Non-Autoregressive Neural Machine Translation

邵晨泽

2019.12.17

https://github.com/ictnlp/BoN-NAT/blob/master/slides.pdf

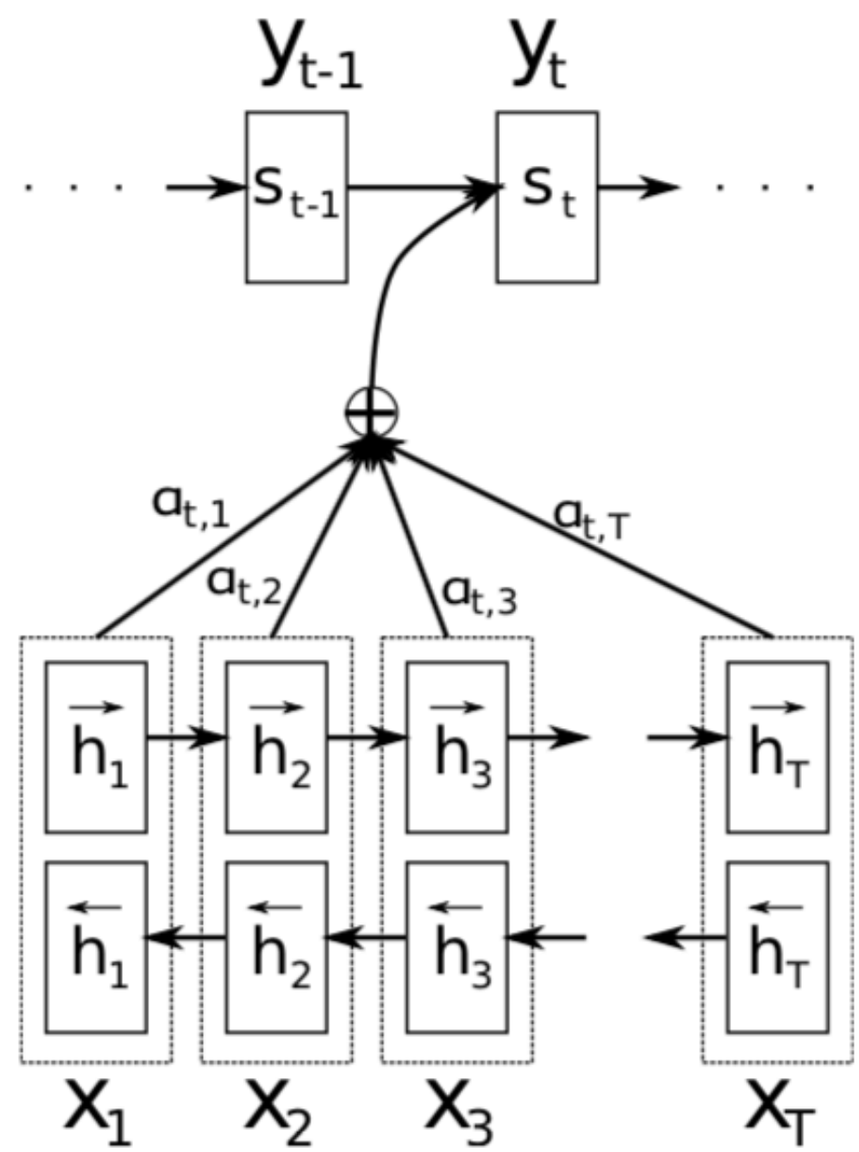
中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES

Outline

- **Introduction**
- Non-Autoregressive Transformer (NAT)
- Progress in NAT
  - Sequence-Level Training
  - Latent Variable
  - Iterative Decoding
  - Knowledge Distillation

Autoregressive Mechanism

- Sequence prediction
- Autoregressive Mechanism: word-by-word generation
- Models: RNN, CNN, Transformer, etc.

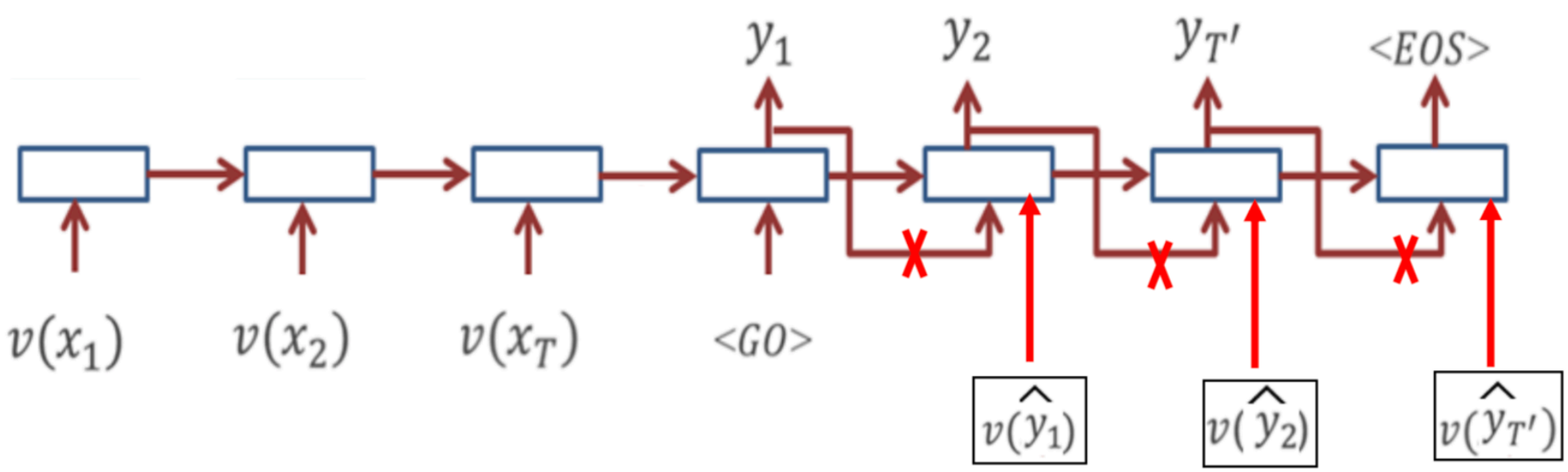


Autoregressive Mechanism

- Training: teacher forcing, the golden target word is fed to the next step.

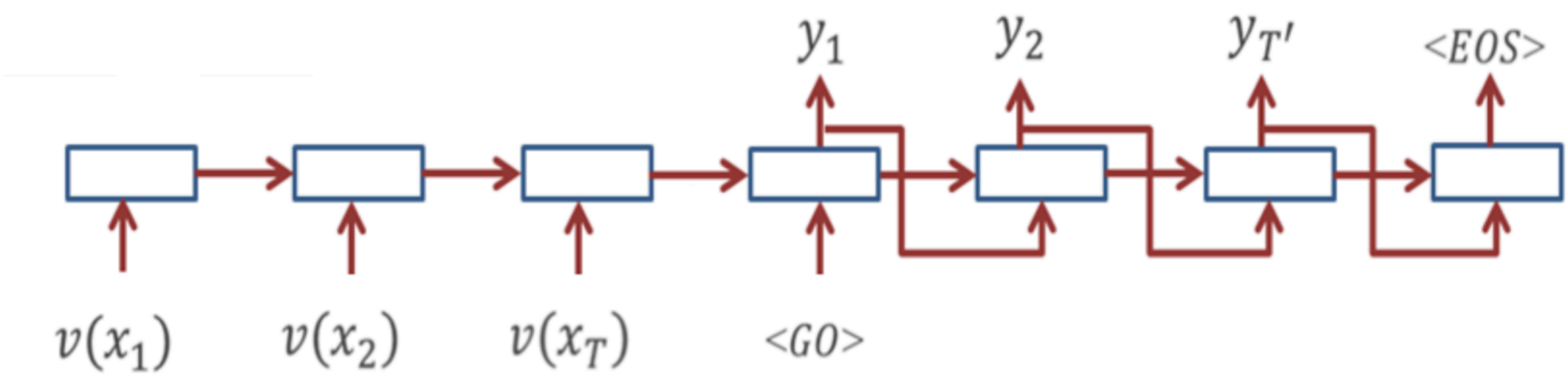
- Loss : word-level cross-entropy loss.

$$\mathcal{L}(\theta) = \sum_{m=1}^{M} \sum_{t=1}^{T} \log(p(y_t^m | \boldsymbol{y}_{<t}^m, \boldsymbol{X}^m, \theta)),$$



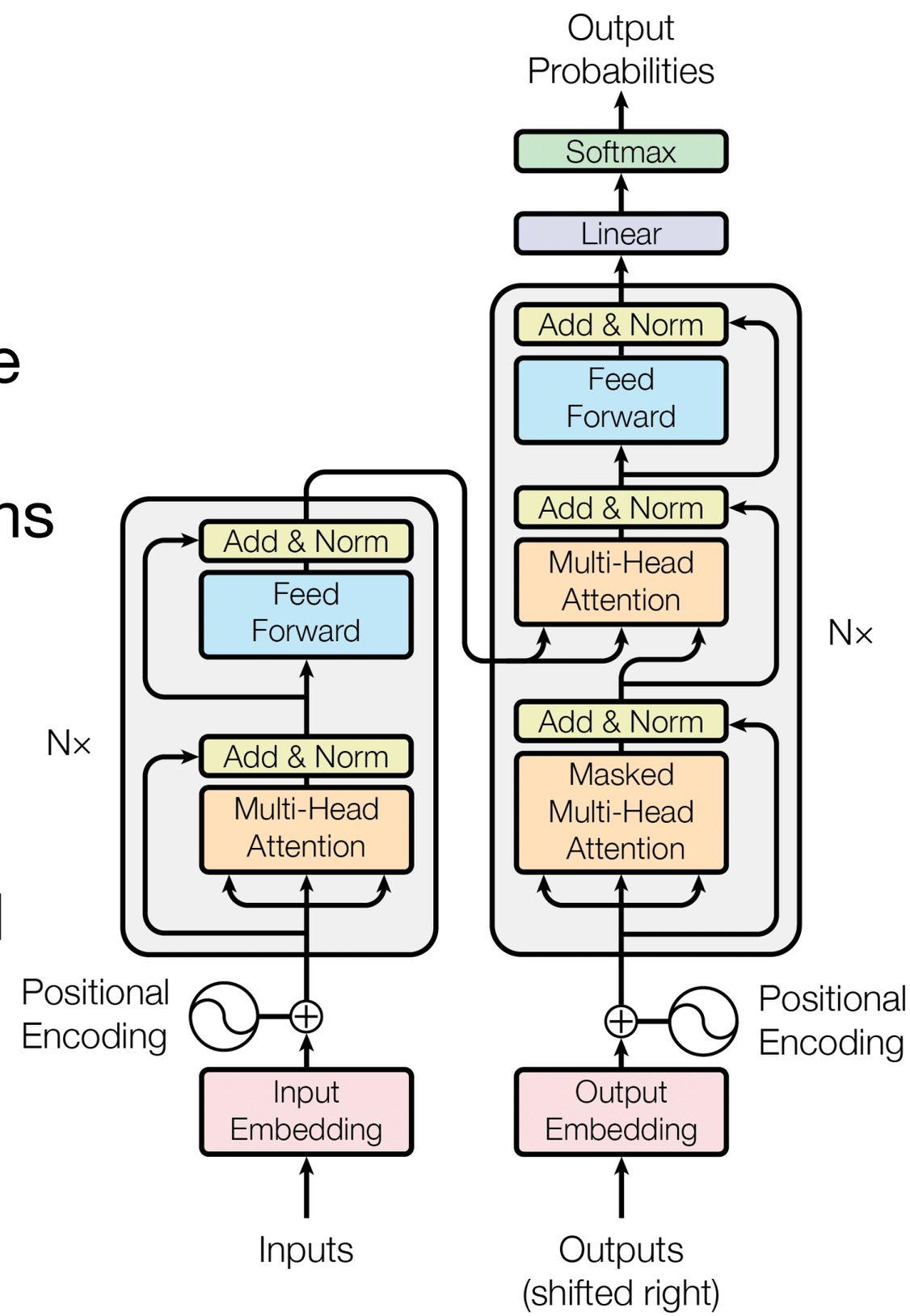
Autoregressive Mechanism

- Testing: word-by-word generation style, feed the model prediction to the next step
- Decoding algorithm: greedy search, beam search, etc.



Transformer

- Transformer: an autoregressive encoder-decoder architecture based on attention mechanisms
- Solve the problem of parallelization, fast in training
- Autoregressive: word-by-word generation, slow in decoding



Outline

- Introduction
- **Non-Autoregressive Transformer (NAT)**
- Progress in NAT
  - Sequence-Level Training
  - Latent Variable
  - Iterative Decoding
  - Knowledge Distillation

Non-autoregressive Transformer

- NAT: Non-autoregressive Transformer、Non-autoregressive Translation
- Non-autoregressive mechanism: independent prediction of target words, generate the whole target sentence simultaneously

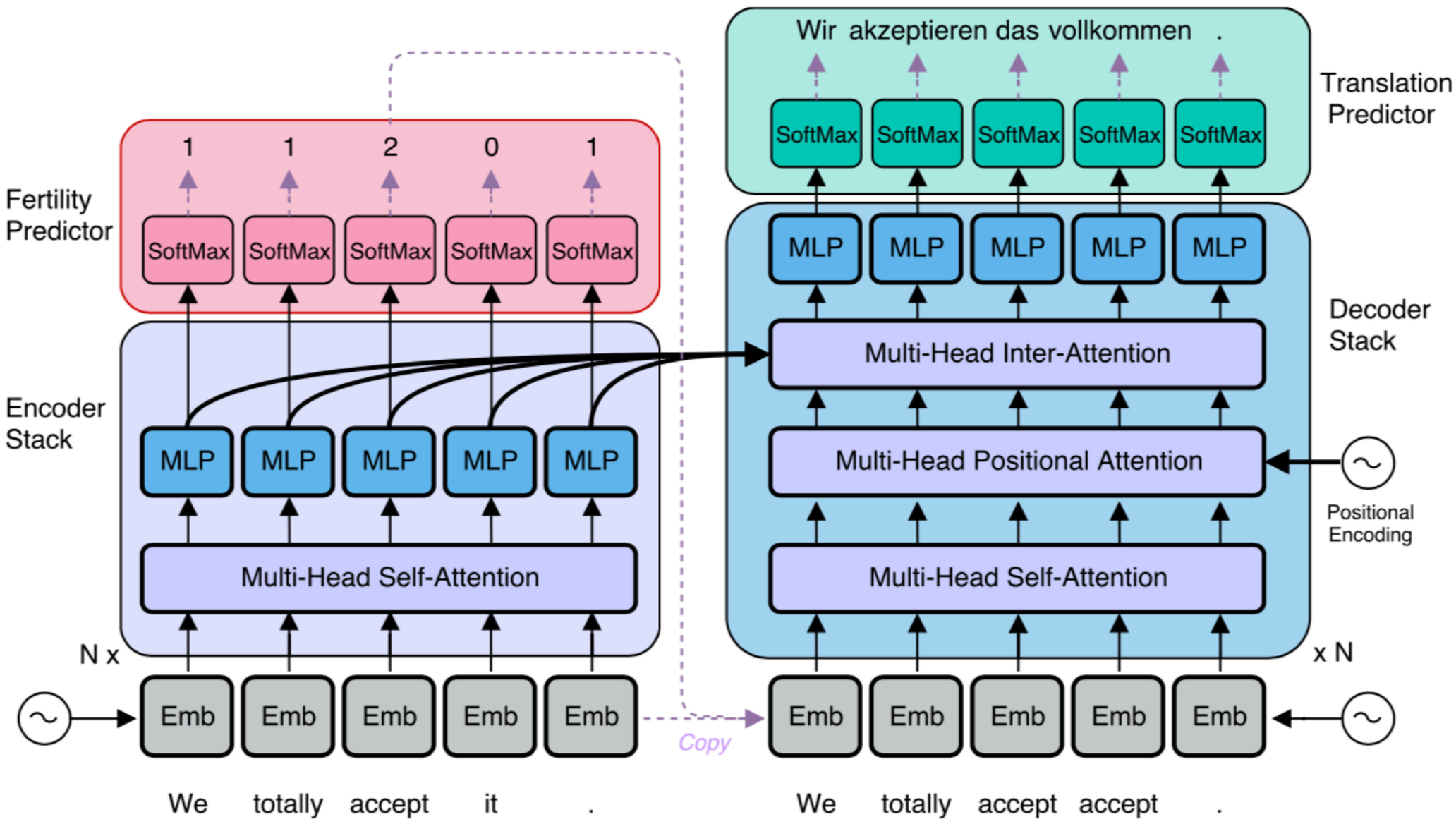
$$P(\boldsymbol{Y}|\boldsymbol{X},\theta) = \prod_{t=1}^{T} p(y_t|\boldsymbol{X},\theta).$$

- Proposed by Gu et al. in ICLR 2018

NON-AUTOREGRESSIVE
NEURAL MACHINE TRANSLATION

Jiatao Gu[†,*], **James Bradbury**[‡], **Caiming Xiong**[‡], **Victor O.K. Li**[†] **& Richard Socher**[‡]
[‡]Salesforce Research
{james.bradbury,cxiong,rsocher}@salesforce.com
[†]The University of Hong Kong
{jiataogu, vli}@eee.hku.hk

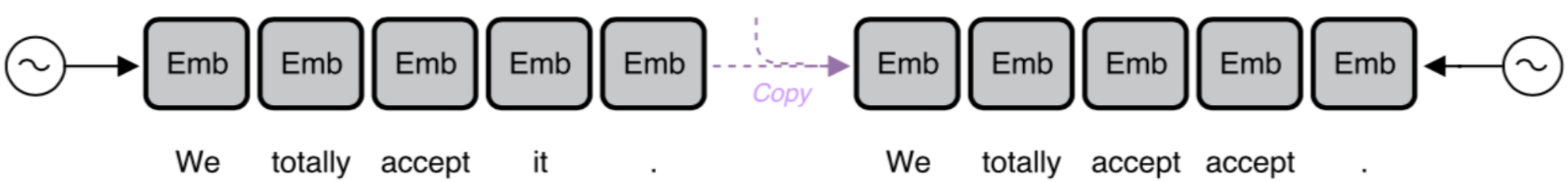
NAT: Model Overview



中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES

NAT: Decoder Inputs

- Decoder inputs: source embedding, copied from encoder inputs (Transformer: target embedding)
- Problem: source length and target length is different
- Copy methods: uniform copy、fertility prediction



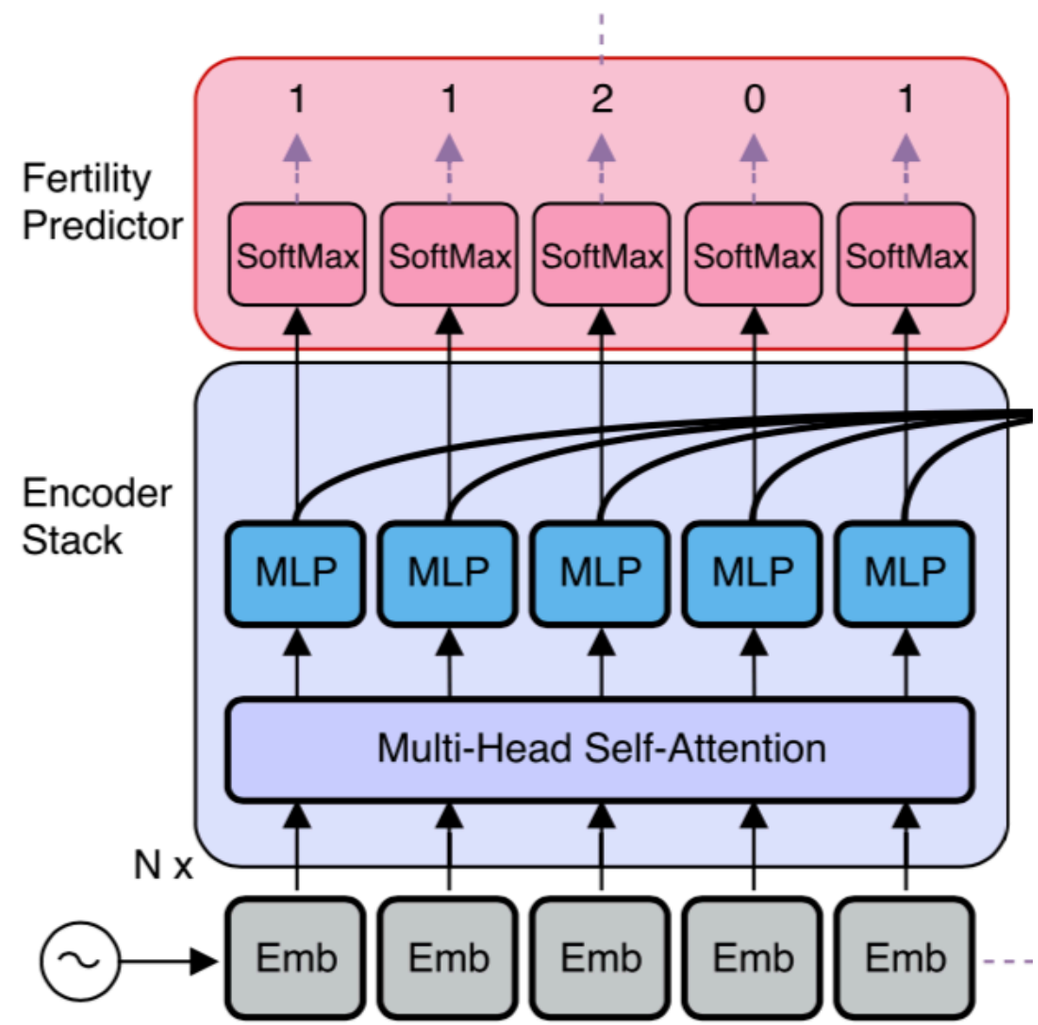
NAT: Decoder Inputs

- uniform: uniformly mapping the source side word embedding as the inputs to the decoder.

$$z_k = \mathcal{E}(x_i),\ i = \left[\frac{T_x}{T_y} t\right],\ t = 1, 2, ..., T_y.$$

- fertility: the copy times of a source word is predicted by the fertility predictor

- uniform copy is more popular



NAT: Length Prediction

- Uniform copy: has to know the target length

$$z_k = \mathcal{E}(x_i),\ i = \left[\frac{T_x}{T_y} t\right],\ t = 1, 2, ..., T_y.$$

- Length predictor: a softmax classifier, predict the target length based on encoder states

- Training: use the ground-truth target length

- Testing: use the predicted target length

NAT: Rescore

- Rescore for NAT: generate many candidates and find the best one, trade-off between quality and speed
  - Uniform: target length ranging from [T-B,T+B] to generate 2B+1 candidates
  - Fertility: sample from the fertility space to generate candidates
  - Sample from other latent variables
- Autoregressive Transformer as scorer

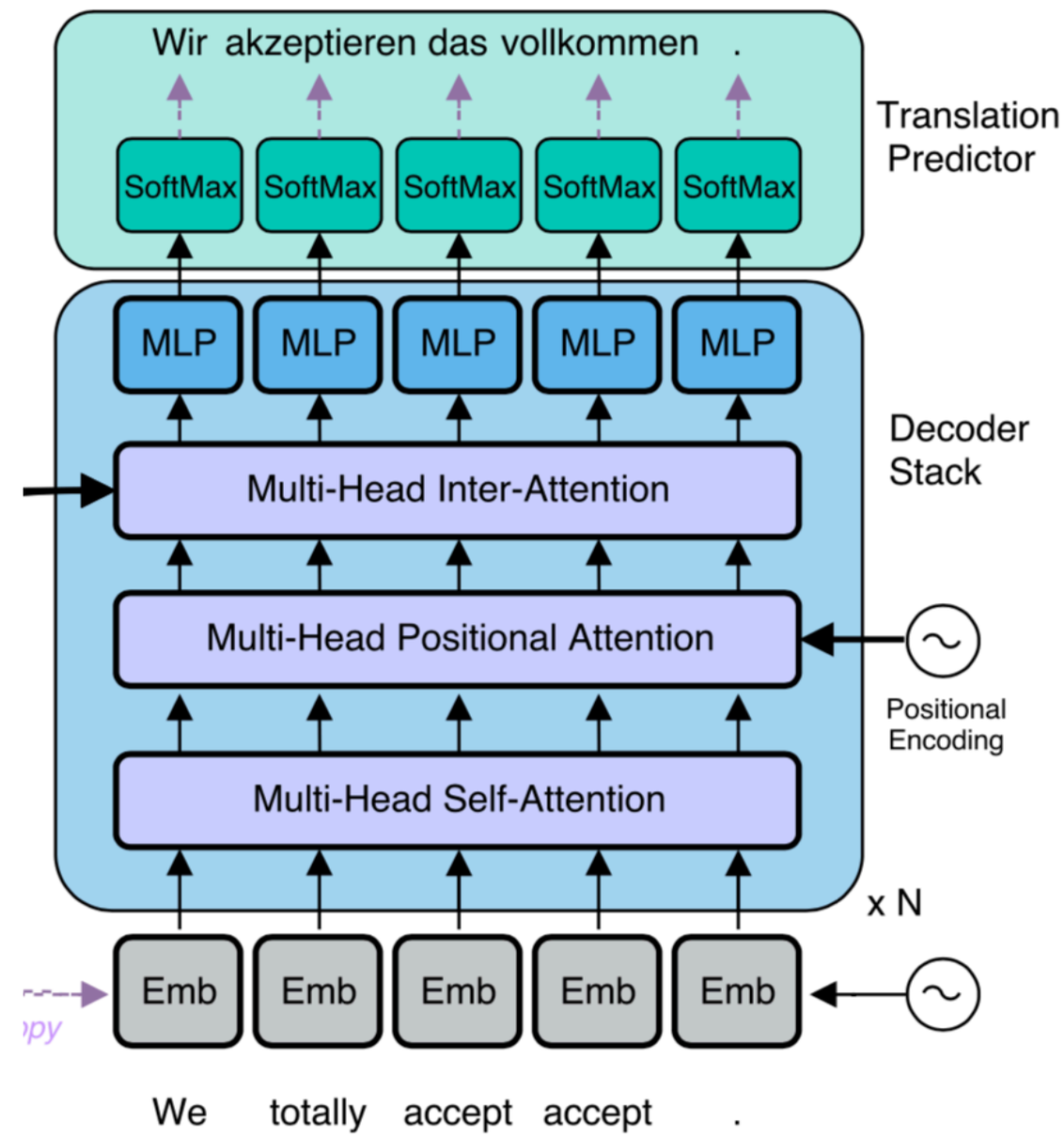
NAT: Positional attention

- Positional attention: an additional positional attention module in each decoder layer

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_{\text{model}}}}\right) \cdot V,$$

- Positional encoding: key and query; decoder states: value

- provides a stronger positional signal



NAT: Knowledge distillation

- Knowledge distillation: train the student model through distilling knowledge from the teacher model
- Sequence-level knowledge distillation: replace the target side of the training corpus by the output of the teacher model
- Student: NAT
- Teacher: autoregressive Transformer

Sequence-Level Knowledge Distillation

Yoon Kim
yoonkim@seas.harvard.edu

Alexander M. Rush
srush@seas.harvard.edu

中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES

NAT: Performance

Models	WMT14		WMT16		IWSLT16		
	En→De	De→En	En→Ro	Ro→En	En→De	Latency / Speedup	
NAT	17.35	20.62	26.22	27.83	25.20	39 ms	15.6×
NAT (+FT)	17.69	21.47	27.29	29.06	26.52	39 ms	15.6×
NAT (+FT + NPD $s = 10$)	18.66	22.41	29.02	30.76	27.44	79 ms	7.68×
NAT (+FT + NPD $s = 100$)	19.17	23.20	29.79	**31.44**	28.16	257 ms	2.36×
Autoregressive ($b = 1$)	22.71	26.39	31.35	31.03	28.89	408 ms	1.49×
Autoregressive ($b = 4$)	23.45	27.02	31.91	31.76	29.70	607 ms	1.00×

Source:	I see wheelchairs bought and sold like used cars .
Target:	ich erlebe , dass Rollstühle gekauft und verkauft werden wie Gebrauchtwagen
AR:	ich sehe Rollstühlen , die wie Autos verkauft und verkauft werden .
NAT:	ich sehe , dass Stühle Stühle und verkauft wie Autos verkauft .
NAT+NPD:	ich sehe Rollühle kauften und verkaufte wie Autos .

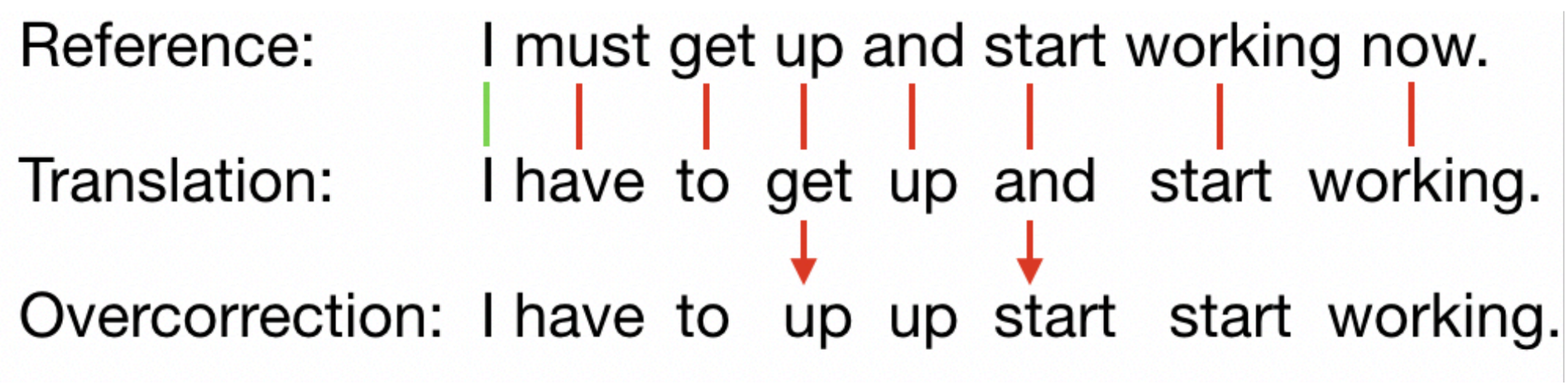
Outline

- Introduction
- Non-Autoregressive Transformer (NAT)
- Progress in NAT
  - **Sequence-Level Training**
  - Latent Variable
  - Iterative Decoding
  - Knowledge Distillation

Sequence-Level Training

- Retrieving Sequential Information for Non-Autoregressive Neural Machine Translation (ACL 2019)

- The cross-entropy loss is not suitable for NAT



Sequence-Level Training

- Training NAT models with sequence-level objectives (e.g., BLEU, GLEU, TER)
- Encourage NAT to generate high quality sentences rather than correct tokens
- Non-differentiable, REINFORCE

$$\mathrm{L}_\theta = -\sum_{\mathbf{Y}} p(\mathbf{Y}|\mathbf{X},\theta)\cdot r(\mathbf{Y})$$

$$\nabla_\theta \mathrm{L}_\theta = -\mathop{\mathbb{E}}_{\mathbf{Y}}[\sum_{t=1}^{T} \nabla_\theta \log(p(y_t|\mathbf{X},\theta))\cdot r(\mathbf{Y})].$$

Sequence-Level Training

- High variance, reduce the variance using some unique properties of NAT

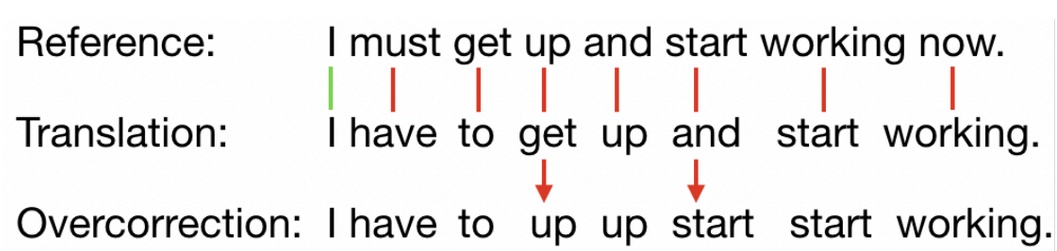
$$\nabla_\theta \mathrm{L}_\theta = -\sum_{t=1}^{T} \mathop{\mathbb{E}}_{y_t}[\nabla_\theta \log(p(y_t|\mathbf{X},\theta)) \cdot r(y_t)]$$

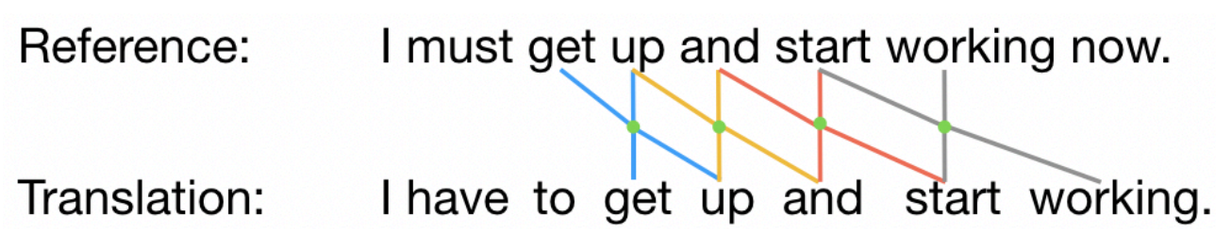
- Reinforce-NAT

$$\nabla_\theta \mathrm{L}_\theta = -\sum_{t=1}^{T}(\sum_{y_t \in T_K} \nabla_\theta p(y_t|\mathbf{X},\theta) \cdot r(y_t)$$
$$+ (1-P_k) \cdot \mathop{\mathbb{E}}_{y_t \sim \tilde{p}}[\nabla_\theta \log(p(y_t|\mathbf{X},\theta)) \cdot r(y_t)])$$

Sequence-Level Training

- Minimizing the Bag-of-Ngrams Difference for Non-Autoregressive Neural Machine Translation (AAAI 2020)
- Model sequential dependency for NAT: n-gram level loss
- Outputs may not be aligned: optimize the bag-of-ngrams
- Bag-of-Ngrams correlates better with the translation quality





Sequence-Level Training

- Minimizing the Bag-of-Ngrams Difference
  - Define Bag-of-Ngrams (BoN) of NAT
  - Give a method to calculate BoN
  - Choose a distance metric to measure BoN Difference
  - Give a method to calculate the metric
  - Bag-of-Ngrams difference as training objective

Sequence-Level Training

- Bag-of-Ngrams (BoN): a vector of size V^n, where V is the vocabulary size. Each entry represents the number of occurrences of an n-gram **g** in sentence **Y**:

$$\mathrm{BoN}_{\boldsymbol{Y}}(\boldsymbol{g}) = \sum_{t=0}^{T-n} 1\{y_{t+1:t+n} = \boldsymbol{g}\}$$

- How to define the BoN of NMT?

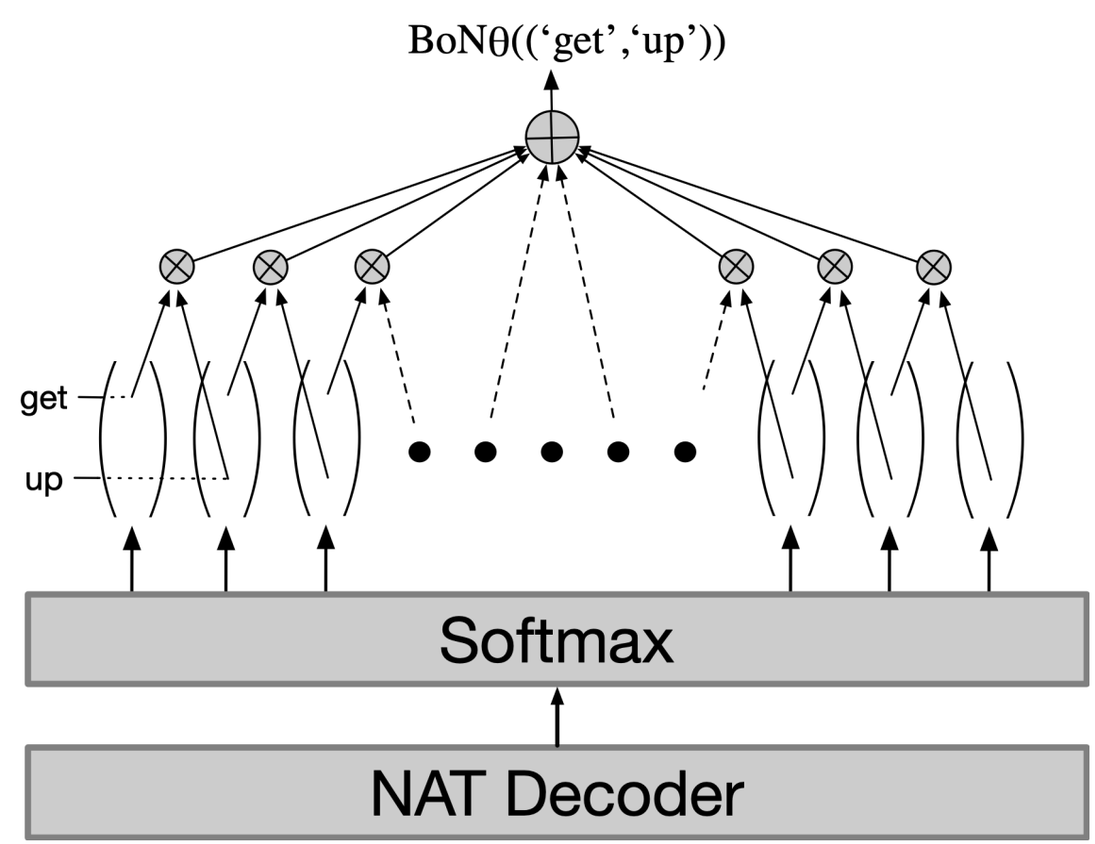
- Consider all possible translations, the expected BoN

$$\mathrm{BoN}_{\theta}(\boldsymbol{g}) = \sum_{\boldsymbol{Y}} P(\boldsymbol{Y}|\boldsymbol{X}, \theta) \cdot \mathrm{BoN}_{\boldsymbol{Y}}(\boldsymbol{g})$$

Sequence-Level Training

- BoN definition $\mathrm{BoN}_\theta(\boldsymbol{g}) = \sum_{\boldsymbol{Y}} P(\boldsymbol{Y}|\boldsymbol{X},\theta) \cdot \mathrm{BoN}_{\boldsymbol{Y}}(\boldsymbol{g})$
- The first difficulty: exponentially large search space
- Use the property of NAT: probabilities at different positions are independent

$$\begin{aligned}
\mathrm{BoN}_\theta(\boldsymbol{g}) &= \sum_{\boldsymbol{Y}} P(\boldsymbol{Y}|\boldsymbol{X},\theta) \cdot \sum_{t=0}^{T-n} 1\{y_{t+1:t+n} = \boldsymbol{g}\} \\
&= \sum_{t=0}^{T-n} \sum_{\boldsymbol{Y}} P(\boldsymbol{Y}|\boldsymbol{X},\theta) \cdot 1\{y_{t+1:t+n} = \boldsymbol{g}\} \\
&= \sum_{t=0}^{T-n} \sum_{\boldsymbol{Y}_{t+1:t+n}} P(\boldsymbol{Y}_{t+1:t+n}|\boldsymbol{X},\theta) \cdot 1\{y_{t+1:t+n} = \boldsymbol{g}\} \\
&= \sum_{t=0}^{T-n} \prod_{i=1}^{n} p(y_{t+i} = g_i|\boldsymbol{X},\theta).
\end{aligned}$$



Sequence-Level Training

- Calculate the BoN difference between NAT and reference (L_1, L_2, cosine, etc.)
- The second difficulty: totally V^n n-grams, costly to calculate and store all of them
- The BoN of reference is very sparse, only a few entries have non-zero values.
- Use the sparsity to simplify the calculation of L_1 distantce

Sequence-Level Training

- Intuition: a sentence of length T has T–n+1 n-grams:

$$\sum_{\boldsymbol{g}} \mathrm{BoN}_{\boldsymbol{Y}}(\boldsymbol{g}) = \sum_{t=0}^{T-n} \sum_{\boldsymbol{g}} 1\{y_{t+1:t+n} = \boldsymbol{g}\} = T - n + 1. \tag{9}$$

$$\begin{aligned} \sum_{\boldsymbol{g}} \mathrm{BoN}_{\theta}(\boldsymbol{g}) &= \sum_{\boldsymbol{g}} \sum_{\boldsymbol{Y}} P(\boldsymbol{Y}|\boldsymbol{X}, \theta) \cdot \mathrm{BoN}_{\boldsymbol{Y}}(\boldsymbol{g}) \\ &= \sum_{\boldsymbol{Y}} P(\boldsymbol{Y}|\boldsymbol{X}, \theta) \cdot \sum_{\boldsymbol{g}} \mathrm{BoN}_{\boldsymbol{Y}}(\boldsymbol{g}) = T - n + 1. \end{aligned} \tag{10}$$

- L_1 distantce: only consider n-grams in the reference

$$\begin{aligned} \mathrm{BoN}\text{-}L_1 &= \sum_{\boldsymbol{g}} |\mathrm{BoN}_{\theta}(\boldsymbol{g}) - \mathrm{BoN}_{\hat{\boldsymbol{Y}}}(\boldsymbol{g})| \\ &= \sum_{\boldsymbol{g}} (\mathrm{BoN}_{\theta}(\boldsymbol{g}) + \mathrm{BoN}_{\hat{\boldsymbol{Y}}}(\boldsymbol{g}) - 2\min(\mathrm{BoN}_{\theta}(\boldsymbol{g}), \mathrm{BoN}_{\hat{\boldsymbol{Y}}}(\boldsymbol{g})) \\ &= 2(T - n + 1 - \sum_{\boldsymbol{g}} \min(\mathrm{BoN}_{\theta}(\boldsymbol{g}), \mathrm{BoN}_{\hat{\boldsymbol{Y}}}(\boldsymbol{g}))). \end{aligned}$$

Sequence-Level Training

- Normalize the L_1 distance to range [0,1]

$$\mathcal{L}_{BoN}(\theta) = \frac{\text{BoN-}L_1}{2(T-n+1)}$$

- BoN-FT: fine-tune the pre-trained NAT model

- BoN-Joint: combine the BoN loss and cross-entropy loss

$$\mathcal{L}_{joint}(\theta) = \alpha \cdot \mathcal{L}_{MLE}(\theta) + (1-\alpha) \cdot \mathcal{L}_{BoN}(\theta)$$

- BoN-Joint+FT: joint training and fine-tuning

Sequence-Level Training

- The correlation between BoN difference and translation quality?

- WMT14 EN->DE dev set, randomly divide the 3000 sentences to 30 groups of size 10

- Calculate the cross-entropy loss, BoN-L_1 loss and BLEU score on each group

- The pearson correlation between BLEU score and losses

Loss function	CE	$n = 1$	$n = 2$	$n = 3$	$n = 4$
Correlation	0.37	0.56	0.70	0.67	0.61

Sequence-Level Training

- The effect of sentence length?
- Divide the devset evenly to short and long sentences
- Calculate the correlation on each part
- Performance of the cross-entropy loss drops sharply, where the BoN loss is robust to long sentences

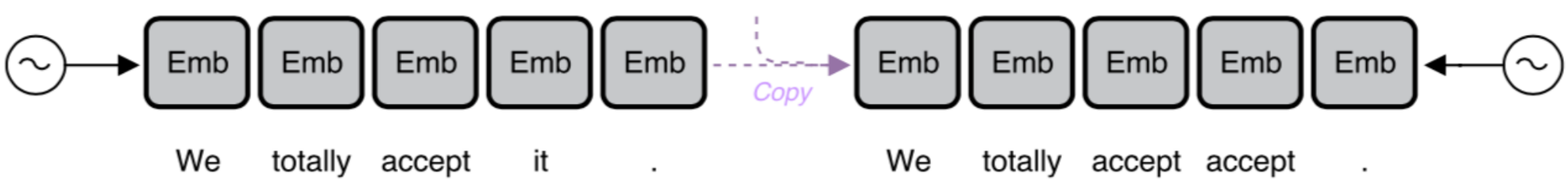
	all	short	long
Cross-Entropy	0.37	0.52	0.21
BoN (n=2)	0.70	0.79	0.81

Outline

- Introduction
- Non-Autoregressive Transformer (NAT)
- Progress in NAT
  - Sequence-Level Training
  - **Latent Variable**
  - Iterative Decoding
  - Knowledge Distillation

Latent Variable

- NON-AUTOREGRESSIVE NEURAL MACHINE TRANSLATION (ICLR 2018)
- Multimodality Problem
  - Thank you -> Danke、Danke schon、Vielen Dank
  - 必须 -> have to、must
- Introduce a latent variable z: fertility



Latent Variable

- FlowSeq: Non-Autoregressive Conditional Sequence Generation with Generative Flow (EMNLP 2019)

- Variational inference

- Optimize the evidence lower bound (ELBO)

$$\log P_\theta(\mathbf{y}|\mathbf{x}) \geq \mathrm{E}_{q_\phi(\mathbf{z}|\mathbf{y},\mathbf{x})}[\log P_\theta(\mathbf{y}|\mathbf{z},\mathbf{x})] - \mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{y},\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})).$$

- Posterior during training, prior during inference

- Flow-based Generative Models

Latent Variable

- Latent-Variable Non-Autoregressive Neural Machine Translation with Deterministic Inference using a Delta Posterior (AAAI 2020)

- Optimize the evidence lower bound (ELBO)

$$\mathcal{L}(\omega, \phi, \theta) = \mathbb{E}_{z \sim q_\phi}\big[\log p_\theta(y|x, z)\big] - \mathrm{KL}\big[q_\phi(z|x, y)||p_\omega(z|x)\big]$$

- Deterministic Iterative Inference

Latent Variable

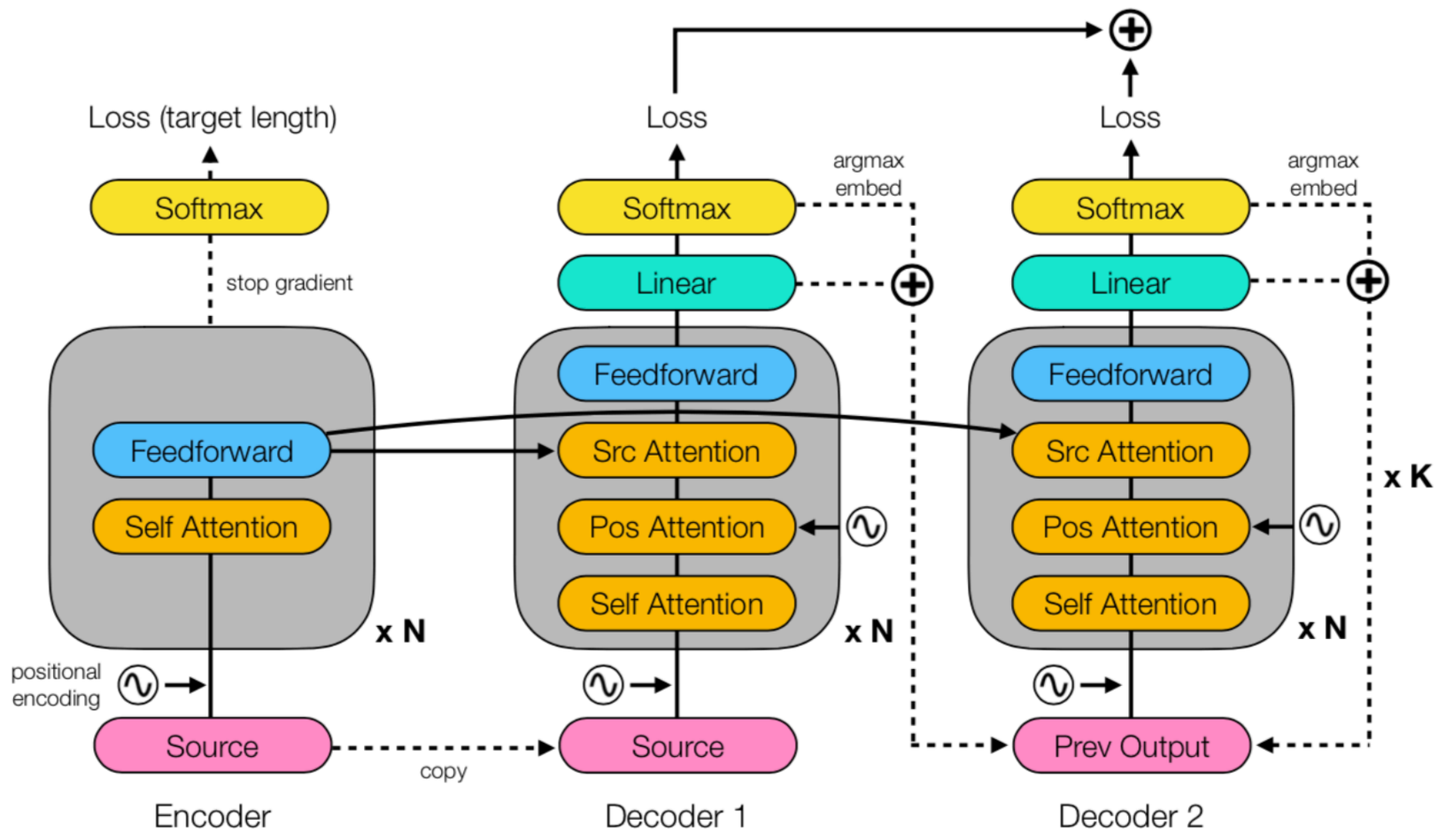
- PNAT: NON-AUTOREGRESSIVE TRANSFORMER BY POSITION LEARNING (ICLR 2020 under review)
- Guiding Non-Autoregressive Neural Machine Translation Decoding with Reordering Information (arxiv)
- Position latent variable
- $Z = \{z1, \cdots, zm\}$ reorders source sentence structure into the target language

Outline

- Introduction
- Non-Autoregressive Transformer (NAT)
- Progress in NAT
  - Sequence-Level Training
  - Latent Variable
  - **Iterative Decoding**
  - Knowledge Distillation

Iterative Decoding

- Deterministic Non-Autoregressive Neural Sequence Modeling by Iterative Refinement (EMNLP 2018)



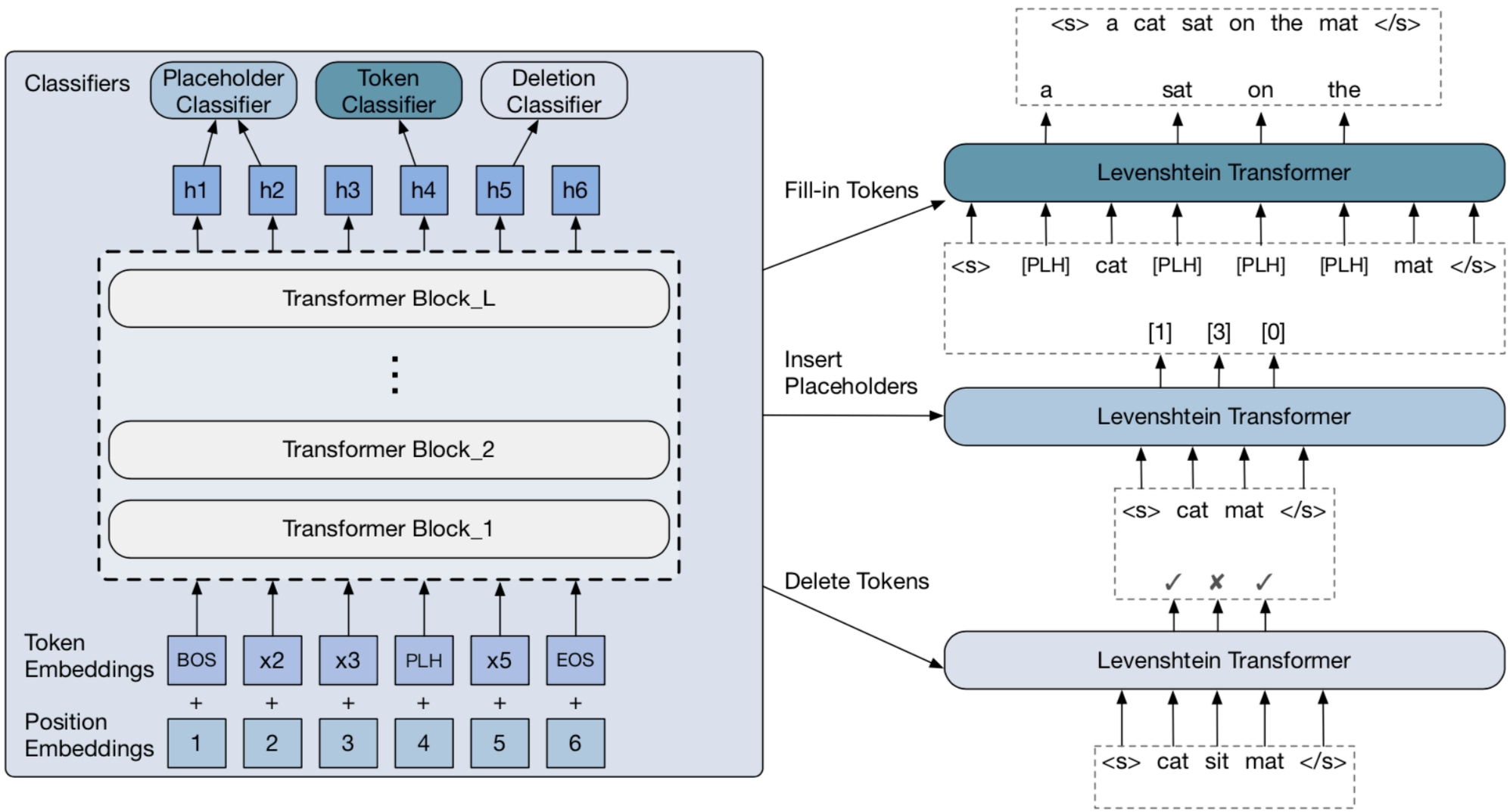
Iterative Decoding

- Mask-Predict: Parallel Decoding of Conditional Masked Language Models (ACL 2019)
- First iteration: mask all tokens
- At each iteration, mask tokens with lowest probabilities and repredict them.

src	Der Abzug der franzsischen Kampftruppen wurde am 20. November abgeschlossen .
$t = 0$	The departure of the French combat completed completed on 20 November .
$t = 1$	The departure of French combat troops was completed on 20 November .
$t = 2$	The withdrawal of French combat troops was completed on November 20th .

Iterative Decoding

- Levenshtein Transformer (NeurIPS 2019)



中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES

Outline

- Introduction
- Non-Autoregressive Transformer (NAT)
- Progress in NAT
  - Sequence-Level Training
  - Latent Variable
  - Iterative Decoding
  - **Knowledge Distillation**

Knowledge Distillation

- Knowledge distillation: train the student model through distilling knowledge from the teacher model
- Student: NAT
- Teacher: autoregressive Transformer
- Sequence-level knowledge distillation: replace the target side of the training corpus by the output of the teacher model

Knowledge Distillation

- Hint-Based Training for Non-Autoregressive Machine Translation (EMNLP 2019)
- Hints from hidden states

$$\mathcal{L}_{hid} = \frac{2}{(T_y - 1)T_y N} \sum_{s=1}^{T_y - 1} \sum_{t=s+1}^{T_y} \sum_{l=1}^{N} \phi(d_{st}, d_{tr})$$

- Hints from word alignments

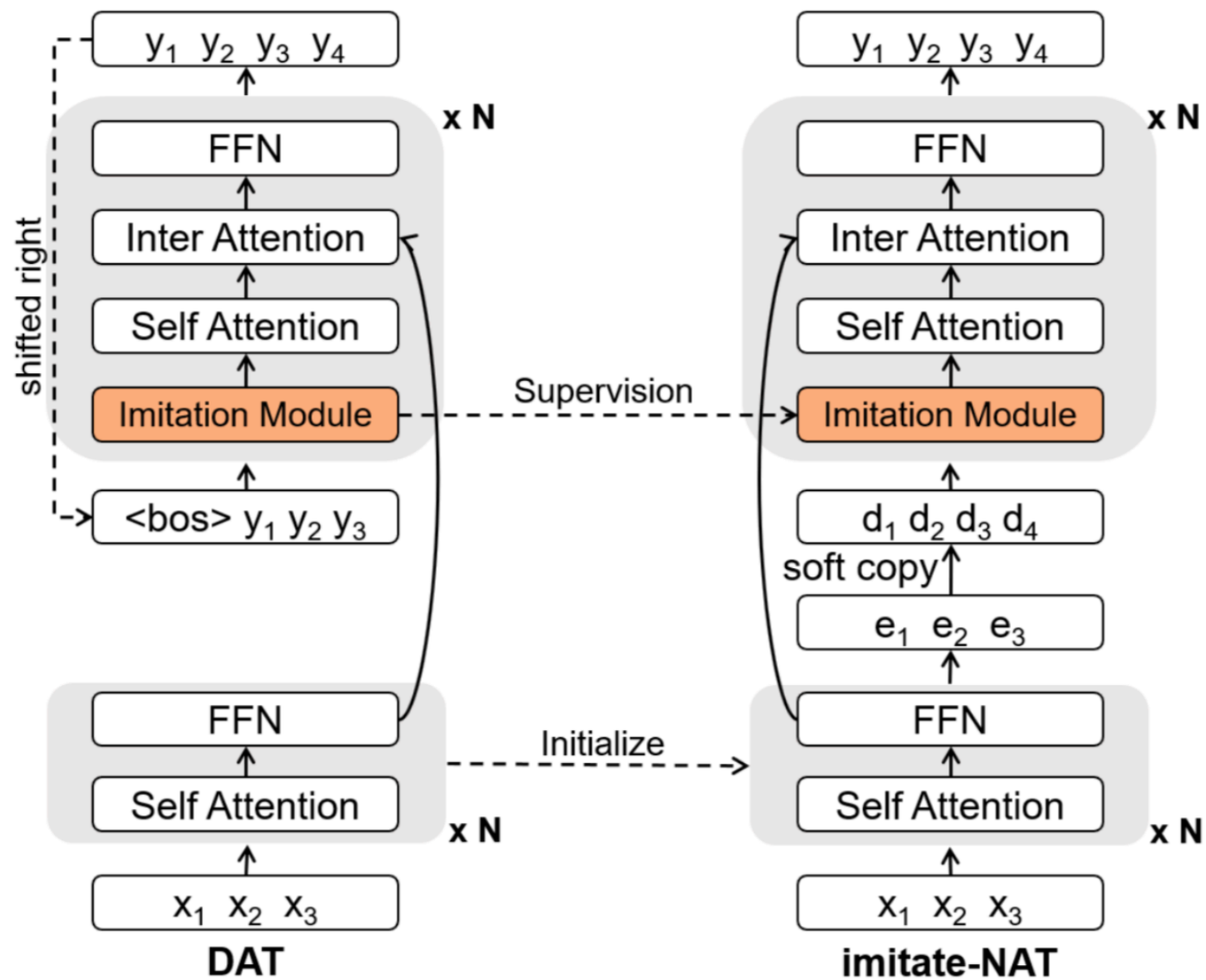
$$\mathcal{L}_{align} = \frac{1}{T_y N H} \sum_{t=1}^{T_y} \sum_{l=1}^{N} \sum_{h=1}^{H} D_{\mathrm{KL}}(a_{t,l,h}^{tr} \| a_{t,l,h}^{st})$$

Knowledge Distillation

- Imitation Learning for Non-Autoregressive Neural Machine Translation (ACL 2019)

- DAT: AT Demonstrator, emits actions at every time step

- Imitate-NAT learns the action from the AT Demonstrator



Knowledge Distillation

- Fine-Tuning by Curriculum Learning for Non-Autoregressive Neural Machine Translation (AAAI 2020)

- Transfer a well-trained AT model to an NAT model through fine-tuning

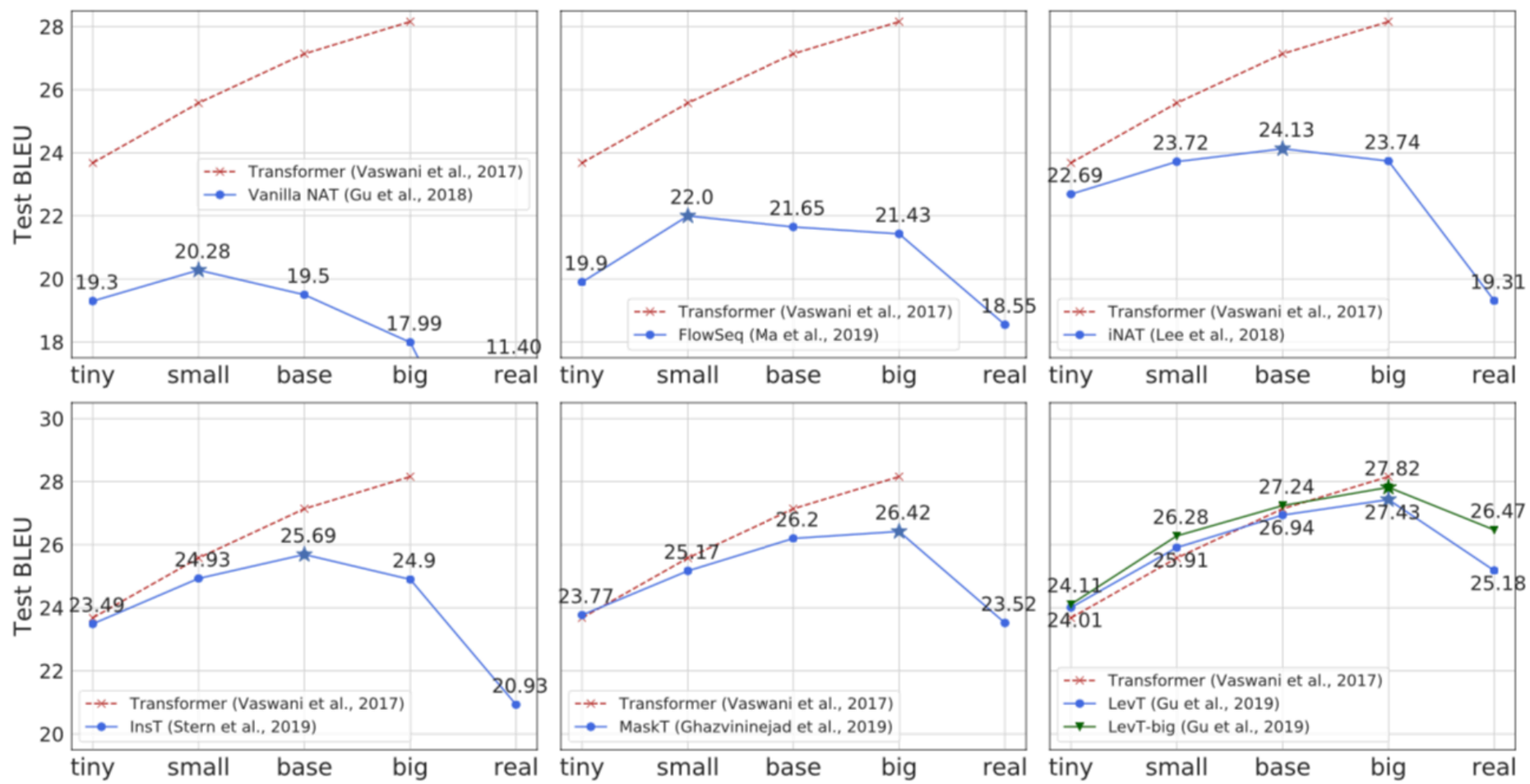
- From $L_{\mathrm{AT}}(x, y; \Theta) = -\sum_{t=1}^{T_y} \log P(y_t | z_{\mathrm{AT}}, x)$, $z_{\mathrm{AT}} = (y_0, \ldots, y_{t-1})$
- To $L_{\mathrm{NAT}}(x, y; \Theta) = -\sum_{t=1}^{T_y} \log P(y_t | z_{\mathrm{NAT}}, x)$, $z_{\mathrm{NAT}} = (\tilde{x}_1, \ldots, \tilde{x}_{T_y})$

- Curriculum for the decoder input

$$z_i = (1 - P_T) \odot z_{\mathrm{AT}} + P_T \odot z_{\mathrm{NAT}}$$

Knowledge Distillation

- UNDERSTANDING KNOWLEDGE DISTILLATION IN NON-AUTOREGRESSIVE MACHINE TRANSLATION (ICLR 2020 under review)



中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES

Thanks

Q&A

中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES