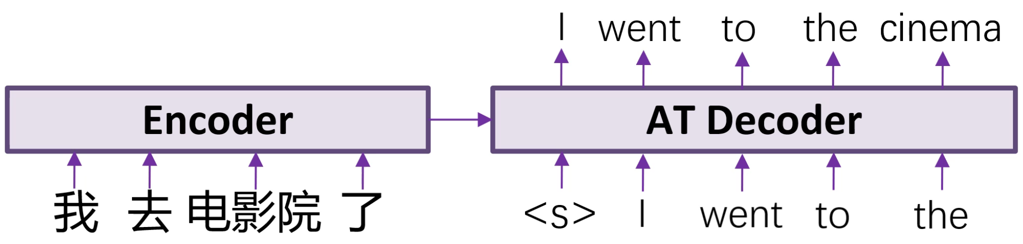
Directed Acyclic Transformer for Non-Autoregressive Machine Translation

Fei Huang, **Hao Zhou**, Yang Liu, Hang Li, Minlie Huang

Background

Autoregressive Translation (AT)

- Generate token by token
- Latency: ~600ms per sample*



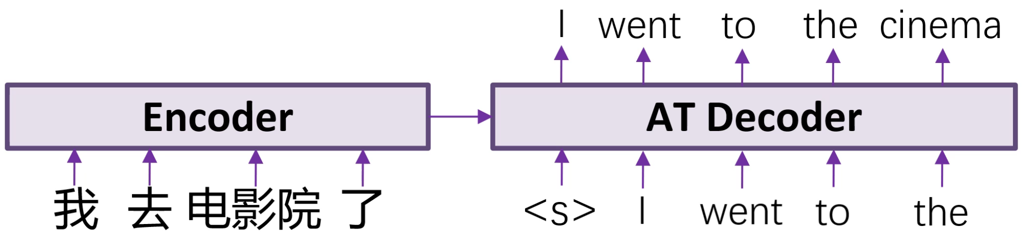
*: Reported by Gu et al. *Non-autoregressive Machine Translation*. ICLR2018.
The latency is evaluated on IWLST16 En-De with batch size=1 on a Nvidia Tesla P100

Background

Reduce the inference latency

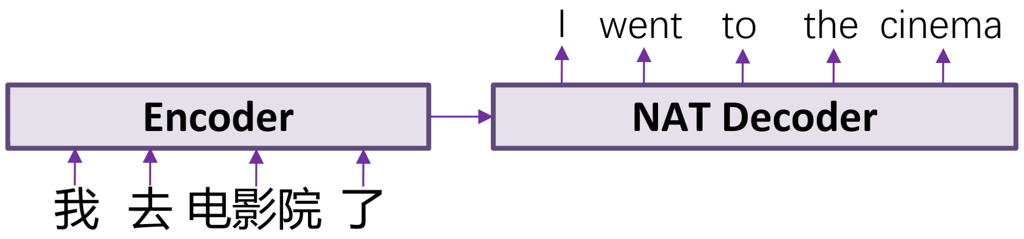
Autoregressive Translation (AT)

- Generate token by token
- Latency: ~600ms per sample*



Non-Autoregressive Translation (NAT)

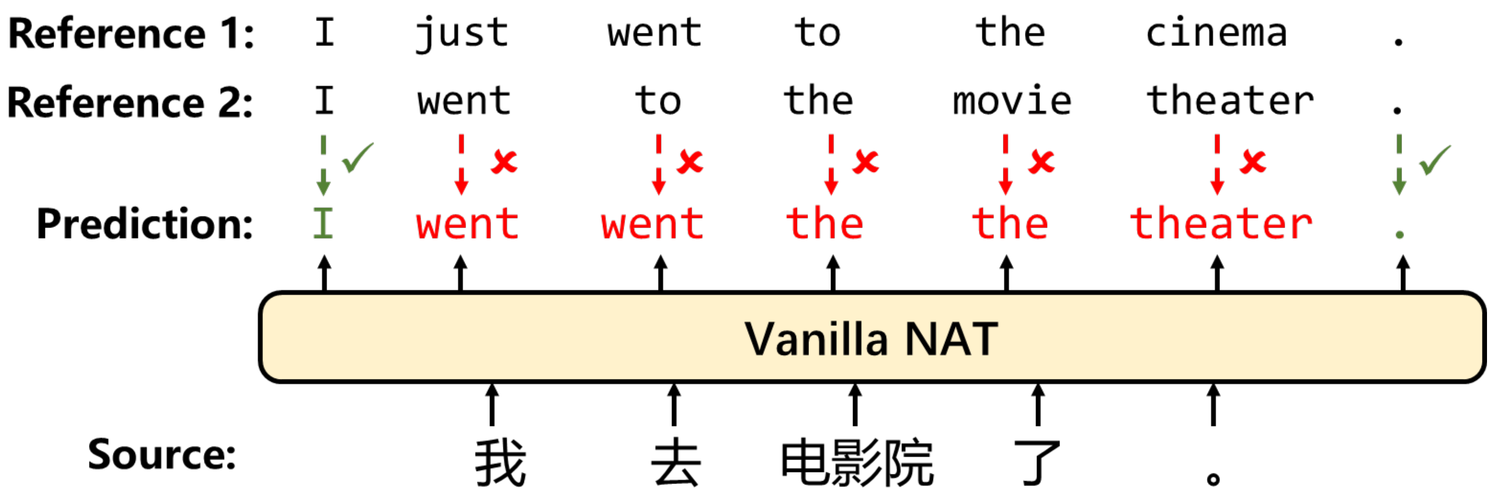
- Generate all tokens in parallel (Gu et al., 2018)
- Latency: ~10x speed up*



*: Reported by Gu et al. *Non-autoregressive Machine Translation*. ICLR2018.
The latency is evaluated on IWLST16 En-De with batch size=1 on a Nvidia Tesla P100

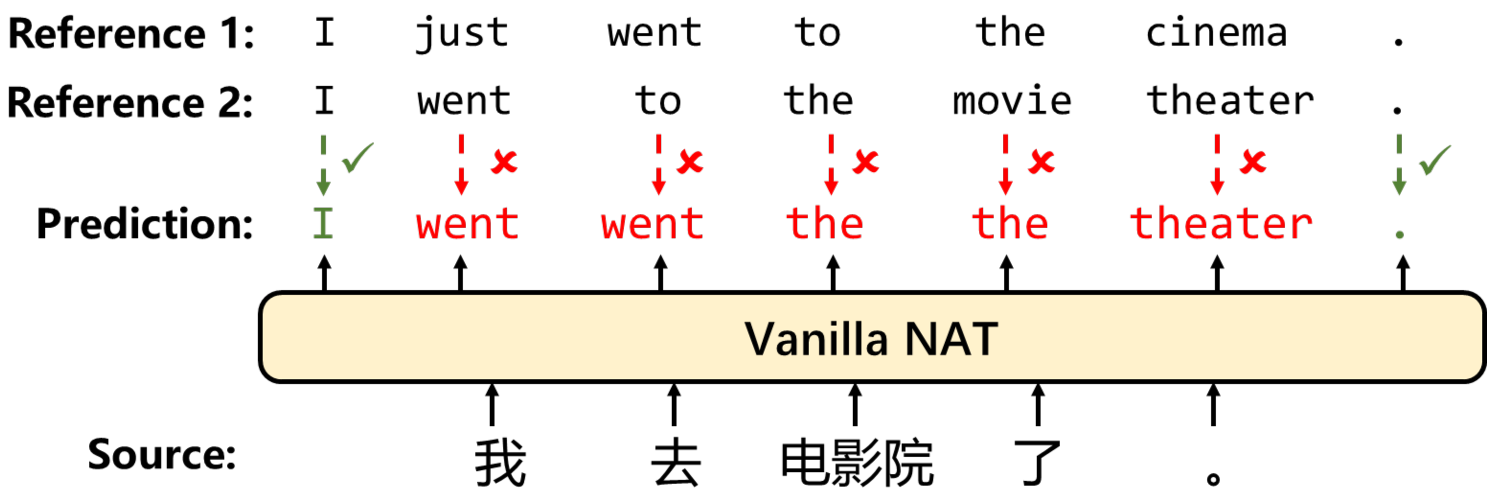
Challenges in NAT

- **Multi-modality Problem:**
  - NATs produce incorrect outputs that mix multiple possible translations



Challenges in NAT

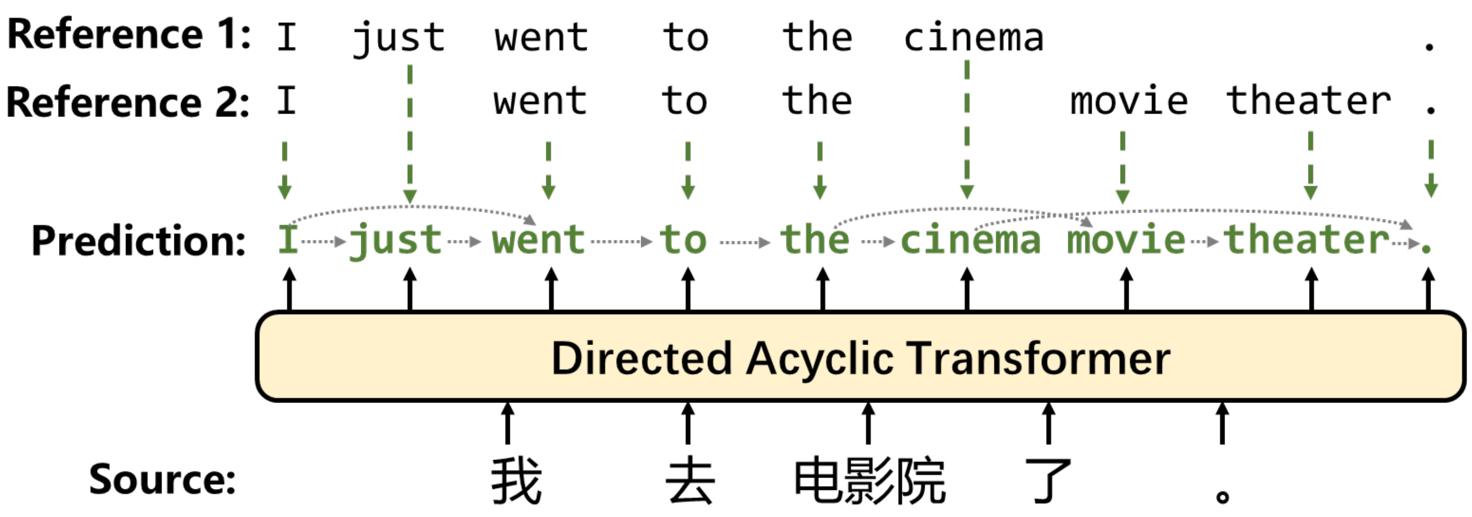
- **Multi-modality Problem:**
  - NATs produce incorrect outputs that mix multiple possible translations



- **Two causes:**
  - **Training**: inconsistent labels in the reference sentences
  - **Inference**: cannot preserve correct lexical dependencies during inference

Our Proposed Method

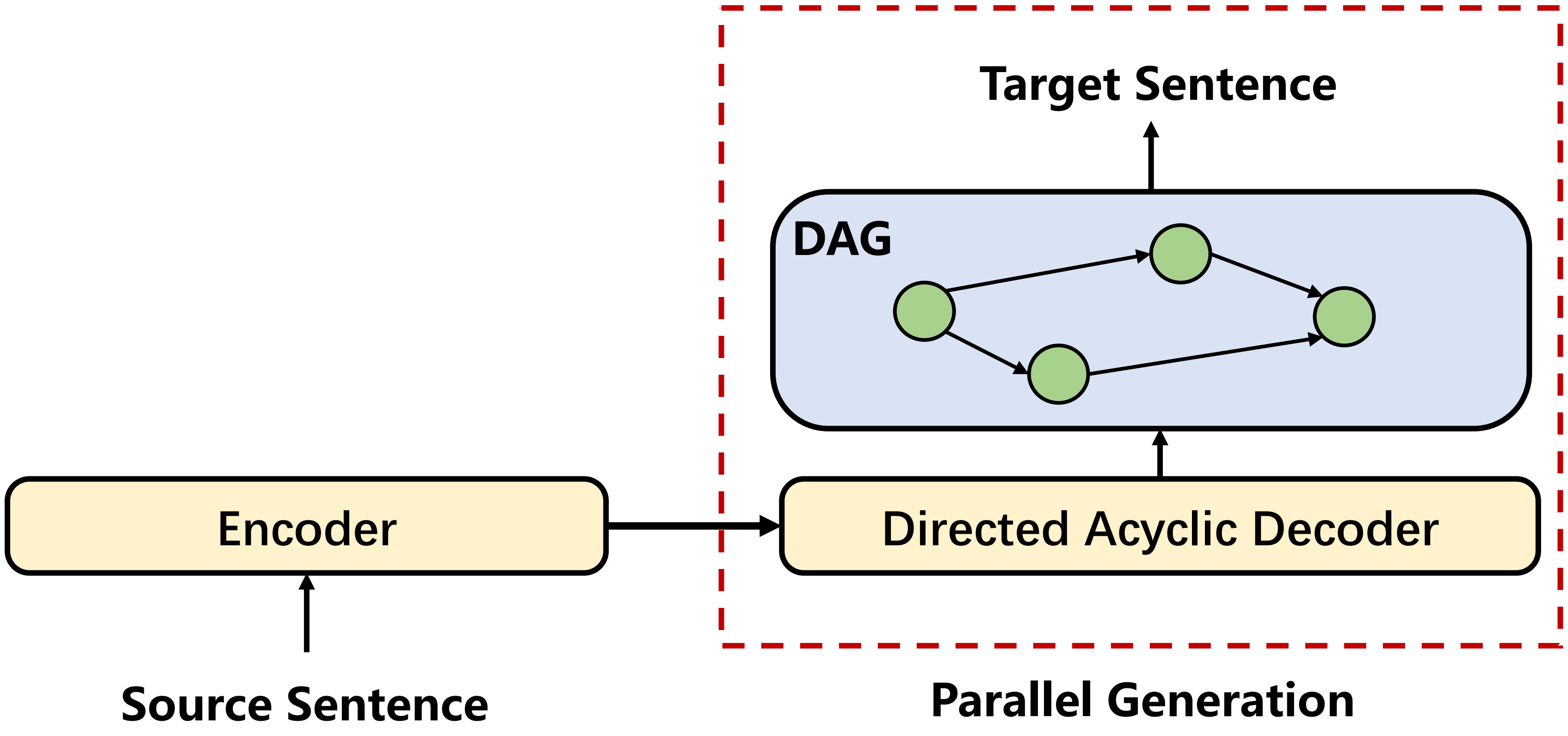
- **Utilize Directed Acyclic Graph (DAG)**
  - to organize the decoding hidden states (and predicted tokens)



  - **In training**: alleviate conflicts by assigning tokens to different vertices
  - **In inference**: recover the translation following predicted transitions

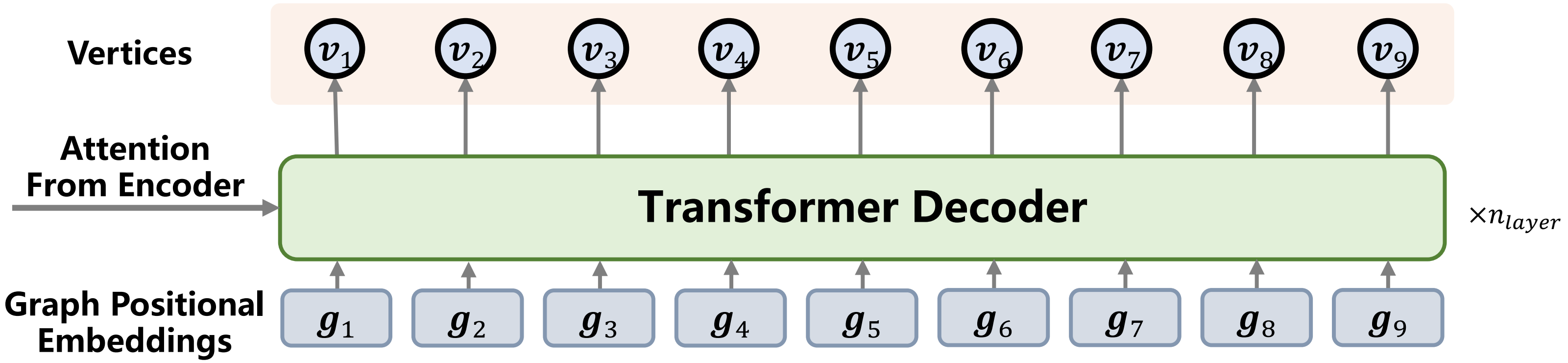
Directed Acyclic Transformer (DA-Transformer)

- Overall Architecture



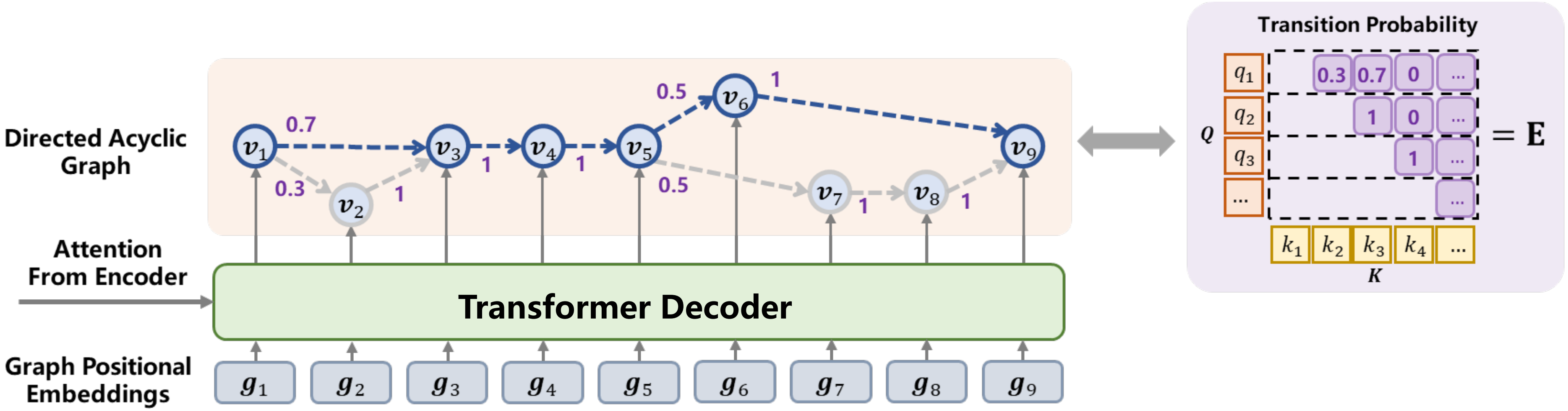
Directed Acyclic Transformer (DA-Transformer)

- Step 1: Obtaining the vertex states $\boldsymbol{V} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_L]^T$



Directed Acyclic Transformer (DA-Transformer)

- Step 2: Predict the transition matrix E and sample a path A

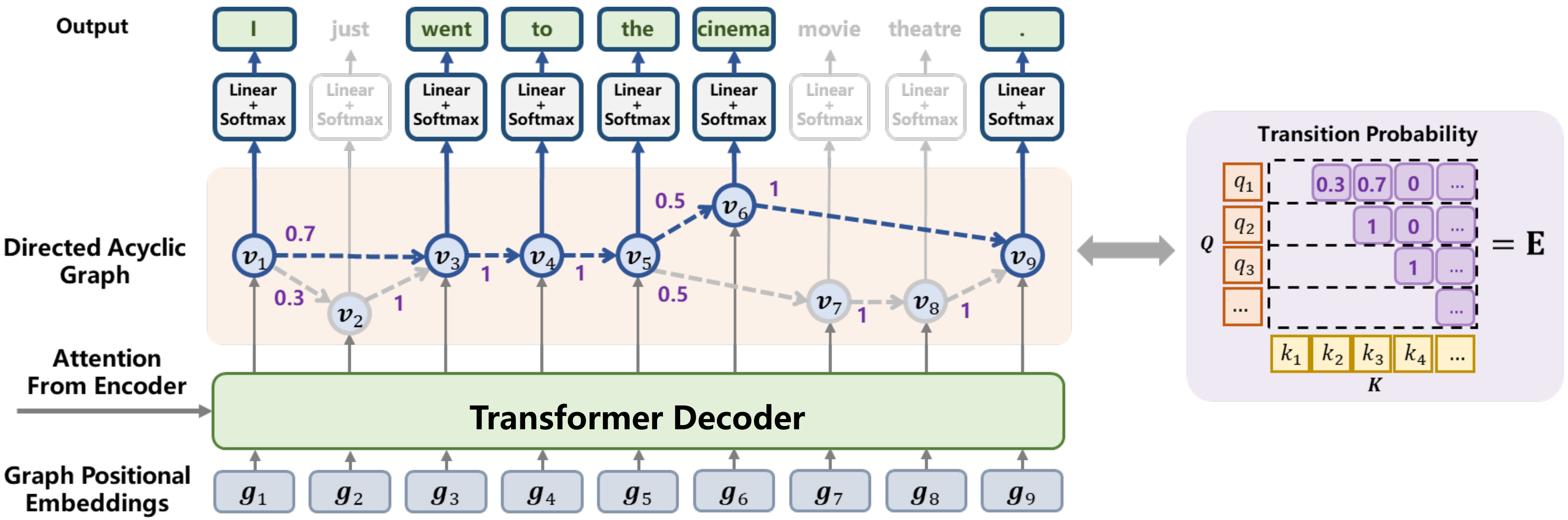


Path $\quad A = \{1, 3, 4, 5, 6, 9\}$

$$P_\theta(A|X) = \prod_{i=1}^{M-1} P_\theta(a_{i+1}|a_i, X) = \prod_{i=1}^{M-1} \mathbf{E}_{a_i, a_{i+1}},$$

Directed Acyclic Transformer (DA-Transformer)

- Step 3: Predict the tokens on the selected path



Path $A = \{1, 3, 4, 5, 6, 9\}$ Reference $Y =$ `I went to the cinema .`

$$P_\theta(Y|A,X) = \prod_{i=1}^{M} P_\theta(y_i|a_i,X) = \prod_{i=1}^{M} \mathrm{softmax}(\mathbf{W_P v}_{a_i})$$

Directed Acyclic Transformer (DA-Transformer)

- Probability Modelling

$$P_\theta(Y|X) = \sum_{A\in\Gamma} P_\theta(Y, A|X) = \sum_{A\in\Gamma} P_\theta(A|X)P_\theta(Y|A, X),$$

All possible paths

Transition Probability $$P_\theta(A|X) = \prod_{i=1}^{M-1} P_\theta(a_{i+1}|a_i, X) = \prod_{i=1}^{M-1} \mathbf{E}_{a_i,a_{i+1}},$$

Token Probability $$P_\theta(Y|A, X) = \prod_{i=1}^{M} P_\theta(y_i|a_i, X) = \prod_{i=1}^{M} \mathrm{softmax}(\mathbf{W}_\mathbf{P}\mathbf{v}_{a_i})$$

DA-Transformer – Training

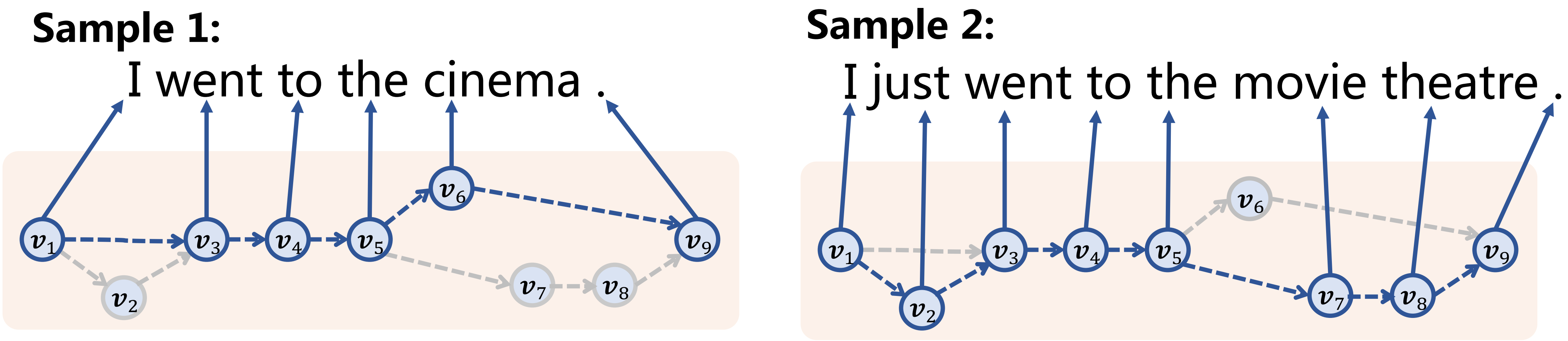
- Cannot obtain multiple references or ground-truth graphs

- **An end-to-end Loss**

$$\mathcal{L} = -\log P_\theta(Y|X) = -\log \sum_{A \in \Gamma} P_\theta(Y, A|X)$$

DA-Transformer – Training

- **Intuitive Explanation**
  - Assign the single reference to paths sparsely
  - Learn the DAG across different training instances

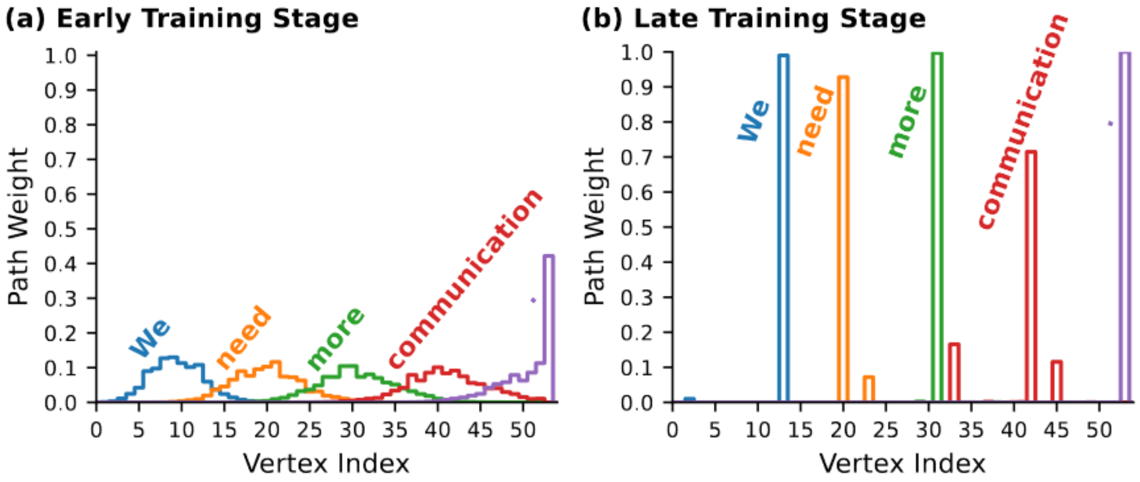


DA-Transformer – Training

$$\mathcal{L} = -\log P_\theta(Y|X) = -\log \sum_{A\in\Gamma} P_\theta(Y, A|X)$$

- Why the objective fits our intuition?

$$\frac{\partial}{\partial\theta}\mathcal{L} = \sum_{A\in\Gamma} w_A \left[\frac{\partial}{\partial\theta}(-\log P_\theta(Y, A|X))\right]$$

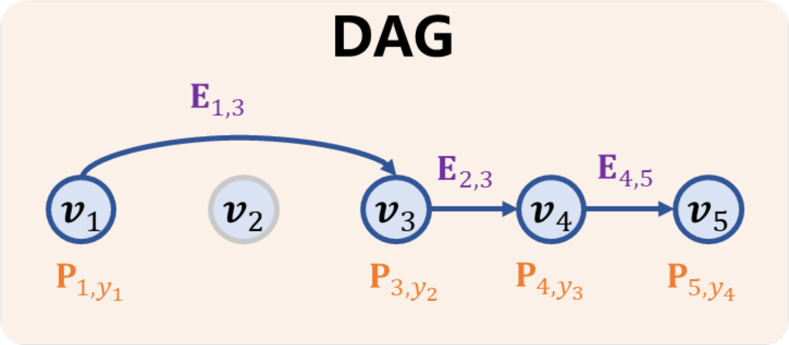


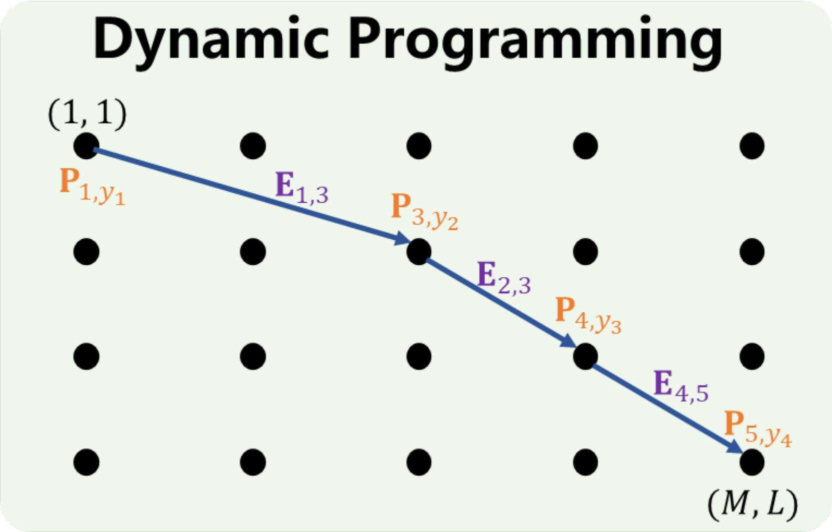
DA-Transformer – Training

$$\mathcal{L} = -\log P_\theta(Y|X) = -\log \boxed{\sum_{A\in\Gamma}} P_\theta(Y, A|X)$$

Enumerate All Paths

- How to get the sum of probability efficiently?





$$A = [1, 3, 4, 5]$$

$$P(Y, A|X) = \mathbf{P}_{1,y_1} + \mathbf{E}_{1,3} + P_{3,y_2} + \mathbf{E}_{2,3} + \mathbf{P}_{4,y_3} + \mathbf{E}_{4,5} + \mathbf{P}_{5,y_4}$$

$$f_{i,u} = \mathbf{P}_{u,y_i} \sum_{v=1}^{u-1} f_{i-1,v}\, \mathbf{E}_{v,u}$$

DA-Transformer – Training

- Dynamic Programming in PyTorch Operations

$$f_{i,u} = \mathbf{P}_{u,y_i} \sum_{v=1}^{u-1} f_{i-1,v}\, \mathbf{E}_{v,u}$$

Algorithm 2 Dynamic Programming Algorithm in Pytorch-like Parallel Pseudocode

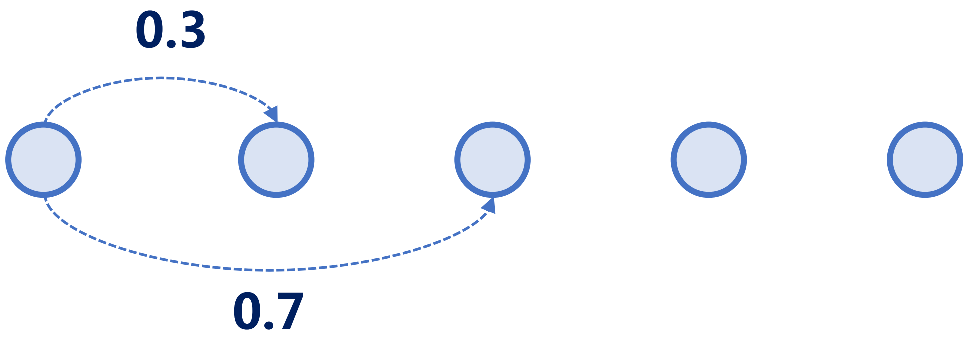
Input: Target Length M, Graph Size L, Target Sentence Y, Transition Matrix $\mathbf{E} \in \mathbb{R}^{L \times L}$, Token Distributions $\mathbf{P} \in \mathbb{R}^{L \times |\mathbb{V}|}$
Initialize a zero matrix $f \in \mathbb{R}^{M \times L}$
$f[1, 1] := 1$
for $i = 2, 3, \cdots, M$ **do**
  $f[i, :] := \mathbf{P}[:, y_i] \otimes (f[i-1, :] \times \mathbf{E})$ # $\otimes$ is the element-wise multiplication, $\times$ is the vector-matrix multiplication
end for
Update the model by minimizing $\mathcal{L} = -\log f[M, L]$.

O(M) PyTorch Operations

DA-Transformer – Inference

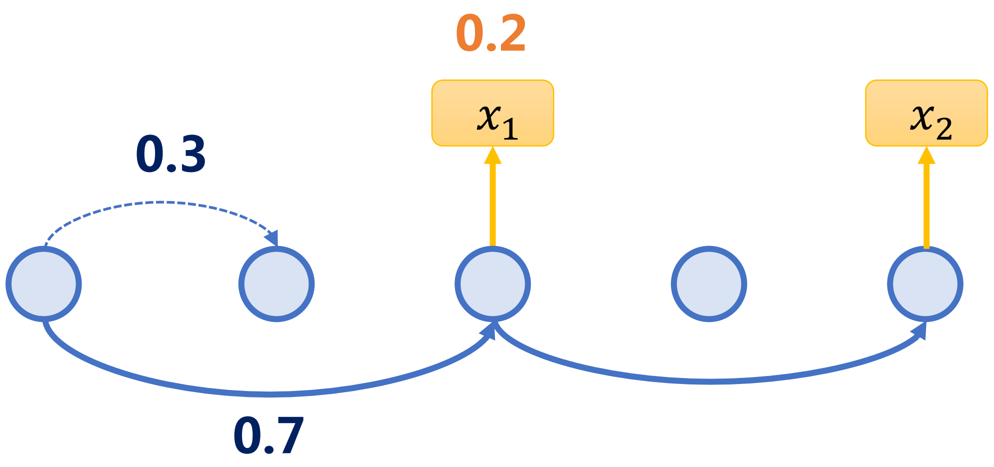
- After parallel prediction, decoding one sentence from the DAG

- Greedy



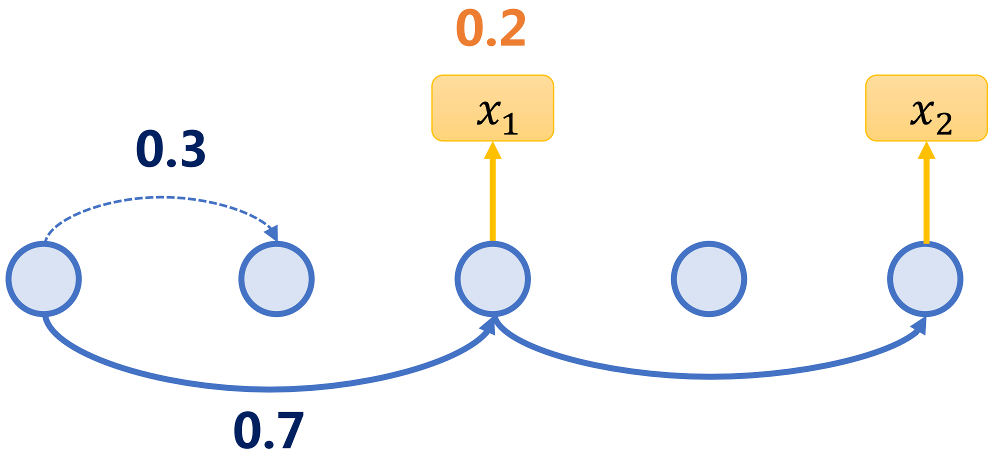
DA-Transformer – Inference

- After parallel prediction, decoding one sentence from the DAG
- Greedy

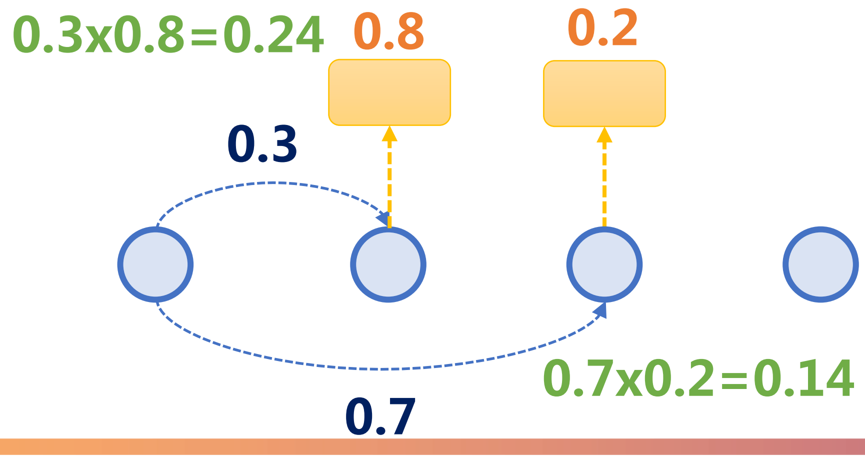


DA-Transformer – Inference

- After parallel prediction, decoding one sentence from the DAG
- Greedy



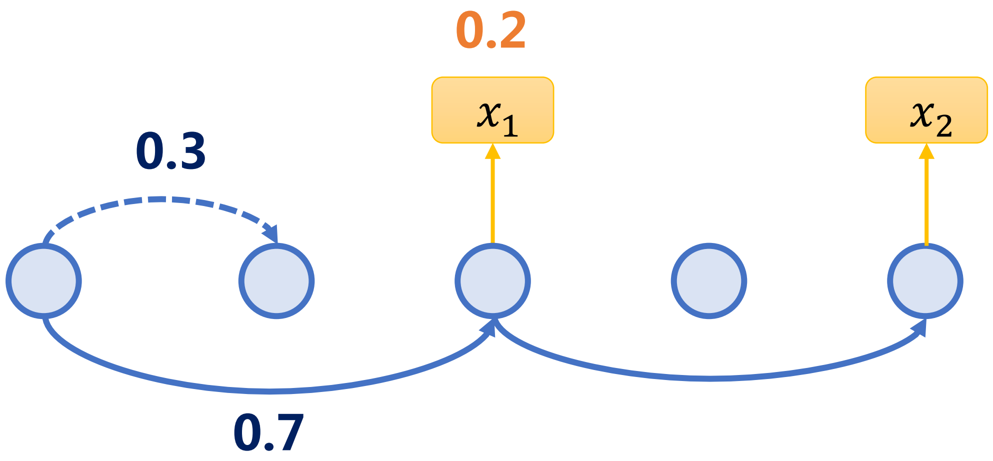
- Lookahead



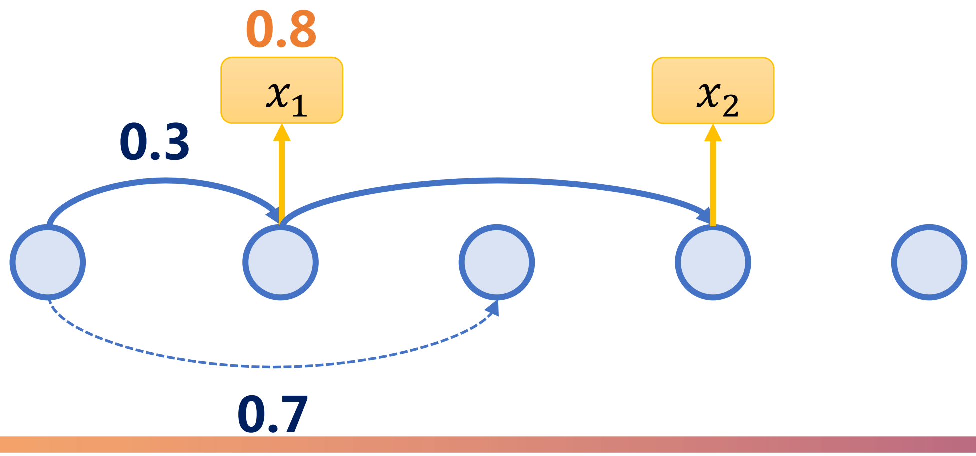
DA-Transformer – Inference

- After parallel prediction, decoding one sentence from the DAG

- Greedy



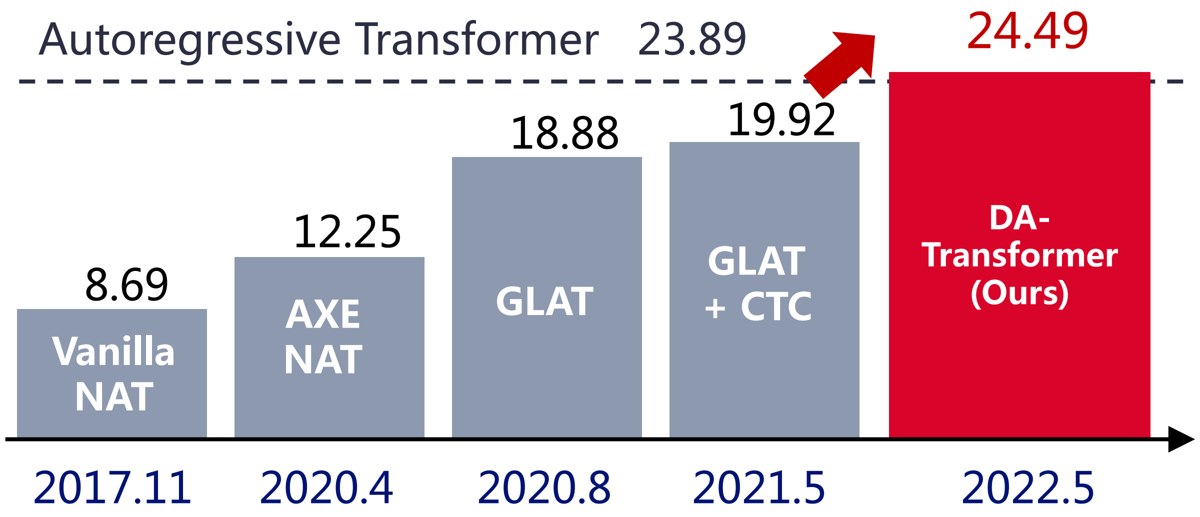
- Lookahead



Main Results

Model	Iter #	Avg Gap ↓ Raw	Avg Gap ↓ KD	Speedup
Transformer (Vaswani et al., 2017)	M	0.45	0.49	1.0x
Transformer (Ours)	M	0	0	1.0x
CMLM (Ghazvininejad et al., 2019)	10	3.00	1.37	2.2x
SMART (Ghazvininejad et al., 2020b)	10	2.67	0.67	2.2x
DisCo (Kasai et al., 2020)	≈4	2.43	0.59	3.5x
Imputer (Saharia et al., 2020)	8	3.07	0.04	2.7x
CMLMC (Anonymous, 2021a)	10	1.35	0.15	1.7x
Vanilla NAT (Gu et al., 2018)	1	15.78	8.26	15.3x
AXE† (Ghazvininejad et al., 2020a)	1	7.36	4.34	14.2x
CTC (Libovický & Helcl, 2018)	1	9.41	3.47	14.6x
GLAT (Qian et al., 2021a)	1	6.05	2.59	15.3x
OaXE† (Du et al., 2021)	1	5.4	2.0	14.2x
CTC + GLAT (Qian et al., 2021a)	1	3.52	1.98	14.6x
CTC + DSLP (Huang et al., 2021)	1	3.44	0.73	14.0x
DA-Transformer + Greedy (Ours)	1	1.47	0.75	14.0x
+ Lookahead	1	1.20	0.58	13.9x
+ BeamSearch	1	0.61	0.18	7.1x
+ BeamSearch + 5-gram LM	1	**0.30**	**0.05**	7.0x

Avg Gap = BLEU gap against the best AT averaged on WMT14 En↔De and WMT17 Zh↔En



BLEU Score (WMT17 Zh-En)

1. Outperforming existing **non-iterative NATs by 3 BLEU** without KD

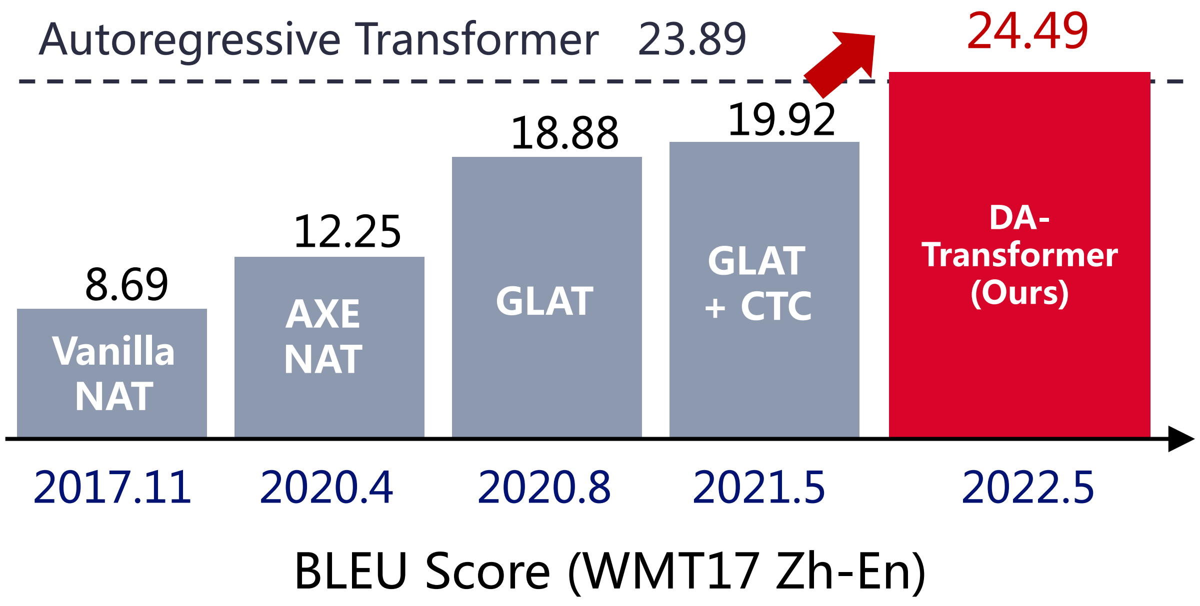
2. Outperforming AT on some datasets

3. Achieving **7x~14x speedups in decoding**

Main Results

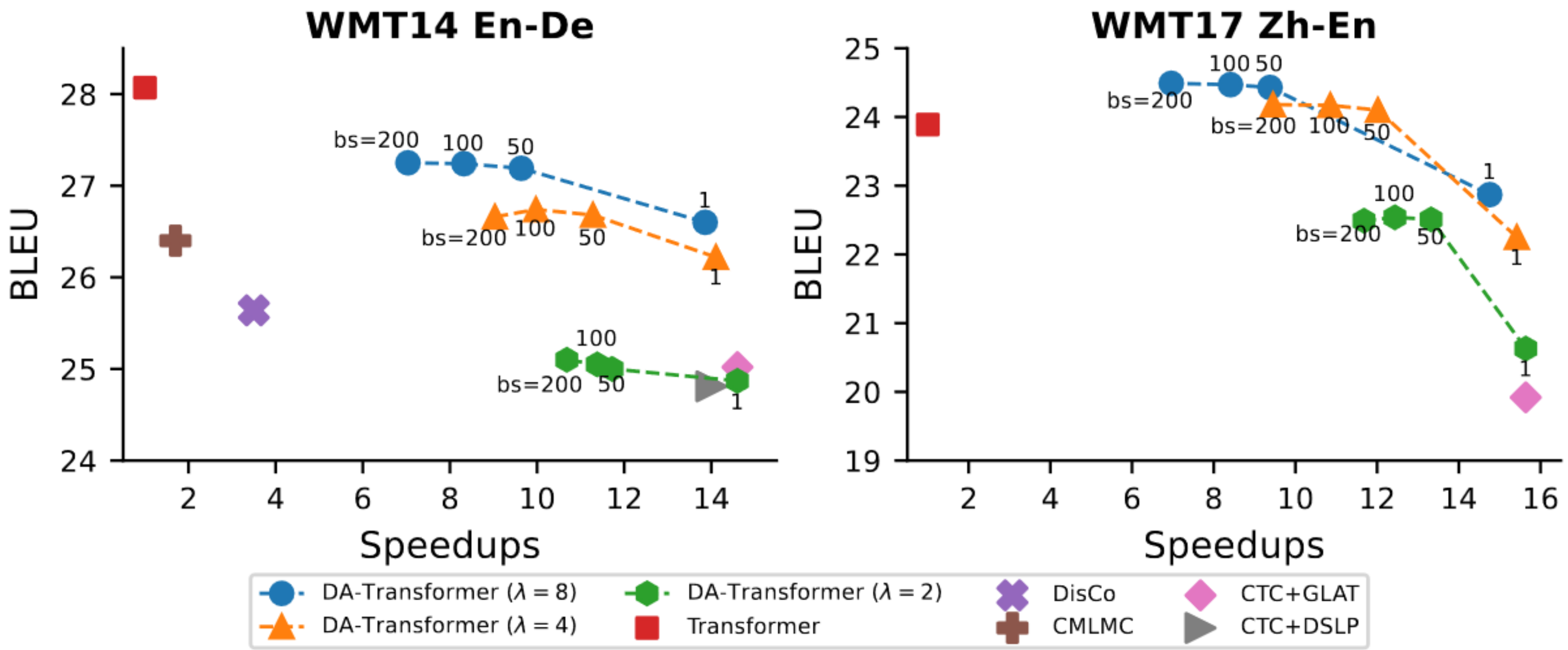
Model	Iter #	Avg Gap ↓ Raw	Avg Gap ↓ KD	Speedup
Transformer (Vaswani et al., 2017)	M	0.45	0.49	1.0x
Transformer (Ours)	M	0	0	1.0x
CMLM (Ghazvininejad et al., 2019)	10	3.00	1.37	2.2x
SMART (Ghazvininejad et al., 2020b)	10	2.67	0.67	2.2x
DisCo (Kasai et al., 2020)	≈4	2.43	0.59	3.5x
Imputer (Saharia et al., 2020)	8	3.07	0.04	2.7x
CMLMC (Anonymous, 2021a)	10	1.35	0.15	1.7x
Vanilla NAT (Gu et al., 2018)	1	15.78	8.26	15.3x
AXE† (Ghazvininejad et al., 2020a)	1	7.36	4.34	14.2x
CTC (Libovický & Helcl, 2018)	1	9.41	3.47	14.6x
GLAT (Qian et al., 2021a)	1	6.05	2.59	15.3x
OaXE† (Du et al., 2021)	1	5.4	2.0	14.2x
CTC + GLAT (Qian et al., 2021a)	1	3.52	1.98	14.6x
CTC + DSLP (Huang et al., 2021)	1	3.44	0.73	14.0x
DA-Transformer + Greedy (Ours)	1	1.47	0.75	14.0x
+ Lookahead	1	1.20	0.58	13.9x
+ BeamSearch	1	0.61	0.18	7.1x
+ BeamSearch + 5-gram LM	1	**0.30**	**0.05**	7.0x

Avg Gap = BLEU gap against the best AT averaged on WMT14 En↔De and WMT17 Zh↔En



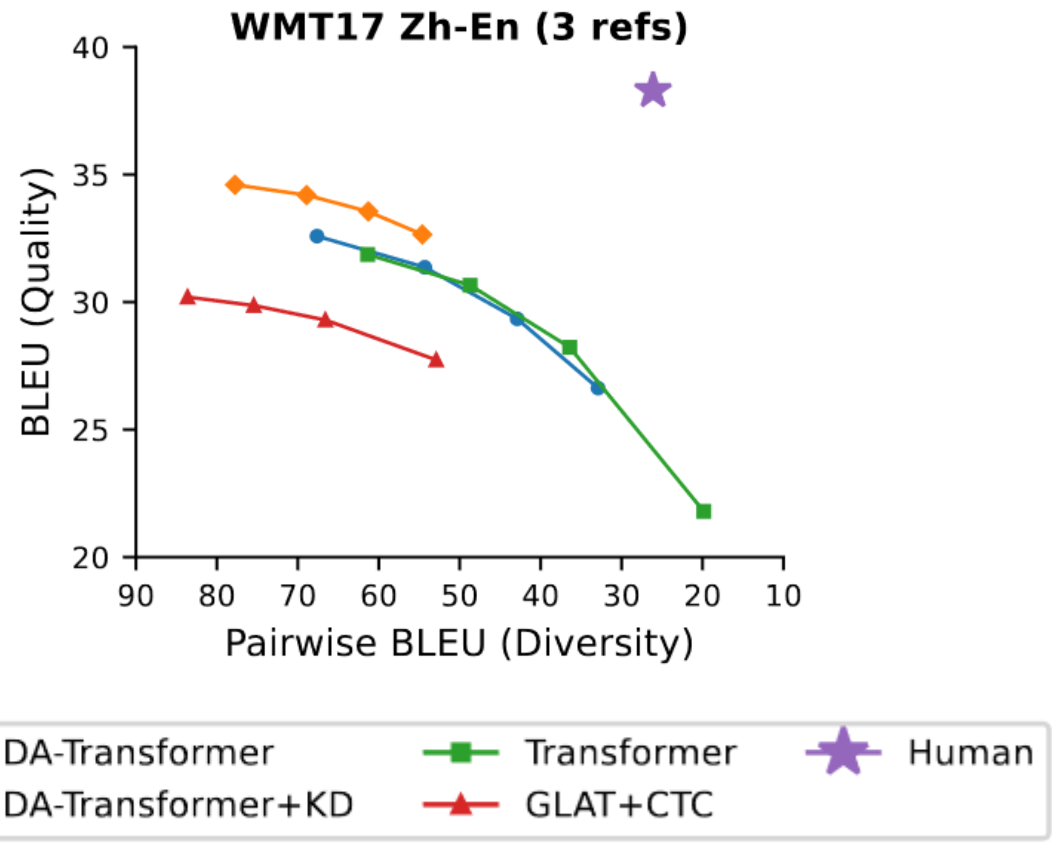
4. **BeamSearch (+ n-gram LM) > Lookahead > Greedy**

Main Results



Provide **flexible tradeoffs between quality and latency** by tuning beam size and graph size

Diverse Generation



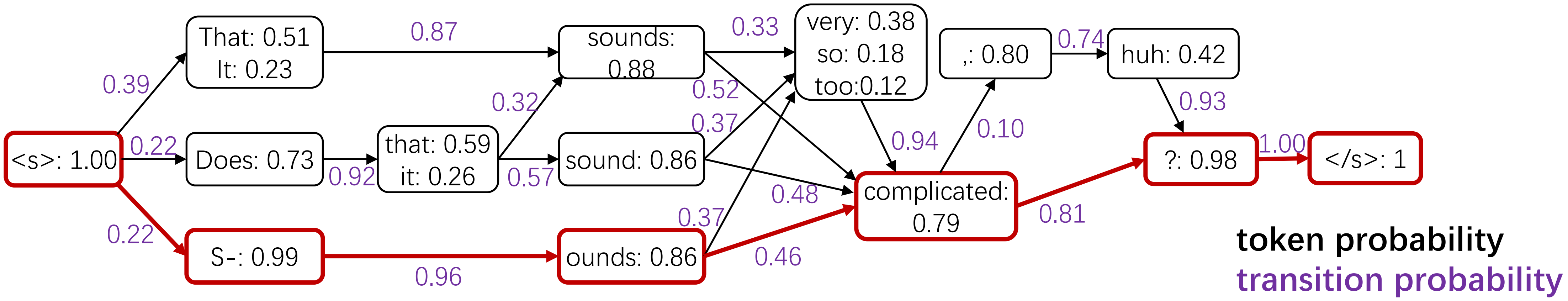
- Facilitate diverse generation by sampling from DAG

Case Study

Source: 听 起来 很 复杂 ？ **Reference:** S- ounds tricky ?

Vanilla NAT: It ounds sounds sounds complicated ?

DA-Transformer:



Rank	Hypotheses of BeamSearch	Score
1	S- ounds complicated ?	-0.55
2	S- ounds very complicated ?	-0.66
3	Does that sound very complicated ?	-0.79
4	S- ounds very complicated , huh ?	-0.94

Fei Huang, Hao Zhou, Yang Liu, Hang Li, Minlie Huang. *Directed Acyclic Transformer for Non-Autoregressive Machine Translation.* **ICML 2022 (CCF-A, Patent Pending).**

Quick Impact

Our Code Released in May

Implement Viterbi decoding #5

Open shaochenze wants to merge 4 commits into `thu-coai:main` from `shaochenze:main`

Pull Request from EMNLP paper

Conversation 0 Commits 4 Checks 0 Files changed 2

shaochenze commented 2 hours ago First-time contributor

Great research and thanks for opening source code! We implement Viterbi decoding algorithms in this pull request, which can find the output that maximizes P(A,Y|X) / |Y|^{beta}. The speed of Viterbi decoding is slightly slower than Lookahead. The performance of Viterbi decoding is between Lookahead and beam search.

Fuzzy Alignments in Directed Acyclic Graph for Non-Autoregressive Machine Translation

Anonymous

22 Sept 2022 (modified: 30 Sept 2022) ICLR 2023 Conference B

ICLR 2023 submission

Show details