LETI: Learning to Generate from Textual Interactions

Xingyao Wang, Hao Peng, Reyhaneh Jabbarvand, Heng Ji
{xingyao6,haopeng,reyhaneh,hengji}@illinois.edu
UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Motivation

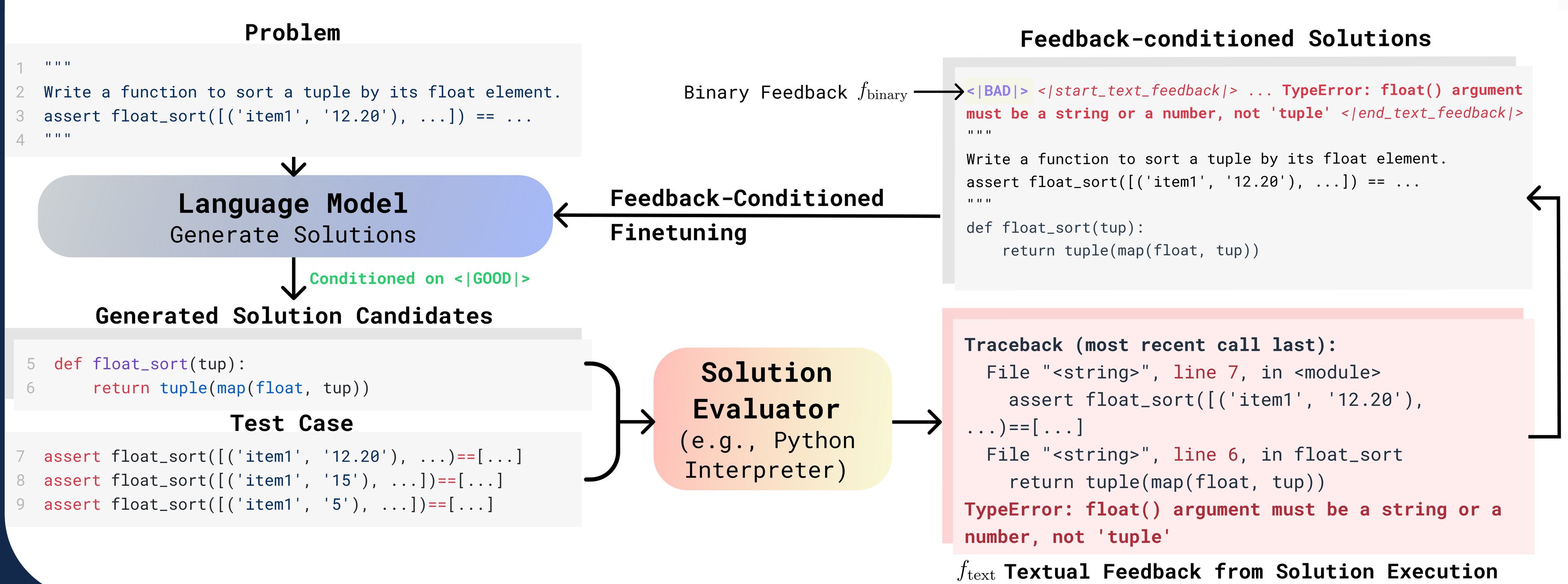
- Fine-tuning language models (LM) can improves task performance.
- Existing techniques commonly fine-tune on input-output pairs (e.g., instruction tuning) or with numerical rewards that gauge the output quality (e.g., RLHF). These coarse-grained labels **tells the model what's good and bad behavior, but not why**!

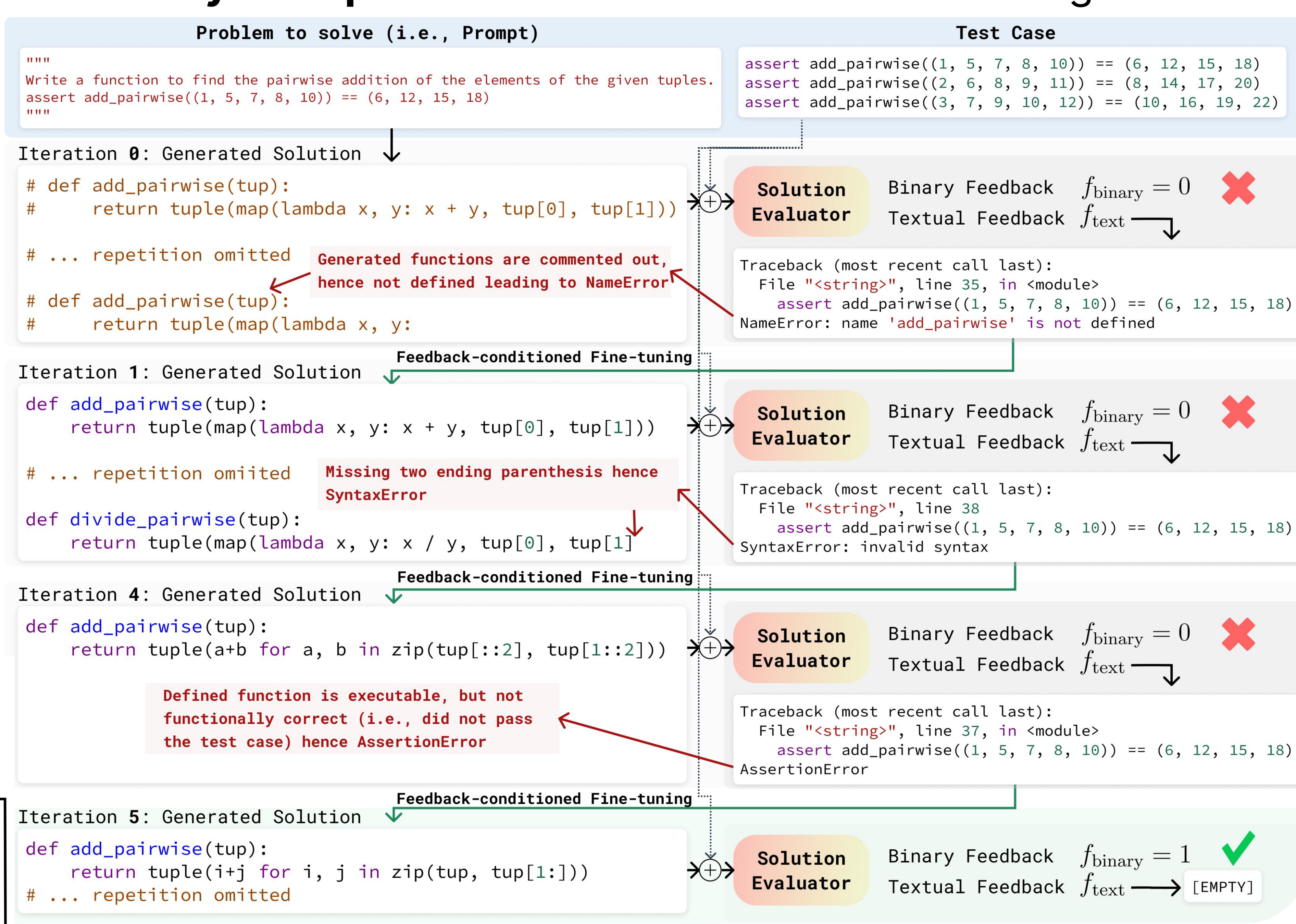
LETI explore LMs' potential to learn from textual interactions that not only *check their correctness with binary labels* but also *pinpoint and explain errors* in their outputs through **textual feedback**.

LETI's Feedback-Conditioned Fine-Tuning (FCFT)

LETI focus on Python code generation. This setting invites a **scalable way to acquire textual feedback**: the error messages and stack traces from code execution using a Python interpreter.

- Given an **instruction**, the LM generate a **solution**. The solution is evaluated by a **solution evaluator** to generate:
  - **Binary feedback**: Use test cases to determine the correctness. Represented as a *special reward token*.
  - **Textual feedback**: error messages and stack traces
- We ***iteratively* finetune** LMs using **language modeling** objective on solution conditioned on (binary+textual) feedback & instruction.
- At inference time, we always conditioned on good binary feedback token to generate solution from a fine-tuned LM.





Regularization of FCFT via Continue Pretraining

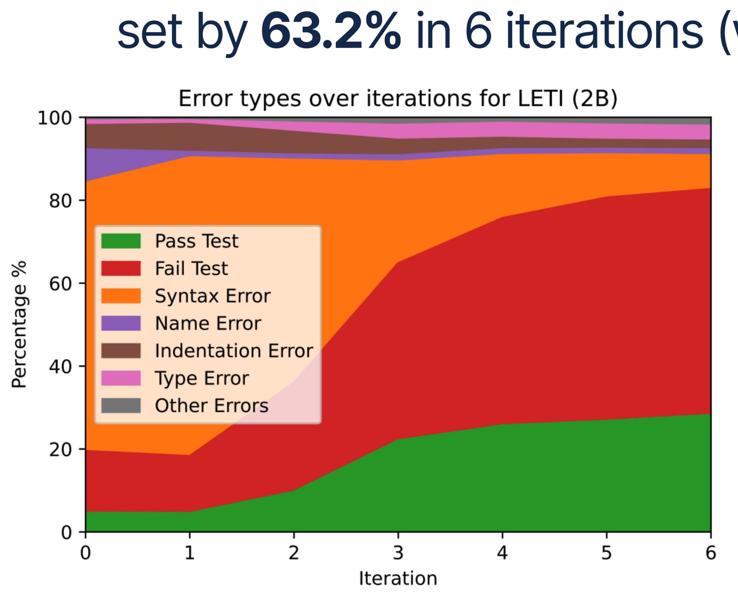
$$\mathcal{L}(\theta) = \underbrace{\frac{1}{|D_{\texttt{FCFT}}|}\sum_{s=F\oplus x\oplus\hat{y}\in D_{\texttt{FCFT}}}}_{\text{(1) Feedback-conditioned Fine-tuning (FCFT)}} \mathcal{L}_{\text{LM}}(s,\theta) + \underbrace{\frac{1}{|D_{\texttt{pre-train}}|}\sum_{s'\in D_{\texttt{pre-train}}}}_{\text{(2) Regularization with pre-training dataset}} \mathcal{L}_{\text{LM}}(s',\theta) \quad (1)$$

$$\mathcal{L}_{\text{LM}}(x,\theta) = -\sum_t \log p_\theta(x_t \mid x_{<t})$$

Evaluation

LETI improve performance by learning to reduce errors

- We train a 2B base LM on MBPP training set and evaluate it on test set.
- LETI increases the proportion of executable code on test set by **63.2%** in 6 iterations (w/o post-processing)!

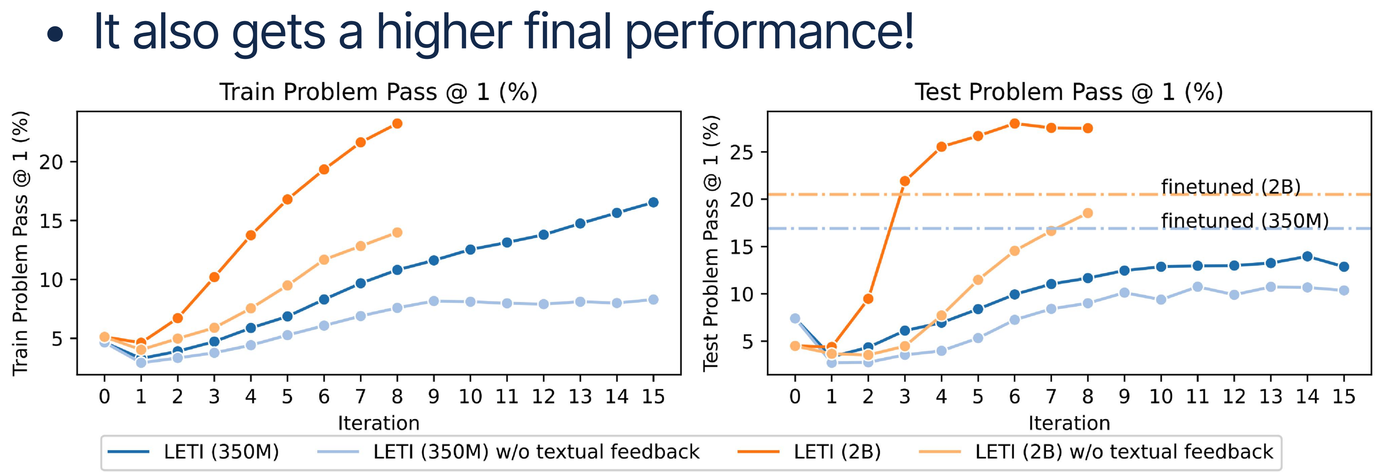


LETI (2B) w/o post-processing	Pre-trained	Fine-tuned
# of AssertionError	**1189**	4356
# of SyntaxError	5179	**652**
# of IndentationError	467	**165**
# of Other Errors	799	**572**
# of Pass Test	366	**2255**
Pass@1 (%)	4.50	**28.00**

LETI (2B) w/ post-processing	Pre-trained	Fine-tuned
# of AssertionError	**3835**	4376
# of SyntaxError	**437**	458
# of NameError	810	**94**
# of Other Errors	**652**	657
# of Pass Test	2266	**2415**
Pass@1 (%)	26.89	**29.53**

Learning from textual feedback is up-to 2x more sample-efficient!

- On a 2B LM, compared to w/o textual feedback, LETI reaches same test performance with 0.5x of the gradient step.
- It also gets a higher final performance!



LETI's performance improvement transfers to other datasets

On LETI models trained on MBPP, we observe **consistent Pass@10 and Pass@100 improvement** across different model sizes on HumanEval.

	HumanEval		
	Pass@1	Pass@10	Pass@100
Pre-trained (350M)	12.56	23.11	35.19
LETI (350M) w/o textual feedback	12.19	21.69	35.62
LETI (350M)	**13.19**	**23.36**	**36.95**
Pre-trained (2B)	**23.70**	36.64	57.01
LETI (2B) w/o textual feedback	19.90	35.62	58.48
LETI (2B)	21.60	37.03	58.28
LETI (2B, trained w/ post-processing)	21.60	**39.51**	**61.46**

LETI retains LM's reasoning and CoT performance

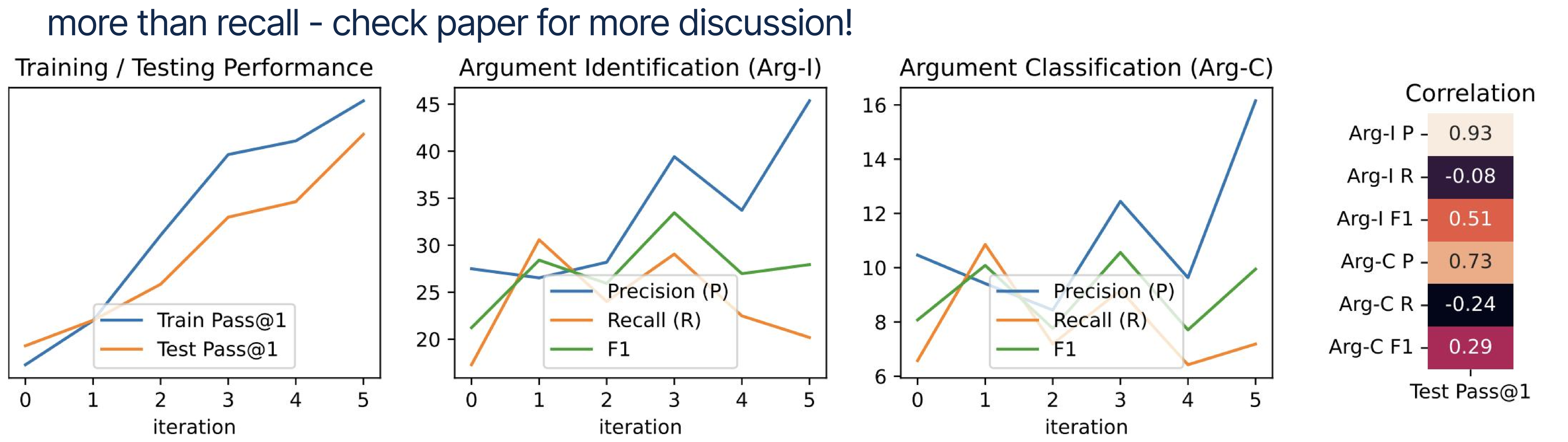
- We observe no significant degradation in out-of-domain reasoning performance (i.e., GSM8K and BBH) after LETI fine-tuning
- Removing regularization degrades performance outside MBPP (e.g., GSM-8K)

	GSM8K	Big-Bench-Hard		
	PaL	direct	CoT*	$\Delta_{\text{CoT-direct}}$
Pre-trained (2B)	40.03	29.67	36.81	7.14
LETI (2B)	38.97	29.41	**37.46**	**8.05**
LETI (2B, w/ post-processing)	**42.99**	**29.81**	36.72	6.91
LETI (2B) w/o textual feedback	41.93	29.23	36.71	7.48
LETI (2B) w/o regularization	32.15	30.06	35.82	5.76
Pre-trained (350M)	13.04	**29.10**	**30.53**	**1.43**
LETI (350M)	**16.68**	28.89	28.86	-0.03
LETI (350M) w/o textual feedback	16.07	28.81	28.72	-0.09
LETI (350M) w/o regularization	7.88	28.00	28.31	0.31

LETI is equally applicable to NLP tasks, If you have a solution evaluator that gives textual feedback

We formulate event argument extraction (EAE) task as a code generation problem [1], and manually designed a rule-based solution evaluator to produce textual feedback.

- LETI can also gradually improve the performance of an EAE task.
- The design of solution evaluator can biases the optimization goal: here it biases the precision more than recall - check paper for more discussion!



[1] Xingyao Wang, Sha Li, and Heng Ji. "Code4Struct: Code Generation for Few-Shot Event Structure Prediction." Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2023.

Code

ILLINOIS

Paper