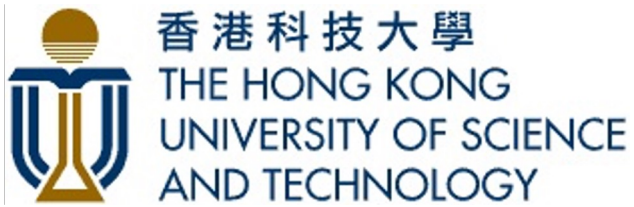
KCTS: Knowledge-Constrained Tree Search Decoding with Token-Level Hallucination Detection

Sehyun Choi, Tianqing Fang, Zhaowei Wang and **Yangqiu Song**
Department of Computer Science and Engineering, HKUST, Hong Kong SAR
{schoiaj, tfangaa, zwanggy, yqsong}@cse.ust.hk



The Hallucination Problem

Definition (by Merriam-Webster)

> *a plausible but false or misleading response generated by an artificial intelligence algorithm*

In this work, we mainly focus on “factually inconsistent” generations.

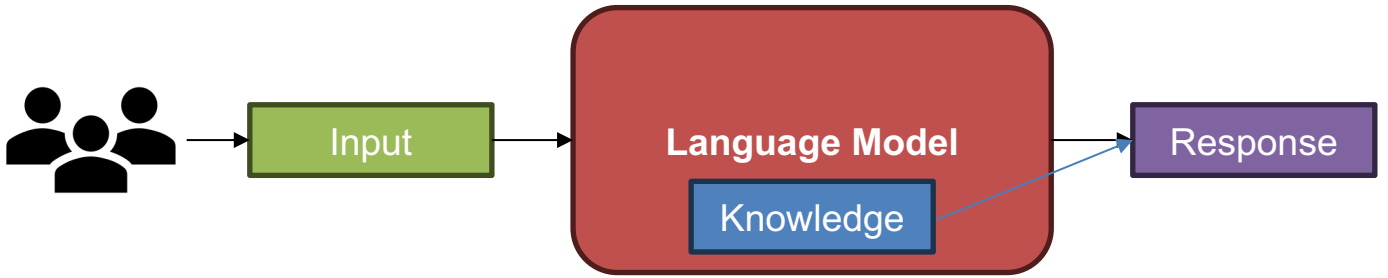
Some examples:

LLaMA-70B	**"Break it or lose it" is a common idiom that means to either take a risk and try to fix a problem or situation, or else lose the opportunity or asset altogether.**
Google PALM	**Joma Tech is a tech company that specializes in AI and machine learning.**

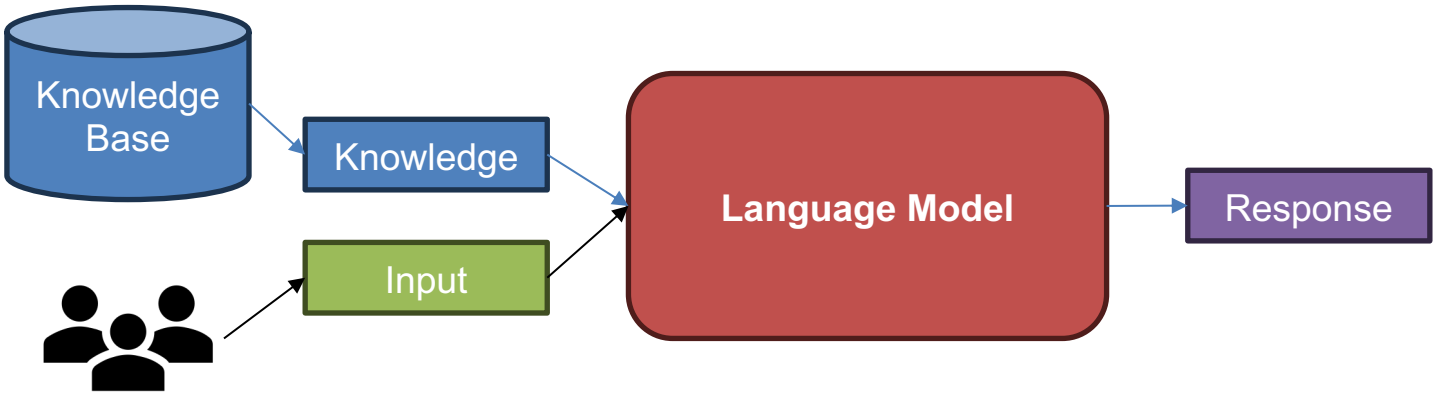
https://www.merriam-webster.com/dictionary/hallucination

The Hallucination Problem: Solutions

1. Memorization



2. Retrieval Augmented Generation (RAG)



Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rock-täschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In Advances in Neural Information Processing Systems, volume 33, pages 9459–9474. Curran Associates, Inc.

The Hallucination Problem: Solutions

Disadvantages: Both still require **training** of the Language Model.

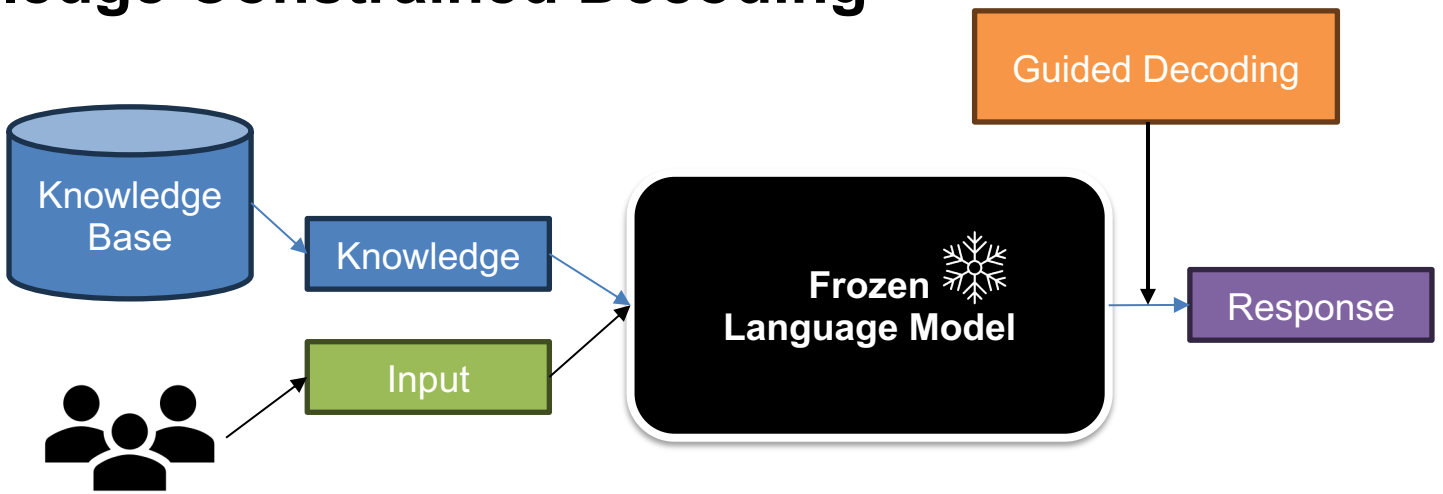
- Growing size of LM: increasing cost
- Non-trainable LLMs (OpenAI ChatGPT, Google PALM, etc.)
- Catastrophic Forgetting: as current models are mostly multi-task models, finetuning on a single task may cause catastrophic forgetting and degenerate the overall performance.

Robert M. French. 1999. Catastrophic forgetting in connectionist networks. Trends in Cognitive Sciences, 3(4):128–135.

Goal

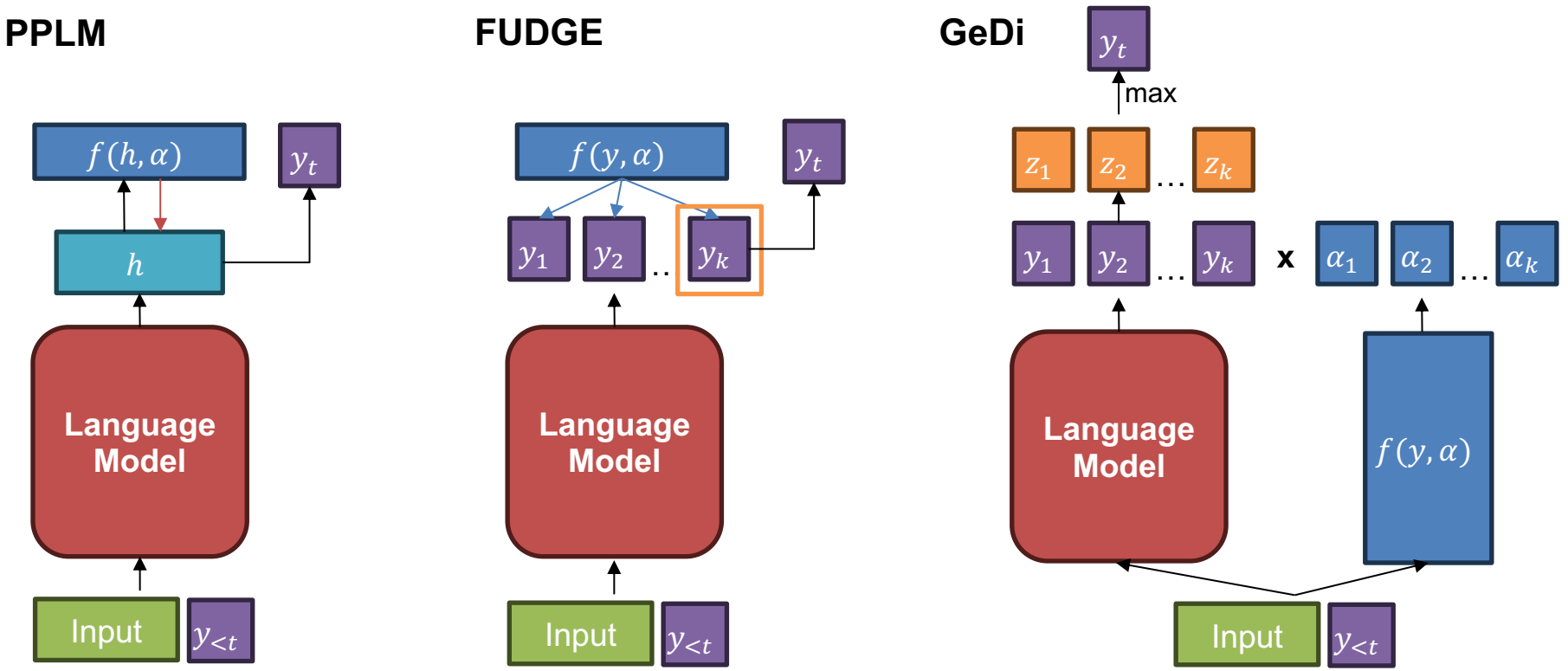
Perform knowledge augmentation **without training** the LM weights.

Knowledge Constrained Decoding



Guided Decoding

Goal: Generate a sequence $\boldsymbol{y}$ that follows the attribute $\boldsymbol{\alpha}$ (guided by $\boldsymbol{f(y, \alpha)}$)



Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In International Conference on Learning Representations.

Kevin Yang and Dan Klein. 2021. FUDGE: Controlled text generation with future discriminators. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3511–3535, Online

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2021. GeDi: Generative discriminator guided sequence generation. In Findings of the Association for Computational Linguistics: EMNLP 2021, pages 4929–4952, Punta Cana, Dominican Republic.

Guided Decoding For Knowledge-Grounded Generation

Knowledge-Groundedness has distinctive features:

- Defined with a reference knowledge (i.e., faithfulness to the reference knowledge)
- Defined on the fully generated sequence.

Proposal:

0. Define $\boldsymbol{f(\alpha, y, k)}$, which denotes the groundedness ($\boldsymbol{\alpha}$) of the generated sequence $\boldsymbol{y}$ with respect to the reference $\boldsymbol{k}$.

1. Approximate sequence-level groundedness to token-level.

2. Use Monte-Carlo Tree Search decoding to better estimate the future impact of current token selection.

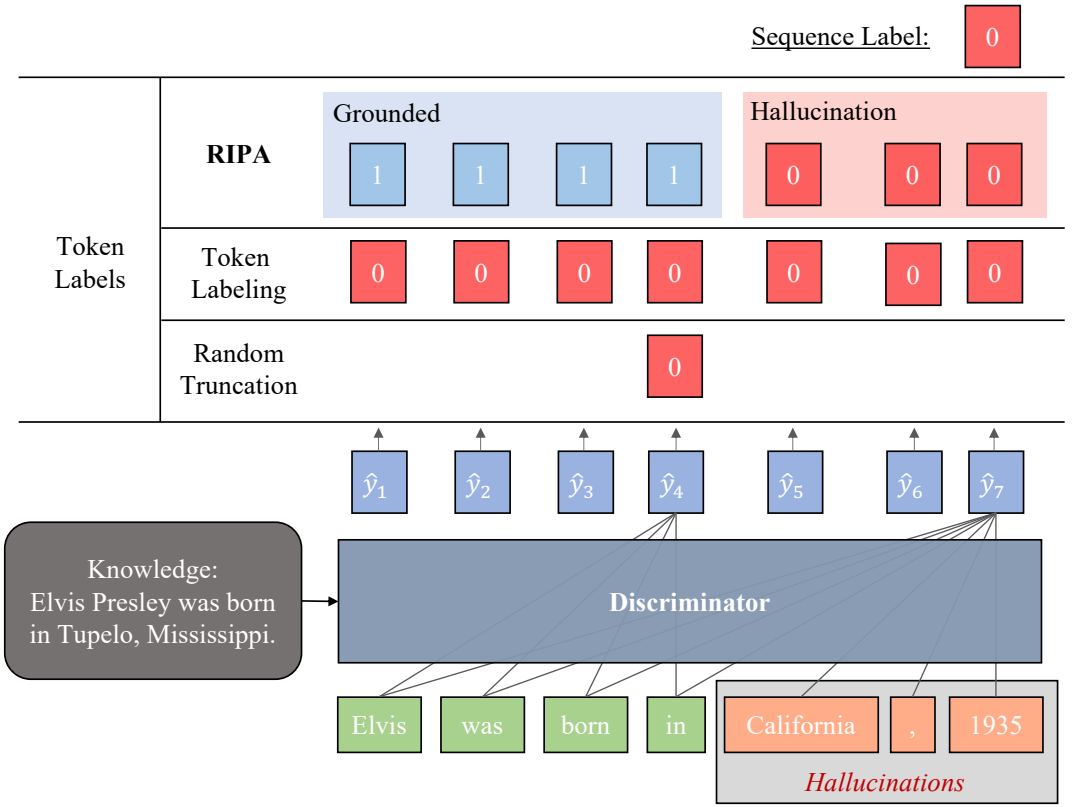
Rémi Coulom. 2007. Efficient selectivity and backup operators in monte-carlo tree search. In Computers and Games, pages 72–83, Berlin, Heidelberg. Springer Berlin Heidelberg.

Proposal 1. RIPA

Reward Inflection-Point Approximation (RIPA)

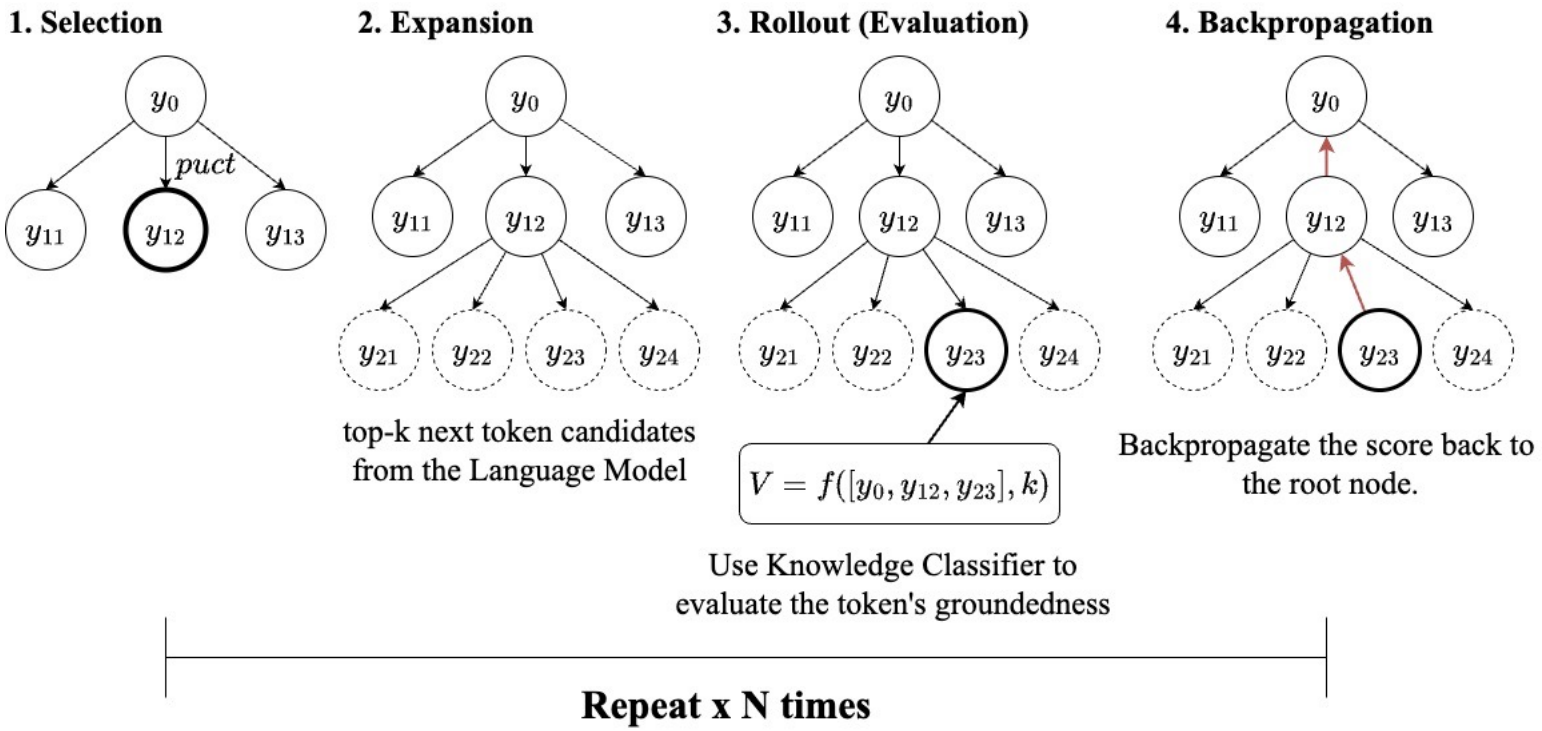
When approximating sequence-level groundedness to token-level, focus on the first position of hallucination.

- Train on all subsequence of the input → sample efficient (*vs. Random Truncation*)
- Does not associate benign tokens before hallucination with the hallucination label. (*vs. Token Labeling*)



Proposal 2. MCTS

Monte-Carlo Tree Search Decoding (MCTS)



- Each node (step 3) is evaluated directly (no rollout) by RIPA Classifier → higher efficiency
- MCTS selects the token that maximizes future score of f based on simulations

Pseudo-Negative Data Generation

Most knowledge-grounded generation benchmarks **do not** include negative data. To train the discriminator, we employed 2 approaches to **pseudo-negative data generation**:

Knowledge Shuffle: Given a positive example (input, knowledge, response) from the dataset, swap the ground truth knowledge with another one randomly sampled from the dataset.

Partial Hallucination: Given a positive example (input, knowledge, response), Perform the knowledge shuffle first, then randomly truncate the response and let the LM complete the response with high temperature sampling.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.
Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, et al. 2022. Scaling instruction-finetuned language models.

Full Method: KCTS

Knowledge-Constrained Tree Search Decoding (KCTS)

→ KCTS = RIPA + MCTS
→ KWD = RIPA + weighted Decoding (FUDGE)

1. We train RIPA with lightweight adapters using LoRA on top of the base LM. (Flan-T5-XL in this study)

2. We decode each token using MCTS with fixed budget (50 simulations), using the groundedness score from the classifier (step 1) to evaluate partial sequences.

Hypothesis: RIPA + MCTS (KCTS) together better estimates full-sequence groundedness, leading to more faithful sequences being generated.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.
Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, et al. 2022. Scaling instruction-finetuned language models.

Experiment

Main Result:

Type	Model	K-Overlap		Token Overlap					UniEval			
		KF1	K-Copy	F1	BLEU	RougeL	ChrF	METEOR	N	C	G	f
LLM	ChatGPT	49.41	39.71	30.32	6.91	26.24	34.95	31.67	57.62	96.41	96.15	95.82
	GPT-3.5	25.91	28.22	22.32	3.01	18.70	27.86	23.06	42.77	98.07	92.42	92.63
SFT	FT5-XL	39.85	37.79	28.08	9.41	25.11	31.17	25.40	76.44	92.36	95.16	97.90
Zero-Shot	FT5-XL	34.50	37.07	21.18	6.81	19.64	24.88	18.53	71.69	82.21	75.70	88.75
	FT5-XXL	28.20	32.33	19.11	5.53	17.55	24.15	17.16	72.37	84.24	75.51	85.89
	T0++	26.94	28.80	17.57	4.13	16.14	19.84	13.37	52.79	85.26	70.14	88.61
Decoding Baselines	FUDGE	55.30	54.04	29.43	11.72	27.35	31.50	26.00	73.68	88.20	83.53	94.54
	NADO	50.20	50.10	27.86	10.57	26.01	29.84	24.51	74.14	88.35	81.10	92.76
	MCTS	55.54	54.21	29.56	11.69	27.48	31.60	26.08	**74.54**	88.16	83.90	95.07
Ours	KWD	**58.19**	56.58	**30.71**	**12.74**	**28.27**	33.40	28.10	70.27	90.51	87.86	97.54
	KCTS	56.06	51.90	30.54	11.42	27.43	**35.22**	**28.92**	62.32	**92.78**	**91.78**	**98.30**

Table 1: Results on WoW Test set (unseen topics). SFT stands for supervised fine-tuning, and FT5 is shorthand for Flan-T5. Under the UniEval metrics, each letter stands for the following: **N** - Naturalness, **C** - Coherence, **G** - Groundedness. For all metrics, a larger number is preferred, except for K-Copy. Note that the performance of LLM in the upper half is for reference only. For each column, **boldface** denotes the best score out of the KCD methods under the FT5-XL backbone, and underline indicates the second best.

Experiment

Main Result:

Type	Model	K-Overlap		Token Overlap					UniEval				MFMA
		KF1	K-Copy	F1	BLEU	RougeL	ChrF	METEOR	Coh.	Cons.	fluency	Relv.	score
LLM	ChatGPT	29.43	17.92	40.45	11.75	27.85	42.96	37.66	93.85	91.67	87.15	87.11	80.62
	GPT-3.5	27.54	16.94	38.96	10.78	26.63	41.17	35.38	92.56	90.33	85.73	85.78	78.74
SFT	FT5-XL	17.04	10.18	32.21	8.74	24.02	30.27	24.47	84.82	86.02	89.90	81.28	64.55
	FT5-XXL	17.45	10.42	31.55	8.43	23.38	29.95	23.91	87.17	88.58	90.00	82.28	68.37
	T0++	22.79	13.65	38.82	13.64	28.06	38.53	33.68	86.57	87.47	89.03	81.09	69.38
Decoding Baselines	FUDGE	18.68	10.70	33.51	9.32	24.83	31.06	24.93	90.52	90.61	83.37	82.00	71.35
	NADO	20.35	11.72	35.10	10.93	26.22	33.50	27.34	92.26	93.72	88.41	84.49	72.01
	MCTS	17.86	10.04	34.59	9.00	25.85	30.90	25.12	94.30	94.28	86.51	85.90	71.28
Ours	KWD	20.39	11.63	36.24	12.30	27.20	34.25	28.46	**96.24**	**96.64**	**91.60**	**88.48**	85.11
	KCTS	**22.97**	13.29	**38.27**	**14.21**	**28.10**	**37.18**	**31.37**	95.85	96.03	90.24	87.16	**85.36**

Table 3: Results on CNN/DM Test set. The guided decoding was conducted with FT5-XL model as the base model. **Coh.**, **Cons.**, and **Relv.** stand for coherence, consistency, and relevance, respectively. As the performance of LLMs is for reference, we highlight the best scores on the last two groups with **boldface** and second-best with underline.

Conclusion

- KCTS is an inference-time decoding algorithm for enhancing knowledge-grounded generations without tuning the LM weights.
- KCTS decoding has shown effectiveness in knowledge-grounded generation tasks.
- KCTS is O(N) times slower than normal generation, where N is the number of simulations per token. This may be improved by using early stopping heuristics proposed for MCTS.

Thanks

Paper

Code

Sehyun Choi, Email: schoiaj@cse.ust.hk