LEVENSHTEIN TRANSFORMER

Jiatao Gu, Changhan Wang, Jake Zhao (Junbo)*

facebook Artificial Intelligence Research

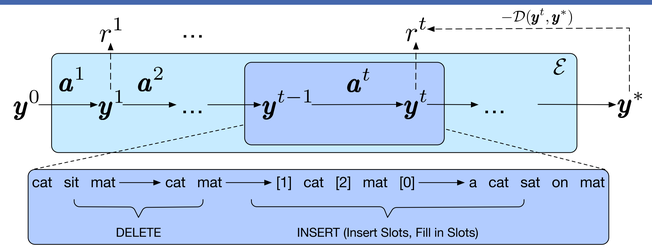
INTRODUCTION

TL;DR We proposed "Levenshtein Transformer" for both sequence generation and refinement based on parallel insertion and deletion operations.

Contribution

- We propose Levenshtein Transformer (LevT) achieving comparable or better results than a strong Transformer baseline in machine translation and summarization, but with much better efficiency;
- We propose a corresponding learning algorithm under the theoretical framework of imitation learning.
- We recognize LevT model as a pioneer attempt to unify sequence generation and refinement.

PROBLEM FORMULATION



Sequence generation as a Markov Decision Process (MDP):

- Basic action = Deletion + Insertion (placeholder, filling-in)

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

IMITATION LEARNING

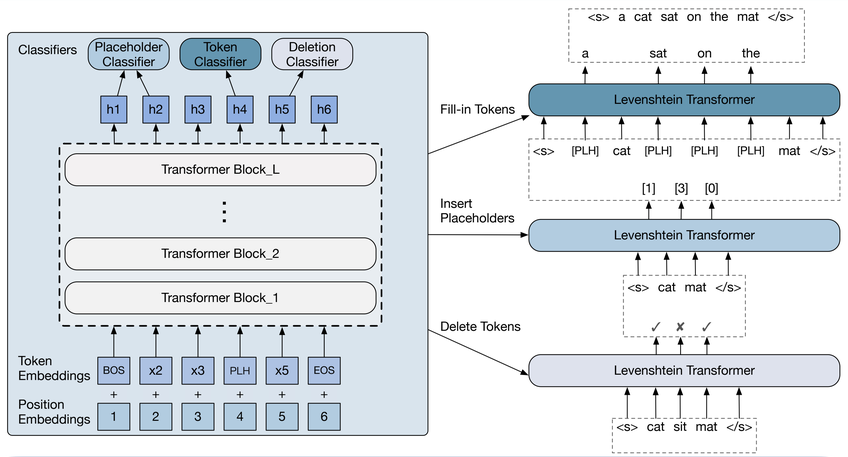
We imitate the behaviors from an expert policy π^*.

- Expert actions are given by comparing the Levenshtein distance with a ground-truth target.

$$\underbrace{\mathbb{E}_{\substack{\boldsymbol{y}_{\text{del}} \sim d_{\bar{\pi}_{\text{del}}} \\ \boldsymbol{d}^* \sim \pi^*}} \sum_{d_i^* \in \boldsymbol{d}^*} \log \pi_\theta^{\text{del}}(d_i^*|i, \boldsymbol{y}_{\text{del}})}_{\textit{Deletion Objective}} + \underbrace{\mathbb{E}_{\substack{\boldsymbol{y}_{\text{ins}} \sim d_{\bar{\pi}_{\text{ins}}} \\ \boldsymbol{p}^*, \boldsymbol{t}^* \sim \pi^*}} \left[\sum_{p_i^* \in \boldsymbol{p}^*} \log \pi_\theta^{\text{plh}}(p_i^*|i, \boldsymbol{y}_{\text{ins}}) + \sum_{t_i^* \in \boldsymbol{t}^*} \log \pi_\theta^{\text{tok}}(t_i^*|i, \boldsymbol{y}'_{\text{ins}}) \right]}_{\textit{Insertion Objective}}$$

- **Learning to delete**: input from [1] the initial input or [2] the output from model's insertion output;
- **Learning to insert**: input from [1] the model's deletion output or [2] randomly deletion from the target;

ARCHITECTURE



LevT is mainly based on TransformerDecoder with additional classifiers for three operations and full self-attention.

Weight Sharing is in default. However, we can also use separate blocks for each classifiers.

We also propose to "**Early Exit**" which attaches the classifiers to an intermediate block instead of the last to save computation.

GENERATION & REFINEMENT EXAMPLES

(a) __The __too __high __rotation __speed __produces __the __reverse __deformation . → __しかし，回転速度が大きすぎると，逆向きの変形が生じる。

(iteration 1)
nothing to delete >>
insert >> [__][回転][回転][すぎ][ると][逆][変形][が生じる][。]
delete >> [__][~~回転~~][回転][すぎ][ると][逆][変形][が生じる][。]
(iteration 2)
insert >> [__][回転][速度が][すぎ][ると][，][逆][変形][が生じる][。]
(iteration 3)
nothing to delete >>
insert >> [__][回転][速度が][すぎ][ると][，][逆][変形][が生じる][。]
nothing to delete, nothing to insert >> **[Terminate]**

(b) __Some __possible __structures __and __circuits __were __proposed __and __verified . → __いくつかの可能な構造と回路を提案し検証した。

(iteration 1)
nothing to delete >>
insert >> [__][可能な][構造][回路][回路][を提案し][，][検証した][。]
delete >> [__][可能な][~~構造~~][回路][回路][を提案し][，][検証した][。]
(iteration 2)
insert >> [__][いくつかの][可能な][構造と][回路][を提案し][，][検証した][。]
nothing to delete, nothing to insert >> **[Terminate]**

(a) In the tag , insert the ActionScript code to create the behavior . → Fügen Sie im Tag den ActionScript-@@ Codes ein , um das Verhalten zu erstellen .

(iteration 1)
delete >> ~~Klicken~~ Sie im ~~Tag des~~ ActionScript-@@ Codes ~~einfügen~~ , um das Verhalten zu erstellen .
insert >> Fügen Sie im Tag des ActionScript-@@ Codes ein , um das Verhalten zu erstellen .
nothing to delete, nothing to insert >> **[Terminate]**

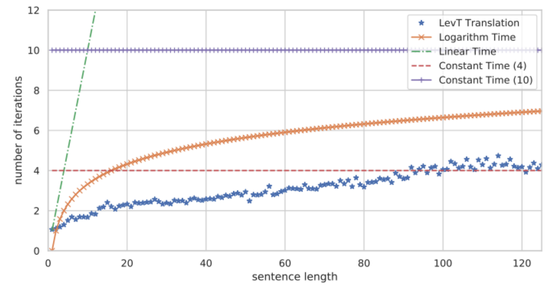
(b) In the tag , insert the ActionScript code to create the behavior . → Verwenden Sie die Schaltfläche " Bearbeiten , " um eine neue Java@@ Script@@ Aktion zu ändern oder zu erstellen .

(iteration 1)
delete >> Verwenden Sie die Schaltfläche " Bearbeiten , " um eine neue ~~JavaScript~~ Aktion zu ändern oder erstellen .
insert >> Verwenden Sie die Schaltfläche " Bearbeiten , " um eine neue JavaScript-@@ Aktion zu ändern oder zu erstellen .
nothing to delete, nothing to insert >> **[Terminate]**

SEQUENCE GENERATION TASKS

We evaluate LevT on machine translation (Ro-En, En-De, En-Ja) and summarization. We also add distillation data as the expert policy.

	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**



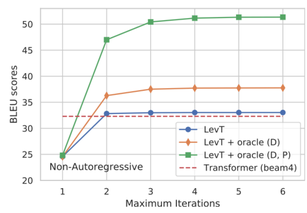
Average number of refinement iterations v.s. length (1~4).

SEQUENCE REFINEMENT TASKS

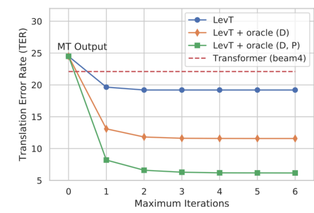
We also evaluate the proposed LevT on automatic post-editing tasks for MT. We test on two simulated and one real datasets.

Dataset		MT system	Do-Nothing	Transformer	Levenshtein Transformer Scratch	Levenshtein Transformer Zero-shot	Levenshtein Transformer 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**

Simulated collaboration with human by combining LevT with oracle "insertion" and "deletion" operations.



(a) Test set BLEU scores for WMT Ro-En



(b) Test set TER scores for Real APE En-De

Please scan for the full paper details and the released code based on Fairseq. If interested, please cite as follows:

Gu, Jiatao, Changhan Wang, and Jake Zhao. "Levenshtein Transformer." *NeurIPS* (2019).

*New York University / Tigerobo Inc.