Text Editing as Imitation Game

REVERSE EXPO

Ning Shi, Bin Tang, Bo Yuan, Longtao Huang, Yewen Pu, Jie Fu, and Zhouhan Lin

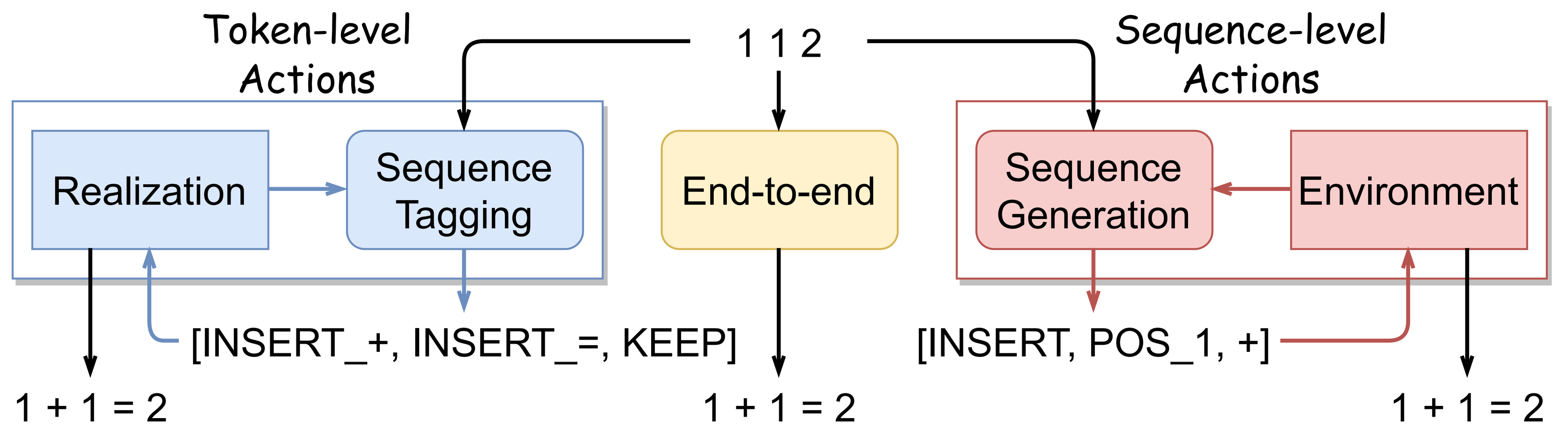
TechAid

Text Editing

Text editing, such as grammatical error correction, arises naturally from imperfect textual data.

Two primary methods to solve text editing:

- End-to-end
- Sequence tagging (token-level actions)



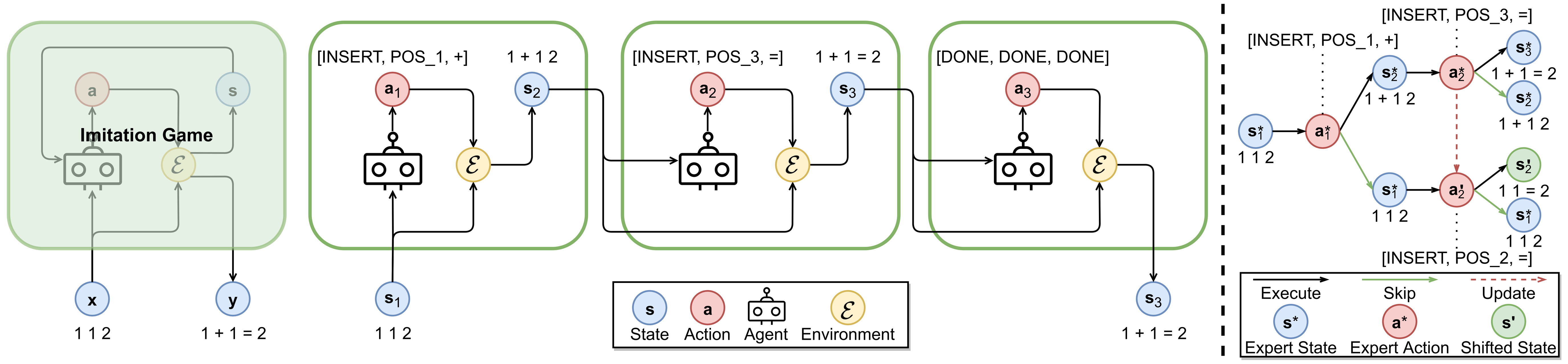
End-to-end

Pros – Has the advantage of simplicity by giving direct input-output pairs.
Cons – Struggles in carrying out localized, specific fixes while ***keeping*** the rest of the sequence intact.

Sequence Tagging

Pros – Appropriate when outputs highly overlap with inputs by assigning no-op (e.g., KEEP).
Cons – Action space is limited to ***token-level***, such as deletion or insertion after a token.

MDP Definition



(a) (b)

Following imitation learning, we tear a text editing $\mathcal{X} \longmapsto \mathcal{Y}$ into recurrent subtasks of ***sequence-level*** action generation $\mathcal{S} \longmapsto \mathcal{A}$ defined by an MDP tuple $\boldsymbol{\mathcal{M}} = (\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{E}}, \boldsymbol{\mathcal{R}})$.

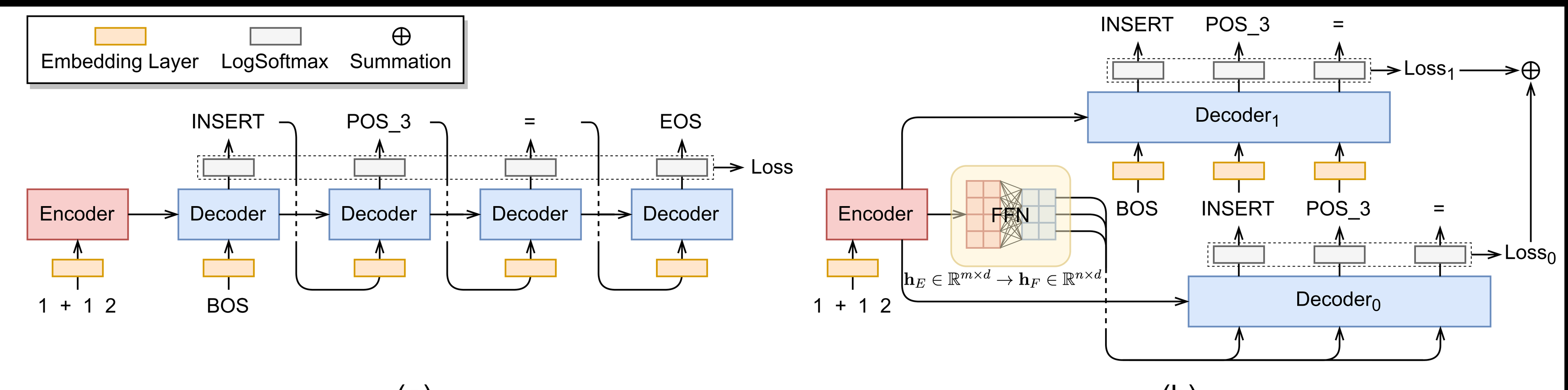
- State $\mathcal{S}$ – is a set of textual sequences s. We think of a source sequence x as the initial state s_1, its target sequence y as the goal state s_T, and every edited sequence in between as an intermediate state s_t.
- Action $\mathcal{A}$ – is a set of action sequences a. Sentence-level actions set free the editing by varying edit metrics E (e.g., Levenshtein distance) as long as $\mathcal{X} \longmapsto \mathcal{Y}$ by $\mathcal{A}_E$.
- Transition matrix $\mathcal{P}$ – can be omitted since we know it is always 1 due to the nature of text editing.
- Environment $\mathcal{E}$ – responds to an action and updates the game state accordingly by $s_{t+1} = \mathcal{E}(s_t, a_t)$.
- Reward function $\mathcal{R}$ – can be omitted as well since we focus on supervised behavior cloning in this work.

Overall, the formulation of text editing turns out to be a simplified imitation game of $\boldsymbol{\mathcal{M}}_{BC} = (\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{E}})$.

Our Contributions

- Frame text editing as imitation game allowing the highest flexibility to design actions at sequence-level.
- Involve Trajectory Generation (TG) to translate input-output data to state-action demonstrations.
- Propose Trajectory Augmentation (TA) to mitigate distribution shift imitation learning often suffers.
- Introduce Dual Decoders (D2), a non-autoregressive decoder, to boost accuracy, efficiency, and robustness.

Dual Decoders (D2) Structure



(a) (b)

Algorithm 1 Trajectory Generation (TG)

Input: Initial state $\mathbf{x}$, goal state $\mathbf{y}$, environment $\mathcal{E}$, and edit metric $\mathbf{E}$.
Output: Trajectories τ.

$\tau \leftarrow \emptyset$
$\mathbf{s} \leftarrow \mathbf{x}$
$ops \leftarrow \mathrm{DP}(\mathbf{x}, \mathbf{y}, E)$
for $op \in ops$ **do**
  $\mathbf{a} \leftarrow \mathrm{Action}(op)$ ▷ Translate operation to action
  $\tau \leftarrow \tau \cup [(\mathbf{s}, \mathbf{a})]$
  $\mathbf{s} \leftarrow \mathcal{E}(\mathbf{s}, \mathbf{a})$
end for
$\tau \leftarrow \tau \cup [(\mathbf{s}, \mathbf{a}_T)]$ ▷ Append goal state and output action
return τ

Algorithm 2 Trajectory Augmentation (TA)

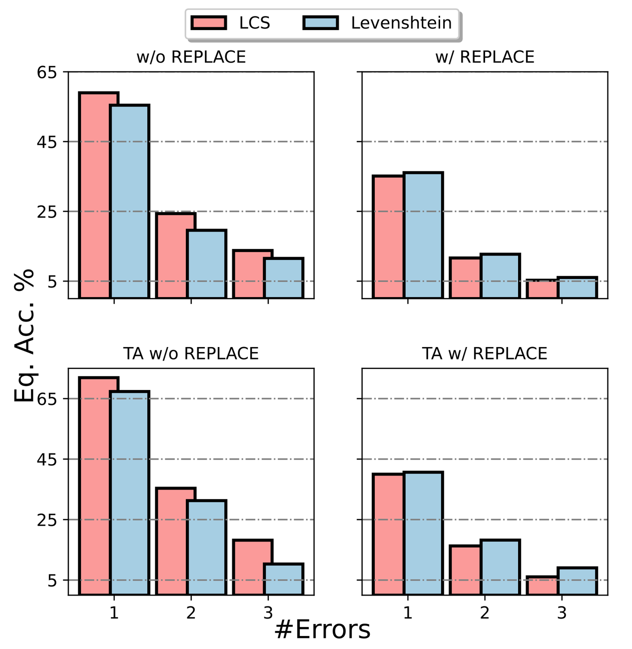
Input: States $\mathbf{S}$, state $\mathbf{s}_t$, expert states $\mathbf{S}^*$, actions $\mathbf{A}$, and environment $\mathcal{E}$.
Output: Augmented states $\mathbf{S}$.

if $|\mathbf{A}| > 1$ **then**
  $\mathbf{a}_t \leftarrow \mathbf{A}.\mathrm{pop}(0)$
  $\mathbf{s}_{t+1} \leftarrow \mathcal{E}(\mathbf{s}_t, \mathbf{a}_t)$
  $\mathbf{S} \leftarrow \mathbf{S} \cup \mathrm{TA}(\mathbf{S}, \mathbf{s}_{t+1}, \mathbf{S}^*, \mathbf{A}, \mathcal{E})$ ▷ Execute action
  $\mathbf{A} \leftarrow \mathrm{Update}(\mathbf{A}, \mathbf{s}_t, \mathbf{s}_{t+1})$
  $\mathbf{S} \leftarrow \mathbf{S} \cup \mathrm{TA}(\mathbf{S}, \mathbf{s}_t, \mathbf{S}^*, \mathbf{A}, \mathcal{E})$ ▷ Skip action
else if $\mathbf{s}_t \notin \mathbf{S}^*$ **then**
  $\mathbf{S} \leftarrow \mathbf{S} \cup [\mathbf{s}_t]$ ▷ Merge shifted state
end if
return $\mathbf{S}$

Arithmetic Equation (AE) Benchmarks

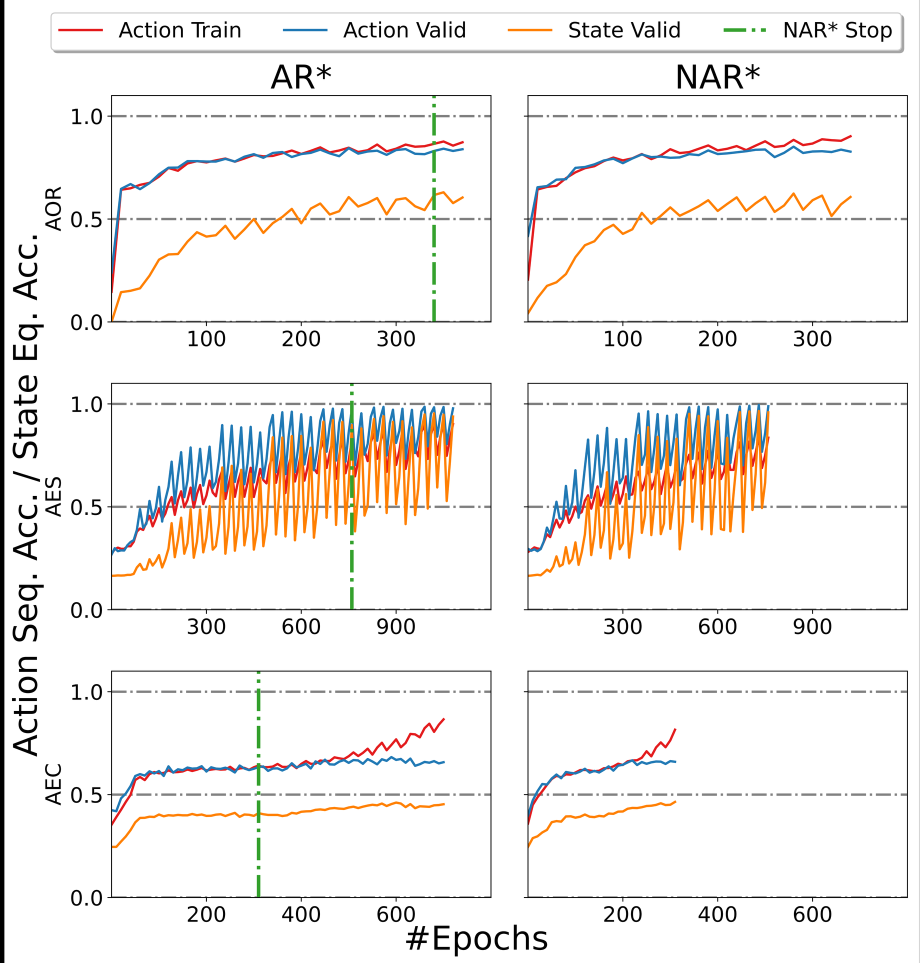
Term	AOR ($N = 10, L = 5, D = 10$K)	AES ($N = 100, L = 5, D = 10$K)	AEC ($N = 10, L = 5, D = 10$K)
Source $\mathbf{x}$	3 6 2 9 3	65 + (25 - 20) - (64 + 32) + (83 - 24) = (- 25 + 58)	- 2 * + 4 10 + 8 / 8 = 8
Target $\mathbf{y}$	- 3 - 6 / 2 + 9 = 3	65 + 5 - 96 + 59 = 33	- 2 + 10 * 8 / 8 = 8
State $\mathbf{s}_t^*$	- 3 - 6 / 2 9 3	65 + 5 - (64 + 32) + (83 - 24) = (- 25 + 58)	- 2 + **4** 10 + 8 / 8 = 8
Action $\mathbf{a}_t^*$	[POS_6, +]	[POS_4, POS_8, 96]	[DELETE, POS_3, POS_3]
Next State $\mathbf{s}_{t+1}^*$	- 3 - 6 / 2 **+** 9 3	65 + 5 - **96** + (83 - 24) = (- 25 + 58)	- 2 + 10 + 8 / 8 = 8
Shifted State $\mathbf{s}_t'$	- 3 - 6 / 2 9 **=** 3	65 + 5 - (64 + 32) + **59** = (- 25 + 58)	- 2 + 4 10 ***** 8 / 8 = 8

Method	AOR ($N = 10, L = 5, D = 10$K)			AES ($N = 100, L = 5, D = 10$K)		AEC ($N = 10, L = 5, D = 10$K)		
	Tok. Acc. %	Seq. Acc. %	Eq. Acc. %	Tok. Acc. %	Eq. Acc. %	Tok. Acc. %	Seq. Acc. %	Eq. Acc. %
End2end	–	–	29.33	84.60	25.20	88.08	57.27	57.73
Tagging	–	–	51.40	87.00	36.67	84.46	46.93	47.33
Recurrence	–	–	58.53	98.63	87.73	83.64	57.47	58.27
Recurrence*	60.30 ± 1.30	27.31 ± 1.33	56.73 ± 1.33	79.82 ± 0.37	22.28 ± 0.52	82.32 ± 0.56	41.72 ± 0.74	42.13 ± 0.75
AR	61.85 ± 0.51	28.83 ± 1.14	59.09 ± 0.95	88.12 ± 2.37	37.05 ± 6.57	$\mathbf{82.61 \pm 0.53}$	45.81 ± 0.36	46.31 ± 0.31
AR*	62.51 ± 0.62	$\mathbf{30.85 \pm 0.41}$	61.35 ± 0.33	99.27 ± 0.32	93.57 ± 2.91	82.29 ± 0.39	45.99 ± 0.49	46.35 ± 0.52
NAR	59.72 ± 0.70	24.16 ± 1.16	51.64 ± 1.97	83.87 ± 1.60	29.49 ± 2.51	80.28 ± 0.76	44.91 ± 1.71	45.40 ± 1.78
NAR*	$\mathbf{62.81 \pm 0.89}$	30.13 ± 1.31	$\mathbf{61.45 \pm 1.61}$	$\mathbf{99.51 \pm 0.13}$	$\mathbf{95.67 \pm 0.93}$	81.82 ± 0.68	$\mathbf{45.97 \pm 1.07}$	$\mathbf{46.43 \pm 1.10}$
AR +TA	62.35 ± 0.61	32.28 ± 0.67	63.56 ± 1.06	88.05 ± 1.20	38.39 ± 3.45	$\mathbf{83.94 \pm 0.42}*$	49.36 ± 1.23	49.83 ± 1.21
AR* +TA	62.58 ± 0.63	33.01 ± 1.31	65.73 ± 1.38	99.44 ± 0.27	95.24 ± 2.38	83.39 ± 0.74	48.95 ± 0.65	49.47 ± 0.73
NAR +TA	61.30 ± 0.86	32.04 ± 1.99	63.75 ± 2.08	90.38 ± 2.21	47.91 ± 8.18	81.36 ± 0.40	48.01 ± 1.07	48.47 ± 1.15
NAR* +TA	$\mathbf{63.48 \pm 0.38}^*$	$\mathbf{34.23 \pm 0.92}^*$	$\mathbf{67.13 \pm 0.99}^*$	$\mathbf{99.58 \pm 0.15}^*$	$\mathbf{96.44 \pm 1.29}^*$	82.70 ± 0.42	$\mathbf{49.64 \pm 0.59}^*$	$\mathbf{50.15 \pm 0.55}^*$



Turning tasks into games that agents feel more comfortable with sheds light on future studies in the direction of reinforcement learning in the context of natural language processing.

Learning Curve



Acknowledgements

This work was supported by Shining Lab and Alibaba Group.
This presentation was partially supported by NSERC Discovery, IVADO, CIFAR through CCAI Chair funding and by the Alberta Machine Intelligence Institute (Amii).