Paper reading

Liang Ding

2020/ 07/ 23

Towards better NAT Encoder & Decoder

1. Jointly Masked S2S model for NAT (ACL2020)

2. NAT with disentangled context Transformer (ICML2020)

Towards better NAT Encoder :

Jointly Masked S2S model for NAT (ACL2020)

Title: Jointly Masked Sequence-to-Sequence Model for Non-Autoregressive Neural Machine Translation (ACL 2020)

Author: Junliang Guo, Linli Xu, Enhong Chen

Comments: 又是一篇跟进MaskPredict的工作，这篇改进Encoder中了ACL（另一篇Jungo对Decoder做了改进,i.e. DisCo中了ICML）。先做了几个实验验证了encoder在NAT中更重要（实际跟之前NMT的结论一样，encoder对性能影响更大）然后就各种mask。在MaskPredict基础上最大的改进是，encoder中类似bert进行mask并预测，decoder采用consecutive masking并套了个ngram loss（n=2），最后再跟AT的loss插值在一起训练。解码的时候跟mask-predict类似，也是iterative refinement，只不过mask的不是单个token，而是训练时采用的ngram。

值得借鉴的是他的preliminary部分如何证明encoder比decoder更重要：1）encoder加层比decoder加层增益更高 2）用pretrained的encoder/decoder是初始化NAT并且fix住，只更新另外的decodder/encoder并观察其收敛速度，发现trainable encoder收敛更慢，也说明encoder更难训 3）解码阶段在encoder input/ output 和 decoder input三个组件上随机替换一些token为噪音，发现BLEU的降低程度 encoder input > encoder output > decoder input。通过这三个分析，得到结论the performance of NAT models rely more on the encoder rather than the decoder。

有个槽点，这篇argue自己和jiatao2017年报出来的AT baseline的latency(607 ms)对比加速了5倍(106 ms)，这个比较不客观。一方面解码时的参数和硬件setting不同不能直接对比607 ms这个结果，此外根据我们的经验，基于MaskPredict架构10次refine一般来说只能提速1.5倍。也难怪twitter上Kenneth Heafield教授会开怼NAT的加速。让我们值得反思除了加速以外，非自回归结构的优势在哪里？另外本文的复现中，我发现BLEU的增益更多源于更多的训练步数(50W vs. 30W)。

By Liang Ding

Towards better NAT Encoder :

Jointly Masked S2S model for NAT (ACL2020)

Motivation: **Encoder is more important and not fully utilized**

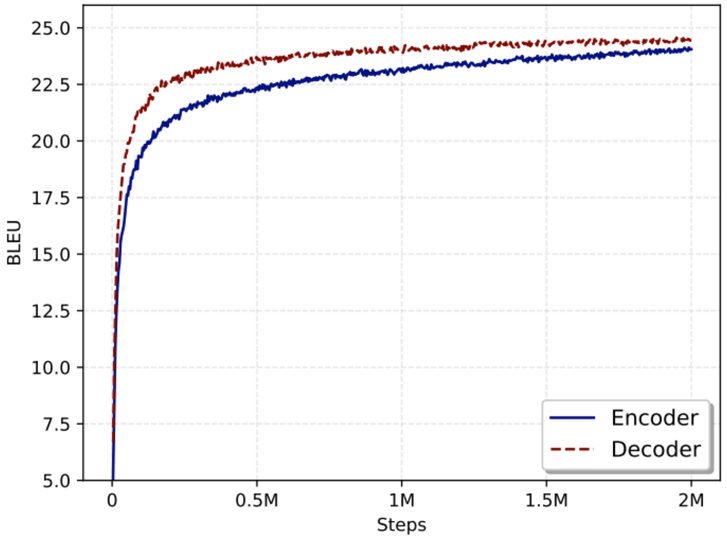
Δ Layers	+5	+10	+15
Δ Enc BLEU	+0.71	+1.05	+1.26
Δ Dec BLEU	+0.12	+0.18	+0.20

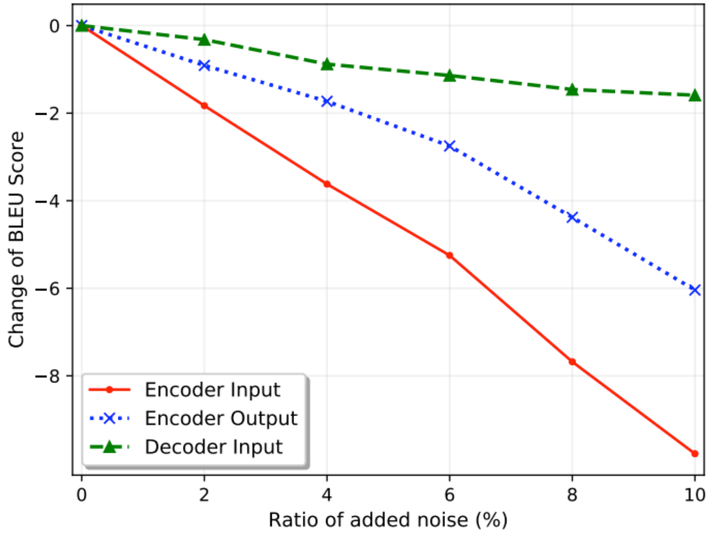
Preliminary studies:

1) **Deep encoder gains more than deep decoder**

2) **Trainable encoder converge slower than trainable decoder, indicating encoder is hard to train**

3) **Add noise to Encoder input/ Encoder output/ Decoder input, △BLEU: E input > E output > D input**

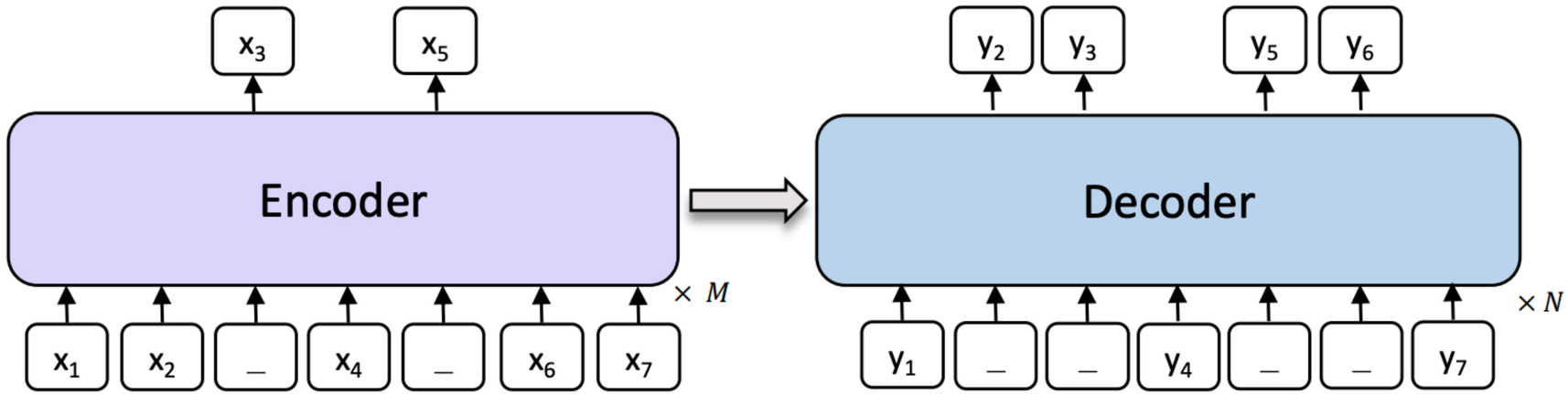




Towards better NAT Encoder :

Jointly Masked S2S model for NAT (ACL2020)

Architecture: **Like MASS, during normal training, encoder as the auto-encoder**



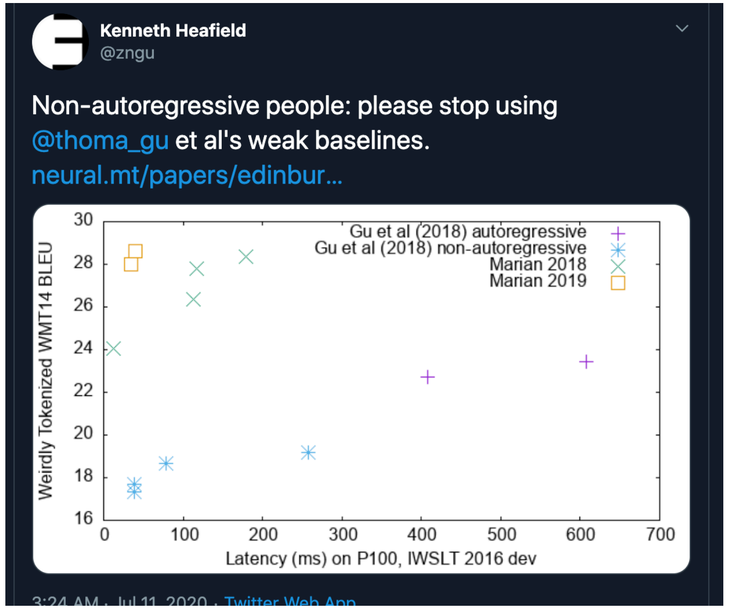
Towards better NAT Encoder :

Jointly Masked S2S model for NAT (ACL2020)

Experiments: better NAT performance on WMT14 En<=>De, WMT16 En<=>Ro and IWSLT14 De->En

Models	WMT14 En−De	WMT14 De−En	WMT16 En−Ro	WMT16 Ro−En	IWSLT14 De−En	Latency	Speedup
Transformer (Vaswani et al., 2017)	28.04*	32.69*	34.13*	34.46*	32.99*	607 ms	1.00×
Transformer (Weak Teacher)	27.40*	31.29*	/	/	/	–	–
NAT-FT (NPD 100) (Gu et al., 2017)	19.17	23.20	29.79	31.44	24.21†	257 ms	2.36×
Imitate-NAT (Wei et al., 2019)	24.15	27.28	31.45	31.81	/	/	/
NAT-Reg (NPD 9) (Wang et al., 2019)	24.61	28.90	/	/	28.04	40 ms	15.1×
FCL-NAT (NPD 9) (Guo et al., 2020)	25.75	29.50	/	/	29.91	38 ms	16.0×
NAT-DCRF (NPD 9) (Sun et al., 2019)	26.07	29.68	/	/	29.99	63 ms	9.63×
NAT-IR ($k = 5$) (Lee et al., 2018)	20.26	23.86	28.86	29.72	/	/	/
NAT-IR ($k = 10$)	21.61	25.48	29.32	30.19	23.94†	404† ms	1.50×
CM-NAT ($k = 4$) (Ghazvininejad et al., 2019)	25.94	29.90	32.53	33.23	30.42*	62* ms	9.79×
CM-NAT ($k = 10$)	27.03	30.53	33.08	33.31	31.71*	161* ms	3.77×
JM-NAT ($k = 4$)	27.05	31.51	32.97	33.21	31.27	45 ms	13.5×
JM-NAT ($k = 10$)	**27.69**	**32.24**	**33.52**	**33.72**	**32.59**	106 ms	5.73×
JM-NAT (WT) ($k = 4$)	26.82	30.59	/	/	/	–	–
JM-NAT (WT) ($k = 10$)	27.31	31.02	/	/	/	–	–

our: ~1.5 x



Towards better NAT Decoder :

NAT with disentangled context Transformer (ICML2020)

Title: Non-autoregressive Machine Translation with Disentangled Context Transformer (ICML 2020)
Author: Jungo Kasai, James Cross, Marjan Ghazvininejad, Jiatao Gu
Comments: Facebook发表在ICML对MaskPredictDecoder进行改进的工作。已有的MaskPredict架构是将部分Y进行Mask（Y_mask），然后基于其余字符Y_obs做生成，该过程可建模为P(Y_mask|X,Y_obs)。本文认为把所有的MASK位置不加区分的用Y_obs来预测，这样对数据利用不充分。因此作者主张不同位置的Y_n应当基于不同的任意Y_observed_n来预测，即P(Y_1|X,Y_obs1),...,P(Y_n|X,Y_obsn)。为避免把当前待预测的位置喂给模型，作者让K和V包含word + hidden信息，Q只包含hidden信息，即

$$
\begin{aligned}
&k_n, v_n = \texttt{Proj}(w_n + p_n) \quad q_n = \texttt{Proj}(p_n) \\
&h_n = \texttt{Attention}(K^n_{\texttt{obs}}, V^n_{\texttt{obs}}, q_n) \\
&K^n_{\texttt{obs}} = \texttt{Concat}\left(\{k_m | Y_m \in Y^n_{\texttt{obs}}\}\right) \\
&V^n_{\texttt{obs}} = \texttt{Concat}\left(\{v_m | Y_m \in Y^n_{\texttt{obs}}\}\right)
\end{aligned}
$$

但是如上连接方式导致在Transformer堆叠的时候，下一层要预测的位置还是会看到包含当前预测位的隐状态，即"信息泄露"：
比如Y={A，B}，第一层位置1和位置2分别condition on B和A来预测A和B，即h1=SAN(k(B),q(A),v(B))；h2=SAN(k(A),q(B),v(A))；第二层位置2可以看到位置1的隐状态，即输出为SAN(k(h1),q(h2),v(h1))，此时h1包含B的信息，用这个包含B的隐状态预测B被称为"信息泄露"。
为了解决堆叠时出现的信息泄露，作者让Layer>1层的K和V只能读取第0层的word + position representation，即:

$$
\begin{aligned}
&k^j_n, v^j_n = \texttt{Proj}(w_n + p_n) \quad q^j_n = \texttt{Proj}(h^{j-1}_n) \\
&h^j_n = \texttt{Attention}(K^{n,j}_{\texttt{obs}}, V^{n,j}_{\texttt{obs}}, q^j_n)
\end{aligned}
$$

该过程可以看作对K和V去内容化(de-contextualize),即disentangled context Transformer。作者认为DisCo如果让Y_obs=Y_<n，那么就能看作AT。
在测试阶段，作者认为MaskPredict每次只更新low confident的几个词汇，没有改善Y_obs词汇。作者提出基于置信度高的去预测并更新所有词汇的easy-first策略，迭代到连续两次输出一致时停止。和MaskPredict不同的是，作者在实验中在{0.1,0.2,0.3}几种dropout ratio中进行了选择而不是固定0.3，在dropout=0.2的情况下，MaskPredict被作者跑出了WMT EnDe 27.39 BLEU（10 iters），作者的DisCo + easyfirst跑到了27.34 BLEU（4.82 iters），DisCo + MaskPredict解码打分为27.06（10 iters），在DeEn,EnRo,RoEn上都有类似的发现，即在iters更少的情况下和MaskPredict可比较的结果。在EnZh,ZhEn,EnFr大规模数据集上发现能用更少的iters勉强超过MaskPredict，作者解释数据量越大效果越明显。整体来说，本文的写作值得学习，也验证了一点：ML系列论文不是那么在意performance。我复现时的小插曲见https://github.com/facebookresearch/DisCo/issues/1
By Liang Ding

Towards better NAT Decoder :

NAT with disentangled context Transformer (ICML2020)

$$P(Y_{\texttt{mask}}|X, Y_{\texttt{obs}}) = \texttt{Transformer}(X, Y_{\texttt{obs}})$$
$$Y_{\texttt{mask}} \sim \texttt{RS}(Y) \quad Y_{\texttt{obs}} = Y \setminus Y_{\texttt{mask}}$$

$$P(Y_n|X, Y^n_{\texttt{obs}}) = \texttt{Transformer}(X, Y^n_{\texttt{obs}})$$
$$Y^n_{\texttt{obs}} \sim \texttt{RS}(Y \setminus Y_n)$$

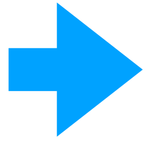
Motivation: **Data/ Context is not fully utilized because all MASK position conditioned on fixed Y_obs**

Methods: **Each Y_n conditioned on different randomly sampled Y_obs**

$$P(Y_1|X, Y^1_{\texttt{obs}}), \cdots, P(Y_N|X, Y^N_{\texttt{obs}}) = \texttt{DisCo}(X, Y)$$

To avoid the model "see" Y_n itself when predicting it.

$$k_n, v_n, q_n = \texttt{Proj}(w_n + p_n)$$
$$h_n = \texttt{Attention}(K, V, q_n)$$
$$K, V = \texttt{Concat}\left(\{k_m\}_{m=1}^N\right), \texttt{Concat}\left(\{v_m\}_{m=1}^N\right)$$

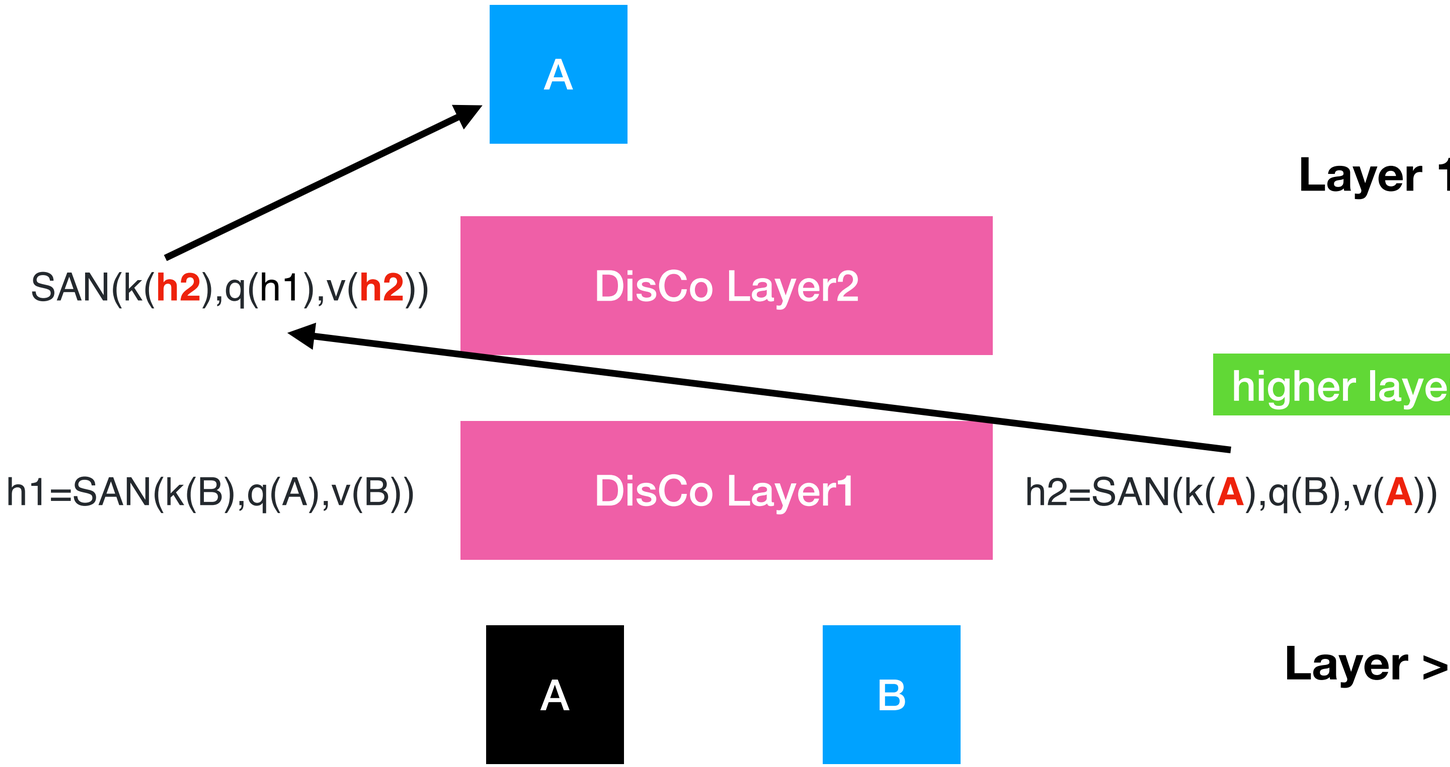


$$k_n, v_n = \texttt{Proj}(w_n + p_n) \quad q_n = \texttt{Proj}(p_n)$$
$$h_n = \texttt{Attention}(K^n_{\texttt{obs}}, V^n_{\texttt{obs}}, q_n)$$
$$K^n_{\texttt{obs}} = \texttt{Concat}\left(\{k_m | Y_m \in Y^n_{\texttt{obs}}\}\right)$$
$$V^n_{\texttt{obs}} = \texttt{Concat}\left(\{v_m | Y_m \in Y^n_{\texttt{obs}}\}\right)$$

Towards better NAT Decoder :

NAT with disentangled context Transformer (ICML2020)

A question that followed: Stacking makes the word Y_n exposure itself in the higher layers



Layer 1:

$$k_n, v_n = \texttt{Proj}(w_n + p_n) \quad q_n = \texttt{Proj}(p_n)$$
$$h_n = \texttt{Attention}(K^n_{\text{obs}}, V^n_{\text{obs}}, q_n)$$
$$K^n_{\text{obs}} = \texttt{Concat}(\{k_m | Y_m \in Y^n_{\text{obs}}\})$$
$$V^n_{\text{obs}} = \texttt{Concat}(\{v_m | Y_m \in Y^n_{\text{obs}}\})$$

higher layer K & V obtains word and position information merely

Layer >1:

$$k^j_n, v^j_n = \texttt{Proj}(w_n + p_n) \quad q^j_n = \texttt{Proj}(h^{j-1}_n)$$
$$h^j_n = \texttt{Attention}(K^{n,j}_{\text{obs}}, V^{n,j}_{\text{obs}}, q^j_n)$$

Towards better NAT Decoder :

NAT with disentangled context Transformer (ICML2020)

A question that followed: Stacking makes the word Y_n exposure itself in the higher layers

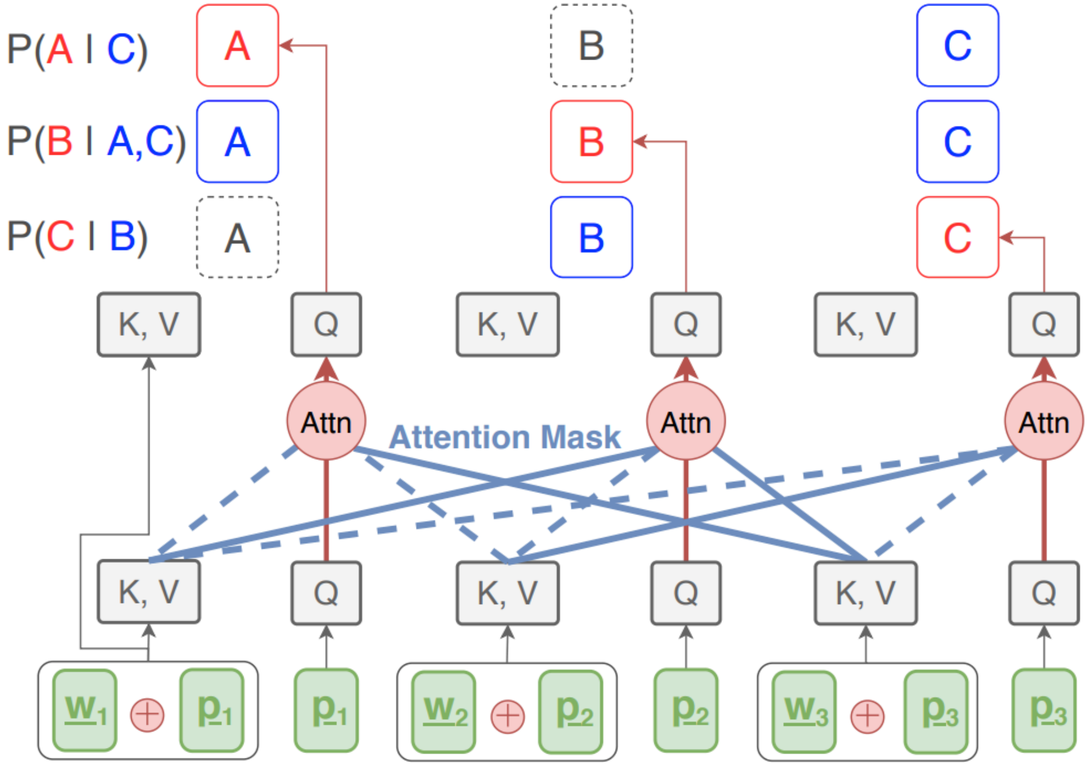
Layer 1:

$$k_n, v_n = \texttt{Proj}(w_n + p_n) \quad q_n = \texttt{Proj}(p_n)$$
$$h_n = \texttt{Attention}(K^n_{\text{obs}}, V^n_{\text{obs}}, q_n)$$
$$K^n_{\text{obs}} = \texttt{Concat}\left(\{k_m | Y_m \in Y^n_{\text{obs}}\}\right)$$
$$V^n_{\text{obs}} = \texttt{Concat}\left(\{v_m | Y_m \in Y^n_{\text{obs}}\}\right)$$

higher layer K & V obtains word and position information merely

Layer >1:

$$k^j_n, v^j_n = \texttt{Proj}(w_n + p_n) \quad q^j_n = \texttt{Proj}(h^{j-1}_n)$$
$$h^j_n = \texttt{Attention}(K^{n,j}_{\text{obs}}, V^{n,j}_{\text{obs}}, q^j_n)$$



Towards better NAT Decoder :

NAT with disentangled context Transformer (ICML2020)

Experiments: En<=>De; En<=>Ro

What ? no improvements! 😅

Model n: # rescored candidates	en→de Step	en→de BLEU	de→en Step	de→en BLEU	en→ro Step	en→ro BLEU	ro→en Step	ro→en BLEU
Gu et al. (2018) ($n = 100$)	1	19.17	1	23.20	1	29.79	1	31.44
Wang et al. (2019) ($n = 9$)	1	24.61	1	28.90	–	–	–	–
Li et al. (2019) ($n = 9$)	1	25.20	1	28.80	–	–	–	–
Ma et al. (2019) ($n = 30$)	1	25.31	1	30.68	1	32.35	1	32.91
Sun et al. (2019) ($n = 19$)	1	26.80	1	30.04	–	–	–	–
Ran et al. (2019)	1	26.51	1	31.13	1	31.70	1	31.99
Shu et al. (2020) ($n = 50$)	1	25.1	–	–	–	–	–	–
Iterative NAT Models								
Lee et al. (2018)	10	21.61	10	25.48	10	29.32	10	30.19
Ghazvininejad et al. (2019) (CMLM)	4	25.94	4	29.90	4	32.53	4	33.23
	10	27.03	10	30.53	10	33.08	10	33.31
Gu et al. (2019b) (LevT)	7+	27.27	–	–	–	–	7+	33.26
Our Implementations								
CMLM + Mask-Predict	4	26.73	4	30.75	4	33.02	4	33.27
CMLM + Mask-Predict	10	**27.39**	10	31.24	10	**33.33**	10	**33.67**
DisCo + Mask-Predict	4	25.83	4	30.15	4	32.22	4	32.92
DisCo + Mask-Predict	10	27.06	10	30.89	10	32.92	10	33.12

Let's do more ! 🧐

Towards better NAT Decoder :

NAT with disentangled context Transformer (ICML2020)

Better inference strategy: Fix iterations MaskPredict ->>> Dynamical iterations EasyFirst

$$i_t = \left\lfloor N \cdot \frac{T-t+1}{T} \right\rfloor$$

$$Y^t_{\text{obs}} = \{Y^{t-1}_j | j \in \underset{n}{\text{topk}}(p^{t-1}_n, k = N - i_t)\}$$

$$Y^t_n, p^t_n = \begin{cases} (\arg)\max_w P(Y_n = w | X, Y^t_{\text{obs}}) \text{ if } Y^t_n \notin Y^t_{\text{obs}} \\ Y^{t-1}_n, p^{t-1}_n \qquad \text{otherwise} \end{cases}$$

$$Y^1_n, p_n = (\arg)\max_w P(Y_n = w | X)$$

Then, we get the *easy-first* order z where $z(i)$ denotes the rank of p_i in descending order. At iteration $t > 1$, we update predictions for all positions by

$$Y^{n,t}_{\text{obs}} = \{Y^{t-1}_i | z(i) < z(n)\}$$

$$Y^t_n, p^t_n = (\arg)\max_w P\left(Y_n = w | X, Y^{n,t}_{\text{obs}}\right)$$

In the t-th iteration, the observed part for each token Y is the tokens with higher probability than it in the iteration (t-1), that is, those characters that are more "simple" than it. Till meeting the repeated sentence~~~

Towards better NAT Decoder :

NAT with disentangled context Transformer (ICML2020)

Experiments: En<=>De; En<=>Ro; En<=>Zh; En->Fr

Let's try again with EasyFirst!

Model n: # rescored candidates	en→de Step	en→de BLEU	de→en Step	de→en BLEU	en→ro Step	en→ro BLEU	ro→en Step	ro→en BLEU
Gu et al. (2018) ($n = 100$)	1	19.17	1	23.20	1	29.79	1	31.44
Wang et al. (2019) ($n = 9$)	1	24.61	1	28.90	–	–	–	–
Li et al. (2019) ($n = 9$)	1	25.20	1	28.80	–	–	–	–
Ma et al. (2019) ($n = 30$)	1	25.31	1	30.68	1	32.35	1	32.91
Sun et al. (2019) ($n = 19$)	1	26.80	1	30.04	–	–	–	–
Ran et al. (2019)	1	26.51	1	31.13	1	31.70	1	31.99
Shu et al. (2020) ($n = 50$)	1	25.1	–	–	–	–	–	–
Iterative NAT Models								
Lee et al. (2018)	10	21.61	10	25.48	10	29.32	10	30.19
Ghazvininejad et al. (2019) (CMLM)	4	25.94	4	29.90	4	32.53	4	33.23
	10	27.03	10	30.53	10	33.08	10	33.31
Gu et al. (2019b) (LevT)	7+	27.27	–	–	–	–	7+	33.26
Our Implementations								
CMLM + Mask-Predict	4	26.73	4	30.75	4	33.02	4	33.27
CMLM + Mask-Predict	10	**27.39**	10	31.24	10	**33.33**	10	**33.67**
DisCo + Mask-Predict	4	25.83	4	30.15	4	32.22	4	32.92
DisCo + Mask-Predict	10	27.06	10	30.89	10	32.92	10	33.12
DisCo + Easy-First	4.82	27.34	4.23	**31.31**	3.29	33.22	3.10	33.25
AT Models								
Vaswani et al. (2017) (base)	N	27.3	–	–	–	–	–	–
Vaswani et al. (2017) (large)	N	28.4	–	–	–	–	–	–
Our Implementations								
AT Transformer Base (EN-RO teacher)	N	27.38	N	31.78	N	34.16	N	34.46
AT Transformer Base + Distillation	N	28.24	N	31.54	–	–	–	–
AT Transformer Large (EN-DE teacher)	N	28.60	N	31.71	–	–	–	–

Model	en→zh Step	en→zh BLEU	zh→en Step	zh→en BLEU
Ghazvininejad et al. (2019)	4	32.63	4	21.90
	10	33.19	10	23.21
Our Implementations				
CMLM + Mask-Predict	4	33.58	4	22.57
CMLM + Mask-Predict	10	34.24	10	23.76
DisCo + Mask-Predict	4	33.61	4	22.42
DisCo + Mask-Predict	10	34.51	10	23.68
DisCo + Easy-First	5.44	**34.63**	5.90	**23.83**
AT Transformer Base	N	34.74	N	23.77
+ Distillation	N	35.09	N	24.53
AT Trans. Large (teacher)	N	35.01	N	24.65

Table 3. WMT14 EN-FR test results.

Model	en→fr Step	en→fr BLEU	Train Time
CMLM + Mask-Predict	4	40.21	53 h
CMLM + Mask-Predict	10	40.55	
DisCo + Mask-Predict	4	39.59	
DisCo + Mask-Predict	10	40.27	37 h
DisCo + Easy-First	4.29	40.66	
Vaswani et al. (2017) (base)	N	38.1	–
Vaswani et al. (2017) (large)	N	41.8	–
Ott et al. (2018) (teacher)	N	43.2	–
AT Transformer Base	N	41.27	28 h
+ Distillation	N	42.03	28 h

Nice, less iters with comparable results!

Q&A