Analyzing RANKGEN Across Different Domains

Jimmy Hoang, Jenny Pang, Alexis Wu

Department of Computer Science, Princeton University

Introduction

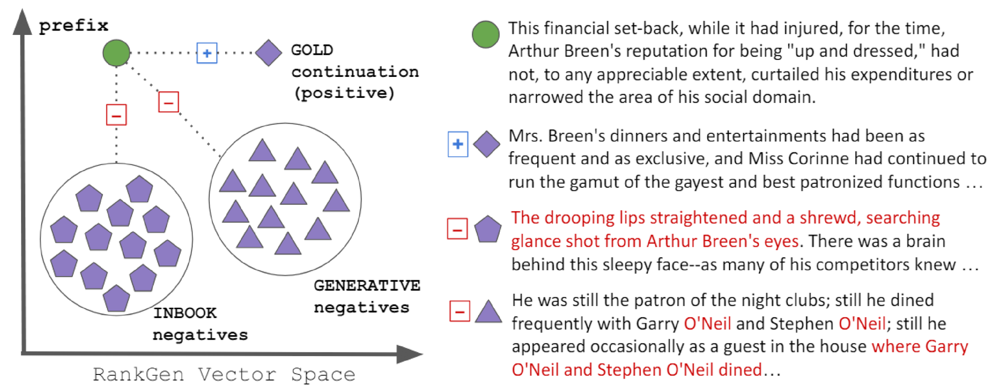
LLMs have gained traction for their utility in prompt-driven text generation. However, these models can produce outputs that are prone to be **repetitive**, **incoherent**, and/or **irrelevant** to the prompt.

- Decoding algorithms like top-k sampling often result in inconsistencies, hallucinations, or factual errors [1].
- Models can focus too much on local context, missing long-range dependencies [2], which can cause coherence and consistency problems in longer texts.

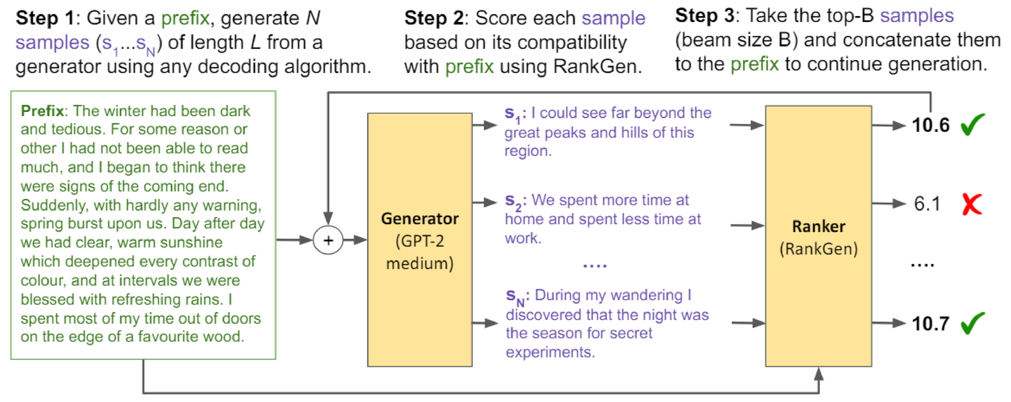
To address these shortcomings, RANKGEN [3] was developed.

RANKGEN

- An encoder model that scores text generations based on individual coherence and consistency with human-written prompts, or prefixes.
- Considers two sequences (prefix and continuation) in contrast to other techniques that only examine singular or local tokens → better captures long-distance relationships.
- RANKGEN's four training domains are PG19, Wikipedia, C4-NewsLike, and C4-WebTextLike.
- Trained using large-scale contrastive learning with **INBOOK** (fluent and sometimes topically-similar but irrelevant and incoherent) and **GENERATIVE** (relevant but potentially containing hallucinations or repetition) negative samples [3]:



- Can be used in a beam search setup [3]:



- Authors found RANKGEN to outperform other decoding algorithms on automatic metrics and human evaluations.

Research Question

Our research serves to broaden the analysis and explore the capabilities of the RANKGEN model.

To what extent does RANKGEN generalize to unseen, specialized domains? Does RANKGEN outperform other decoding strategies for these domains?

Datasets

RANKGEN's training domains were PG19 (Western literature, reference works, and periodicals), Wikipedia (cleaned page articles), and the C4-NewsLike and C4-WebTextLike datasets (subsets of a cleaned version of Common Crawl's web crawl corpus). To enrich our understanding of RANKGEN's capabilities, we conduct experiments using four domain-specific datasets, each with its own novel characteristics:

- **Government Reports:** Long-content form data; content will use more policy and bureaucratic related vocabulary than in training data.
- **Mathematical Proofs:** Sequential data; proofs must follow a certain order to be logical.
- **Python code completion:** Sequential data; code syntax significantly differs from standard English.
- **Poetry:** Does not strictly abide by standard English grammar rules and can have multiple interpretations.

Methodology

RANKGEN Setup

- RANKGEN-base version, trained on all four original domains to maximize generalizability
- GPT2-Medium as the generator language model
- Beam search size: 2, number of samples: 10, sample length: 20 tokens

Dataset Construction

- Using the NLTK and GPT2 tokenizers, constructed 256-token prefixes and 128-token target continuations
- Constructed 128-token prefixes and 64-token targets for the poetry dataset because of its shorter datapoints. (RANKGEN is robust to prefix length [3].)

Evaluation

- MAUVE scoring as an automatic metric to measure similarity between human-written and generated text
- Human evaluation survey with blind A/B testing on RANKGEN vs. top-k GovReport generation quality

Results

Automatic Metrics (MAUVE Scores)

Dataset	RANKGEN-base	top-k
Wikipedia	86.8	73.8
GovReport	87.1	76.7
MetaMathQA	2.5	8.7
Python code	0.8	1.4
Poetry	42.9	28.8

- RANKGEN's GovReport generations are relevant to the prefix, coherent, and not repetitive:

Prefix	Ground-truth continuation	RANKGEN continuation
GovReport, ... Table 1 shows seven effects commonly associated with climate change that DOD has documented. According to a 2010 National Research Council report on making informed decisions about climate change and our October 2009 report on climate change adaptation, most decision makers need a basic set of information to understand and make choices about how to adapt to the effects of climate change. ...	In November 2015, we found that in order for climate information to be useful, it must be tailored to meet the needs of each decision maker, such as an engineer responsible for building a bridge in a specific location, a county planner responsible for managing development over a larger region, or a federal official managing a national-scale program. ...	This information is needed to understand and make a decision about what actions to take or what measures to take as climate change increases. One way DOD has demonstrated its ability to understand and make informed decisions about climate change is by creating several interactive online climate change resource databases, ...

- RANKGEN is unable to solve the given math problem, beginning to make many unnecessary assumptions. Other generations are incorrect proofs or proposals for additional questions:

Prefix	Ground-truth continuation	RANKGEN continuation
MetaMathQA, Carl caused a car accident that resulted in $40,000 worth of property damage and $70,000 worth of medical bills. If Carl's insurance company will pay 80% of the costs, leaving Carl to pay the remaining 20%, how much does Carl personally owe?	The total cost of the car accident is $40,000 + $70,000 = $110,000. Carl's insurance company will pay 80% of the costs, so they will cover 80% of $110,000 = $88,000. This means Carl is responsible for the remaining 20% of the costs, which is 20% of $110,000 = $22,000. Therefore, Carl personally owes $22,000. #### 22000 The answer is: 22000	Let's assume Carl is not insured by any insurance company. Let's assume that Carl only lives in the Chicago area for a short period of time. Carl has a good credit score and is currently unemployed. What is Carl's insurance claim history and what does this tell us about Carl's ability to pay his insurance claim? Let's assume that Carl is a current employee of a non-bank financial company and has an auto insurance policy.

- RANKGEN is unable to follow and provide proper Python code to answer the given instructions:

Prefix	Ground-truth continuation	RANKGEN continuation

Python code instructions, Create a Python script to find the maximum and minimum values for each sublist in a list	def findMinMax(list):
	min_list = []
	max_list = []
	
	for sublist in list:
	min_list.append(min(sublist))
	max_list.append(max(sublist))
	return min_list, max_list
	
	list = [[8, 4], [6, 2, 3], [1, 9, 7]]
	min_list, max_list = find
		(or a tuple of lists, or tuple of lists) #!/usr/bin/env python """ A simple script to find the maximum and minimum values for each sublist in a list or a tuple of lists """ # Find the maximum value from first list list = list [- 1] print'The maximum value is {}'. format (list) print'The minimum value is {}'. format (list) return max (list) # Find the minimum value from first list list = list [- 1] print'The minimum value is {}'.

- For poetry, RANKGEN generates a passage that contains some words relevant to the prefix but is nonsensical:

Prefix	Ground-truth continuation	RANKGEN continuation
Poetry, Let the bird of loudest lay On the sole Arabian tree Herald sad and trumpet be, To whose sound chaste wings obey. But thou shrieking harbinger, Foul precurrer of the fiend, Augur of the fever's end, To this troop come thou not near. From this session interdict Every fowl of tyrant wing, Save the eagle, feather'd king; Keep the obsequy so strict.	Let the priest in surplice white, That defunctive music can, Be the death-divining swan, Lest the requiem lack his right.	Every bird of tyrician wing,ortium The fowl of the tyranny wing;DeliveryDate These steeds of the bird of thy wing,ocally Thy fount and your lord.

Results (cont.)

Human Evaluation

- 9 out of 10 questions had a majority in favor of RANKGEN vs. top-k. The last question was a tie.
- On average, a participant preferred RANKGEN for 70% of questions. 58.33% of participants preferred RANKGEN for any given question.
- Distribution of reasons for choosing RANKGEN:

Reasons relating the prefix with the generation (70.55%)	
More topically relevant to the prefix	22.70%
Better continuity / flow / chronology	38.04%
Does not contradict prefix	1.84%
Stylistically closer to prefix	7.98%
Reasons related only to the generated text (29.45%)	
Better commonsense understanding	7.98%
Less repetitive	9.20%
More grammatical	6.13%
Less contradictions	0.00%
More coherent / other	6.13%

Conclusion & Future Work

We replicated [3] on the Wikipedia dataset and saw consistent results. We also investigated RANKGEN's generalizability to other domains by testing on four different datasets. Our key results and findings were as follows:

- RANKGEN achieved high MAUVE and human evaluation scores on government reports, indicating strong generalization to other long-form English content.
- Low MAUVE scores and poor generations for the mathematical proof, Python code instruction, and poetry datasets indicate RANKGEN does not generalize well to domains deviating from standard English syntax and/or those reliant on sequential content.

We suggest the following as future work to verify and further extend our findings:

- Training RANKGEN on math proofs, code outputs, or poetry to investigate whether using text from these specialized domains will improve the quality of the generated output.
- Conducting follow-up human evaluations with more participants and also inviting graduate-level students, lecturers, and domain experts for evaluation.

References

[1] H. Zhang, D. Duckworth, D. Ippolito, and A. Neelakantan. Trading off diversity and quality in natural language generation, 2021.
[2] S. Sun, K. Krishna, A. Mattarella-Micke, and M. Iyyer. Do long-range language models actually use long-range context?, 2021.
[3] K. Krishna, Y. Chang, J. Wieting, and M. Iyyer. Rankgen: Improving text generation with large ranking models, 2022.