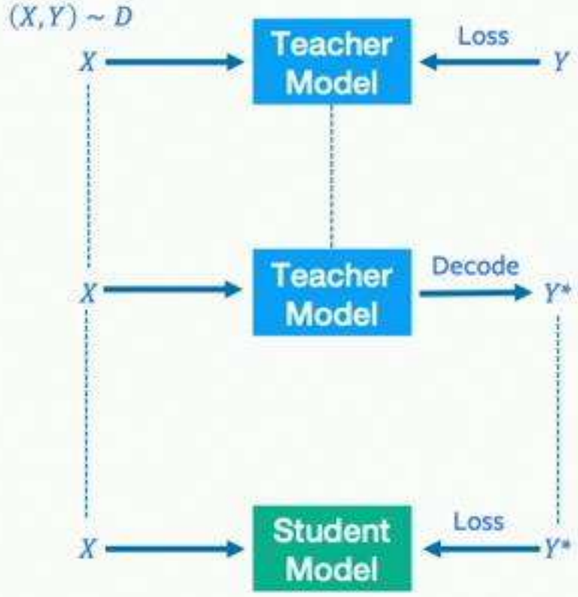


- As additional supporting metrics, we also plot the BLEU score (compared to the real data), showing it also correlates the data quality well.

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!

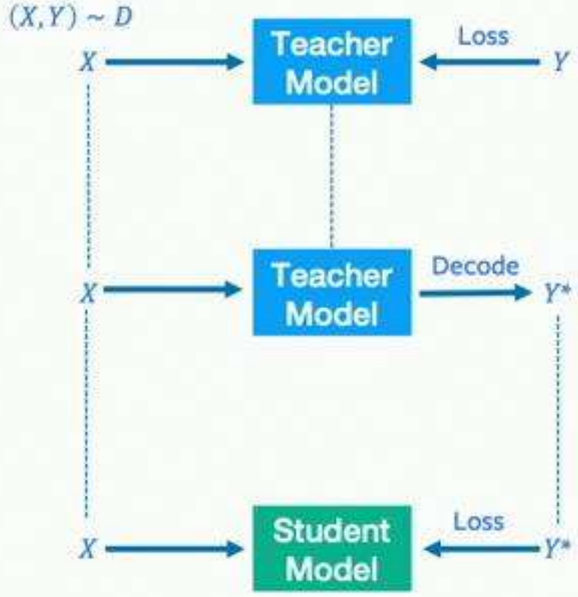
Source	For more than 30 years , Josef Winkler has been writing from the heart , telling of the hardships of his childhood and youth .
Distilled Target	Seit mehr als 30 Jahren schreibt Josef Winkler aus dem Herzen und erzählt von der Not seiner Kindheit und Jugend .
Real Target	Josef Winkler schreibt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

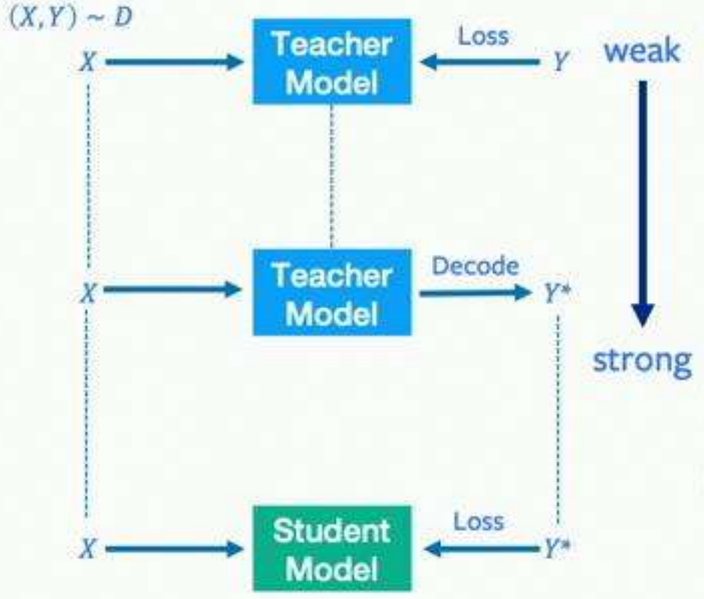


Decoding Method	$C(d)$	$F(d)$	BLEU
sampling	3.623	3.354	6.6
sampling (Top 10)	2.411	2.932	14.6
greedy	1.960	2.959	18.9
beam search	1.902	2.948	**19.5**

Experiments

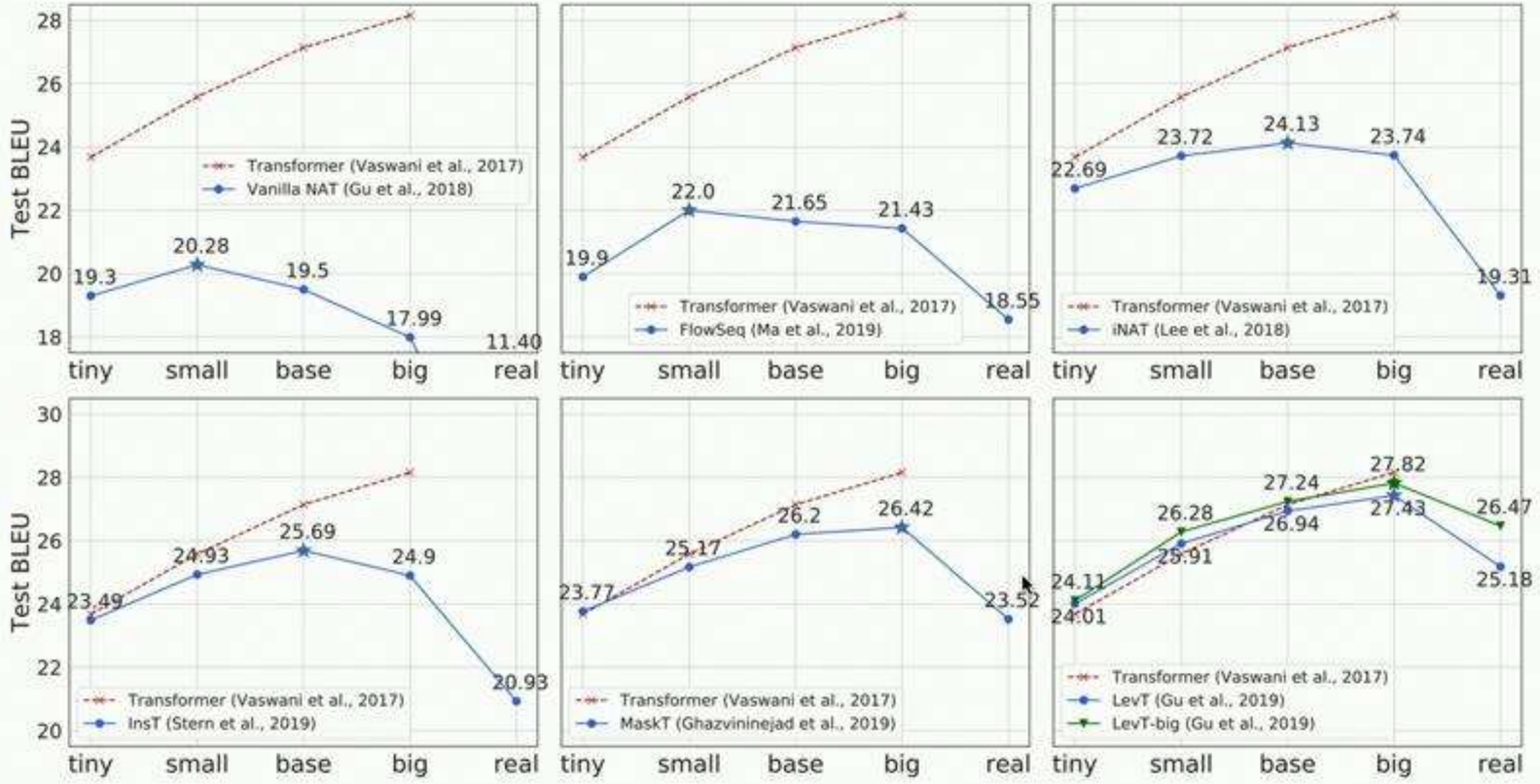
Analysis of the NAT models

- 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.



weak

strong

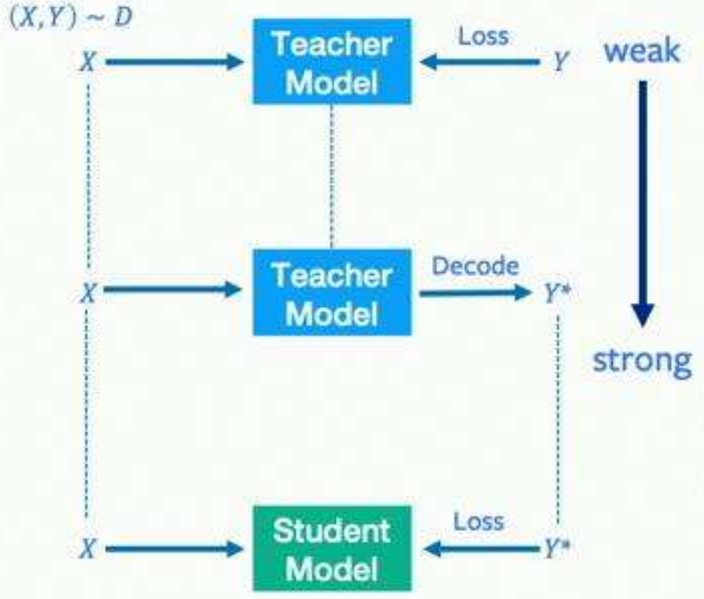


weak → strong

Experiments

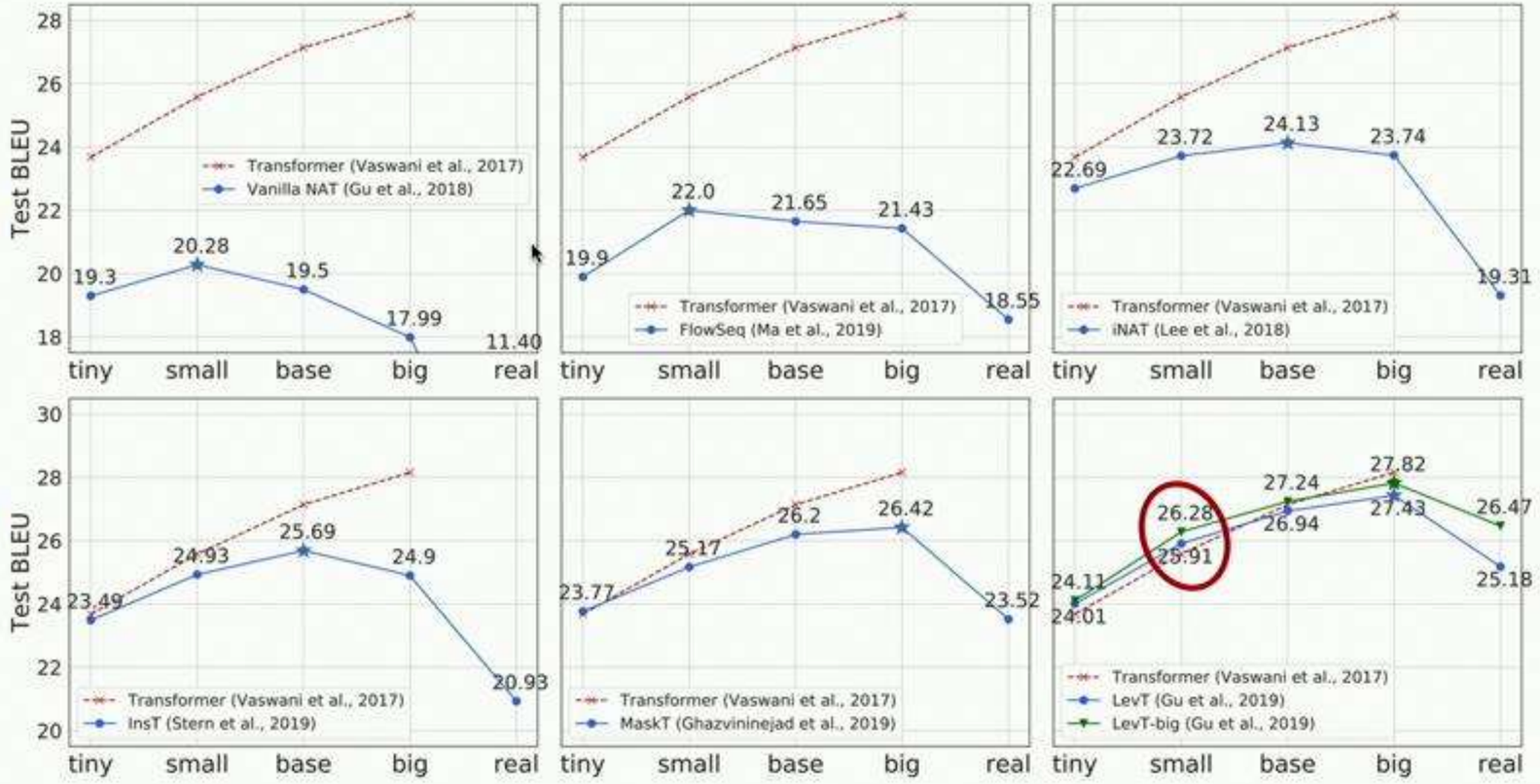
Analysis of the NAT models

- 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

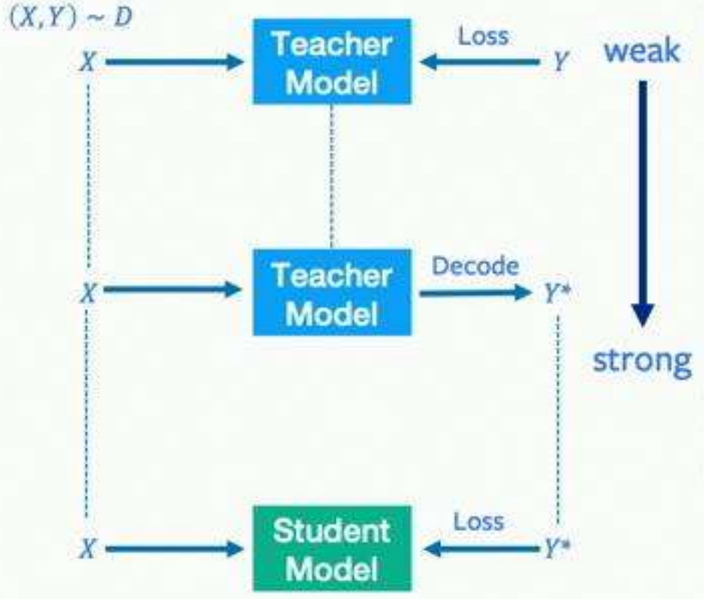


weak → strong

Experiments

Analysis of the NAT models

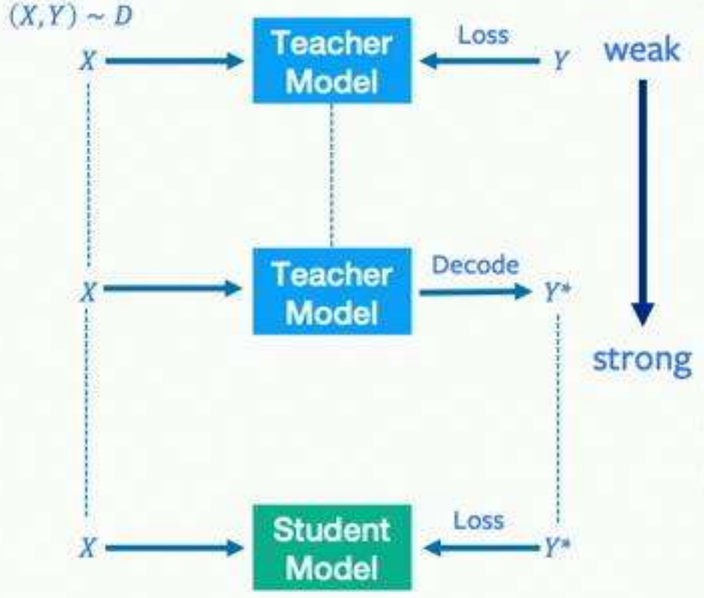
- All NAT performance curves give the same pattern: when increasing the capacity of the teacher model, distillation results first improve and then drop.
- The best performance of NAT models – from lower capacity ones to higher capacity ones – is achieved with distilled data of lower complexity to higher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

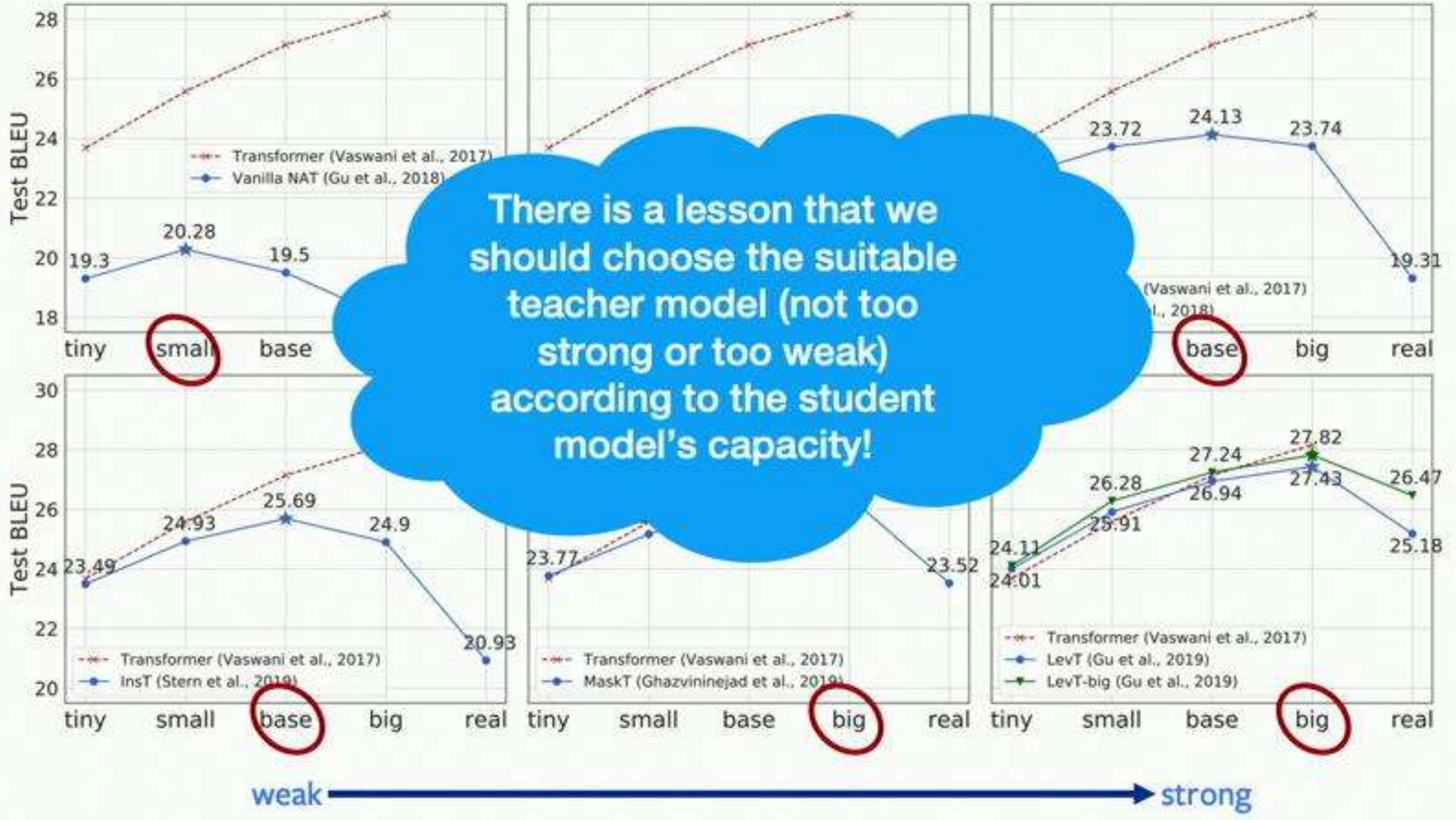


Experiments

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





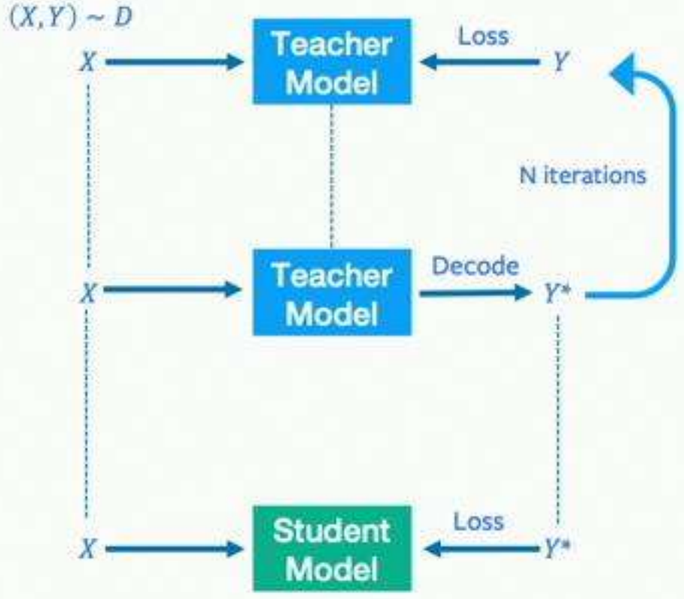
Experiments

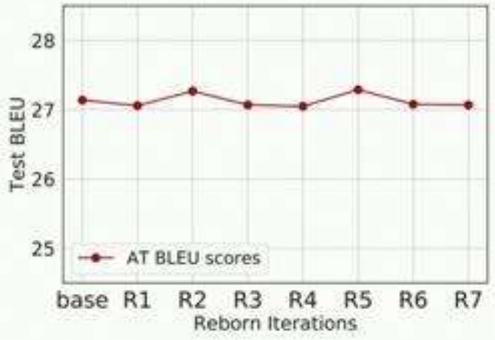
Improvements for WEAK student models

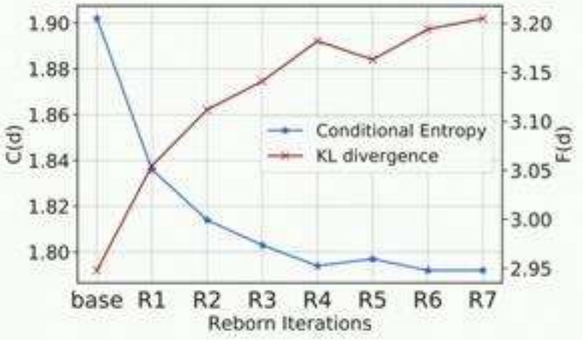
- Take the vanilla NAT model as an example.

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 simple solution: it repeatedly distill the teacher model by its own output for multiple iterations and use the final output to train the student model.



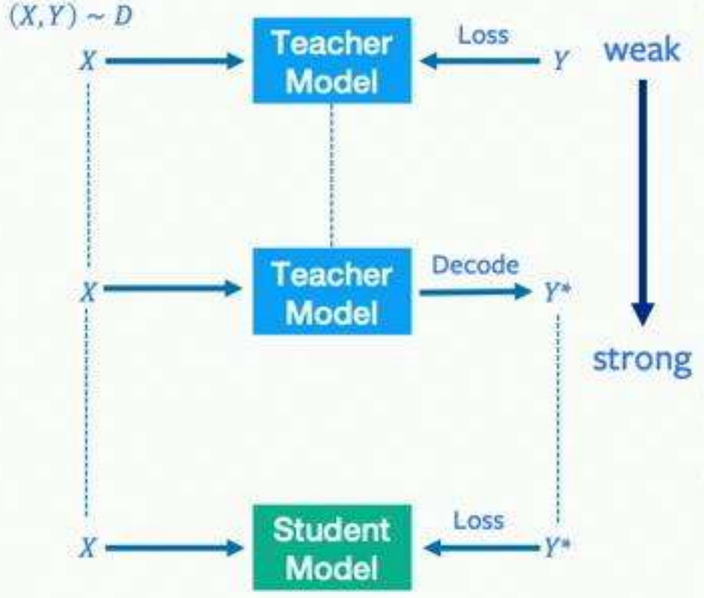


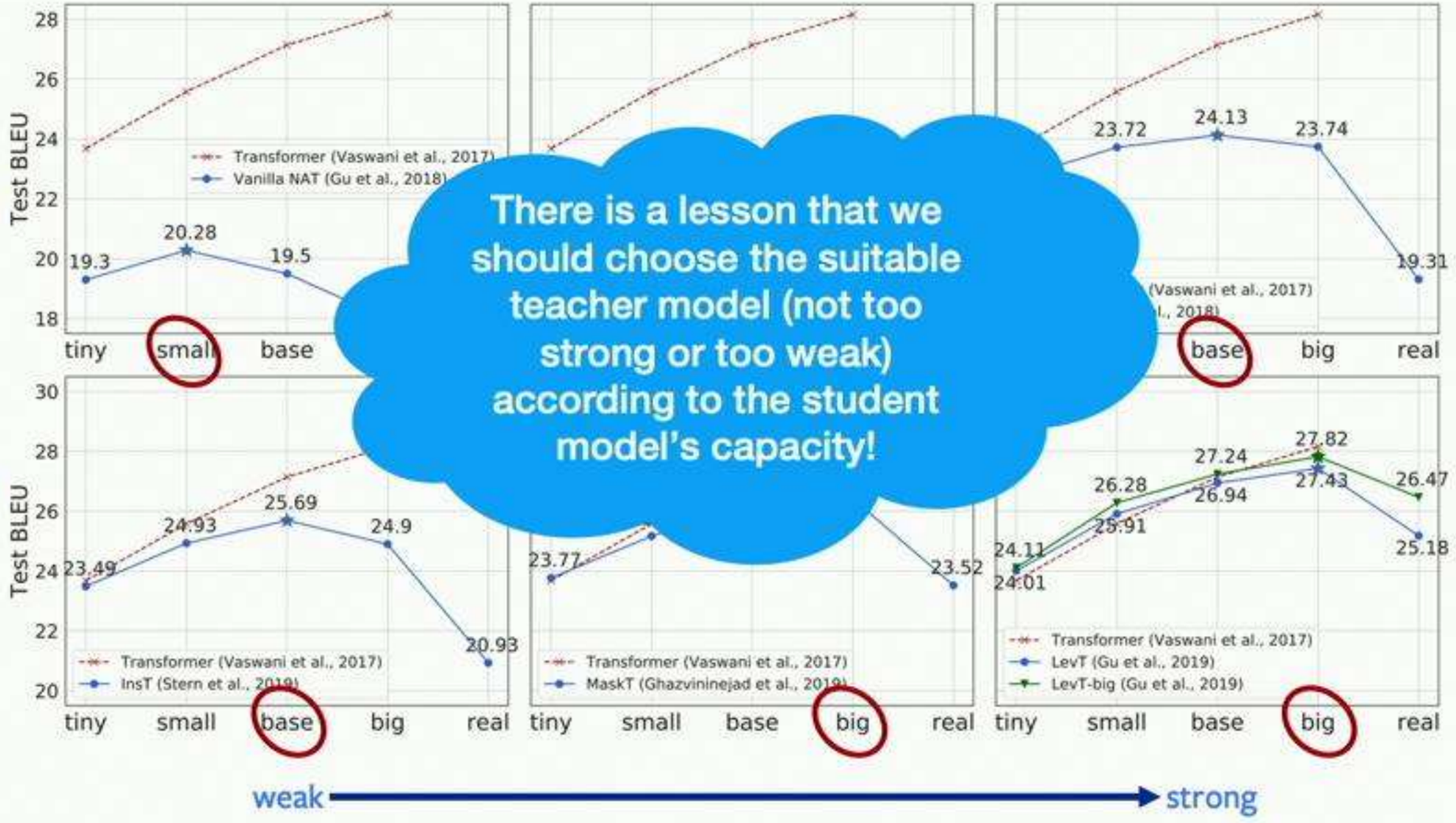


Experiments

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

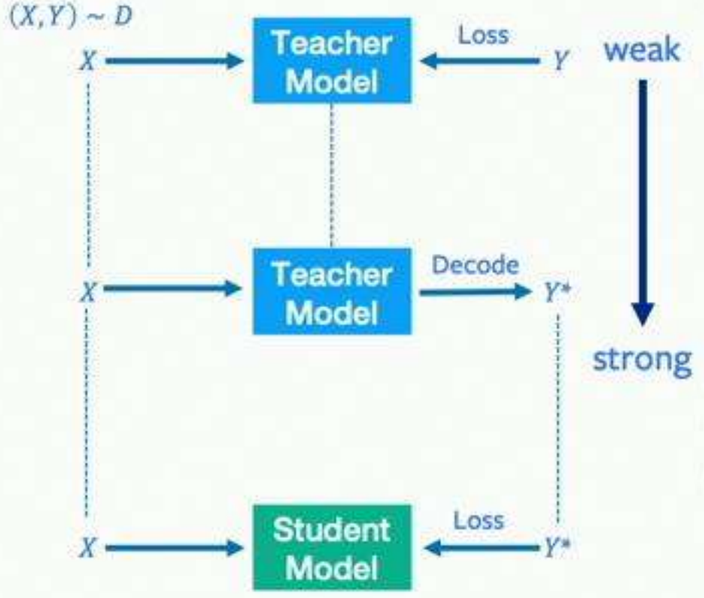




Experiments

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



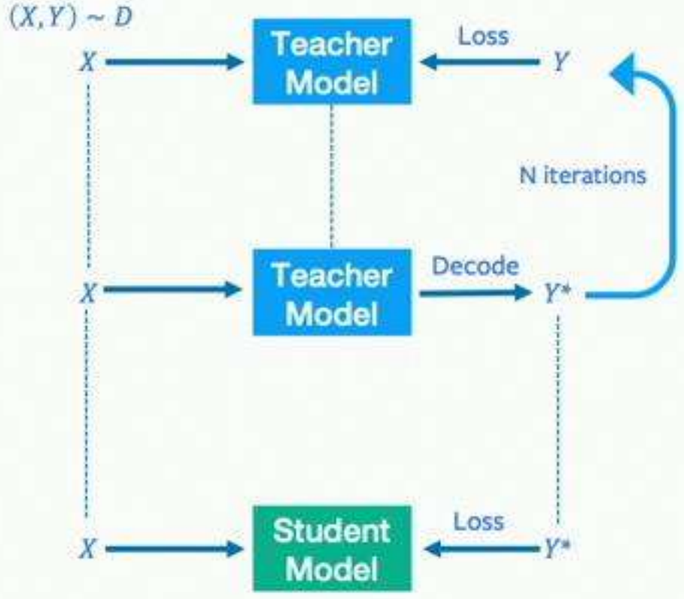
Experiments

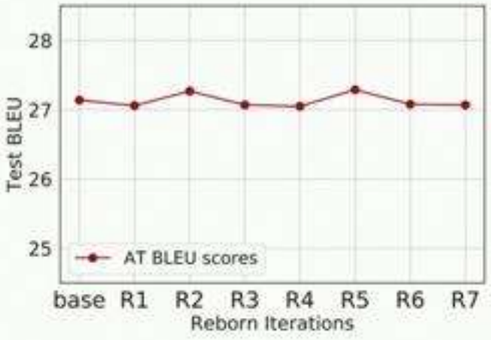
Improvements for WEAK student models

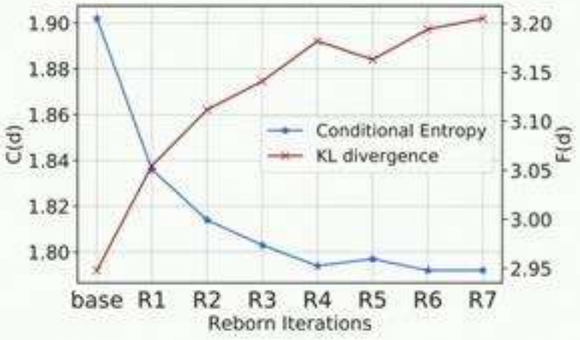
- Take the vanilla NAT model as an example.

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 simple solution: it repeatedly distill the teacher model by its own output for multiple iterations and use the final output to train the student model.







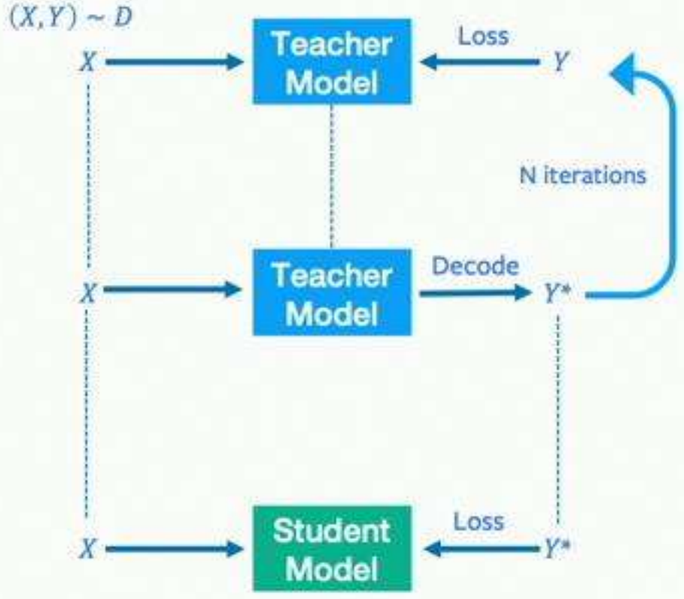
Experiments

Improvements for WEAK student models

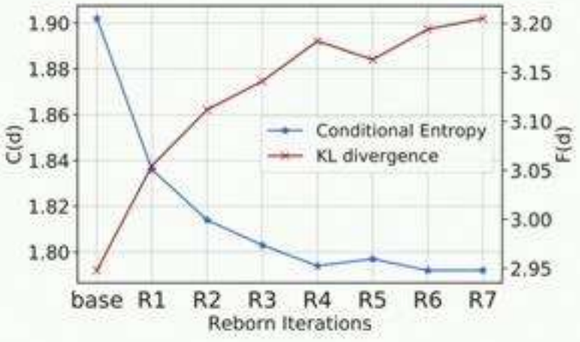
- Take the vanilla NAT model as an example.

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… 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.



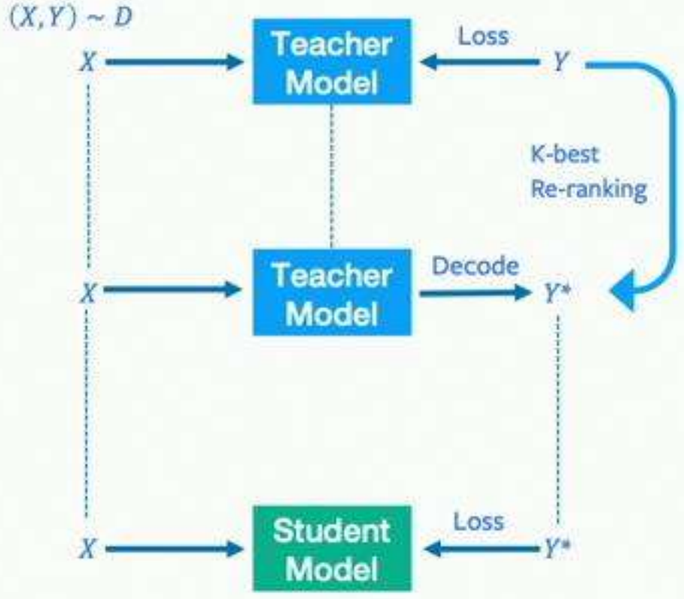
Experiments

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

TinyBERT: Distilling BERT For Natural Language Understanding review

TinyBERT is an under review paper, and it is a paper from Huawei's Noah's Ark Lab. The code is on GitHub huawei-noah/Pretrained-Language-Model/TinyBERT . The arxiv link is https://arxiv.org/abs/1909.10351 .

This paper applies KD for inference acceleration and model size compression, and attempts distillation from the teacher BERT model to the student BERT model. This model is 7.5x smaller and 9.4x faster in inference, while preserving more than 96% performance.

Introduction

- Usually Compression tries quantization, pruning, knowledge distillation, etc.
- Among them, TinyBERT is focused on KD, and the authors paid much attention to how to design the loss.
- Use the following representations to find the loss
  - output of embedding layer
  - hidden states and attention matrices
  - logit output of prediction layer
- main contributions
  - Newly proposed transformer distillation method
  - Using a two-stage distillation method
  - 96% performance of the teacher model is preserved.

Transformer Distillation

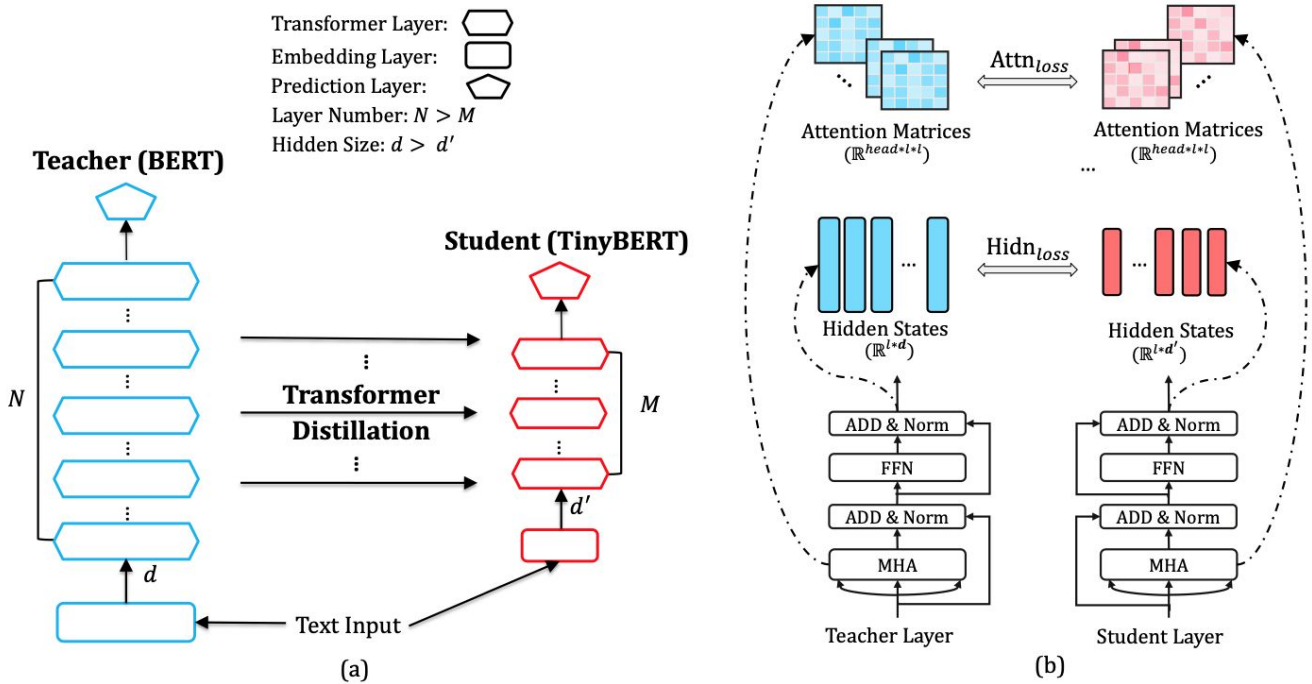


Figure 1: An overview of Transformer distillation: (a) the framework of Transformer distillation, (b) the details of Transformer-layer distillation consisting of Attn$_{loss}$(attention based distillation) and Hidn$_{loss}$(hidden states based distillation).

Problem Formulation

- Teacher has N Transformer Layers
- Student has M Transformer Layers
- The output of the embedding layer is regarded as layer 0.
- Distillation goes in the direction of minimizing the equation below.

$$l_{model} = \sum_{m=0}^{M+\mathrm{One}} \lambda_m l_{layer}(S_m, T_{g(m)})$$

Transformer Layer Distillation

- Loss for attention mechanism

$$l_{attn} = \frac{\mathrm{One}}{h} sum_{i=\mathrm{One}}^{h} \mathrm{MSE}(\mathbf{A}_i^S, \mathbf{A}_i^T)$$

- The peculiar thing is that the matrix before softmax is put as the input of the loss function, which is said to be faster to converge.
  - If you would like to comment on this point, I think it is because a lot of information about low attention values is lost after softmax is given.
- Loss for hidden states

$$l_{hidn} = \mathrm{MSE}(\mathbf{H}^S \cdot \mathbf{W}_h, \mathbf{H}^T)$$

Other Distillation

Embedding Layer Distillation

- Loss for embedding layer

$$l_{embd} = \mathrm{MSE}(\mathbf{E}^S \cdot \mathbf{W}_e, \mathbf{E}^T)$$

Prediction Layer Distillation

- Loss for Prediction Layer

$$l_{pred} = -\mathrm{softmax}(z^T) \cdot \mathrm{log\_softmax}(\frac{Z^S}{t})$$

- In the prediction layer, a penalty was given to the logit of the student model, but in the actual experiment, t = 1 was the best.

Training Process

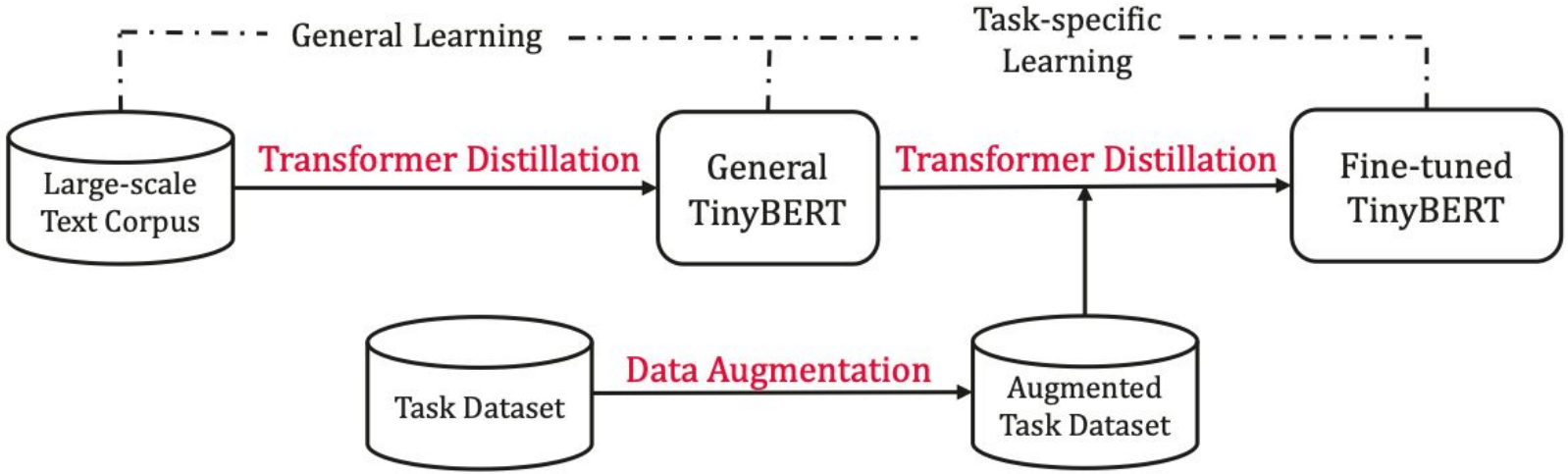


Figure 2: The illustration of TinyBERT learning

Training Process (cont.)

- Try general-distillation and task specific distillation
- General distillation is a stage that proceeds excluding Prediction Layer Distillation for the original Bert model that is not fine-tuned.
- For task specific distillation, distillation is performed by applying data augmentation.

TinyBERT Results

Table 2: Results are evaluated on the test set of GLUE official benchmark. All models are learned in a single-task manner. "-" means the result is not reported.

System	MNLI-m	MNLI-mm	QQP	SST-2	QNLI	MRPC	RTE	CoLA	STS-B	Average
$\text{BERT}_{\text{BASE}}$ (Google)	84.6	83.4	71.2	93.5	90.5	88.9	66.4	52.1	85.8	79.6
$\text{BERT}_{\text{BASE}}$ (Teacher)	83.9	83.4	71.1	93.4	90.9	87.5	67.0	52.8	85.2	79.5
$\text{BERT}_{\text{SMALL}}$	75.4	74.9	66.5	87.6	84.8	83.2	62.6	19.5	77.1	70.2
Distilled $\text{BiLSTM}_{\text{SOFT}}$	73.0	72.6	68.2	90.7	-	-	-	-	-	-
BERT-PKD	79.9	79.3	70.2	89.4	85.1	82.6	62.3	24.8	79.8	72.6
DistilBERT	78.9	78.0	68.5	91.4	85.2	82.4	54.1	32.8	76.1	71.9
TinyBERT	82.5	81.8	71.3	92.6	87.7	86.4	62.9	43.3	79.9	76.5

Table 3: The model sizes and inference time for baselines and TinyBERT. The number of layers does not include the embedding and prediction layers.

System	Layers	Hidden Size	Feed-forward Size	Model Size	Inference Time
$\text{BERT}_{\text{BASE}}$ (Teacher)	12	768	3072	109M($\times$1.0)	188s($\times$1.0)
Distilled $\text{BiLSTM}_{\text{SOFT}}$	1	300	400	10.1M($\times$10.8)	24.8s($\times$7.6)
BERT-PKD/DistilBERT	4	768	3072	52.2M($\times$2.1)	63.7s($\times$3.0)
TinyBERT/$\text{BERT}_{\text{SMALL}}$	4	312	1200	14.5M($\times$7.5)	19.9s($\times$9.4)

Ablation Study

Table 5: Ablation studies of different procedures (i.e., TD, GD, and DA) of the two-stage learning framework. The variants are validated on the dev set.

System	MNLI-m	MNLI-mm	MRPC	CoLA	Average
TinyBERT	82.8	82.9	85.8	49.7	75.3
No GD	82.5	82.6	84.1	40.8	72.5
No TD	80.6	81.2	83.8	28.5	68.5
No DA	80.5	81.0	82.4	29.8	68.4

Table 6: Ablation studies of different distillation objectives in the TinyBERT learning. The variants are validated on the dev set.

System	MNLI-m	MNLI-mm	MRPC	CoLA	Average
TinyBERT	82.8	82.9	85.8	49.7	75.3
No Embd	82.3	82.3	85.0	46.7	74.1
No Pred	80.5	81.0	84.3	48.2	73.5
No Trm	71.7	72.3	70.1	11.2	56.3
No Attn	79.9	80.7	82.3	41.1	71.0
No Hidn	81.7	82.1	84.1	43.7	72.9

Table 7: Results (dev) of different mapping strategies.

System	MNLI-m	MNLI-mm	MRPC	CoLA	Average
TinyBERT (Uniform-strategy)	82.8	82.9	85.8	49.7	75.3
TinyBERT (Top-strategy)	81.7	82.3	83.6	35.9	70.9
TinyBERT (Bottom-strategy)	80.6	81.3	84.6	38.5	71.3

Thank you for your attention!

@madrugado