ACL 2025 VIENNA

RoToR: Towards More Reliable Responses for Order-Invariant Inputs

Soyoung Yoon[1*] **Dongha Ahn**[1,2] **Youngwon Lee**[1] **Minkyu Jung**[2]
HyungJoo Jang[2] **Seung-won Hwang**[1†]
[1]Seoul National University [2]Channel Corporation
{soyoung.yoon, seungwonh}@snu.ac.kr

Overview

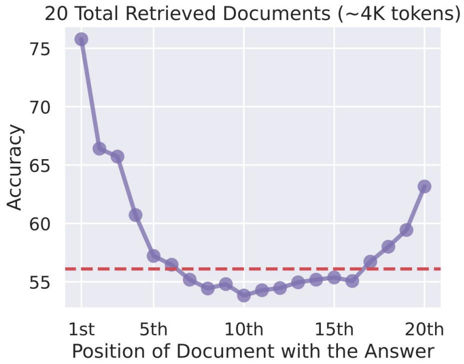
We introduce **RoToR:** which ensures robustness to the order of input contexts by modifying attention. This is done in a zero-shot manner, by (1) Global Sorting + Circular Position IDs and (2) Selective Routing for Mixed Inputs, which achieve SOTA robustness on 3 benchmarks and lower FLOPs v.s. Baselines (PINE)

1. Motivation: positional bias for listwise inputs
2. Limitations of prior works
3. Contributions of **RoToR** with **Selective Routing**
4. Experimental results

Motivation: Positional Bias for listwise inputs

- Lost-in-the-Middle (RAG)
- First-choice bias (75%) in LLM-as-a-judge
- MMLU rank shifts by 8 with shuffle
- Need neutral handling for sets, tables, multiple-choice questions

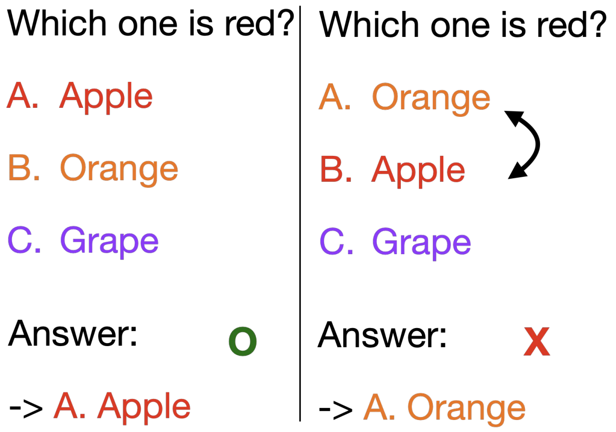
Which one is red?	Which one is red?
A. Apple	A. Orange
B. Orange	B. Apple
C. Grape	C. Grape
Answer: O	Answer: X
-> A. Apple	-> A. Orange



Liu et al., Lost in the Middle: How Language Models Use Long Contexts
Zheng et al., Judging llm-as-a-judge with mt-bench and chatbot arena.
Alzahrani et la., When Benchmarks are Targets: Revealing the Sensitivity of Large Language Model Leaderboards

Prior works to enforce invariance for listwise inputs

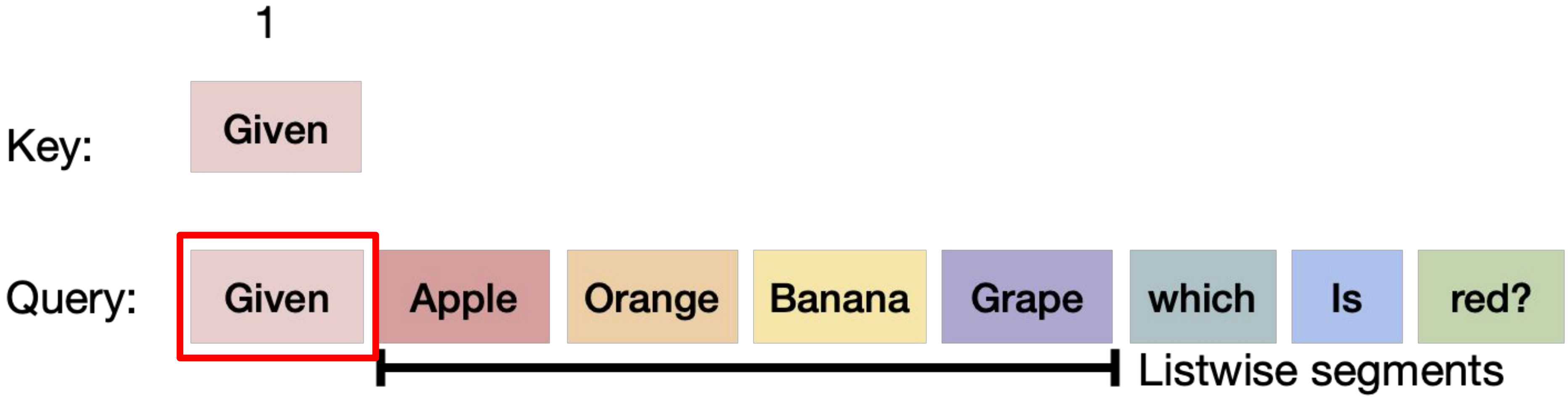
- Self-consistency (swap A/B in LLM-as-a-judge, ...) -> Needs N! forwards or approximations
- Attention alteration methods
  - PCW, Set-based Prompting
  - PINE



Wang et al., Eliminating Position Bias of Language Models: A Mechanistic Approach

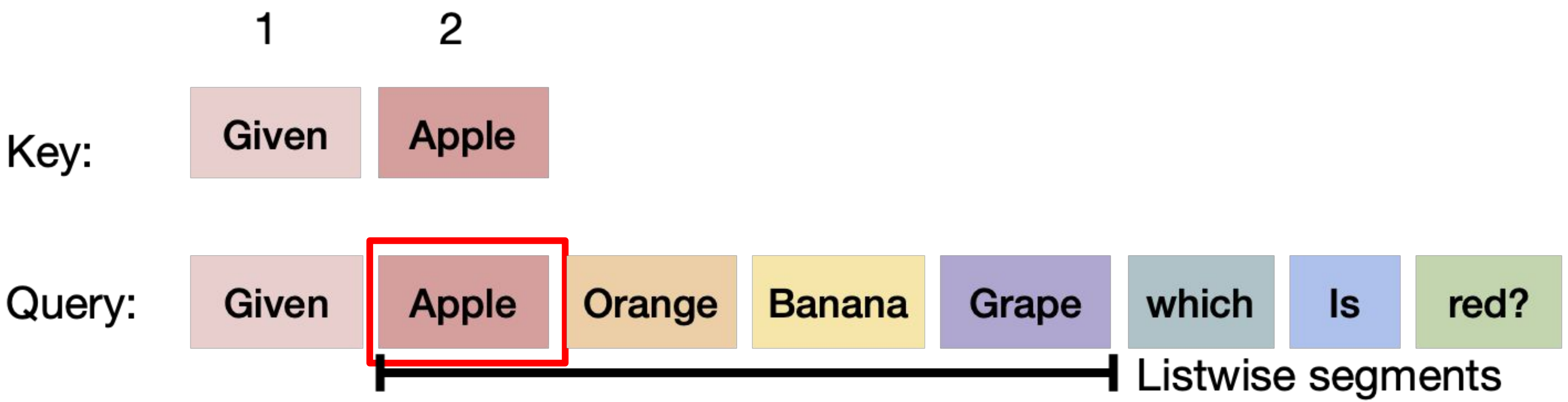
Original Model

Example: enforcing invariance via altering self-attention



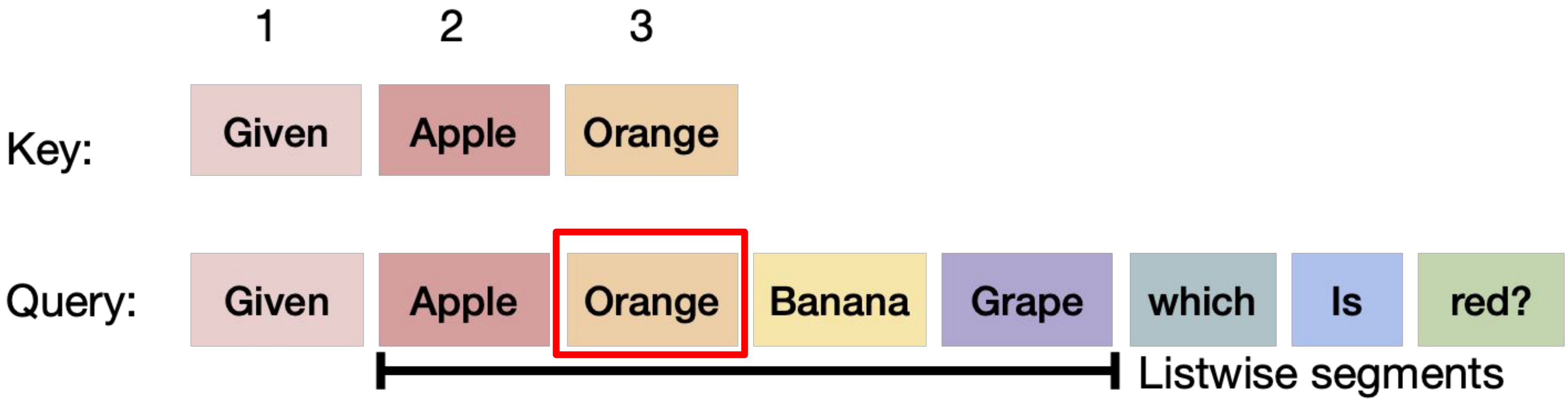
Original Model

Example: enforcing invariance via altering self-attention



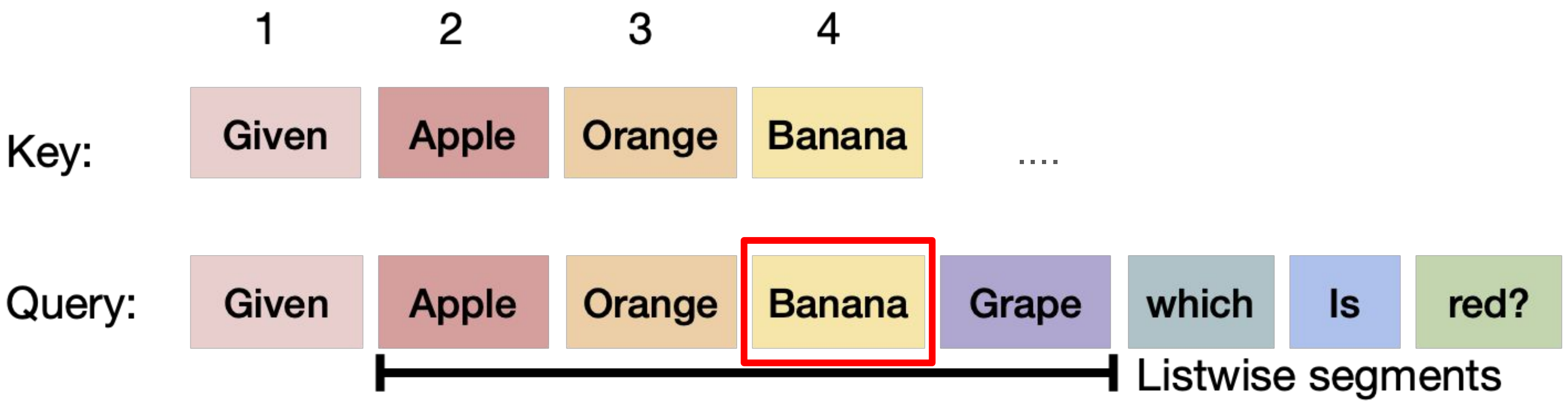
Original Model

Example: enforcing invariance via altering self-attention

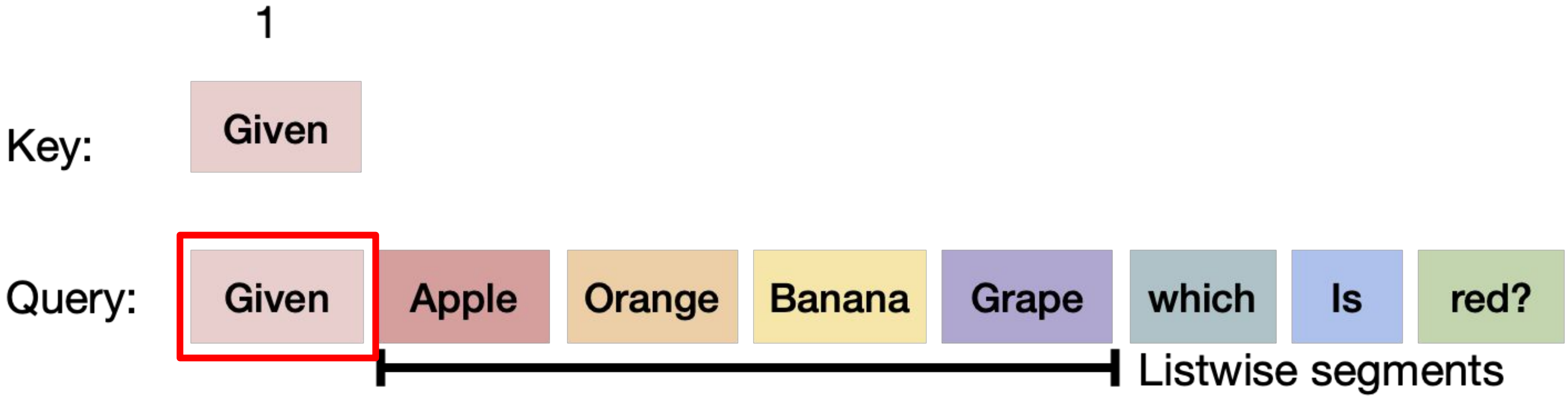


Original Model

Example: enforcing invariance via altering self-attention

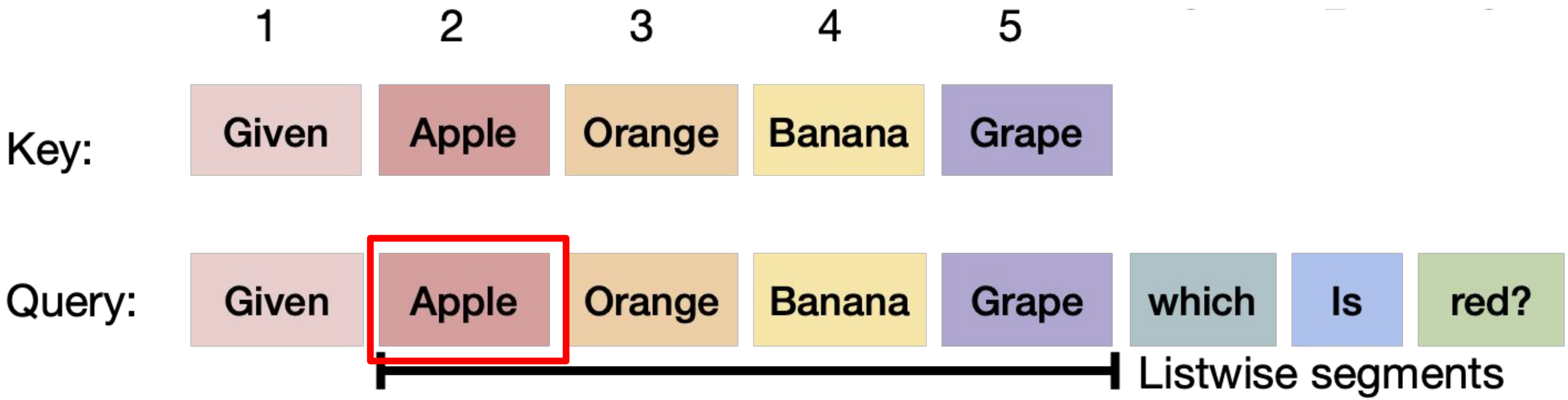


Example: enforcing invariance via altering self-attention



Example: enforcing invariance via altering self-attention

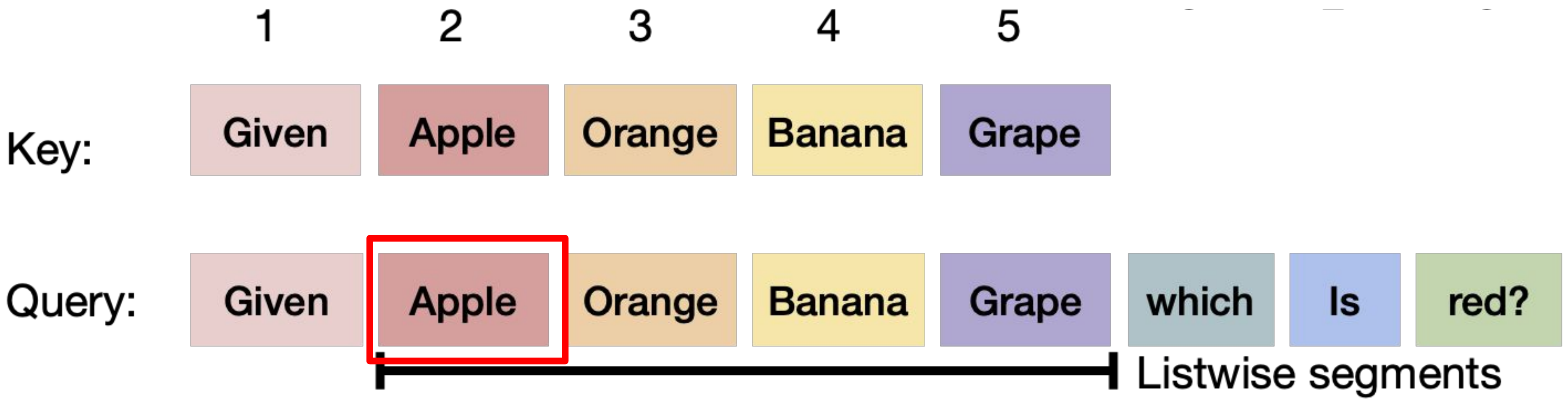
Causal X , open **ALL** attention for **segments (bidirectional)**



Invariant Model

Example: enforcing invariance via altering self-attention

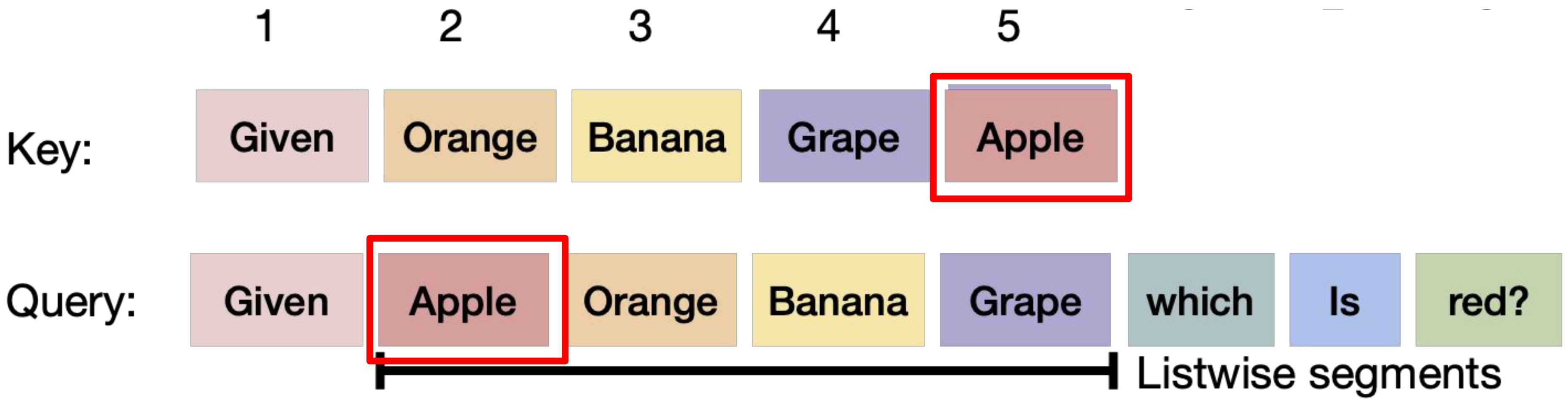
Causal X , open **ALL** attention for **segments (bidirectional)**



But, the position of query tokens should be placed last to follow the causal nature!

Example: enforcing invariance via altering self-attention

- Query token **Last**

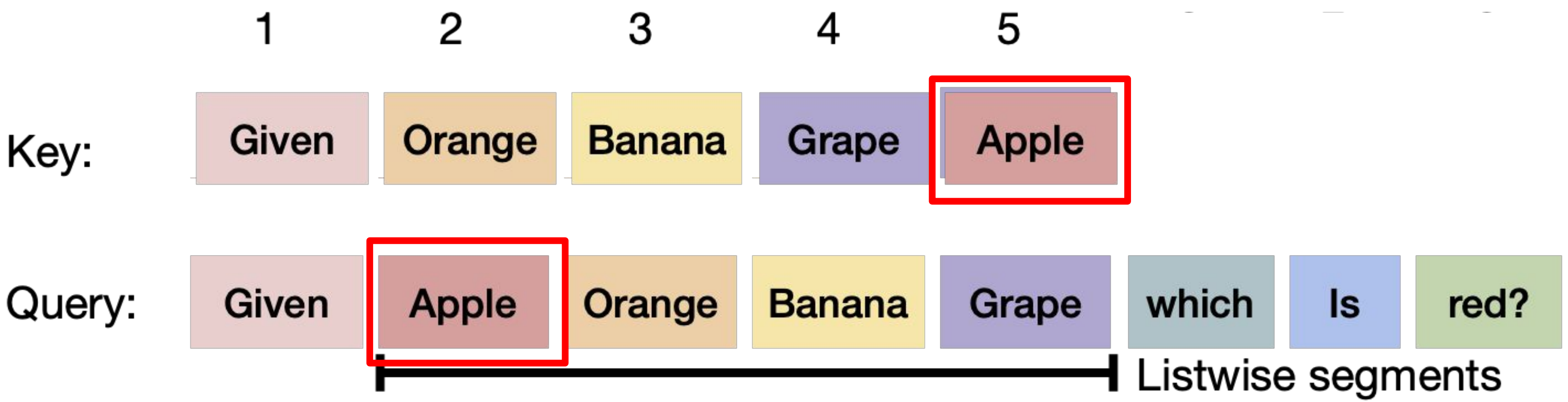


But, the position of query tokens should be placed last to follow the causal nature!

Invariant Model

Example: enforcing invariance via altering self-attention

- Query token **Last**

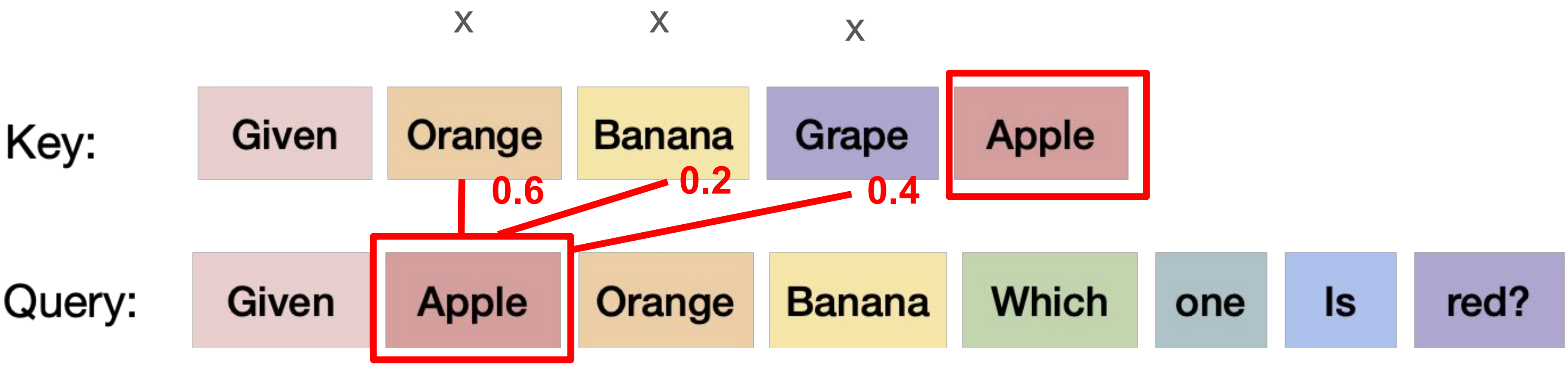


Also, the order of segments should not depend on the initial ordering (apple -> orange -> banana) of segments!

Example: enforcing invariance via altering self-attention

- Query token **Last**
- Order of segments

independent on initial ordering



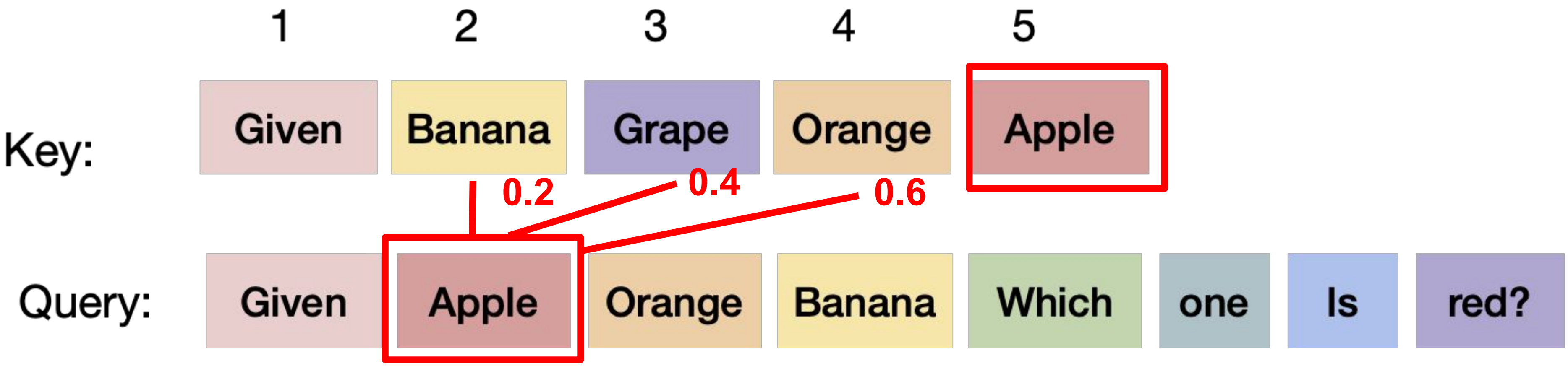
How? compute pairwise attention (relevance) among segments **without** positional ID

Invariant Model: PINE's way

Example: enforcing invariance via altering self-attention

Re-order segments so that **relevant** segment get **closer** to query segment!

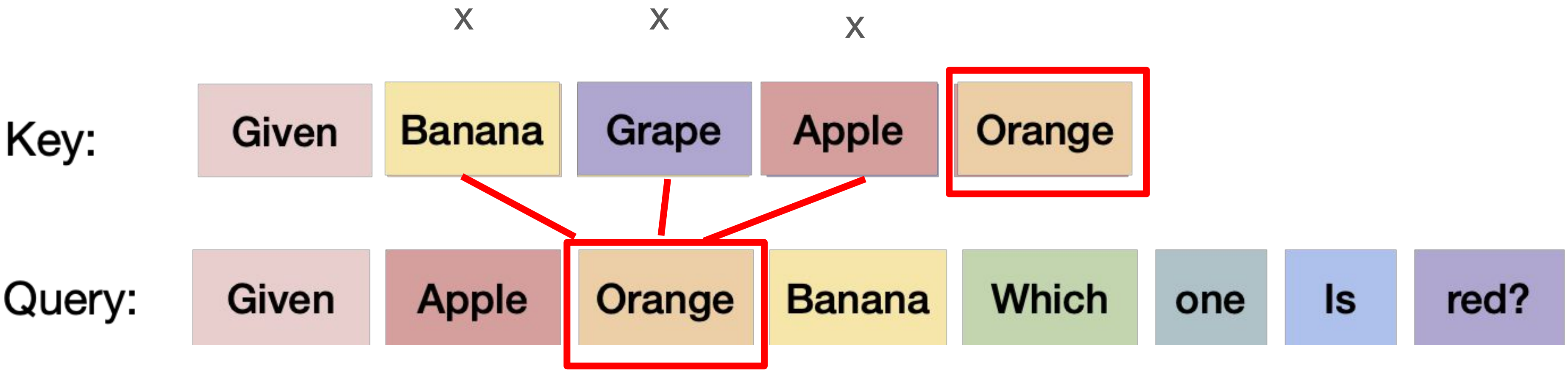
- Query token **Last**
- Order of segments **independent** on initial ordering



Example: enforcing invariance via altering self-attention

- Query token **Last**
- Order of segments

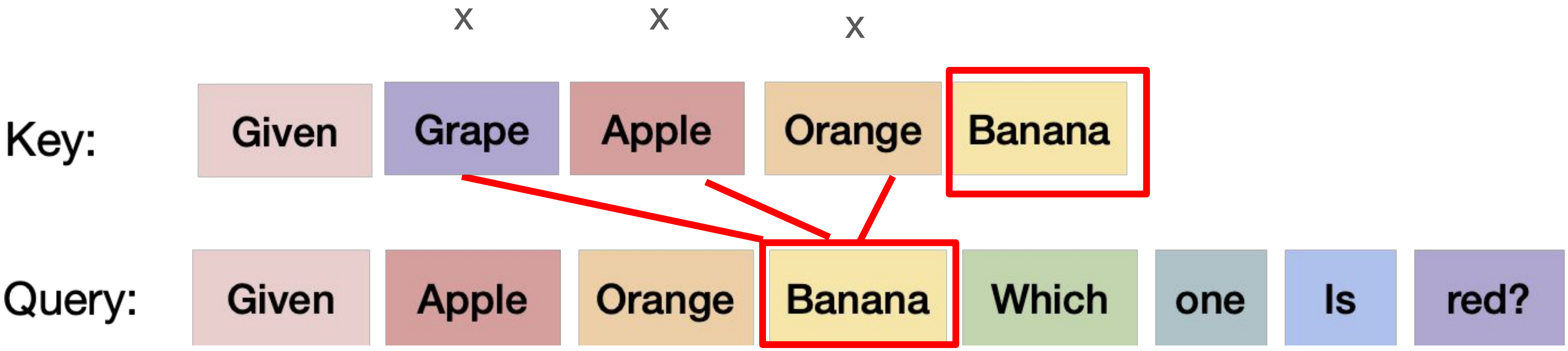
independent on initial ordering



Problem: need to re-calculate pairwise relevance labels for every query tokens

Example: enforcing invariance via altering self-attention

- Query token **Last**
- Order of segments **independent** on initial ordering



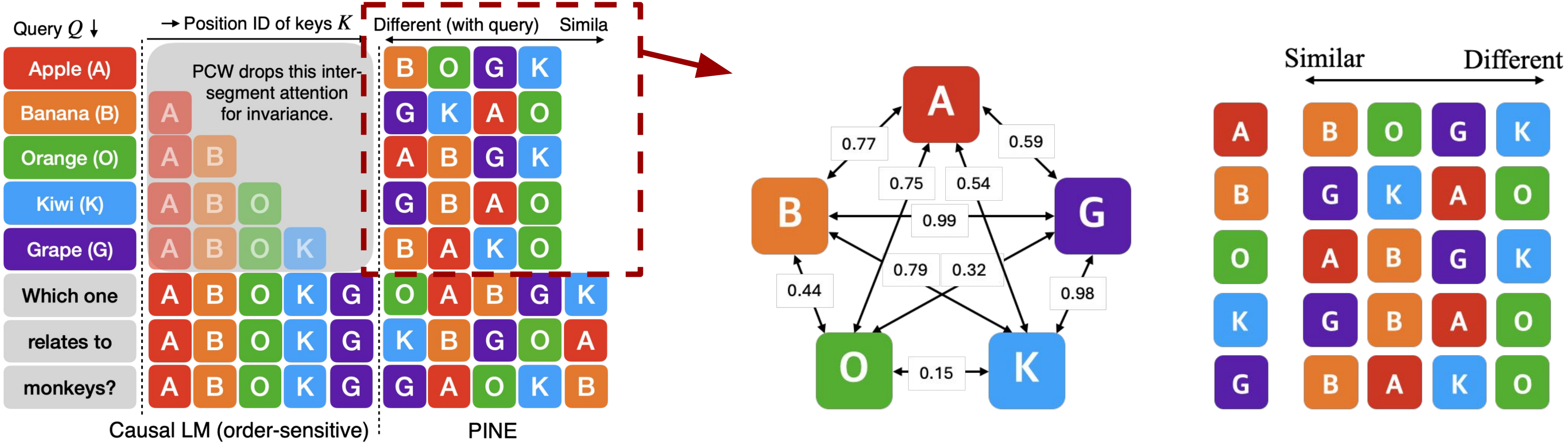
Problem: need to re-calculate pairwise relevance labels for every query tokens

Methodology: Order-invariant causal LMs

- PINE: Bidirectional processing with Q-K similarity

  - Has to obtain the same attention representation, regardless of initial ordering of segments
  - Places query IDs last, sorts other segments in a order-invariant way

Self-attention patterns (x = query, y = key) across order-invariant models

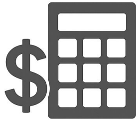


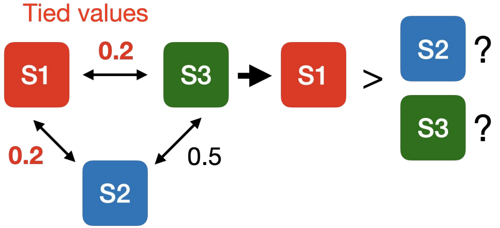
Limitations of prior works

-> Zero-shot order-invariant LMs have been proposed, but had limitations in two aspects:

1. Training and inference distribution mismatch

- PCW, Set-based prompting: No cross-segment context
- PINE: per-query sort -> O(O(n^2) + instability)
- Frequent ID changes cause **OOD behavior** -> drops its ability
- Computationally expensive (per-query KV attention compute)
- Numerical Instability (arising from attention assignment)



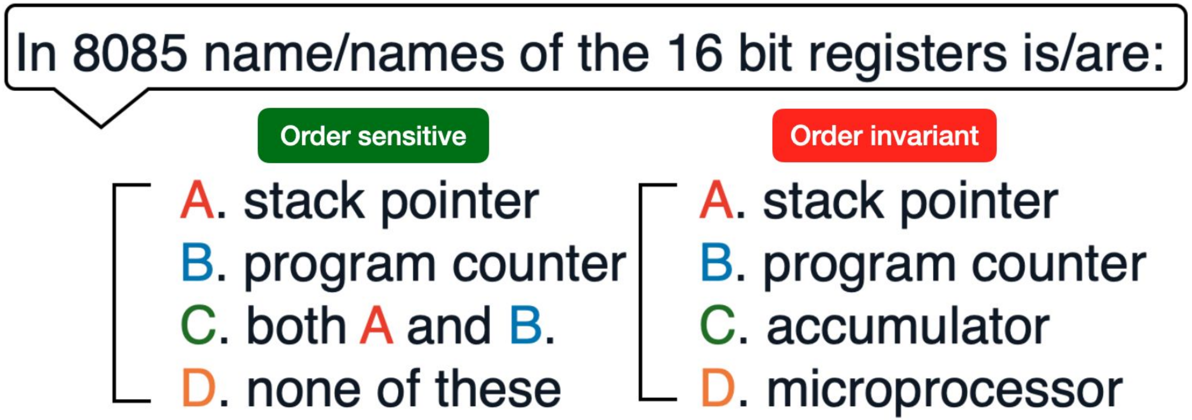


-> Zero-shot order-invariant LMs have been proposed, but had limitations in two aspects:

Limitations of prior works

2. Fail to extend to real-life scenarios (order-invariant + order-sensitive)

- Does not consider hybrid cases (e.g., MMLU)
- Cannot mix order-sensitive segments



Methodology: Order-invariant causal LMs

- Solution: RoToR

  - Keep the bidirectional structure, but alter the position assignment in a simple and stable way!
  - Define a single global ordering + circular arrangement

  Circular arrangement: Reuse global ordering by allocating them in a circular way!

    - shift global orders so that query token gets last, but relative ordering of others is maintained

RoToR, global ordering + circular arrangement

How? simple **Hierarchical Lexical Sorting*** of segments depending on their **tokenized IDs**

Apple [**16,** 1]

Orange [12, **8**]

Grape [12, 5, 2]

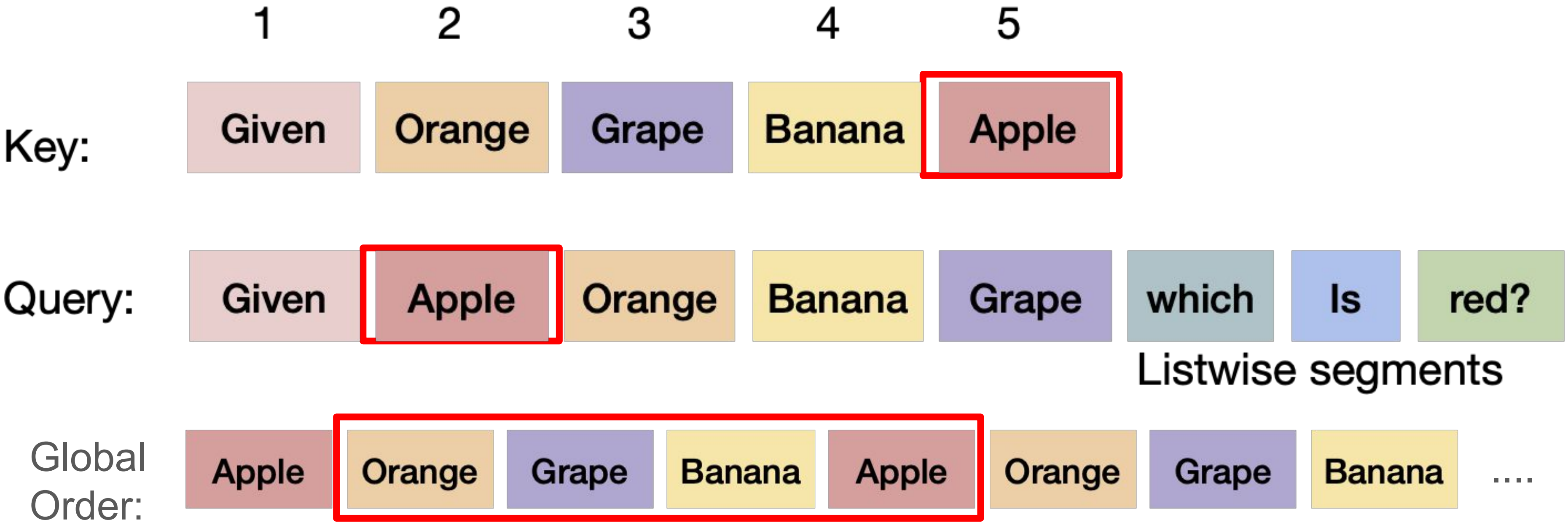
Banana [12, 3]

16 > 12
8 > 5 > 3

Global order: Apple **>** Orange **>** Grape **>** Banana

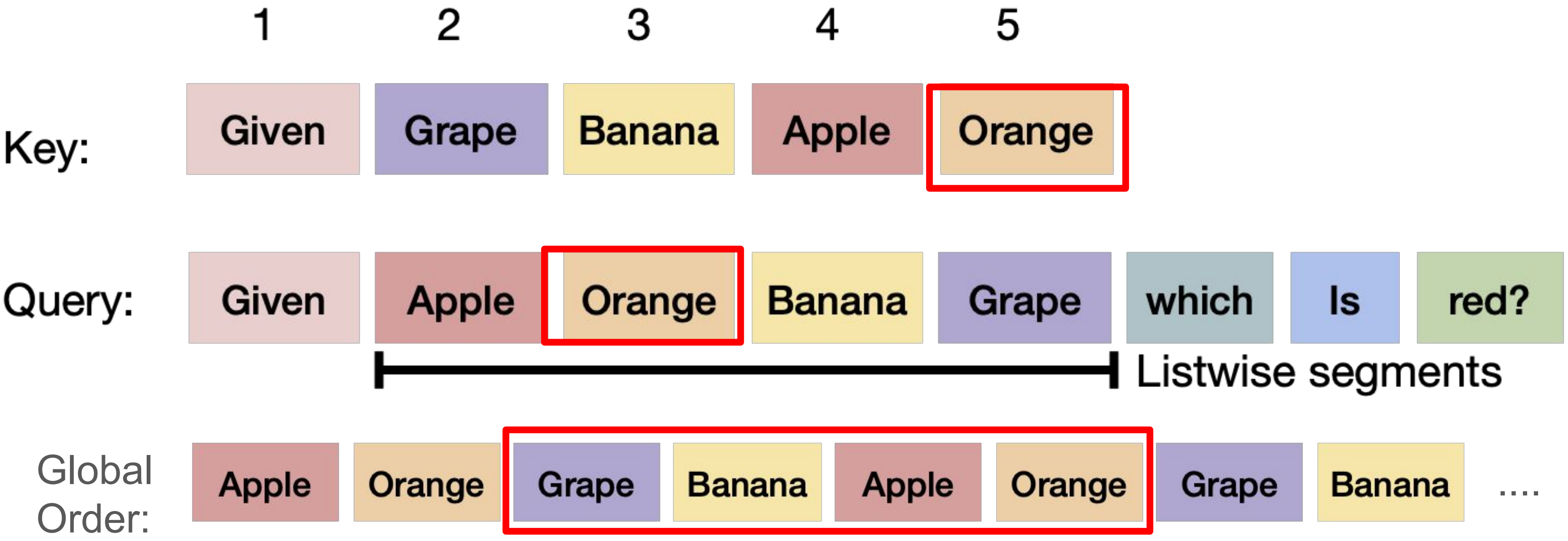
* We also experiment with other global sorting methods, such as reranking-based and freqency-based

Example: enforcing invariance via altering self-attention



Invariant Model: RoToR's way

Example: enforcing invariance via altering self-attention



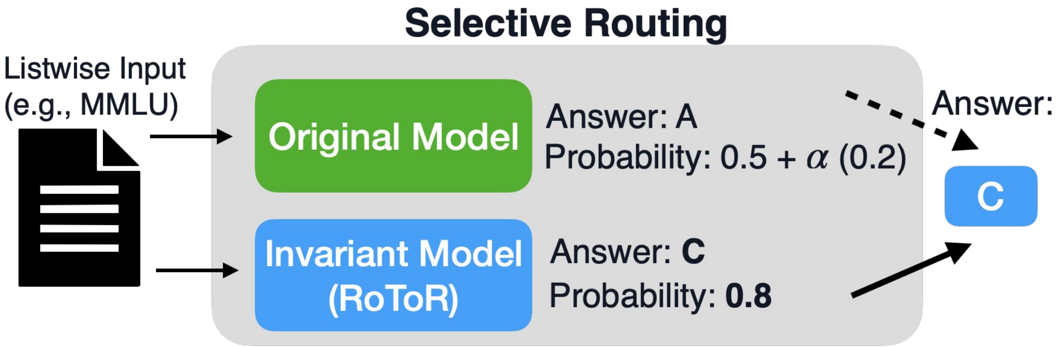
RoToR - Key Contributions

1. Training and inference distribution mismatch

- Stable, order-invariant solution (RoToR)
- Query-agnostic global ordering with minimal positional ID modifications

2. Fail to extend to hybrid cases

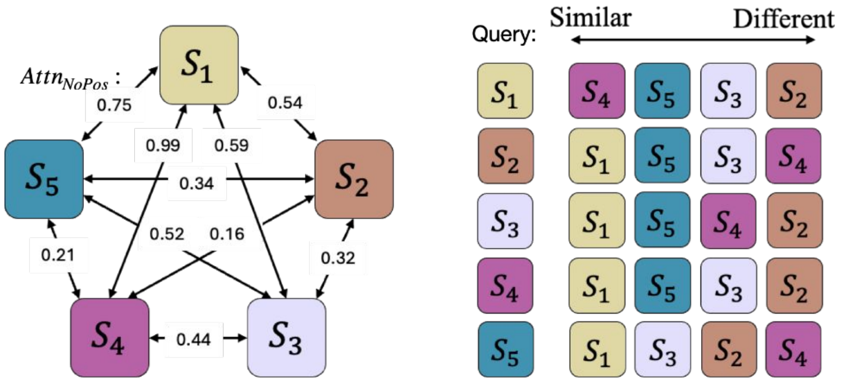
- Selective Routing, which switches between original / invariant LMs based on confidence



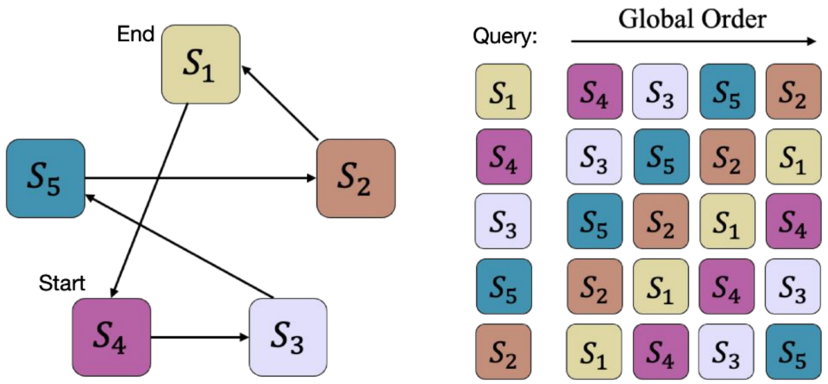
RoTоR v.s. PINE (Schematic)

PINE: query-dependent grid, RoToR: fixed order, rotate per query

-> Stable IDs, zero collisions, less computation



PINE

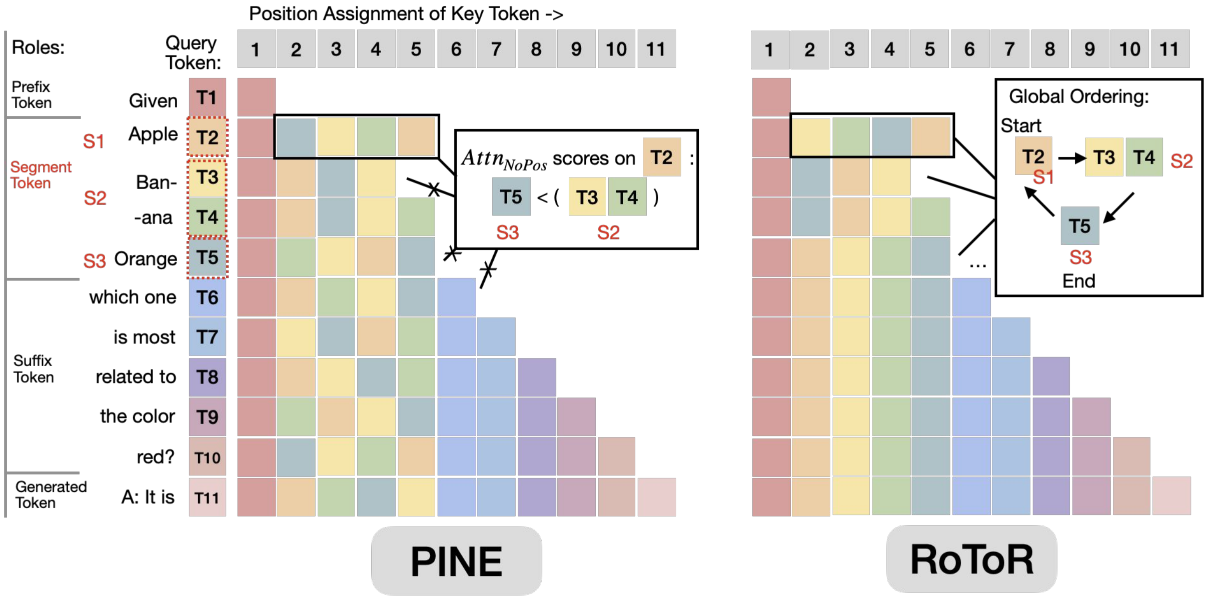


RoToR

RoToR v.s. PINE (Schematic)

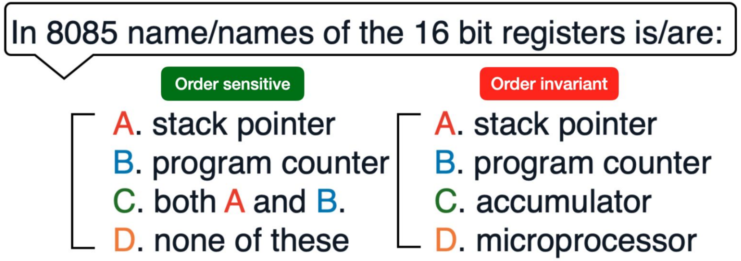
PINE: query-dependent grid, RoToR: fixed order, rotate per query

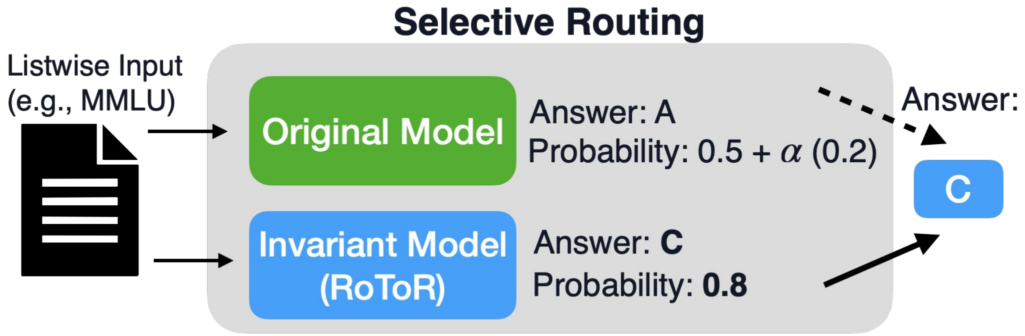
-> Stable IDs, zero collisions, less computation



Selective Routing: extend to hybrid cases (e.g., MMLU)

- Compute confidence of Original & RoToR outputs
- Choose higher p + α (α=0.2)





Experimental Setup

- Benchmarks:

- Lost-in-the-Middle (LitM)
- Knowledge Graph QA (KGQA): Mintaka
- MMLU: selective routing cases
- LongBench: long context scenarios (Appendix)

- Model backbones:

- Llama-3.1-8B/70B
- Qwen-1.5-4/7/72B-Chat

- **Metrics:** best_subspan_em (LitM), EM, F1, Acc. (KGQA), Acc. (MMLU)

- Methods: Original (order-sensitive), PCW, Set-based prompting, PINE, RoToR

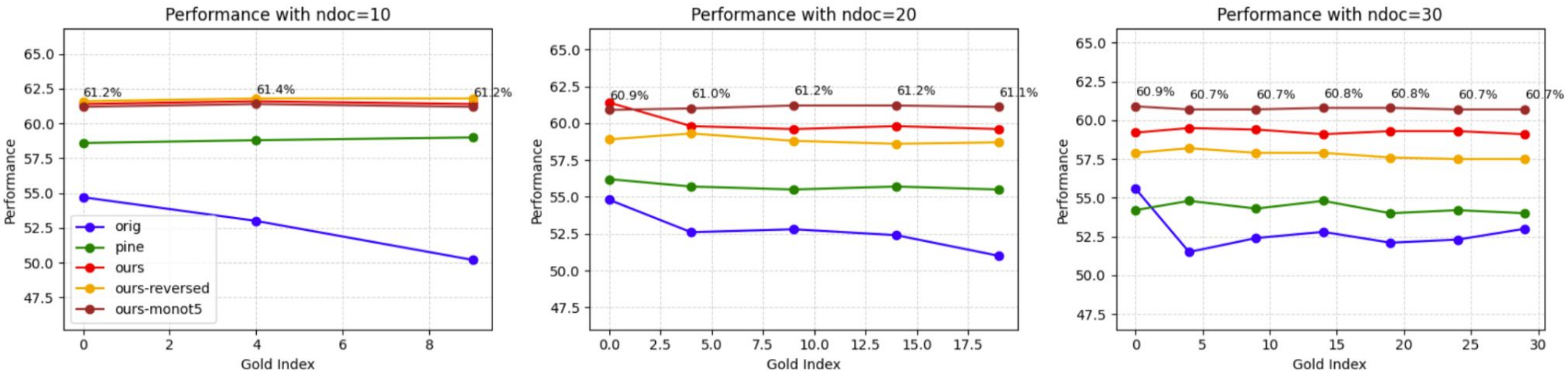
Efficiency Gains (over PINE)

- Less computation:
  - Overhead FLOPs ↓ 98 % (72B)
- Faster:
  - E2E Latency ↓ 23-43 % on LitM
- Reduces OOD:
  - Perplexity ↓; collision rate 0 %

Model	Benchmark	PINE	RoToR	Reduction
(a) Overhead FLOPs, relative to original model				
Llama-3.1-8B-Instruct	MMLU, $N = 4$	0.59×	**0.55**×	7.6%
	LitM, $N = 10$	7.07×	**4.81**×	31.9%
	LitM, $N = 30$	22.43×	**15.05**×	32.9%
Llama-3.1-70B-Instruct	KGQA, $N = 30$	1.27×	**0.94**×	26.0%
	KGQA, $N = 50$	1.82×	**1.29**×	29.0%
Qwen1.5-72B-Chat	KGQA, $N = 30$	0.45×	**0.01**×	98.0%
	KGQA, $N = 50$	0.58×	**0.03**×	94.8%
(b) End-to-end latency (s)				
Llama-3.1-70B-Instruct	LitM, $N = 10$	57,352	**44,219**	22.9%
	LitM, $N = 20$	87,091	**58,680**	32.6%
Llama-3.1-8B-Instruct	MMLU, $N = 4$	7,371	**6,608**	10.4%
	LitM, $N = 10$	18,551	**14,264**	23.1%
	LitM, $N = 30$	41,664	**23,569**	43.4%
(c) Perplexity & Collision rate, (on LitM)				
Llama-3.1-8B-Instruct	Perplexity ($N = 20$)	6.91	**6.65**	–
	Collision rate ($N = 30$)	42.3%	**0 (None)**	–

Table 4: **Unified efficiency comparison of RoToR vs. PINE,** reporting (a) Additional FLOPs, (b) Latency, and (c) Perplexity & Collision rate. Columns list each metric for PINE and RoToR, and the relative reduction. Yellow rows separate sub-sections.

Results: Lost-in-the Middle (LitM)



- Original Model fluctuates performance
- Ours (RoToR): maintains stable & higher performance than other order-invariant models

On Llama-3.1-8B-Inst.

Results: Lost-in-the Middle (LitM)

- Full results

Total ndoc (segments)	10			20					30						
Gold idx at:	0	4	9	0	4	9	14	19	0	4	9	14	19	24	29
Llama-3.1-8B-Instruct															
Original	54.7	53.0	50.2	54.8	52.6	52.8	52.4	51.0	55.6	51.5	52.4	52.8	52.1	52.3	53.0
PCW	12.4	11.9	12.2	3.7	4.0	4.0	4.0	3.9	2.3	1.8	2.0	2.0	2.1	2.0	2.0
Set-Based Prompting	42.5	42.5	42.5	26.3	26.3	26.3	26.3	26.3	14.1	14.1	14.1	14.1	14.1	14.1	14.1
PINE	58.6	58.8	59.0	56.2	55.7	55.5	55.7	55.5	54.2	54.8	54.3	53.7	54.8	54.2	54.0
RoToR-lexical	61.4	61.6	61.6	**61.4**	59.8	59.6	59.6	59.8	59.2	59.5	59.4	59.1	59.0	59.3	59.1
RoToR-reversed lexical	**61.6**	**61.8**	**61.8**	58.9	59.3	58.8	58.6	58.7	57.9	58.2	57.9	57.4	57.9	57.6	57.5
RoToR-MonoT5	61.2	61.4	61.2	60.9	**61.0**	**61.2**	**61.2**	**61.2**	**60.9**	**60.7**	**60.7**	**60.7**	**60.8**	**60.8**	**60.7**
RoToR-Freq.	61.0	61.1	61.1	60.4	60.3	58.6	60.2	60.0	59.3	60.4	59.7	59.5	59.5	59.6	59.2
Qwen1.5-4B-Chat															
Original	61.3	54.8	53.1	**59.5**	49.1	47.9	45.9	48.3	**56.8**	45.6	44.9	44.6	45.3	43.5	48.3
PINE	57.2	57.4	57.0	48.6	48.2	48.2	48.1	48.9	46.4	45.9	46.7	46.6	46.4	46.4	46.3
RoToR	58.5	58.4	58.1	49.9	49.7	49.6	49.8	49.9	44.6	44.8	44.7	44.7	44.9	44.8	44.7
RoToR-MonoT5	**58.9**	**58.5**	**58.7**	52.2	**52.1**	**52.1**	**52.2**	**52.6**	50.6	**50.7**	**50.5**	**50.6**	**50.5**	**50.6**	**50.4**
RoToR-Freq.	56.7	56.9	56.9	51.9	51.5	51.8	51.6	52.4	46.8	46.7	46.7	46.4	47.0	46.8	46.6
Qwen1.5-7B-Chat															
Original	**72.5**	63.3	62.9	**72.5**	58.5	56.1	56.0	58.2	**73.1**	58.6	55.8	53.3	53.2	52.5	57.5
PINE	65.4	65.5	66.3	59.1	59.4	59.1	58.6	59.2	58.0	55.3	55.7	56.3	55.1	55.8	56.1
RoToR	68.6	68.7	68.6	62.6	62.9	62.7	63.0	62.7	57.0	57.3	59.7	57.4	57.3	**62.8**	57.0
RoToR-MonoT5	68.8	**69.4**	**69.0**	65.2	**65.5**	**65.0**	**64.9**	**65.0**	62.6	**62.8**	**62.9**	**62.7**	**62.9**	**62.8**	**62.5**
RoToR-Freq.	68.2	68.4	68.4	62.6	62.9	62.8	62.7	62.3	59.5	59.8	59.7	59.6	59.7	59.7	59.7

Results: KGQA

- Top-30 and Top-50 knowledge triples per query
- Test before / after shuffling segments to see robustness
- RoToR obtains lower stdev (better stability) + higher performance than PINE
- Trend persists for > 70B model variants

	Llama-3.1-8B-Instruct						Qwen1.5-4B-Chat						Qwen1.5-7B-Chat					
	N = 30			N = 50			N = 30			N = 50			N = 30			N = 50		
Method	Acc.	EM	F1	Acc.	EM	F1	Acc.	EM	F1	Acc.	EM	F1	Acc.	EM	F1	Acc.	EM	F1
Initial, no shuffling of segments																		
Original	50.2	44.0	51.9	50.0	44.0	51.7	30.7	27.9	34.9	31.6	28.6	35.8	31.5	27.8	35.4	31.7	28.0	35.7
PINE	51.5	45.0	52.6	51.6	45.1	52.6	31.6	28.7	35.6	31.6	28.8	35.3	32.3	28.8	36.4	32.0	28.5	35.9
RoToR	**53.1**	**46.5**	**54.1**	52.9	46.0	53.6	32.0	29.0	35.7	**32.7**	**29.6**	**36.2**	**34.3**	**29.8**	**37.7**	**34.3**	**30.1**	**38.0**
RoToR-MonoT5	51.6	45.0	52.5	52.4	45.4	52.8	**32.3**	29.1	**36.2**	32.3	29.3	35.9	32.9	28.4	36.3	32.9	28.9	36.6
RoToR-Freq.	52.6	46.1	53.7	**53.1**	**46.4**	**53.7**	**32.3**	**29.2**	36.0	32.3	29.2	35.9	33.7	29.5	37.2	33.5	29.5	37.2
After shuffling segments, averaged over 3 seeds																		
Original	49.5	43.3	51.0	49.7	43.5	51.0	30.1	27.5	34.7	30.3	27.6	35.0	31.4	27.3	35.0	31.6	27.9	35.5
↪ stdev. (±)	0.07	0.14	0.17	0.34	0.28	0.46	0.41	0.34	0.43	0.26	0.24	0.35	0.26	0.28	0.29	0.40	0.56	0.42
PINE	51.8	45.2	52.8	51.8	45.3	52.7	31.5	28.7	35.6	31.5	28.7	35.3	32.3	28.8	35.7	31.7	28.2	35.7
↪ stdev. (±)	0.05	0.07	0.16	0.15	0.16	0.19	0.20	0.18	0.13	0.17	0.20	0.21	0.17	0.20	0.13	0.18	0.16	0.14
RoToR	**52.8**	**46.2**	**53.8**	52.7	45.9	53.5	31.8	28.8	35.5	**32.5**	**29.6**	**36.1**	**34.2**	**29.9**	**37.7**	**34.2**	**30.1**	**38.0**
↪ stdev. (±)	0.05	0.05	0.02	0.05	0.09	0.04	0.05	0.02	0.09	0.11	0.06	0.09	0.09	0.07	0.06	0.06	0.05	0.04
RoToR-MonoT5	51.6	45.0	52.6	52.2	45.2	52.8	**32.4**	29.2	**36.3**	32.3	29.4	35.9	33.0	28.8	36.5	32.8	28.8	36.5
↪ stdev. (±)	0.12	0.06	0.10	0.16	0.18	0.18	0.04	0.02	0.13	0.16	0.13	0.07	0.12	0.09	0.07	0.16	0.09	0.07
RoToR-Freq.	52.5	45.9	53.5	**53.1**	**46.4**	**53.7**	32.3	**29.3**	36.0	32.4	29.3	**36.1**	33.8	29.6	37.4	33.7	29.6	37.4
↪ stdev. (±)	0.10	0.15	0.11	0.02	0.07	0.03	0.13	0.16	0.09	0.09	0.04	0.06	0.04	0.00	0.09	0.04	0.16	0.22

Results: MMLU (selective routing)

	Llama-3.1-8B-Instruct			Qwen1.5-4B-Chat			Qwen1.5-7B-Chat		
Method	**Init.**	**Rev.**	**Avg.**	**Init.**	**Rev.**	**Avg.**	**Init.**	**Rev.**	**Avg.**
Orig.	68.3	64.8	65.5 ± 1.0	53.6	51.9	52.6 ± 0.6	**60.1**	56.6	58.6 ± 0.9
PCW	57.0	55.1	56.1 ± 1.1		–			–	
Set-Based Prompting	31.1	33.0	31.6 ± 0.8		–			–	
PINE	64.8	63.3	63.6 ± 0.7	50.5	49.3	49.4 ± 0.5	57.0	54.4	55.8 ± 0.9
RoToR	63.2	62.6	62.8 ± 0.7	49.6	47.7	48.3 ± 0.7	56.5	55.8	56.2 ± 0.6
↪ + S.R.	**68.5**	65.1	65.7 ± 0.9	53.7	51.8	$\mathbf{52.6 \pm 0.6}$	**60.1**	**57.4**	$\mathbf{58.8 \pm 0.7}$
RoToR - MonoT5	64.2	62.9	63.5 ± 0.5	49.7	47.6	48.7 ± 0.7	56.2	54.4	55.5 ± 0.7
↪ + S.R.	68.4	65.2	65.8 ± 0.9	**53.8**	51.9	$\mathbf{52.6 \pm 0.6}$	**60.1**	57.3	58.7 ± 0.8
RoToR - Freq.	64.3	63.6	63.8 ± 0.6	49.9	47.6	48.7 ± 0.5	56.4	54.7	55.7 ± 0.7
↪ + S.R.	**68.5**	**65.3**	$\mathbf{65.8 \pm 0.8}$	53.7	**52.3**	$\mathbf{52.6 \pm 0.6}$	60.0	57.3	58.6 ± 0.8
RoToR + S.R. (Oracle)	75.0	71.9	72.7 ± 1.0	61.8	60.1	61.1 ± 1.0	68.1	66.2	67.2 ± 0.7

- Order-invariant models fail (than the original model) with single use (expected)
- Selective Routing shows improved performance and stability across input re-orderings
- High S.R. (Oracle) value indicates high potential for further accuracy gains by optimizing choices on routing methods

Summary

We propose RoToR: a simple, effective order-invariant LM that..

- Can be applied to **any** zero-shot decoder-only model (with RoPE)
- Global sort + circular IDs mitigate positional bias
- Selective Routing enables practical use
- Paper: https://arxiv.org/pdf/2502.08662
- Code: github.com/soyoung97/RoToR

Code

Paper

Thank you!

Appendix

Appendix: computational overhead

PINE requires two additional operations: (1) computing attention scores without rotary position embeddings ($\mathcal{O}(n^2 d)$) and (2) sorting k segments for each query token ($\mathcal{O}(nk \log k)$), totaling $\boldsymbol{\mathcal{O}(n^2 d + nk \log k)}$ (Wang et al., 2024)[4].

our lexicographical sorting requires only a single global sort of k segments ($\mathcal{O}(k \log k)$), each with length $\mathcal{O}(n)$, achieving $\boldsymbol{\mathcal{O}(nk \log k)}$ and eliminating the expensive $\mathcal{O}(n^2 d)$ term entirely. This can be further optimized to $\boldsymbol{\mathcal{O}(nk)}$ using radix sort.[5]

Appendix: Example Input/Output

lost in the middle

Prefix:
<|begin_of_text|><|start_header_id|>system<|end_header_id|>

You are a helpful, respectful and honest assistant. Always answer as helpfully as possible, while being safe. Please ensure that your responses are socially unbiased and positive in nature. If a question does not make any sense, or is not factually coherent, explain why instead of answering something not correct. If you don't know the answer to a question, please don't share false information.<|eot_id|><|start_header_id|>user<|end_header_id|>

Write a high-quality answer for the given question using only the provided search results (some of which might be irrelevant).

Parallel texts:
Document [1](Title: List of Nobel laureates in Physics) The first ...
...
Document [10](Title: Nobel Prize in Chemistry) on December 10, the ...

Suffix:
Question: who got the first nobel prize in physics<|eot_id|><|start_header_id|>assistant
<|end_header_id|>

Figure 7: Example input for the lost in the middle dataset.

Appendix: Example Input/Output

Mintaka

Prefix:
<|begin_of_text|><|start_header_id|>system<|end_header_id|>

Below are the facts in the form of the triple meaningful to answer the question. Answer the given question in a JSON format, such as "Answer": "xxx". Only output the JSON, do NOT say any word or explain.

<|eot_id|><|start_header_id|>user<|end_header_id|>

Parallel texts:
(Super Bowl XLII, winner, New York Giants)
(Super Bowl XLII, participating team, New York Giants)
(Super Bowl XLII, point in time, time: +2008-02-03)
(Super Bowl XLII, followed by, Super Bowl XLIII)
(Super Bowl XLII, location, State Farm Stadium)
...
(Super Bowl XLII, sport, American football)
(Super Bowl XLII, instance of, Super Bowl)

Suffix:
Question: which team did the super bowl xlii mvp play for?, Answer: <|eot_id|><|start_header_id|> assistant <|end_header_id|>

Gold Answer(s):
('NYG', 'Giants', 'NY Giants', 'New York Giants')

Example generated output:
{"Answer": "New York Giants"} (Parsed to: New York Giants)

Figure 10: Example input for the Mintaka dataset.

Appendix: Example Input/Output

MMLU

Prefix:
The following are multiple choice questions (with answers) about moral disputes.

Norcross agrees that if a being is incapable of moral reasoning, at even the most basic level, then it cannot be

Parallel texts:
A. a being of value.
B. an object of moral sympathy.
C. a moral agent.
D. a moral patient.

Suffix:
Answer:

Figure 11: Example input for the MMLU benchmark.

Appendix: LongBench-2WikiMultiHopQA

		Llama 3.1-8B-Instruct				Qwen 1.5-7B-Chat			
Order	Method	0–4k	4–8k	8k+	Total	0–4k	4–8k	8k+	Total
	Count	25	131	144	300	23	121	156	300
Initial (e.g., 1,2,3,4,5)	Orig.	48.3	56.8	34.0	45.1	65.6	47.9	26.7	38.2
	PINE	51.0	47.6	–	–	70.2	45.1	–	–
	RoToR	**59.0**	52.7	**41.8**	**48.0**	**75.7**	**47.8**	**31.0**	**41.2**
Reversed (e.g., 5,4,3,2,1)	Orig.	57.0	51.5	39.0	46.0	53.4	43.3	**34.2**	39.3
	PINE	43.0	49.8	–	–	64.1	**48.9**	–	–
	RoToR	**59.0**	**52.0**	**41.0**	**47.3**	**72.8**	47.6	30.8	**40.8**
Center flip (e.g., 3,2,1,5,4)	Orig.	47.0	47.7	35.6	41.8	61.0	40.6	32.7	38.1
	PINE	46.3	49.2	–	–	70.2	43.5	–	–
	RoToR	**59.0**	**52.5**	**41.5**	**47.8**	**77.1**	**47.3**	**30.9**	**41.0**

Table 9: F1 scores (%) on LongBench–2WikiMultihopQA with ∼10k-token contexts. "Count" is the number of examples per length bucket; "–" denotes out-of-memory.

Appendix: Selective Routing ratio

	Llama-3.1-8B-Instr.			**Qwen1.5-4B-Chat**			**Qwen1.5-7B-Chat**		
Sorting	Init.	Rev.	Avg.	Init.	Rev.	Avg.	Init.	Rev.	Avg.
Lexical	7.0	8.5	7.3 ± 0.8	5.9	6.2	6.2 ± 0.4	10.3	10.6	9.9 ± 0.6
MonoT5	6.9	7.6	6.7 ± 1.5	8.0	12.5	9.8 ± 2.1	10.7	10.9	10.7 ± 0.7
Freq.	6.4	6.7	6.9 ± 0.5	8.5	10.9	9.4 ± 1.6	10.7	11.1	11.1 ± 0.8

Table 11: Selection ratio (%) of the RoToR variant under SR. Higher values indicate more frequent routing to RoToR.