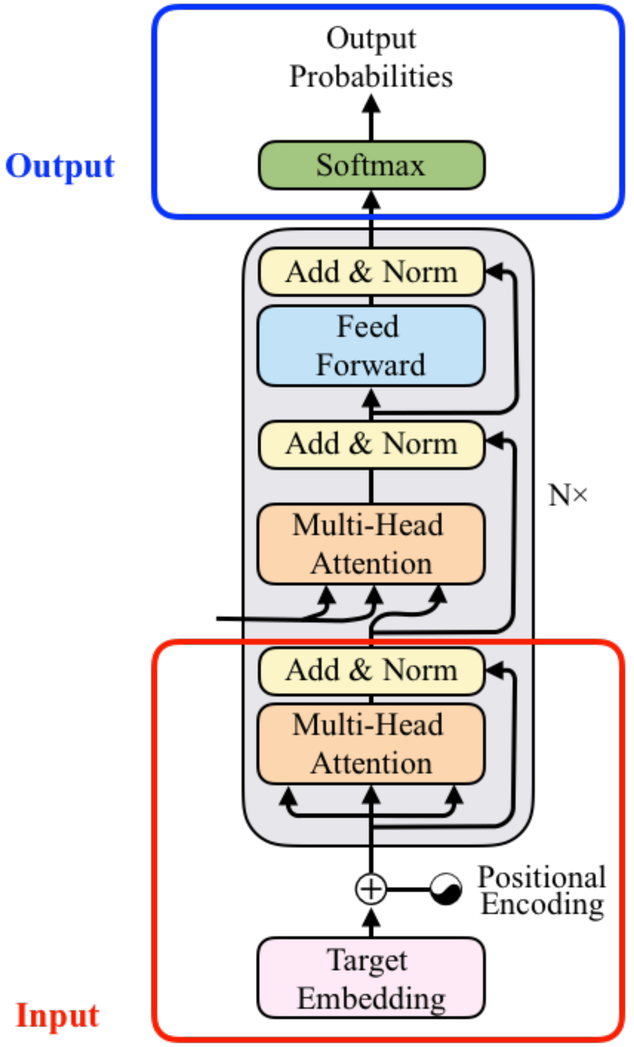
On Conditional Generation: Non-autoregressive and Knowledge Distillation

Longyue Wang, Xuebo Liu and Liang Ding

2019/12/31

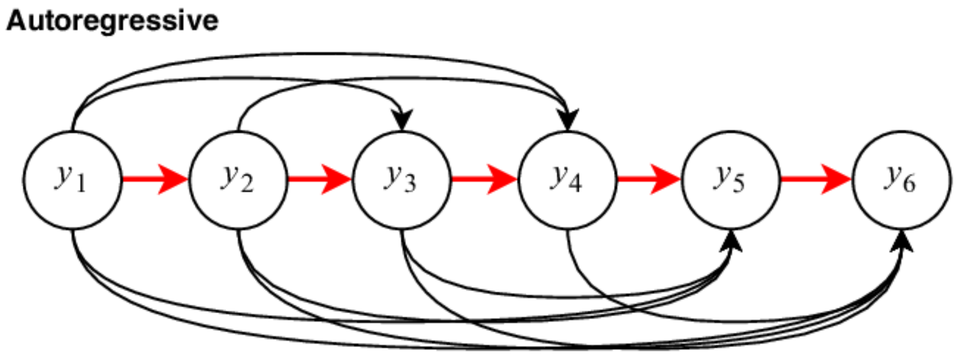
Conditional Generation

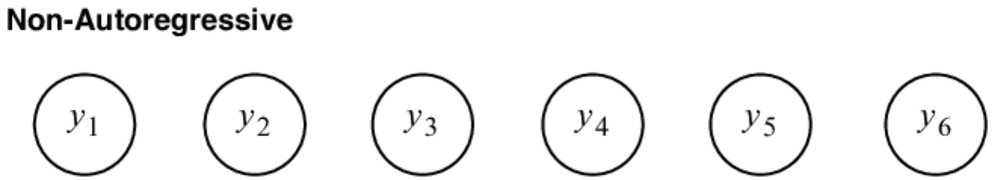
- Decoder contains *input targets* and *output probabilities* parts:
  - Input targets contributes to extract source knowledge for generation
  - Output probabilities guides learning direction by generated outputs and labels
- NA vs. KD
  - Input (architecture): non-autoregressive (NA)
  - Output (data limitation): knowledge distillation (KD)



Non-autoregressive

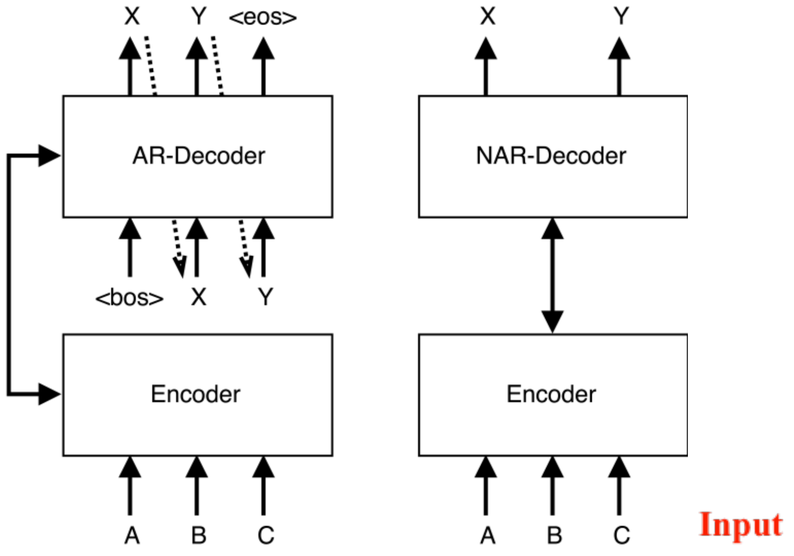
- Background
  - In **statistics**, autoregressive is one method to deal with time series, using x_1 to x_t-1 to predict x_t
  - In **machine learning**, autoregressive and recurrent networks
  - In **natural language**, RNN and Transformer are autoregressive
  - Old limitation: O(N) -> O(1)

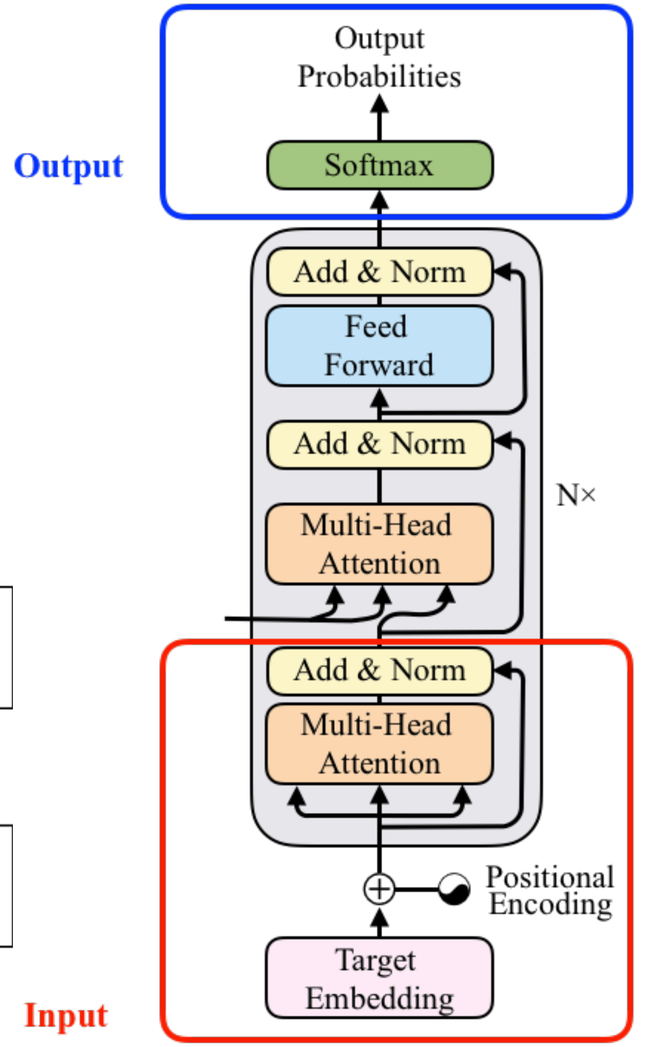




Non-autoregressive

- Background
  - In **machine translation**, to speed up the decoding process
  - Gu et al. (2018) starts the new research direction: non-autoregressive
  - New limitation: repeated translation (fluency) and incomplete translation (adequacy)



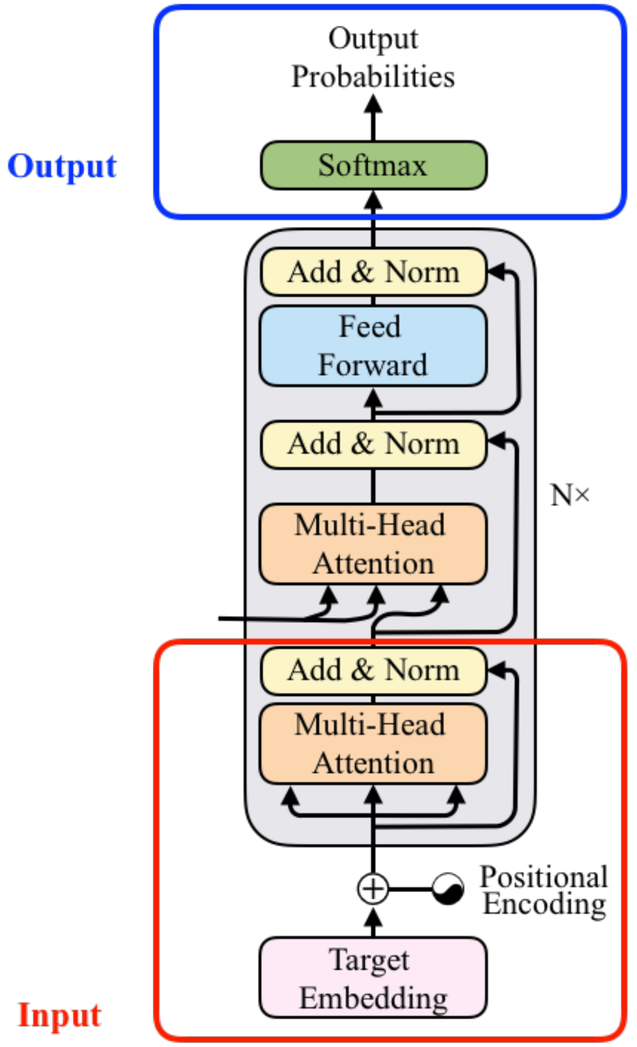


Related Work

- **Model Architecture**
  - Ma et al. FlowSeq: Non-Autoregressive Conditional Sequence Generation with Generative Flow. EMNLP 2019.
  - Gu et al. **Levenshtein Transformer**. NIPS2019.
- **Training Objectives and Frameworks**
  - Lee et al. Deterministic Non-Autoregressive Neural Sequence Modeling by Iterative Refinement. EMNLP 2018.
  - Shao et al. Retrieving Sequential Information for Non-Autoregressive Neural Machine Translation. ACL 2019.
- **Decoding**
  - Gu et al. Non-Autoregressive Neural Machine Translation. ICLR 2018.
  - Guo et al. Non-Autoregressive Neural Machine Translation with Enhanced Decoder Input. AAAI 2019.
  - Wang et al.Non-Autoregressive Machine Translation with Auxiliary Regularization. AAAI 2019.
  - Wei et al. Imitation Learning for Non-Autoregressive Neural Machine Translation. ACL 2019.
  - Ghazvininejad et al. **Mask-Predict: Parallel Decoding of Conditional Masked Language Models**. EMNLP2019.

Knowledge Distillation

- Background
  - In **thermodynamics**, process of separating the components
  - In **machine learning**, Caruana 2006 use it for model compression; Hinton 2015 use it for knowledge transfer
  - Limitation (old motivation):
    1. soft target with high information entropy can help hard target
    2. the bottleneck of output is limited by data
    3. soft target can work like a regularizer



Knowledge Distillation

- Background
  - In **our work**, domain knowledge distillation for better pattens

Input	143 li yuan wai xin bo zhou ting huan zhe jing song lai yi yuan, fu su cun huo jin 2 li (1.4%).
Reference	In the other 143 patients occurring sudden arrest of heart beat outside hospital, only 2 survived (1.4%).
Baseline	In the other 143 patients who received cardiac arrest, only 2 survived (1.4%).
+Trans.	In the 143 patients admitted to hospital, only 2 (1.4%) survived for resuscitation.
+Distill.	In the other 143 patients who suffered a sudden arrest of heart beat outside hospital, only 2 (1.4%) survived.
+Discri.	In the other 143 patients who suffered sudden arrest of heart beat outside hospital, only 2 survived (1.4%).

Related Work

- Two levels in applications
  - word-level
  - sequence-level

$$L_{\text{WORD-KD}}(\theta) = -\sum_{i=1}^{N}\sum_{j=1}^{J}\sum_{k=1}^{|\nu|} q(t_j = k \mid \mathbf{s_i}, t_{<j}) \times \log p_\theta(t_j = k \mid \mathbf{s_i}, t_{<j})$$

$$L_{\text{SEQ-KD}}(\theta) = -\sum_{i=1}^{N}\sum_{\mathbf{t}\in\tau} q(\mathbf{t} \mid \mathbf{s_i}) \log p_\theta(\mathbf{t} \mid \mathbf{s_i})$$

- Kim and Rush (2016) showed that word-level and sequence-level distillation are roughly orthogonal and probably improve student models in different ways.
- Limitation (new motivation):
  - word-level KD exploits more signals by soft targets
  - sequence-leve KD gives more determinative signals by high-frequent targets

Related Work

- In our work, we tested two methods
  - domain adaptation
  - word-level > sequence-level

#	Model	BLEU
1	Transformer	30.90
2	+ Distillation (sequence)	31.45
3	+ Distillation (word)	31.51
4	+ Domain Transformation	32.34
5	+ Domain Distillation (sequence)	32.70
6	+ Domain Distillation (word)	33.05
7	+ Domain Discrimination	33.18
8	+ Domain Distillation (word)	34.25

Related Work

- Distilling the Knowledge in a Neural Network (NeurIPS 2014 Workshop)
- Sequence-Level Knowledge Distillation (EMNLP 2016)
- Understanding Knowledge Distillation in Non-Autoregressive MT (ICLR 2020)
- Data-Dependent Gaussian Prior Objective for Language Generation (ICLR 2020)

Paper Reading on Non-Autoregressive Transformer

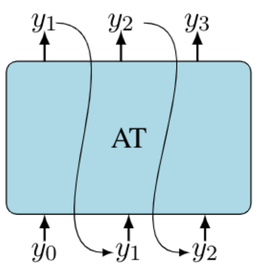
Longyue Wang, Xuebo Liu, Liang Ding

December 31, 2019

Why Non-Autoregressive Transformer?

- AT objective function
  - Generate sentence word-by-word
  - Sequential dependency
- NAT objective function
  - Generate words in parallel
  - No sequential dependency
- NAT leads to
  - 😁 Inference speedup
  - 😣 Performance degradation

- Autoregressive Transformer (AT)

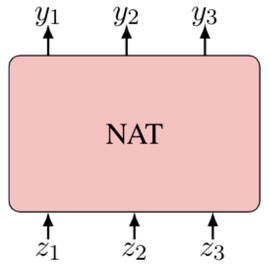


Encoder output

$$y_j = \mathbb{D}(\mathbb{E}(\mathbf{x}), \mathbf{y}_{1:j-1})$$

Intractable

- Non-autoregressive Transformer (NAT)



$$y_j = \mathbb{D}(\mathbb{E}(\mathbf{x}), \mathbf{z})$$

Latent variable

- Adaptive-length source input
- Transformations of encoder output
- Placeholders

Goal: NAT aims to strike the right balance between higher inference speedup and lower performance degradation.

NAT: One of the hottest topic of generation

- Year 2018
  - (Salesforce Research) Gu et al. Non-Autoregressive Neural Machine Translation. ICLR 2018.
  - (Google Brain) Kaiser et al. Fast Decoding in Sequence Models Using Discrete Latent Variables. ICML 2018.
  - (NYU) Lee et al. Deterministic Non-Autoregressive Neural Sequence Modeling by Iterative Refinement. EMNLP 2018.
  - (Alibaba) Wang et al. Semi-Autoregressive Neural Machine Translation. EMNLP 2018.
  - (Charles University) Libovicky and Helcl. End-to-End Non-Autoregressive Neural Machine Translation with Connectionist Temporal Classification. EMNLP 2018.
  - (Google Brain) Stern et al. Blockwise Parallel Decoding for Deep Autoregressive Models. NIPS 2018.

NAT: One of the hottest topic of generation

- Year 2019
  - (MSRA) Wang et al. Non-Autoregressive Machine Translation with Auxiliary regularization. AAAI 2019.
  - (MSRA) Guo et al. Non-Autoregressive Neural Machine Translation with Enhanced Decoder Input. AAAi 2019.
  - (ICT) Shao et al. Retrieving Sequential Information for Non-Autoregressive Neural Machine Translation. ACL 2019.
  - (Peking University) Ma et al. Imitation Learning for Non-Autoregressive Neural Machine Translation. ACL 2019.
  - (CMU) Ma et al. FlowSeq: Non-Autoregressive Conditional Sequence Generation with Generative Flow. EMNLP 2019
  - (MSRA) Li et al. Hint-Based Training for Non-Autoregressive Machine Translation. EMNLP 2019.
  - (Facebook) Ghazvininejad et al. Mask-Predict: Parallel Decoding of Conditional Masked Language Models. EMNLP 2019.
  - (Facebook) Gu et al. Levenshtein Transformer. NIPS 2019.

NAT: One of the hottest topic of generation

- Year 2019
  - (MSRA) Wang et al. Non-Autoregressive Machine Translation with Auxiliary regularization. AAAI 2019.
  - (MSRA) Guo et al. Non-Autoregressive Neural Machine Translation with Enhanced Decoder Input. AAAi 2019.
  - (ICT) Shao et al. Retrieving Sequential Information for Non-Autoregressive Neural Machine Translation. ACL 2019.
  - (Peking University) Ma et al. Imitation Learning for Non-Autoregressive Neural Machine Translation. ACL 2019.
  - (CMU) Ma et al. FlowSeq: Non-Autoregressive Conditional Sequence Generation with Generative Flow. EMNLP 2019
  - (MSRA) Li et al. Hint-Based Training for Non-Autoregressive Machine Translation. EMNLP 2019.
  - (Facebook) Ghazvininejad et al. Mask-Predict: Parallel Decoding of Conditional Masked Language Models. EMNLP 2019.
  - (Facebook) Gu et al. Levenshtein Transformer. NIPS 2019.

Mask-Predict

- Description
  - An encoder-decoder architecture trained with a masked language model objective (BERT)
- Key idea
  - Training: Random mask target tokens and then predict the masked tokens
  - Inference: Parallel generate the targets and iteratively refine them

- Training
  - Modify the mask matrix of the self-attention layer of decoder
    - Target tokens can attend to future tokens, i.e. building a bidirectional decoder
  - Randomly mask the target inputs by replacing them with the <mask> placeholder
    - Only compute the prediction loss of the <mask> tokens
- Inference-iterative refinement
  - Predict the target length by source output representation
  - Generate intermediate target representations and re-predict those low-confidence predictions

Inference-An Example

Given a source sentence Der Abzug der franzsischen Kampftruppen wurde am 20. November abgeschlossen

Predict target length N=12

Padded target input <mask> <mask> <mask> <mask> <mask> <mask> <mask> <mask> <mask> <mask> <mask> <mask>

1st Prediction

The	departure	of	the	French	combat	completed	completed	on	20	November	.
0.8	0.3	0.2	0.3	0.5	0.5	0.4	0.1	0.3	0.6	0.6	0.9

Refine $N \cdot \frac{T-t}{T} = 8$ tokens

Padded target input The <mask> <mask> <mask> <mask> <mask> <mask> <mask> <mask> 20 November .

Total Round=3

2nd Prediction

The	departure	of	French	combat	troops	was	completed	on	20	November	.
0.8	0.5	0.8	0.8	0.9	0.7	0.7	0.6	0.8	0.6	0.6	0.9

Refine $N \cdot \frac{T-t}{T} = 4$ tokens

Padded target input The <mask> of French combat troops was <mask> on <mask> <mask> .

3rd Prediction (Final Prediction) The withdrawal of French combat troops was completed on November 20th .

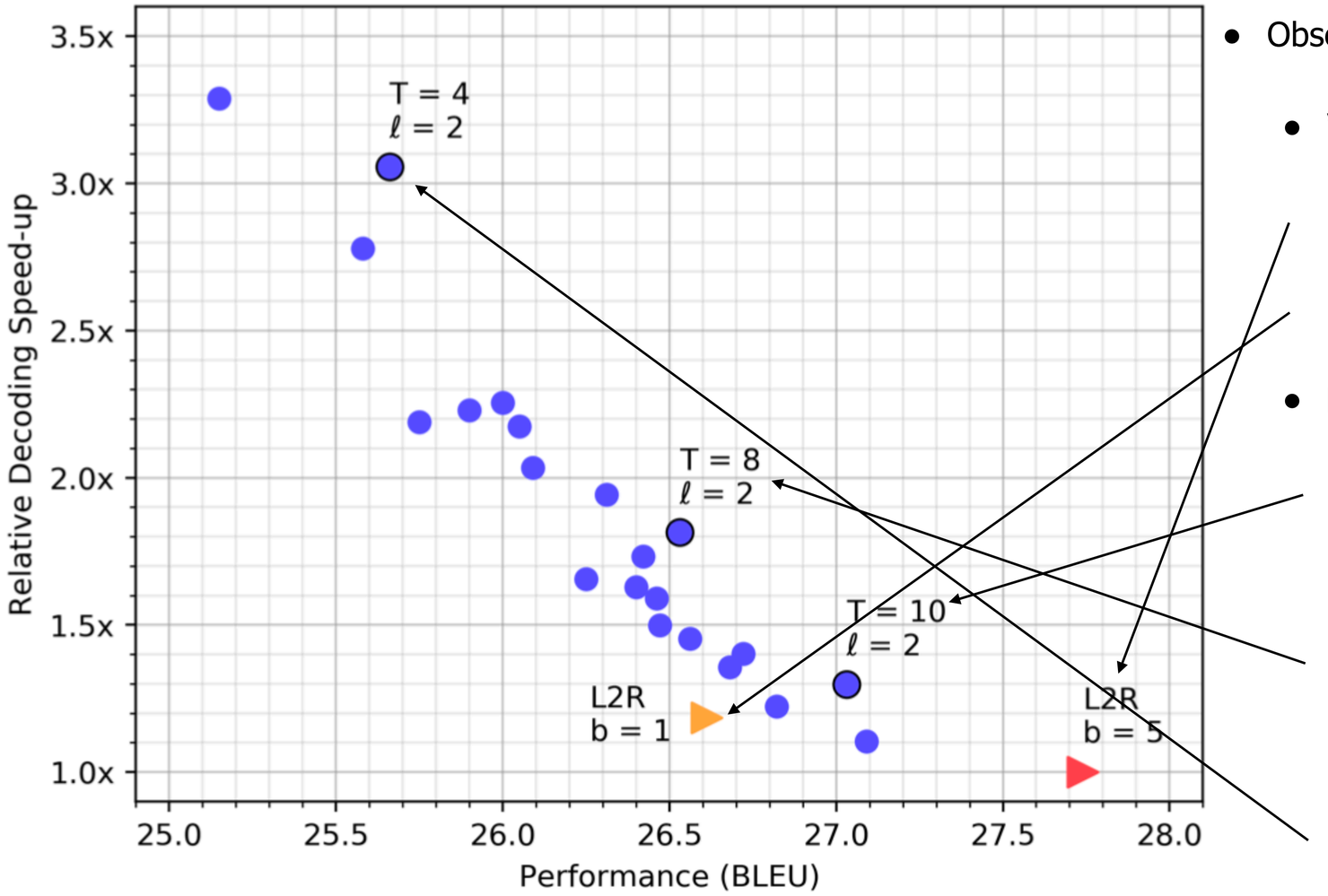
Settings&Results-BLEU Score

- Task
  - Medium-scale: WMT'14 English<->German (4.5M)
  - Small-scale: WMT'16 Romanian<->English (600K)
  - Large-scale:WMT'17 Chinese<->English (20M)
- Settings
  - Small: (model) 512 / (FFN hidden) 512
  - Base: (model) 512 / (FFN hidden) 2048
  - 16*V100
    - Batch size=8192*16

Model	Dimensions (Model/Hidden)	Iterations	WMT'14 EN-DE	WMT'14 DE-EN	WMT'16 EN-RO	WMT'16 RO-EN
NAT w/ Fertility (Gu et al., 2018)	512/512	1	19.17	23.20	29.79	31.44
CTC Loss (Libovický and Helcl, 2018)	512/4096	1	17.68	19.80	19.93	24.71
Iterative Refinement (Lee et al., 2018)	512/512	1	13.91	16.77	24.45	25.73
	512/512	10	21.61	25.48	29.32	30.19
(Dynamic #Iterations)	512/512	?	21.54	25.43	29.66	30.30
Small CMLM with Mask-Predict	512/512	1	15.06	19.26	20.12	20.36
	512/512	4	**24.17**	**28.55**	**30.00**	30.43
	512/512	10	**25.51**	**29.47**	**31.65**	**32.27**
Base CMLM with Mask-Predict	512/2048	1	18.05	21.83	27.32	28.20
	512/2048	4	**25.94**	**29.90**	**32.53**	**33.23**
	512/2048	10	**27.03**	**30.53**	**33.08**	**33.31**
Base Transformer (Vaswani et al., 2017)	512/2048	N	27.30	— —	— —	— —
Base Transformer (Our Implementation)	512/2048	N	27.74	31.09	34.28	33.99
Base Transformer (+Distillation)	512/2048	N	27.86	31.07	— —	— —
Large Transformer (Vaswani et al., 2017)	1024/4096	N	28.40	— —	— —	— —
Large Transformer (Our Implementation)	1024/4096	N	28.60	31.71	— —	— —

Model	Dimensions (Model/Hidden)	Iterations	WMT'17 EN-ZH	WMT'17 ZH-EN
Base CMLM with Mask-Predict	512/2048	1	24.23	13.64
	512/2048	4	32.63	21.90
	512/2048	10	33.19	23.21
Base Transformer (Our Implementation)	512/2048	N	34.31	23.74
Base Transformer (+Distillation)	512/2048	N	34.44	23.99
Large Transformer (Our Implementation)	1024/4096	N	35.01	24.65

Results-Inference Speedup



- Observations
  - Vanilla Transformer with knowledge distillation
    - Beam=5; BLEU=27.74; Speedup=1.0x
    - Beam=1; BLEU=26.56; Speedup=1.2x
  - Mask-Predict with knowledge distillation
    - T=10 refinements; Length beam l=2
      - BLEU=27.03; Speedup=1.3x
    - T=8 refinements; Length beam l=2
      - BLEU=26.53; Speedup=1.7x
    - T=4 refinements; Length beam l=2
      - BLEU=25.68; Speedup=3.1x

My Experimental Results

- Reproducibility
  - The BLEU score and speedup are completely truthful
- Can wider or deeper model help Mask-Predict?
  - YES
  - Deeper: Encoder layer=12
  - Wider: Model=768; Hidden=3072
  - Both settings can improve ~0.5 BLEU scores on the WMT'14 English-German translation task

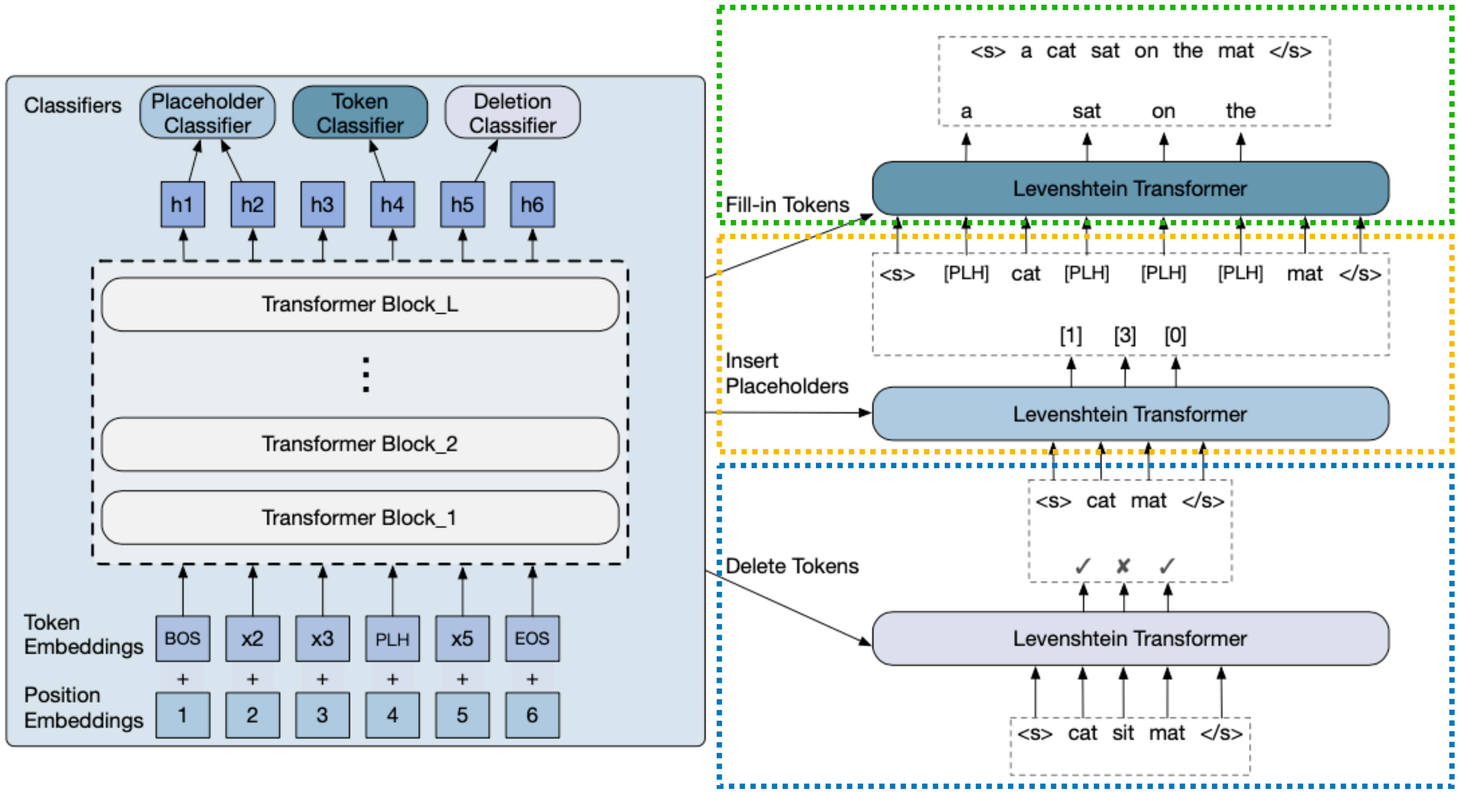
Discussion

- Pros
  - One of the simplest NAT architecture
    - Most of NAT architectures are over-designed
  - The training objective of Mask-Predict is much similar to the inference objective than that of AT
  - Better utilize the knowledge from multilingual BERT or other pre-trained models with similar training objective
  - Still under-explored, many components can be improved
- Cons
  - Also face the exposure bias issue
    - The refinement prediction is conditioned on the previous predictions
  - The length is fixed in the inference process
    - Inflexible
    - Limit the further improvement of translation performance
    - The reason why Levenshtein Transformer

Levenshtein Transformer

- Description
  - Optimize Mask-Predict with insertion and deletion actions
- Key idea
  - A refinement iteration
    - Delete tokens
    - Insert placeholders
    - Predict placeholders (Fill-in Tokens)
- Stopping condition
  - Unchanged refinement & Maximum Round

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



Training

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

Delete

Insert

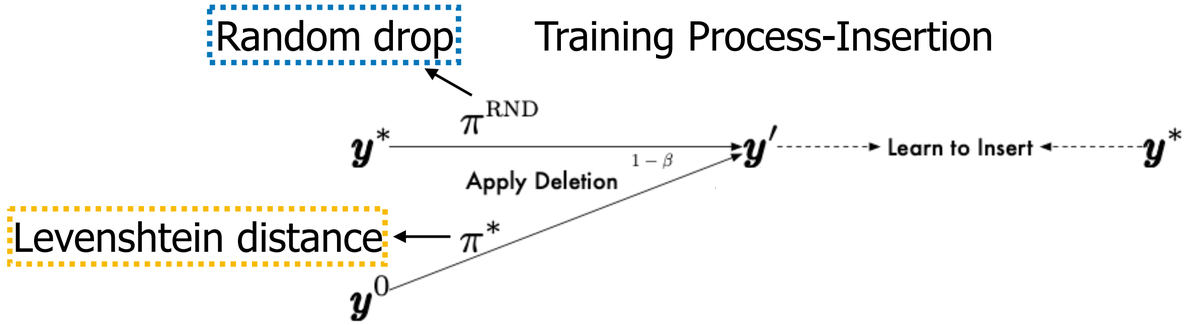
Predict

$$\pi_\theta^{\text{del}}(d|i,\boldsymbol{y}) = \text{softmax}\left(\boldsymbol{h}_i \cdot A^\top\right), \quad i = 1, \ldots n-1$$

$$\pi_\theta^{\text{plh}}(p|i,\boldsymbol{y}) = \text{softmax}\left(\texttt{concat}(\boldsymbol{h}_i, \boldsymbol{h}_{i+1}) \cdot B^\top\right), \quad i = 0, \ldots n-1$$

$$\pi_\theta^{\text{tok}}(t|i,\boldsymbol{y}) = \text{softmax}\left(\boldsymbol{h}_i \cdot C^\top\right), \quad \forall y_i = \texttt{<PLH>}$$

- Insertion (Prediction) Learning
  - y': Random drop words from y^* (ground-truth) or apply deletion to y^0 (noise-added ground-truth or model predictions)
  - Learn(y^*,y')



Training

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

Delete

Insert

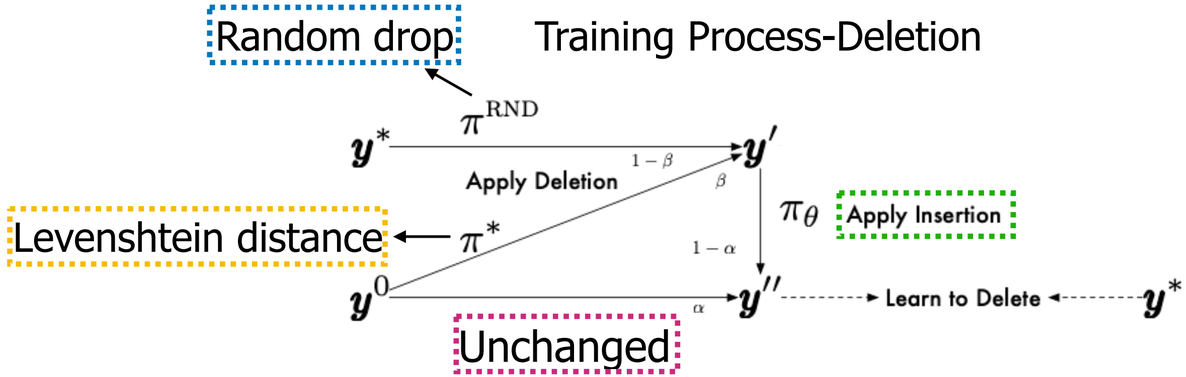
Predict

$$\pi_\theta^{\text{del}}(d|i,\boldsymbol{y}) = \text{softmax}\left(\boldsymbol{h}_i \cdot A^\top\right), \quad i = 1, \ldots n-1$$

$$\pi_\theta^{\text{plh}}(p|i,\boldsymbol{y}) = \text{softmax}\left(\texttt{concat}(\boldsymbol{h}_i, \boldsymbol{h}_{i+1}) \cdot B^\top\right), \quad i = 0, \ldots n-1$$

$$\pi_\theta^{\text{tok}}(t|i,\boldsymbol{y}) = \text{softmax}\left(\boldsymbol{h}_i \cdot C^\top\right), \quad \forall y_i = \texttt{<PLH>}$$

- Insertion (Prediction) Learning
  - y': Random drop words from y^* (ground-truth) or apply deletion to y^0 (noise-added ground-truth or model predictions)
  - Learn(y^*,y')
- Deletion Learning
  - y': Random drop words from y^* or apply deletion to y^0
  - y'': Apply insertion to y' or unchanged y^0
  - Learn(y^*,y'')



Settings&Results

- Task
  - Machine Translation
    - Ro-En (600K); En-De (4.5M); En-Ja (2M)
  - Summarization
    - Gigaword (3.8M)
- Settings
  - Base: (model) 512 / (FFN hidden) 2048
  - 8*V100
    - Batch size=8192*8

	Dataset	Metric	Transformer greedy	Transformer beam4	Levenshtein Transformer oracle	Levenshtein Transformer distillation
Quality ↑	Ro-En	BLEU	31.67	32.30	33.02	**33.26**
	En-De	BLEU	26.89	27.17	25.20	**27.27**
	En-Ja	BLEU	42.86	**43.68**	42.36	43.17
	Gigaword	ROUGE-1	37.31	**37.87**	36.14	37.40
		ROUGE-2	18.10	**18.92**	17.14	18.33
		ROUGE-L	34.65	**35.13**	34.34	34.51
Speed ↓	Ro-En	Latency (ms) / I_{DEC}	326 / 27.1	349 / 27.1	97 / 2.19	**90 / 2.03**
	En-De	Latency (ms) / I_{DEC}	343 / 28.1	369 / 28.1	126 / 2.88	**92 / 2.05**
	En-Ja	Latency (ms) / I_{DEC}	261 / 22.6	306 / 22.6	112 / 2.61	**106 / 1.97**
	Gigaword	Latency (ms) / I_{DEC}	116 / 10.1	149 / 10.1	98 / 2.32	**84 / 1.73**

Decode (Refine) iterations

- Student Levenshtein Transformer can even beat its AT teacher
- Levenshtein Transformer provides considerable inference speedup

Results on Automatic Post-Editing

- Automatic Post-Editing (APE)
  - Data

Dataset		MT-Train	APE-Train	Valid	Test	Vocabulary
Synthetic	WMT'16 Ro-En	300,000	308,319	1999	1999	34,983
	WMT'14 En-De	2,250,000	2,250,967	3000	3003	37,009
	WAT'17 En-Ja	1,000,000	1,000,000	1790	1812	17,952 / 17,801
Real	WMT'17 APE En-De	4,391,180	526,368 (fake) + 24,000 (real)	2000	2000	40,349

- Fake: Noise-added data
- Real: Human PE data

  - Results

Dataset		MT system	Do-Nothing	Transformer	Levenshtein Transformer		
					Scratch	Zero-shot	Fine-tune
Synthetic	Ro-En	PBMT	27.5 / 52.6	28.9 / 52.8	29.1 / **50.4**	**30.1** / 51.7	–
		NMT	26.2 / 56.5	26.9 / 55.6	**28.3** / **53.6**	28.0 / 55.8	–
	En-De	PBMT	15.4 / 69.4	22.8 / 61.0	**25.8** / **56.6**	16.5 / 69.6	–
	En-Ja	NMT	37.7 / 48.0	41.0 / 44.9	**42.2** / **44.3**	39.4 / 47.5	–
Real	En-De	PBMT	62.5 / 24.5	67.2 / 22.1	66.9 / 21.9	59.6 / 28.7	**70.1** / **19.2**

- Do-Nothing: Without APE
- Transformer: Transformer-based APE system
- Scratch: LT-based APE system
- Zero-shot: Directly using LT-based NMT
- Fine-tune: Tune LT-based NMT with APE data

😄: **No En-De NMT baselines**

Discussion

- Pros
  - Alleviate the fixed-length limitation of Mask-Predict
    - Flexible in dealing with the length of predictions
  - Performance & Speedup
    - Slight performance drop
    - ~3x speedup
  - The training objective of LT is very similar to the inference objective
  - Potential in input-output-similar generation tasks
    - Grammatical error correction
- Cons
  - Each refinement iteration consisting of three sub-actions
    - Delete-Insert-Predict
    - Jointly predict?

NAT and knowledge distillation

- Current NAT methods benefit a lot from sentence-level knowledge distillation from teachers
  - Reduce multi-modality
  - Does NAT really need it?
    - On the WMT'16 Ro-En Translation task
      - Pure & Clean raw data
      - Mask-Predict
        - Raw: 32.86; Dist: 33.08
      - Levenshtein Transformer
        - Raw: 33.02; Dist: 33.26

Mask-Predict

Iterations	WMT'14 EN-DE		WMT'16 EN-RO	
	Raw	Dist	Raw	Dist
$T = 1$	10.64	**18.05**	21.22	**27.32**
$T = 4$	22.25	**25.94**	31.40	**32.53**
$T = 10$	24.61	**27.03**	32.86	**33.08**

Levenshtein Transformer

	Dataset	Metric	Transformer		Levenshtein Transformer	
			greedy	beam4	oracle	distillation
Quality ↑	Ro-En	BLEU	31.67	32.30	33.02	**33.26**
	En-De	BLEU	26.89	27.17	25.20	**27.27**
	En-Ja	BLEU	42.86	**43.68**	42.36	43.17
	Gigaword	ROUGE-1	37.31	**37.87**	36.14	37.40
		ROUGE-2	18.10	**18.92**	17.14	18.33
		ROUGE-L	34.65	**35.13**	34.34	34.51
Speed ↓	Ro-En	Latency (ms) / I_{DEC}	326 / 27.1	349 / 27.1	97 / 2.19	**90 / 2.03**
	En-De	Latency (ms) / I_{DEC}	343 / 28.1	369 / 28.1	126 / 2.88	**92 / 2.05**
	En-Ja	Latency (ms) / I_{DEC}	261 / 22.6	306 / 22.6	112 / 2.61	**106 / 1.97**
	Gigaword	Latency (ms) / I_{DEC}	116 / 10.1	149 / 10.1	98 / 2.32	**84 / 1.73**

Conditional Generation from Knowledge Distillation Perspective

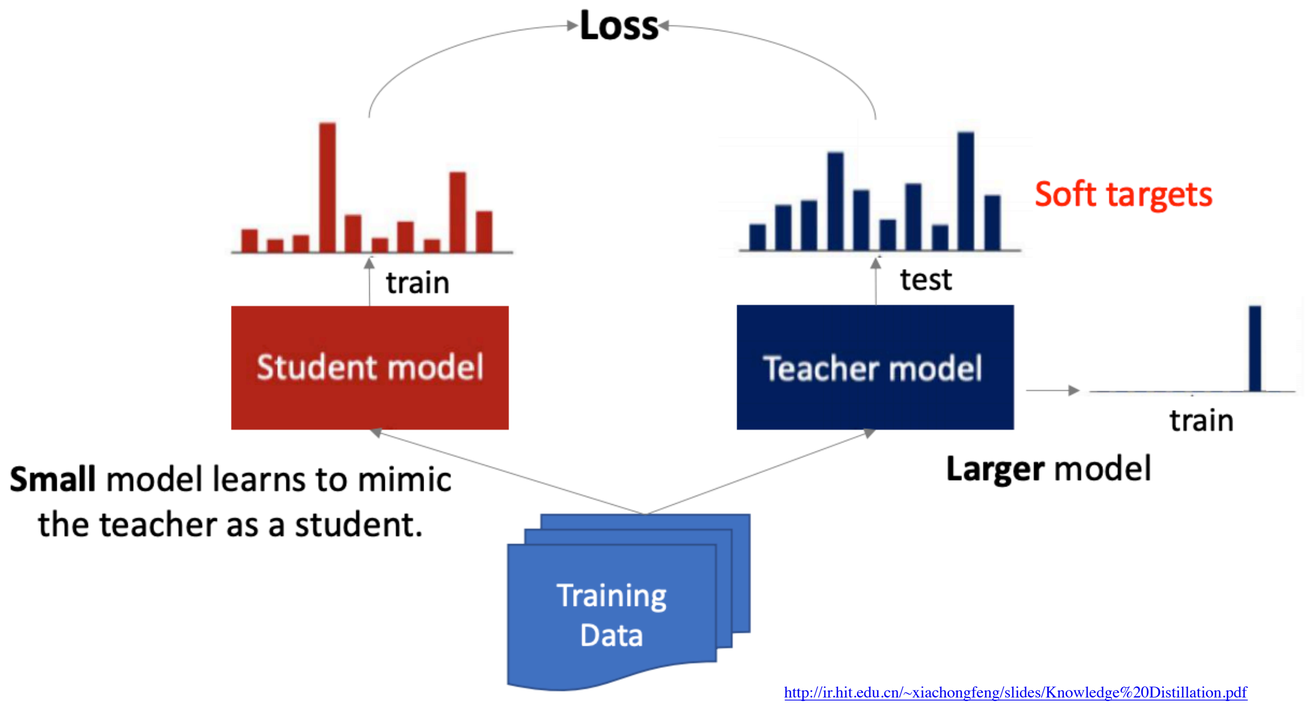
Longyue Wang, Xuebo Liu, Liang Ding
December 31, 2019

- Distilling the Knowledge in a Neural Network (NIPS 2014, Deep Learning Workshop)
- Sequence-level Knowledge Distillation (EMNLP 2016)
- Understanding Knowledge Distillation in Non-autoregressive MT (ICLR 2020)
- Data-dependent Gaussian Prior Objective for Language Generation (ICLR 2020)

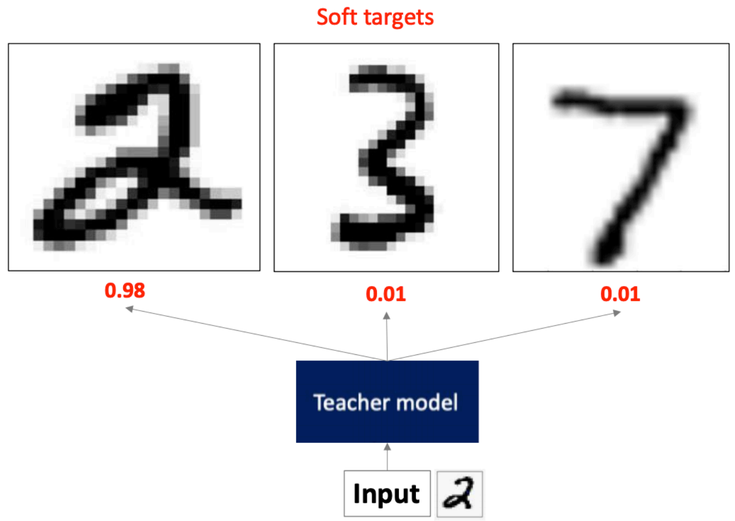
- Distilling the Knowledge in a Neural Network (NIPS 2014, Deep Learning Workshop)
- Sequence-level Knowledge Distillation (EMNLP 2016)
- Understanding Knowledge Distillation in Non-autoregressive MT (ICLR 2020)
- Data-dependent Gaussian Prior Objective for Language Generation (ICLR 2020)

Knowledge Distillation

Knowledge distillation (KD) is a process of distilling or transferring the knowledge from a (set of) large, cumbersome model(s) to a lighter, easier-to-deploy single model, without significant loss in performance.



http://ir.hit.edu.cn/~xiachongfeng/slides/Knowledge%20Distillation.pdf



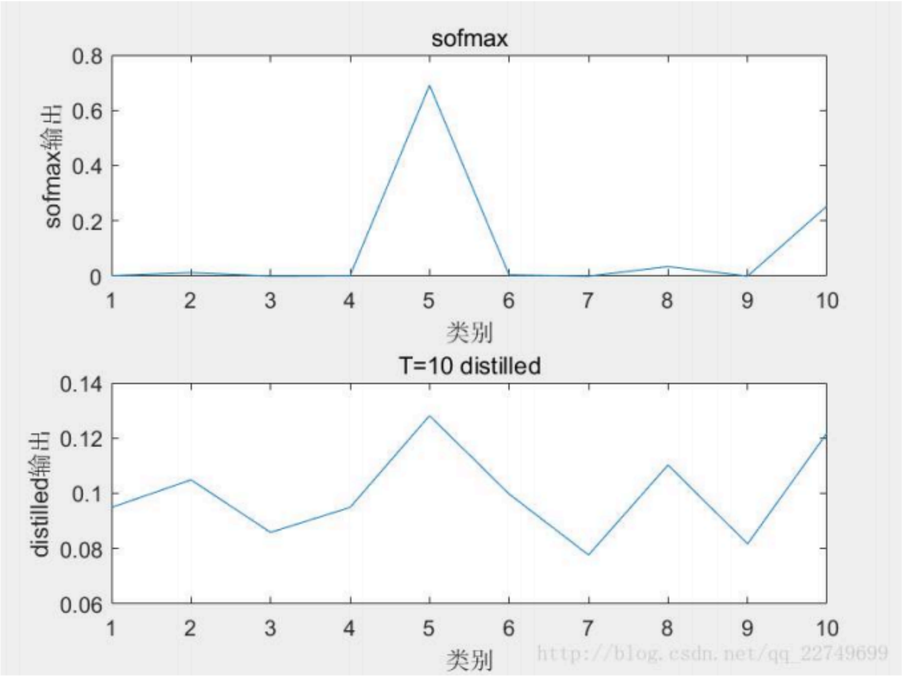
[1] Hinton et al. Distilling the Knowledge in a Neural Network. In NIPS 2014 Deep Learning Workshop.

Soft distilled target

Logits

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

Temperature



https://blog.csdn.net/qq_22749699/article/details/79460817

[1] Hinton et al. Distilling the Knowledge in a Neural Network. In NIPS 2014 Deep Learning Workshop.

Conclusions & Thought

- Discrete distribution (one-hot)->Continuous distribution (soft-target)
- Model compression / Inference speed-up
- Soft targets in KD increases entropy for classification task
- KD is a new perspective that can be used for any task
- Is KD a kind of data augmentation approach (e.g., classification task)? but sometimes it increases the deterministics (e.g., sentence level KD)

[1] Li et al. Data-dependent Gaussian Prior Objective for Language Generation. In ICLR 2019. (Oral)

- Distilling the Knowledge in a Neural Network (NIPS 2014, Deep Learning Workshop)
- Sequence-level Knowledge Distillation (EMNLP 2016)
- Understanding Knowledge Distillation in Non-autoregressive MT (ICLR 2020)
- Data-dependent Gaussian Prior Objective for Language Generation (ICLR 2020)

Sequence Level Knowledge Distillation

- **Negative Log Likelihood for multi-class classifier:**

$$\mathcal{L}_{\text{NLL}}(\theta) = -\sum_{k=1}^{|\mathcal{V}|} \mathbb{1}\{y = k\} \log p(y = k \,|\, x; \theta)$$

- **Pre-trained teacher model** θ_T
  - KD Loss:

$$\mathcal{L}_{\text{KD}}(\theta; \theta_T) = -\sum_{k=1}^{|\mathcal{V}|} q(y = k \,|\, x; \theta_T) \times \log p(y = k \,|\, x; \theta)$$

  - Interpolation Loss:

$$\mathcal{L}(\theta; \theta_T) = (1 - \alpha)\mathcal{L}_{\text{NLL}}(\theta) + \alpha \mathcal{L}_{\text{KD}}(\theta; \theta_T)$$

- Word-level KD

$$\mathcal{L}_{\text{WORD-KD}} = -\sum_{j=1}^{J}\sum_{k=1}^{|\mathcal{V}|} q(t_j = k \,|\, \mathbf{s}, \mathbf{t}_{<j}) \times \log p(t_j = k \,|\, \mathbf{s}, \mathbf{t}_{<j})$$

- Sequence-level KD

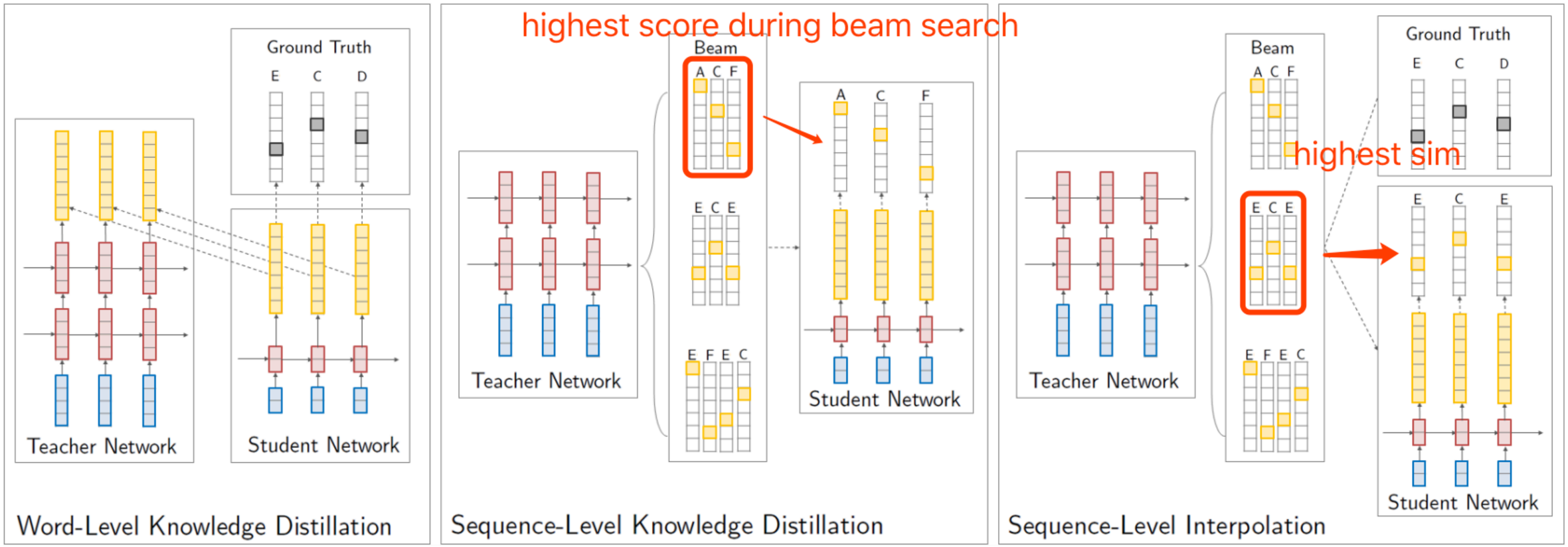
$$\begin{aligned}\mathcal{L}_{\text{SEQ-KD}} &\approx -\sum_{\mathbf{t} \in \mathcal{T}} \mathbb{1}\{\mathbf{t} = \hat{\mathbf{y}}\} \log p(\mathbf{t} \,|\, \mathbf{s}) \\ &= -\log p(\mathbf{t} = \hat{\mathbf{y}} \,|\, \mathbf{s})\end{aligned}$$

- Sequence-level Interpolation KD

$$\mathcal{L} = (1 - \alpha)\mathcal{L}_{\text{SEQ-NLL}} + \alpha \mathcal{L}_{\text{SEQ-KD}}$$

[1] Kim and Rush. Sequence-level Knowledge Distillation. In EMNLP 2016.

Sequence Level Knowledge Distillation



[1] Kim and Rush. Sequence-level Knowledge Distillation. In EMNLP 2016.

Experiment results

Model	$\text{BLEU}_{K=1}$	$\Delta_{K=1}$	$\text{BLEU}_{K=5}$	$\Delta_{K=5}$	PPL	$p(\mathbf{t} = \hat{\mathbf{y}})$
English → German WMT 2014						
Teacher Baseline 4×1000 (Params: 221m)	17.7	–	19.5	–	6.7	1.3%
Baseline + Seq-Inter	19.6	+1.9	19.8	+0.3	10.4	8.2%
Student Baseline 2×500 (Params: 84m)	14.7	–	17.6	–	8.2	0.9%
Word-KD	15.4	+0.7	17.7	+0.1	8.0	1.0%
Seq-KD	18.9	**+4.2**	19.0	+1.4	22.7	16.9%
Baseline + Seq-Inter	18.5	+3.6	18.7	+1.1	11.3	5.7%
Word-KD + Seq-Inter	18.3	+3.6	18.5	+0.9	11.8	6.3%
Seq-KD + Seq-Inter	18.9	**+4.2**	19.3	**+1.7**	15.8	7.6%
Seq-KD + Word-KD	18.7	+4.0	18.9	+1.3	10.9	4.1%
Seq-KD + Seq-Inter + Word-KD	18.8	+4.1	19.2	+1.6	14.8	7.1%
Student Baseline 2×300 (Params: 49m)	14.1	–	16.9	–	10.3	0.6%
Word-KD	14.9	+0.8	17.6	+0.7	10.9	0.7%
Seq-KD	18.1	+4.0	18.1	+1.2	64.4	14.8%
Baseline + Seq-Inter	17.6	+3.5	17.9	+1.0	13.0	10.0%
Word-KD + Seq-Inter	17.8	+3.7	18.0	+1.1	14.5	4.3%
Seq-KD + Seq-Inter	18.2	+4.1	18.5	+1.6	40.8	5.6%
Seq-KD + Word-KD	17.9	+3.8	18.8	+1.9	44.1	3.1%
Seq-KD + Seq-Inter + Word-KD	18.5	**+4.4**	18.9	**+2.0**	97.1	5.9%
Thai → English IWSLT 2015						
Teacher Baseline 2×500 (Params: 47m)	14.3	–	15.7	–	22.9	2.3%
Baseline + Seq-Inter	15.6	+1.3	16.0	+0.3	55.1	6.8%
Student Baseline 2×100 (Params: 8m)	10.6	–	12.7	–	37.0	1.4%
Word-KD	11.8	+1.2	13.6	+0.9	35.3	1.4%
Seq-KD	12.8	+2.2	13.4	+0.7	125.4	6.9%
Baseline + Seq-Inter	12.9	+2.3	13.1	+0.4	52.8	2.5%
Word-KD + Seq-Inter	13.0	+2.4	13.7	+1.0	58.7	3.2%
Seq-KD + Seq-Inter	13.6	+3.0	14.0	+1.3	106.4	3.9%
Seq-KD + Word-KD	13.7	+3.1	14.2	+1.5	67.4	3.1%
Seq-KD + Seq-Inter + Word-KD	14.2	**+3.6**	14.4	**+1.7**	117.4	3.2%

[1] Kim and Rush. Sequence-level Knowledge Distillation. In EMNLP 2016.

- Distilling the Knowledge in a Neural Network (NIPS 2014, Deep Learning Workshop)
- Sequence-level Knowledge Distillation (EMNLP 2016)
- Understanding Knowledge Distillation in Non-autoregressive MT (ICLR 2020)
- Data-dependent Gaussian Prior Objective for Language Generation (ICLR 2020)

Research Questions & Contributions

- **How does distillation reduce the "modes"? can we measure this reduction quantitatively?**
- **The performance of NAT models is largely bounded by the choice of AT teacher. can we close the performance gap with standard AT models?**

- **visualize how modes are reduced by distillation**
- **propose metrics for measuring complexity and faithfulness**
- **find a strong correlation between the capacity of NAT model and the optimal dataset complexity**
- **propose approaches to adjust the complexity of the distilled data to match the model's capacity**

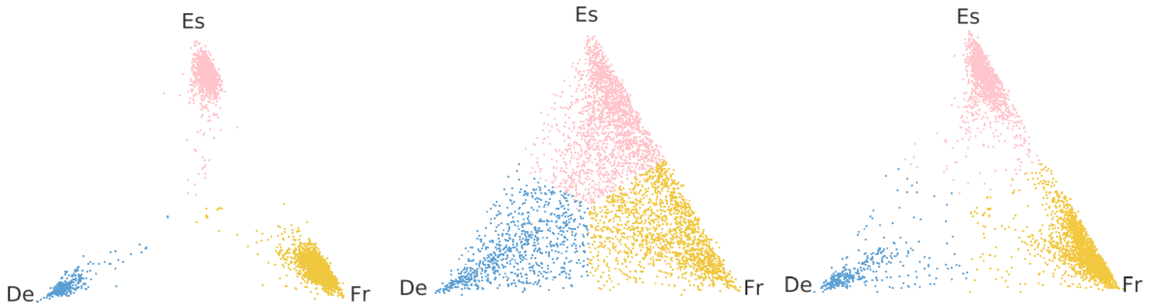
[1] Zhou et al. Understanding Knowledge Distillation in Non-autoregressive Machine Translation. In ICLR 2020.

KD's reduction of the "modes"

- **Dataset:**
  - Three explicit output modes: En-De/Es/Fr
- **Models:**
  - NAT (Gu et. al., 2018) & AT (Vaswani et. al., 2017)
  - 300, 000 steps using maximum likelihood
- **Approach:**
  - translate the EN sentences in the val&test sets
  - visualize the estimated probability distribution of language classes (average of the posterior probability of each token

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

$$\Delta^2 = \{(p_{\text{de}}, p_{\text{es}}, p_{\text{fr}}) | p_k \in (0,1), p_{\text{de}} + p_{\text{es}} + p_{\text{fr}} = 1\}$$



(a) AT Baseline (b) NAT Baseline (d) NAT Distill

- AT model prefers to generate the whole sequence in one language
- Training with reduced modes is essential for NAT

[1] Zhou et al. Understanding Knowledge Distillation in Non-autoregressive Machine Translation. In ICLR 2020.

Quantitative measures for parallel data

- **Complexity (conditional entropy)**

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

- **faithfulness (cross entropy v.s. real data)**

KL-divergence of the alignment distribution between the real parallel data

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

$$\begin{aligned}
\mathcal{H}(\mathbf{Y}|\mathbf{X} = \boldsymbol{x}) &= \sum_{\boldsymbol{y} \in \mathcal{Y}} p(\boldsymbol{y}|\boldsymbol{x}) \log p(\boldsymbol{y}|\boldsymbol{x}) \\
&\approx \sum_{\boldsymbol{y} \in \mathcal{Y}} \prod_{t=1}^{T_y} p(y_t|\boldsymbol{x}))(\sum_{t=1}^{T_y} \log p(y_t|\boldsymbol{x})) && \text{asm.1: conditional independence} \\
&\approx \sum_{t=1}^{T_y} \sum_{y_t \in \mathcal{A}(\boldsymbol{x})} p(y_t|\mathrm{Align}(y_t)) \log p(y_t|\mathrm{Align}(\mathrm{y_t})) && \text{asm.2: alignment model} \\
&= \sum_{t=1}^{T_x} \mathcal{H}(y|x = x_t)
\end{aligned}$$

- *Assumption* 1: We assume the target tokens are independent given the source sentence. Then the conditional entropy of a sentence can be converted into the sum of entropy of target words conditioned on the source sentence $\boldsymbol{x}$.
- *Assumption* 2: We assume the distribution of $p(y_t|\boldsymbol{x})$ follows an alignment model (Dyer et al., 2013)[3] where y_t is is generated from the word alignment distribution $p(y_t|\mathrm{Align}(\mathrm{y_t}))$. This makes it possible to simplify the conditional entropy to the sum of entropy of target words conditioned on the aligned source words.

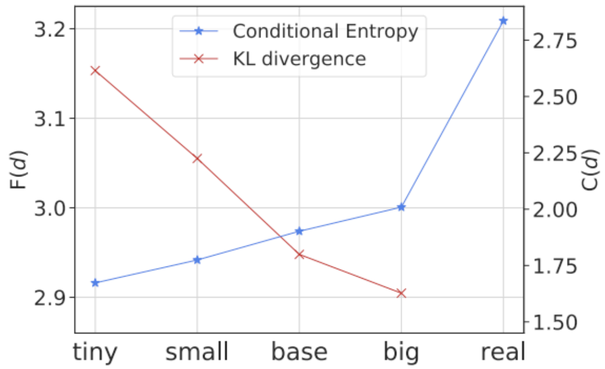
[1] Zhou et al. Understanding Knowledge Distillation in Non-autoregressive Machine Translation. In ICLR 2020.

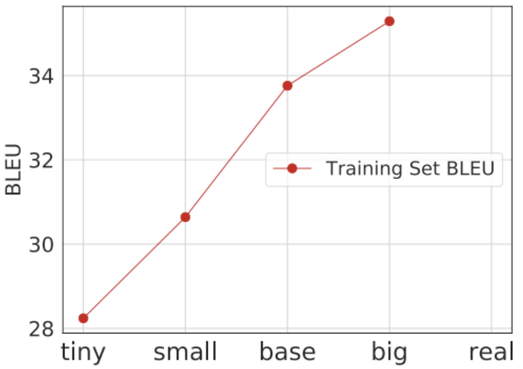
C & F correlate with the performance of an NAT model

- **Settings and BLEU scores**

- **Vanilla NAT (Gu et al., 2018):** Similarly to §3.1, we use a simplified version where the decoder's inputs are directly copied from the encoder without considering latent variables.
- **FlowSeq (Ma et al., 2019):** FlowSeq adopts normalizing flows (Kingma & Dhariwal, 2018) as the latent variables to model the mappings from source sentences to a latent space.
- **NAT with Iterative Refinement (iNAT, Lee et al., 2018):** iNAT extends the vanilla NAT by iteratively reading and refining the translation. The number of iterations is set to 10 for decoding.
- **Insertion Transformer (InsT, Stern et al., 2019):** InsT adopts a similar architecture as iNAT while generating the sequence by parallel insertion operations. Here, we only consider InsT trained with uniform loss as described in the original paper.
- **MaskPredict (MaskT, Ghazvininejad et al., 2019):** MaskT adopts a masked language model (Devlin et al., 2018) to progressively generate the sequence from an entirely masked input. The number of iterations is set to be 10.
- **Levenshtein Transformer (LevT, Gu et al., 2019):** LevT uses similar architectures as in InsT and MaskT while generating based on both insertion and deletion operations. We experiment with a base and big LevT model (LevT and LevT-big in Tab. 2).

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	≈3	$3k \ll n$



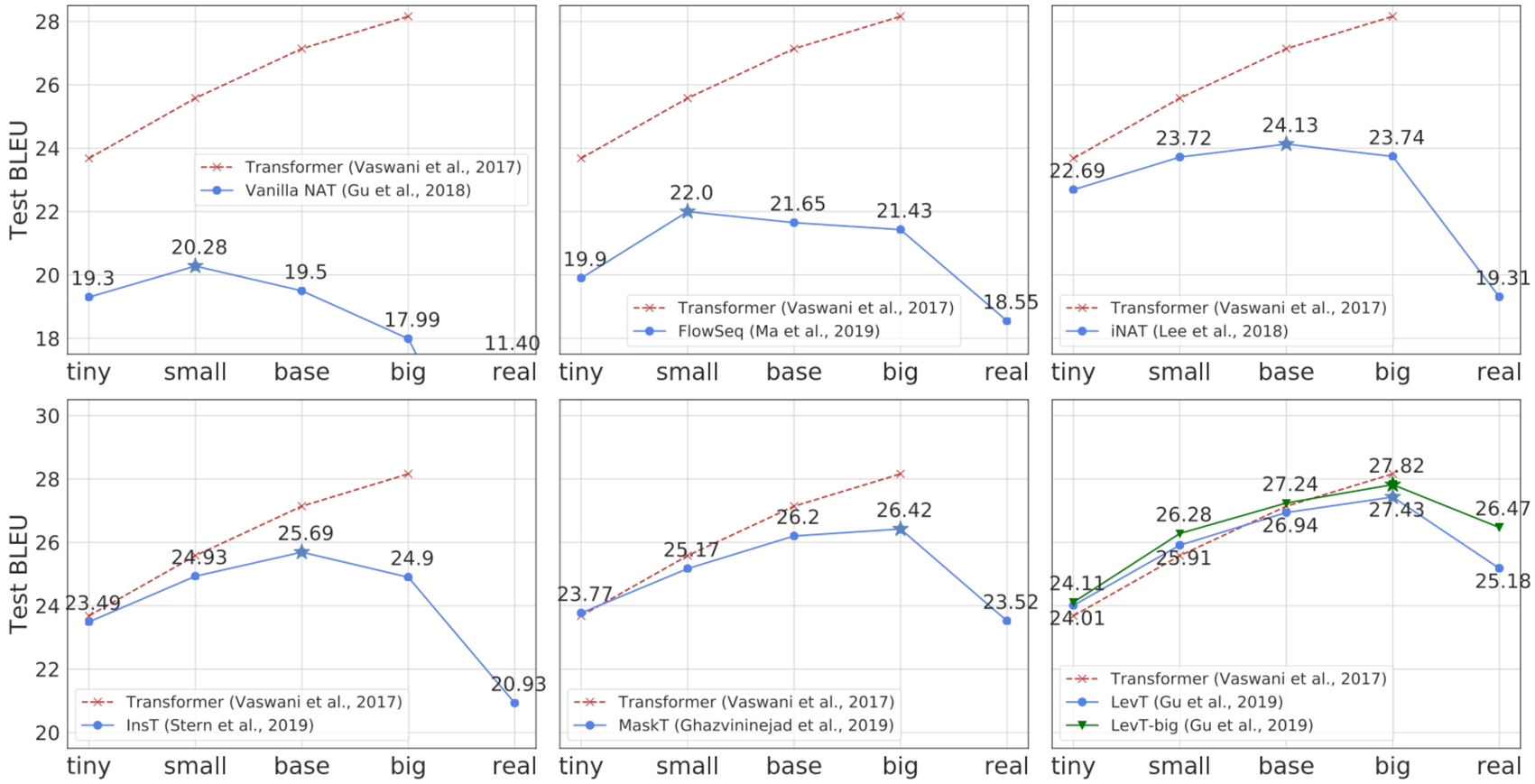


AT capacity ↑=> C ↑, multi-modality ↑ , more faithfully BLEU score of the distilled data ↑

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**

[1] Zhou et al. Understanding Knowledge Distillation in Non-autoregressive Machine Translation. In ICLR 2020.

C & F correlate with the performance of an NAT model

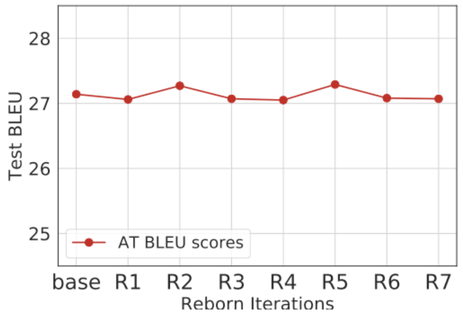


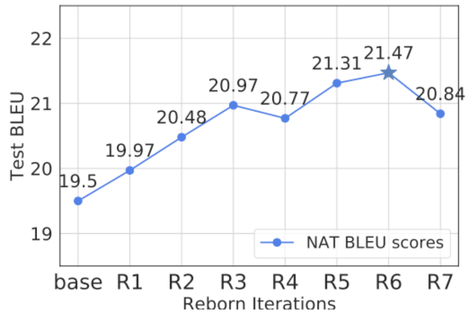
weaker NAT students prefer distilled data with smaller C

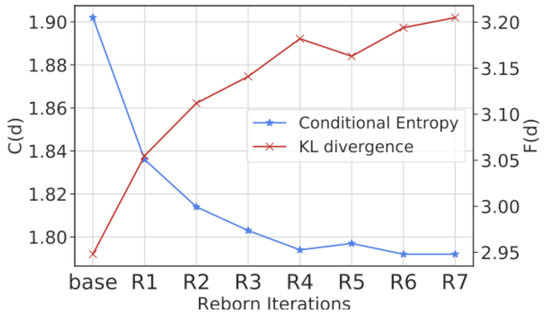
[1] Zhou et al. Understanding Knowledge Distillation in Non-autoregressive Machine Translation. In ICLR 2020.

Techniques to simplify the data further (improve KD)

Born-Again Networks. (Furlanello et al., 2018)





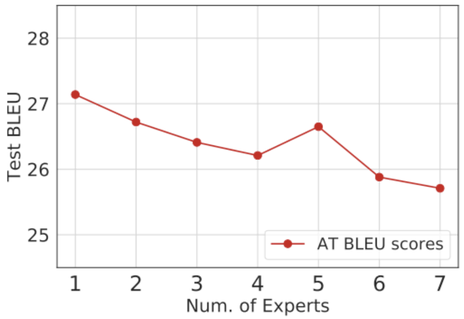


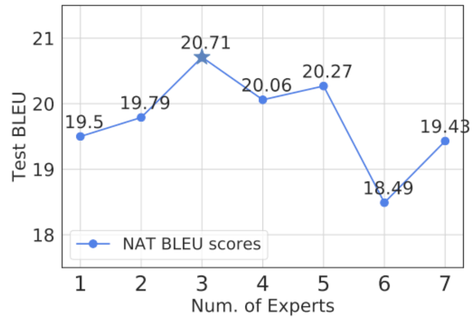
Sequence-Level Interpolation. Kim & Rush (2016)

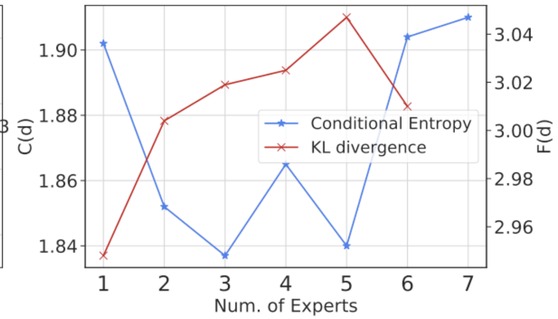
d	$C(d)$	$F(d)$	BLEU
base	1.902	2.948	26.94
base-inter	1.908	2.916	**27.32**

Mixture-of-Experts. (MoE; Shen et al. (2019))

Low C & High F augment KD







[1] Zhou et al. Understanding Knowledge Distillation in Non-autoregressive Machine Translation. In ICLR 2020.

Conclusions

- Systematic experiments to explain **why knowledge distillation is crucial** to NAT training
- **Two metrics** to measure **complexity** and **faithfulness**
- **AT teacher** produces a **less multimodal distribution** that is easier to model with NAT
- We need a teacher model with **low complexity and high faithfulness**

[1] Zhou et al. Understanding Knowledge Distillation in Non-autoregressive Machine Translation. In ICLR 2020.

Thoughts

- **How to directly train a teacher model with low complexity and high faithfulness to facilitate KD or NA generation?**
  - Faithfulness ↑: through reconstruction?
  - Complexity ↓: IB theory / RL may help?
- **Use the metrics (complexity and faithfulness) of training data to interpret Back Translation?**

[1] Zhou et al. Understanding Knowledge Distillation in Non-autoregressive Machine Translation. In ICLR 2020.

- Distilling the Knowledge in a Neural Network (NIPS 2014, Deep Learning Workshop)
- Sequence-level Knowledge Distillation (EMNLP 2016)
- Understanding Knowledge Distillation in Non-autoregressive MT (ICLR 2020)
- Data-dependent Gaussian Prior Objective for Language Generation (ICLR 2020)

Problems of MLE in language generation tasks

- **Exposure bias**

  not exposed to the full range of errors during training

- **Loss mismatch**

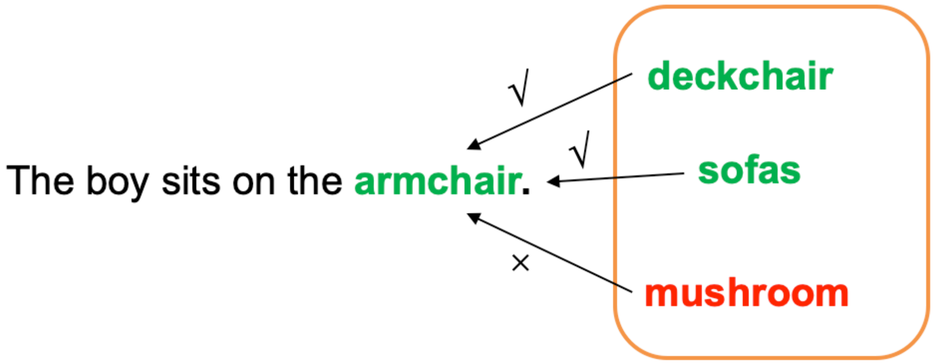
  maximize the log-likelihood in training / BLEU in testing

- **Generation diversity**

  dull, generic, repetitive and short-sighted

- **<u>Negative diversity ignorance</u>**

  fails to assign proper scores to different incorrect model outputs, which means that all incorrect outputs are treated equally during training



MLE:
once-for-all matching

We need a token-wise structure to accurately assign scores to diverse (especially negative) model predictions.

[1] Li et al. Data-dependent Gaussian Prior Objective for Language Generation. In ICLR 2020.

Gaussian Prior Objective[1]

- Goal
  - Alleviate the "negative diversity ignorance" issue of MLE objective function
- Key idea
  - Add a simple yet effective model-agnostic prior objective:

$$\mathcal{L}_{\mathcal{O}}(\boldsymbol{\theta}, q) = KL(q(\boldsymbol{y}) \| p_\theta(\boldsymbol{y}|\boldsymbol{x})),$$

$$\min_\theta \mathcal{L}_{\mathbf{MLE}}(\boldsymbol{\theta}) + \lambda \mathcal{L}_{\mathcal{O}}(\boldsymbol{\theta}, q),$$

- Research question

① **How to calculate** $q(y)$**?**

- Method
  - Obtain $\tilde{f}(\tilde{y}, y^*)$ by cosine similarity of pre-trained embedding distance
  - Smooth $\tilde{f}(\tilde{y}, y^*)$ to $f(\tilde{y}, y^*)$ by Gaussian probability density function
  - Calculate $q(y)$ by a temperature-based softmax function

$$q(y^*) = \frac{exp(f(\tilde{\boldsymbol{y}}, y^*)/T)}{\sum_j exp(f(\tilde{y}_j, y^*)/T)},$$

[1] Li et al. Data-dependent Gaussian Prior Objective for Language Generation. In ICLR 2020.

Experiments

- **Supervised NMT**
  - WMT14 English-German (EN-DE) 4.43M
  - WMT14 English-French (EN-FR) 36M
  - WMT16 English-Romanian (EN-RO) 0.6M / 2.8M

System	EN–DE	EN–FR	EN–RO	EN–RO + STD
Vaswani et al. (2017) (base)	27.30	38.10	-	-
Vaswani et al. (2017) (big)	28.40	41.00	-	-
Transformer (base)	27.35	38.44	33.22	36.68
+ **D2GPo**	**27.93++**	**39.23++**	**34.00+**	**37.11+**
Transformer (big)	28.51	41.05	33.45	37.55
+ **D2GPo**	**29.10+**	**41.77++**	**34.13+**	**37.92+**

[1] Li et al. Data-dependent Gaussian Prior Objective for Language Generation. In ICLR 2020.

Experiments

- **Unsupervised NMT**
  - Baseline: MASS (Song et. al., 2019) pre-trained on EN, FR, DE and RO monol data samples from WMT17-18 News Crawl
  - Fine-tune with D2GPo loss for enhancements

Method	EN–FR	FR–EN	EN–DE	DE–EN	EN–RO	RO–EN
Artetxe et al. (2017)	15.13	15.56	6.89	10.16	-	-
Lample et al. (2017)	15.05	14.31	9.75	13.33	-	-
Yang et al. (2018)	16.97	15.58	10.86	14.62	-	-
Lample et al. (2018)	25.14	24.18	17.16	21.00	21.18	19.44
XLM (Lample & Conneau, 2019)	33.40	33.30	27.00	34.30	33.30	31.80
MASS (Song et al., 2019)	37.50	34.90	28.30	35.20	35.20	33.10
MASS + **D2GPo**	**37.92**	**34.94**	**28.42**	**35.62**	**36.31**	**33.41**

[1] Li et al. Data-dependent Gaussian Prior Objective for Language Generation. In ICLR 2020.

Experiments

- **Text Summarization**
  - Annotated Gigaword corpus (3.8M training/ 400K validation/ 2000 test)
  - Fine-tune with D2GPo loss for enhancements

	Model	ROUGE-1	ROUGE-2	ROUGE-L
Supervised	RNN-based seq2seq	35.50	15.54	32.45
	Nallapati et al. (2016)	34.97	17.17	32.70
Semi-supervised	MLM pre-training (Song et al., 2019)	37.75	18.45	34.85
	DAE pre-training (Song et al., 2019)	35.97	17.17	33.14
	MASS pre-training (Song et al., 2019)	38.73	19.71	35.96
	MASS + **D2GPo**	**39.23**	**20.11**	**36.48**

[1] Li et al. Data-dependent Gaussian Prior Objective for Language Generation. In ICLR 2020.

Experiments

- **Storytelling**
  - Baseline: Hierarchical story generation (Fan et. al., 2018)
  - Fine-tune with D2GPo loss for enhancements

Model	Params	Valid Perplexity	Test Perplexity
GCNN LM	123.4 M	54.50	54.79
GCNN + self-attention LM	126.4 M	51.84	51.18
LSTM seq2seq	110.3 M	46.83	46.79
Conv seq2seq	113.0 M	45.27	45.54
Conv seq2seq + self-attention	134.7 M	37.37	37.94
Ensemble: Conv seq2seq + self-attention	270.3 M	36.63	36.93
Fusion: Conv seq2seq + self-attention	255.4 M	36.08	36.56
Conv seq2seq + self-attention + **D2GPo**	134.7 M	**35.56**	**35.74**
Fusion: Conv seq2seq + self-attention + **D2GPo**	255.4 M	**33.82**	**33.90**

Table 4: Perplexity on WRITINGPROMPTS.

[1] Li et al. Data-dependent Gaussian Prior Objective for Language Generation. In ICLR 2020.

Experiments

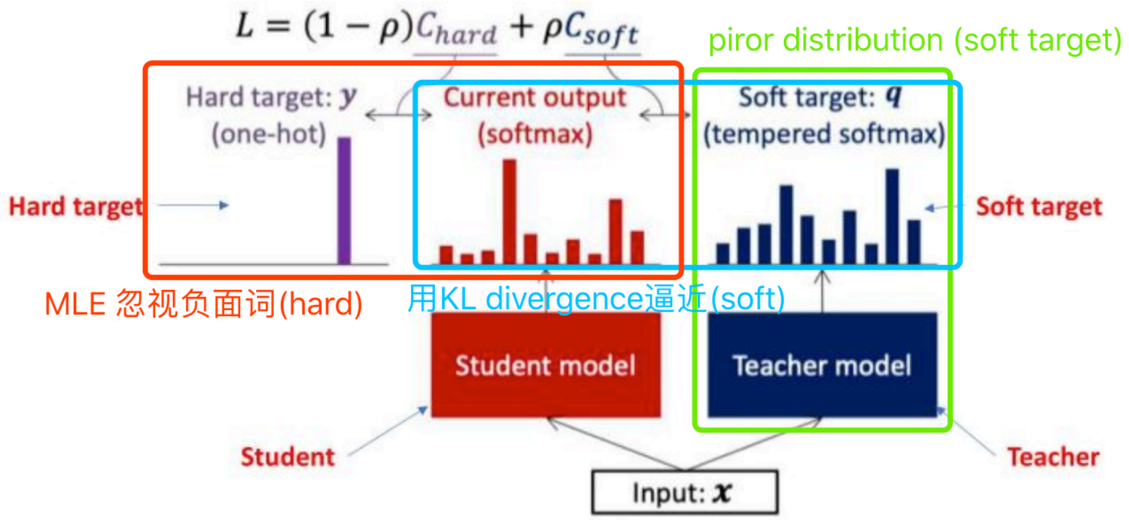
- **Image Captioning**
  - Dataset: MSCOCO 2014
  - Fine-tune with D2GPo loss for enhancements

	BLEU-1	BLEU-4	METEOR	ROUGE-L	CIDEr	SPICE
Att2in (Rennie et al., 2017)	-	31.3	26.0	54.3	101.3	-
Att2all (Rennie et al., 2017)	-	30.0	25.9	53.4	99.4	-
Baseline: Top-down	74.5	33.4	26.1	54.4	105.4	19.2
Baseline + **D2GPo**	**75.2**	**33.6**	**26.3**	**55.1**	**106.6**	**19.7**
Baseline + SCST	77.8	34.4	26.6	56.1	114.3	19.9
Baseline + SCST + **D2GPo**	**78.0**	**34.7**	**26.8**	**56.3**	**116.8**	**20.2**

[1] Li et al. Data-dependent Gaussian Prior Objective for Language Generation. In ICLR 2020.

Conclusions & Thoughts

- Imposes the model-agnostic **prior** from (linguistic) data over the sequence prediction models.
- Alleviating the **negative diversity ignorance** issue, thereby improving the performance of several NLG tasks.



- Can the soft-target (prior) interact with model dynamically?
- Two similar words in the word embedding space are more related words than synonyms. To be precise, this method does increase the generation diversity, but, sentences with a high score assigned by D2GPo are not necessarily highly faithful.
- The essential idea of this paper is very close to KD…

[1] Li et al. Data-dependent Gaussian Prior Objective for Language Generation. In ICLR 2020.

- Distilling the Knowledge in a Neural Network (NIPS 2014, Deep Learning Workshop)
- Sequence-level Knowledge Distillation (EMNLP 2016)
- Understanding Knowledge Distillation in Non-autoregressive MT (ICLR 2020)
- Data-dependent Gaussian Prior Objective for Language Generation (ICLR 2020)

Conclusions

- DK is an **elegant idea** that can be employed in **different scenarios** with **multiple granularities**.
- DK is **helpful for conditional generation** (especially **NA generation**) at the **data level** (Kim, EMNLP2016; Gu, ICLR2018; Zhou, ICLR2020) and **objective level** (Kim, EMNLP2016; Zhou, ICLR2020) etc.