Fully Non-autoregressive Neural Machine Translation: Tricks of the Trade

(Jiatao Gu and Xiang Kong, 2020)

出口　祥之

deguchi@ai.cs.ehime-u.ac.jp

2021/01/20　二宮研論文輪読会

Links

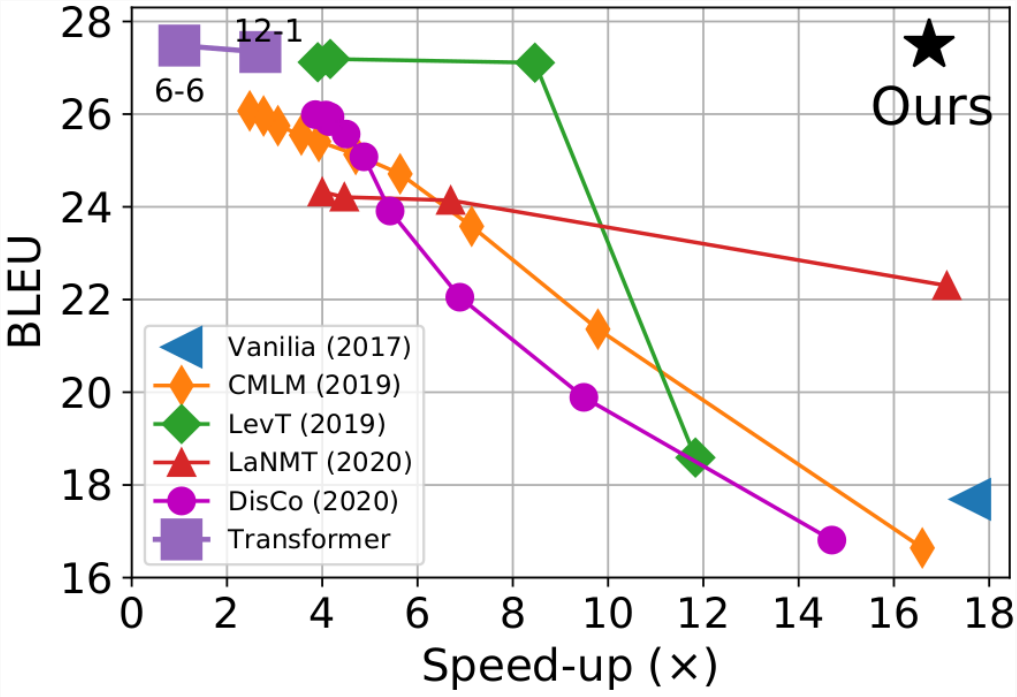
📖 Paper

https://arxiv.org/abs/2012.15833

Introduction

NAT が Autoregressive Transformer と comparable に

- WMT14 En-De: 27.49 BLEU (%)，翻訳速度 16.5x
  - Autoregressive Transformer から性能を劣化させずに翻訳速度を改善



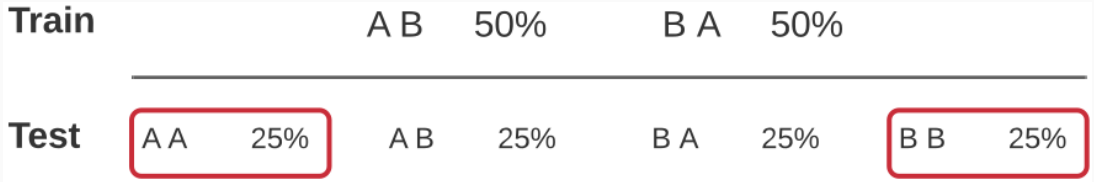
Motivation

NAT とは何か，なぜ性能が劣化しやすいか

- AT と NAT の比較

	AT	NAT
生成確率	$p_\theta(y_t \mid y_{<t}, x_{1:T'})$	$\prod_{t=1}^{T} p_\theta(y_t \mid x_{1:T'})$
生成時間	$O(T)$	$O(1)$

- NAT の問題点: 目的言語文中の単語共起を捉えられない



  - 解決策?: Iterative refinement (← non “fully-NAT”...)

Proposed Method

single forward で生成できる “fully-NAT”

- 従来研究で提案されてきた数々の NAT モデルは出力単語間の **依存削減 (dependency reduction)** を目的として設計
  - cf. 数々の NAT モデル: `https://github.com/kahne/NonAutoregGenProgress`
- 複数の手法を組み合わせて依存削減することにより，fully-NAT で AT の性能に追い付く

Data: Knowledge Distillation (KD)

訓練データの目的言語文を教師モデルの出力に置換

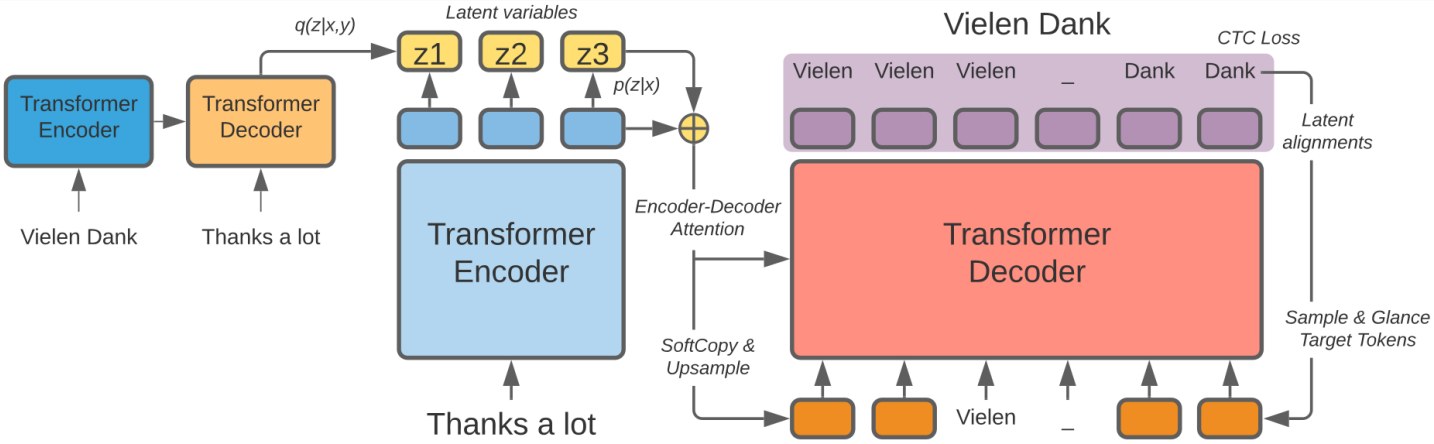
- 最も効果の高い依存削減
- 目的言語文のノイズが減る
- 原言語文との対応関係がより決定的に
- 教師モデルのサイズは NAT モデルに応じて適切に選ぶ必要がある

NAT と KD についての解析 (ICLR 2020):

`https://jiataogu.me/publication/understand-distillation`

Model: Latent Variables

Shu et al. (2020) のモデルを採用



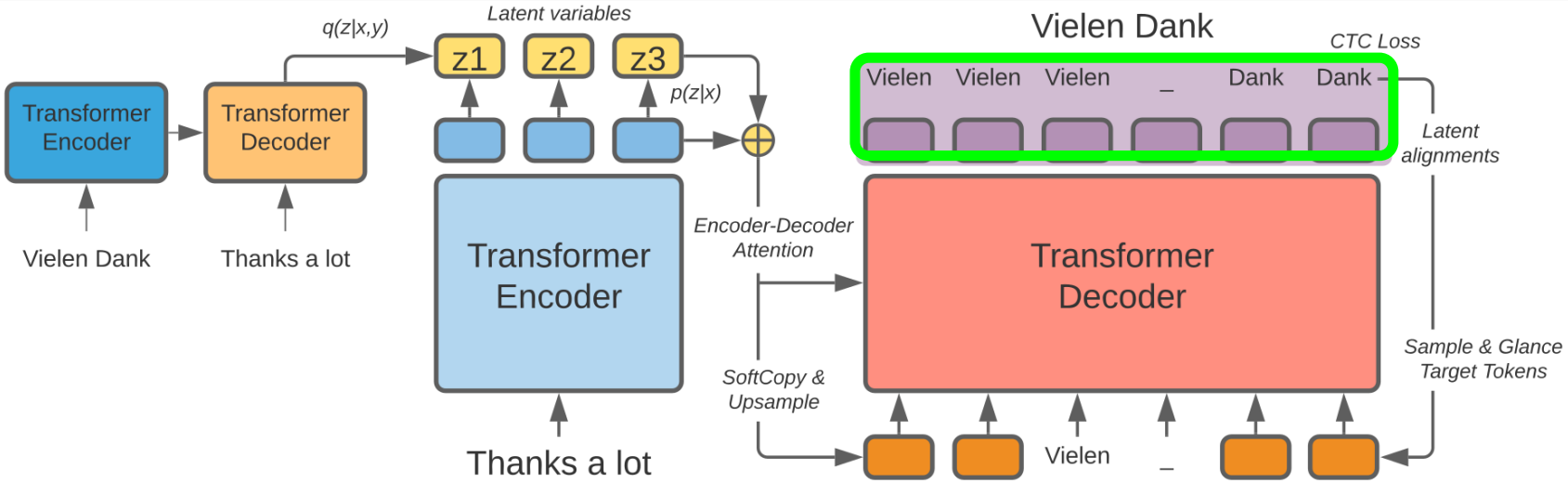
生成確率: $p_\theta(\boldsymbol{y}|\boldsymbol{x}) = \int_{\boldsymbol{z}} p_\theta(\boldsymbol{z}|\boldsymbol{x}) \prod_{t=1}^{T} p_\theta(y_t|\boldsymbol{z}, \boldsymbol{x}) d\boldsymbol{z}$

ELBO: $\underbrace{\mathbb{E}_{\boldsymbol{z}\sim q_\phi}\left[\log p_\theta(\boldsymbol{y}|\boldsymbol{z}, \boldsymbol{x})\right]}_{\text{likelihood}} - \mathcal{D}_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{y}) \parallel p_\theta(\boldsymbol{z}|\boldsymbol{x}))$

- 事後確率 $q_\phi(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{y})$ を得るために Encoder-Decoder モデルを使用
- θ と ϕ の間で embedding layer のみパラメータ共有

Loss Function: Latent Alignments

cross entropy (CE) loss の代わりに CTC loss を採用



- CE: 出力の位置のずれに対して敏感
- CTC: 出力位置に柔軟性をもたせる (Saharia et al. 2020)

$$\log p_\theta(\boldsymbol{y}|\boldsymbol{x}) = \log \sum_{\boldsymbol{a}\in\Gamma(\boldsymbol{y})} p_\theta(\boldsymbol{a}|\boldsymbol{x}),\ \boldsymbol{a}: \text{latent alignments}$$

Learning: Glancing Targets

訓練時に目的言語文をランダムにマスクして入力

(Ghazvininejad et al. 2019)

- 訓練時の目的関数:

  $\log p_\theta(\boldsymbol{y}|\boldsymbol{x}) \rightarrow \log p_\theta(\boldsymbol{y}|\boldsymbol{m} \odot \boldsymbol{y}, \boldsymbol{x}), \boldsymbol{m} \sim \gamma(l, \boldsymbol{y}), l \sim \mathcal{U}_{|\boldsymbol{y}|}$

  - $\boldsymbol{m}$: マスク
  - γ : マスクトークン数 l を受け取るサンプル関数
- カリキュラム学習によって訓練 (GLAT) (Qian et al. 2020)
  - 訓練時と翻訳時のギャップを埋める
  - $l \sim g(f_{ratio} \cdot \mathcal{D}(\hat{\boldsymbol{y}}, \boldsymbol{y}))$
    - $\mathcal{D}$ はモデル予測 $\hat{\boldsymbol{y}}$ と正解との差 (レーベンシュタイン距離など)
    - f_{ratio} はマスク割合 (ハイパーパラメータ)
    - 本論文では g にポアソン分布を使用

Learning: Glancing Targets

- 訓練時のデコーダ入力長
  - GLAT (Qian et al. 2020) : 参照訳のトークン長
  - 提案モデル (CTC-based) : 出力系列長 $\geq$ 入力系列長
    - viterbi aligned tokens: $\hat{\boldsymbol{a}} = \underset{\boldsymbol{a} \in \Gamma(\boldsymbol{y})}{\arg\max}\, p_\theta(\boldsymbol{a}|\boldsymbol{x})$

Summary

Methods	Distillation	Latent Variables	Latent Alignments	Glancing Targets
What it can do?	simplifying the training data	model any types of dependency in theory	handling token shifts in the output space	ease the difficulty of learning hard examples
What it cannot?	uncertainty exists in the teacher model	constrained by the modeling power of the used latent variables	unable to model non-monotonic dependency, e.g. reordering	training / testing phase mismatch
Potential issues	sub-optimal due to the teacher's capacity	difficult to train; posterior collapse	decoder inputs must be longer than targets	difficult to find the optimal masking ratio

Experiments

Dataset

- WMT14 EN $\leftrightarrow$ DE (4.0M)
- WMT16 EN $\leftrightarrow$ RO (610k)
- WMT20 JA $\rightarrow$ EN (13M (filterd))

Knowledge Distillation

- 教師モデルによって生成した目的言語文から学習
  - WMT14 EN $\leftrightarrow$ DE, WMT16 EN $\leftrightarrow$ RO: Transformer *base*
  - WMT20 JA $\rightarrow$ EN: Transformer *big*
- ビーム幅 5 のビーム探索で目的言語文を生成

Experiments

Decoding

- 各位置で最大確率を持つトークンを生成後，Γ^{-1} によって出力を獲得
- noisy parallel decoding (NPD) (Gu et al. 2018)
- ビーム探索や n-gram 言語モデルと組み合わせ

$$\log p_\theta(\boldsymbol{y}|\boldsymbol{x}) + \alpha \log p_{\mathrm{LM}}(\boldsymbol{y}) + \beta \log |\boldsymbol{y}|$$

Baselines: Autoregressive Transformer (AT)

- *base*
- *big*
- *Deep Encoder-Shallow Decoder* (12-1) (Kasai et al. 2021)

Evaluation

翻訳性能 BLEU

翻訳速度

- $\mathcal{L}_1^{\mathrm{GPU}}$：並列計算可能な計算機上における 1 文の翻訳速度
- $\mathcal{L}_1^{\mathrm{CPU}}$：並列計算不可能な計算機上における 1 文の翻訳速度
- $\mathcal{L}_{\mathrm{max}}^{\mathrm{GPU}}$：並列計算可能な計算機上におけるミニバッチ単位 (複数文) の翻訳速度

Implementation Details

- VAE: エンコーダ最終層出力から $z \in \mathbb{R}^{T' \times 8}$ を計算
  - z は線形変換してエンコーダ出力に加えられる
  - posterior network には 3 層の Transformer
  - KL-annealing
- CTC: 原言語文の 3 倍のトークン数をデコーダに入力
  - *SoftCopy* (Wei et al. 2019)
- GLAT: mask ratio $f_{\text{ratio}} = 0.5$

その他は論文参照

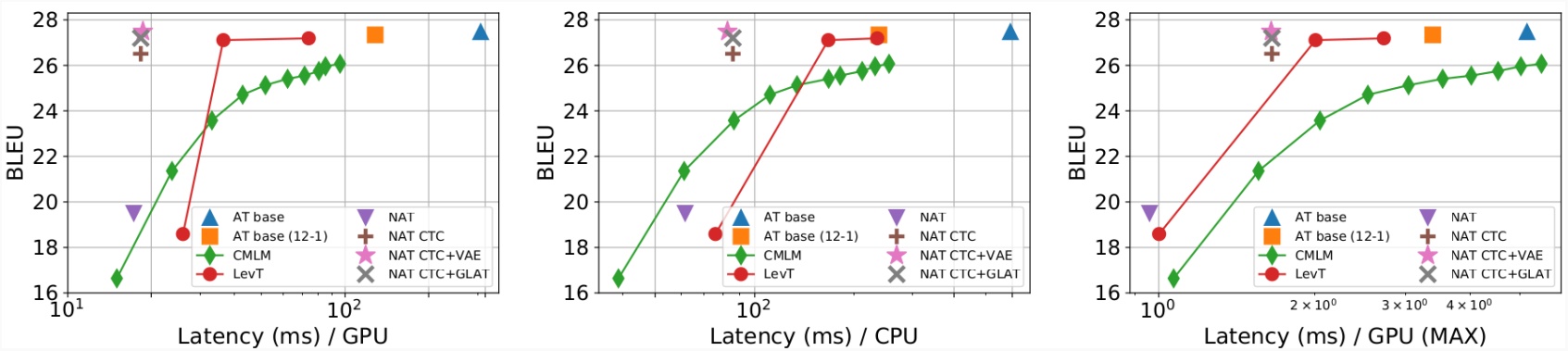
Results: WMT14 EN $\leftrightarrow$ DE, WMT16 EN $\leftrightarrow$ RO

Models		Iter.	Speed	WMT'14 EN-DE	WMT'14 DE-EN	WMT'16 EN-RO	WMT'16 RO-EN
AT	Transformer *base* (teacher)	N	1.0×	**27.48**	**31.39**	**33.70**	**34.05**
	Transformer *base* (12-1)	N	2.4×	26.21	30.80	33.17	33.21
	+ KD	N	2.5×	27.34	30.95	33.52	34.01
Iterative NAT	iNAT (Lee et al., 2018)	10	1.5×	21.61	25.48	29.32	30.19
	Blockwise (Stern et al., 2018)	$\approx N/5$	3.0×	27.40	-	-	-
	InsT (Stern et al., 2019)	$\approx \log N$	4.8×	27.41	-	-	
	CMLM (Ghazvininejad et al., 2019)*	10	1.7×	27.03	30.53	33.08	33.31
	LevT (Gu et al., 2019)	Adv.	4.0×	27.27	-	-	33.26
	KERMIT (Chan et al., 2019)	$\approx \log N$	-	27.80	30.70	-	-
	LaNMT (Shu et al., 2020)	4	5.7×	26.30	-	-	29.10
	SMART (Ghazvininejad et al., 2020b)*	10	1.7×	27.65	31.27	-	-
	DisCO (Kasai et al., 2020a)*	Adv.	3.5×	27.34	31.31	33.22	33.25
	Imputer (Saharia et al., 2020)*	8	3.9×	**28.20**	**31.80**	**34.40**	**34.10**
Fully NAT	Vanilla-NAT (Gu et al., 2018a)	1	15.6×	17.69	21.47	27.29	29.06
	LT (Kaiser et al., 2018)	1	3.4×	19.80	-	-	-
	CTC (Libovický and Helcl, 2018)	1	-	16.56	18.64	19.54	24.67
	NAT-REG (Wang et al., 2019)	1	-	20.65	24.77	-	-
	Bag-of-ngrams (Shao et al., 2020)	1	10.0×	20.90	24.60	28.30	29.30
	Hint-NAT (Li et al., 2018)	1	-	21.11	25.24	-	-
	DCRF (Sun et al., 2019)	1	10.4×	23.44	27.22	-	-
	Flowseq (Ma et al., 2019)	1	1.1 ×	23.72	28.39	29.73	30.72
	ReorderNAT (Ran et al., 2019)	1	16.1×	22.79	27.28	29.30	29.50
	AXE (Ghazvininejad et al., 2020a)*	1	15.3×	23.53	27.90	30.75	31.54
	EM+ODD (Sun and Yang, 2020)	1	16.4×	24.54	27.93	-	-
	GLAT (Qian et al., 2020)	1	15.3×	25.21	29.84	31.19	32.04
	Imputer (Saharia et al., 2020)*	1	18.6×	25.80	28.40	32.30	31.70
	Ours (Fully NAT)	1	17.6×	11.40	16.47	24.52	24.79
	+ KD	1	17.6×	19.50	24.95	29.91	30.25
	+ KD + CTC	1	16.8×	26.51	30.46	33.41	34.07
	+ KD + CTC + VAE	1	16.5×	**27.49**	31.10	**33.79**	33.87
	+ KD + CTC + GLAT	1	16.8×	27.20	**31.39**	33.71	**34.16**

Results: WMT20 JA $\rightarrow$ DE

	Configuration	BLEU (Δ)	BP	$\mathcal{L}_1^{\text{GPU}}$ (Speed-up)		$\mathcal{L}_1^{\text{CPU}}$ (Speed-up)	
AT	*big* (teacher)	21.07	0.920	345 ms	1.0 $\times$	923 ms	1.0 $\times$
	base	18.91	0.908	342 ms	1.0 $\times$	653 ms	1.4 $\times$
	base (12-1)	15.47	0.806	152 ms	2.3 $\times$	226 ms	4.0 $\times$
	base (12-1) + KD	18.76	0.887	145 ms	2.4 $\times$	254 ms	3.6 $\times$
NAT	KD + CTC	16.93 (+0.00)	0.828	17.3 ms	19.9 $\times$	84 ms	11.0 $\times$
	KD + CTC + VAE	18.73 (+1.80)	0.862	16.4 ms	21.0 $\times$	83 ms	11.1 $\times$
	w. *BeamSearch20*	19.80 (+2.87)	0.958	28.5 ms	12.1 $\times$	99 ms	9.3 $\times$
	w. *BeamSearch20 + 4-gram LM*	**21.41 (+4.48)**	0.954	31.5 ms	11.0 $\times$	106 ms	8.7 $\times$
	w. *NPD5*	18.88 (+1.95)	0.866	34.9 ms	9.9 $\times$	313 ms	2.9 $\times$
	w. *NPD5 + BeamSearch20 + 4-gram LM*	**21.84 (+4.91)**	0.962	57.6 ms	6.0 $\times$	284 ms	3.2 $\times$

Quality v.s. Latency



Ablation Study (on WMT14 EN $\rightarrow$ DE)

KD	AXE	CTC	VAE	RND	GLAT	BLEU
						11.40
✓						19.50
	✓					16.59
✓	✓					21.66
		✓				18.18
✓		✓				**26.51**
		✓	✓			23.58
✓	✓		✓			22.19
✓		✓	✓			**27.49**
✓	✓			✓		22.74
✓	✓				✓	24.67
✓		✓		✓		26.16
		✓			✓	21.81
✓		✓			✓	**27.20**
✓		✓	✓		✓	**27.21**

Ablation Study (on WMT14 EN $\rightarrow$ DE)

Models		Distillation *base*	Distillation *big*	BLEU	Speed-up
AT	*base*			27.43	1.0×
	big			28.14	0.9×
	base (12-1)			26.12	2.4×
		✓		27.34	2.5×
			✓	27.83	2.4×
NAT	*base*			23.58	16.5×
		✓		27.49	16.5×
			✓	27.56	16.5×
	big		✓	**27.89**	15.8×

Upsampling Ratio (λ) for CTC Loss

λ	BLEU	$\mathcal{L}_1^{\text{GPU}}$	$\mathcal{L}_{\max}^{\text{GPU}}$	$\mathcal{L}_1^{\text{CPU}}$
1.5	26.16	17.9 ms	**0.95 ms**	**66.6 ms**
2.0	26.39	17.5 ms	1.03 ms	71.6 ms
2.5	**26.54**	17.6 ms	1.16 ms	76.9 ms
3.0	26.51	**17.0 ms**	1.32 ms	81.8 ms

Conclusion

fully NAT の性能が AT に追い付いた

- 4 つの *dependency reduction* によって SOTA な fully NAT モデルを設計
  - Data: Knowledge Distillation
  - Model: Latent Variables
  - Loss Function: Latent Alignments
  - Learning: Glancing Targets