基于向量检索增强的LLM

博士后 纪焘

A5029

taoji@fdu.edu.cn

基于向量检索增强的LLM

博士后 纪焘

A5029

taoji@fdu.edu.cn

LLM θ

(通用知识，常识...)

基于向量检索增强的LLM

博士后 纪焘

A5029
taoji@fdu.edu.cn

LLM θ

(通用知识，常识...)

外部知识库
(长尾，
长距离，
实时更新，
隐私保护...)

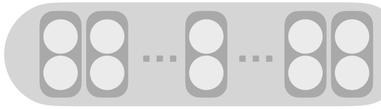
基于向量检索增强的LLM

博士后 纪焘

A5029
taoji@fdu.edu.cn

LLM θ

(通用知识，常识...)



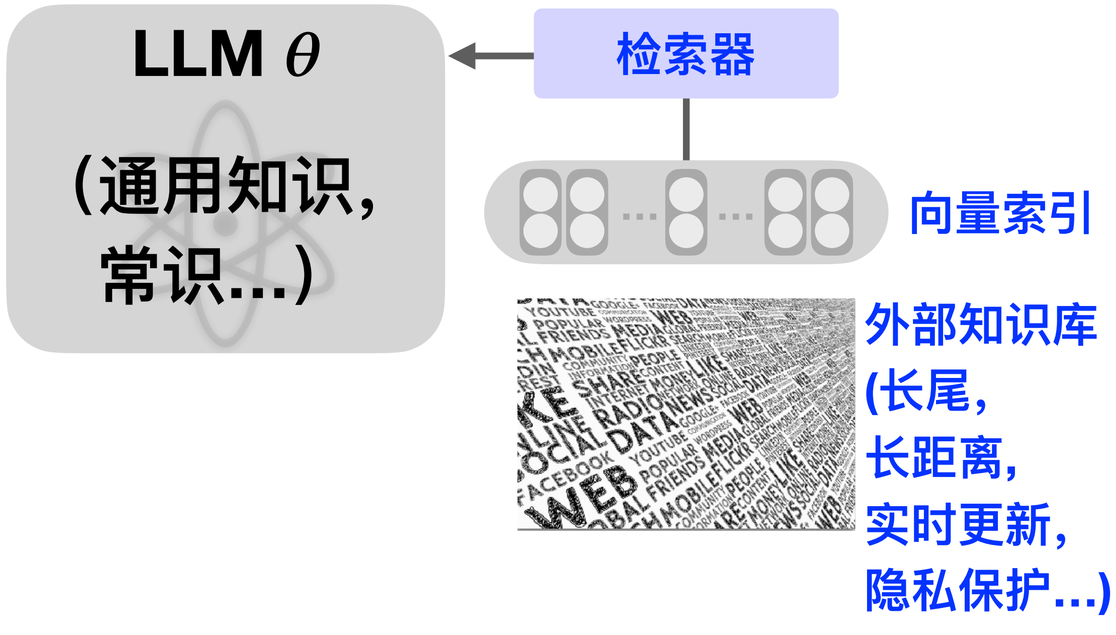
向量索引

外部知识库
(长尾，
长距离，
实时更新，
隐私保护...)

基于向量检索增强的LLM

博士后 纪焘

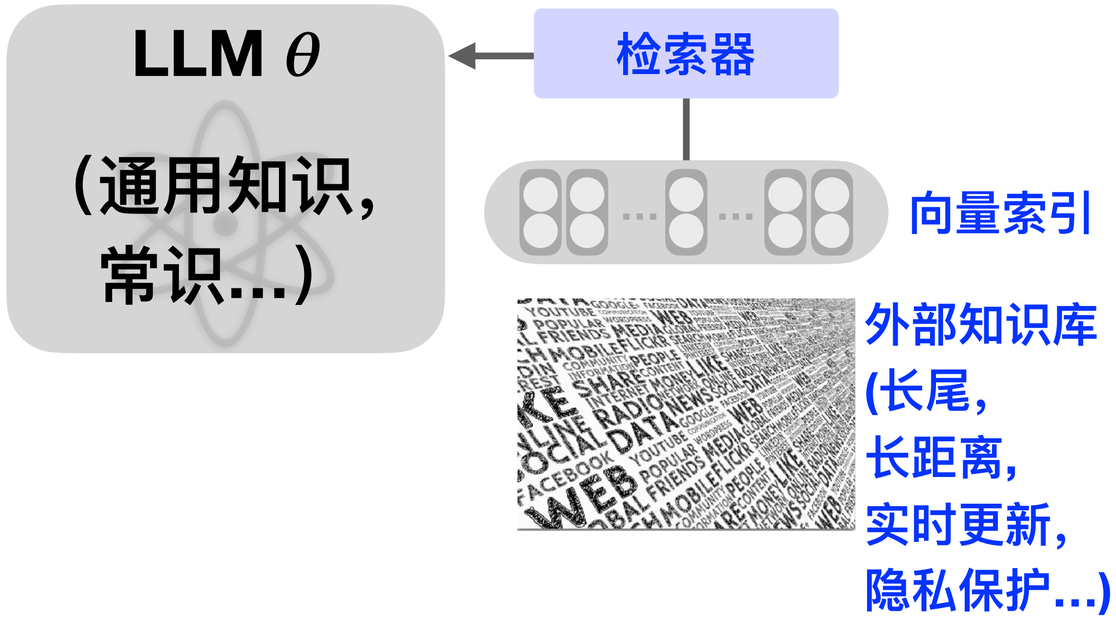
A5029
taoji@fdu.edu.cn



基于向量检索增强的LLM

博士后 纪焘

A5029
taoji@fdu.edu.cn



检索+LLM框架

1. Input
2. hidden
3. output

ACL 2023 Tutorial:

Retrieval-based Language Models and Applications

Akari Asai, Sewon Min, Zexuan Zhong, Danqi Chen

https://acl2023-retrieval-lm.github.io/

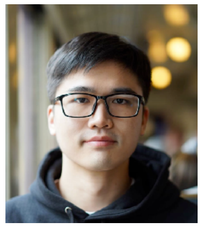
July 9, 2023

Akari Asai

PhD student
@UW

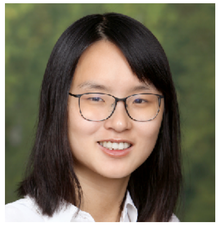
Sewon Min

PhD student
@UW



Zexuan Zhong

PhD student
@Princeton



Danqi Chen

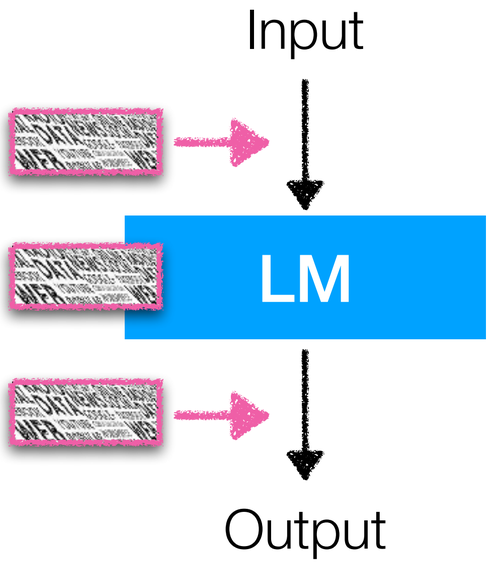
Faculty
@Princeton

What to retrieve?

Query

Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?



When to retrieve?

w/ retrieval

The capital city of Ontario is Toronto.

w/ retrieval w/ r w/r w/r w/ r w/r w/r

The capital city of Ontario is Toronto.

w/ retrieval w/r w/r

The capital city of Ontario is Toronto.

Input Layer: Prompt Learning

arXiv-2023-05

LaMP: When Large Language Models Meet Personalization

Alireza Salemi[1], Sheshera Mysore[1], Michael Bendersky[2], Hamed Zamani[1]
[1]University of Massachusetts Amherst
[2]Google Research

Input Layer: Prompt Learning

arXiv-2023-05

LaMP: When Large Language Models Meet Personalization

Alireza Salemi[1], Sheshera Mysore[1], Michael Bendersky[2], Hamed Zamani[1]
[1]University of Massachusetts Amherst
[2]Google Research

What to retrieve?

- **Chunks** ✓
- Tokens
- Others

How to use retrieval?

- **Input layer** ✓
- Intermediate layers
- Output layer

When to retrieve?

- **Once** ✓
- Every *n* tokens (*n*>1)
- Every token

LaMP Benchmark

- **Personalized Text Classification**

  (1) Personalized Citation Identification
  (2) Personalized News Categorization
  (3) Personalized Product Rating

- **Personalized Text Generation**

  (4) Personalized News Headline Generation
  (5) Personalized Scholarly Title Generation
  (6) Personalized Email Subject Generation
  (7) Personalized Tweet Paraphrasing

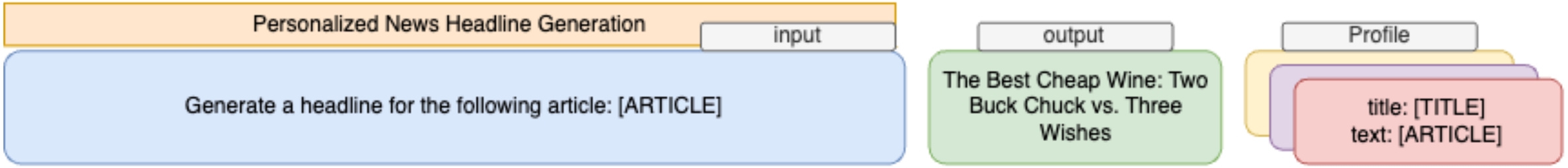
LaMP Benchmark

- **Personalized Text Classification**

  (1) Personalized Citation Identification
  (2) Personalized News Categorization
  (3) Personalized Product Rating

- **Personalized Text Generation**

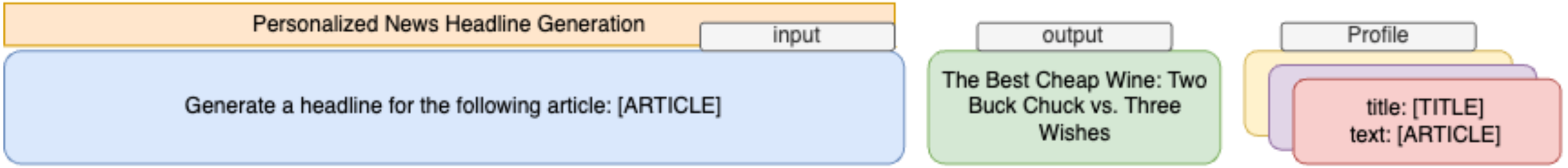
  (4) Personalized News Headline Generation
  (5) Personalized Scholarly Title Generation
  (6) Personalized Email Subject Generation
  (7) Personalized Tweet Paraphrasing



LaMP Benchmark

- **Personalized Text Classification**
  - (1) Personalized Citation Identification
  - (2) Personalized News Categorization
  - (3) Personalized Product Rating
- **Personalized Text Generation**
  - (4) Personalized News Headline Generation
  - (5) Personalized Scholarly Title Generation
  - (6) Personalized Email Subject Generation
  - (7) Personalized Tweet Paraphrasing

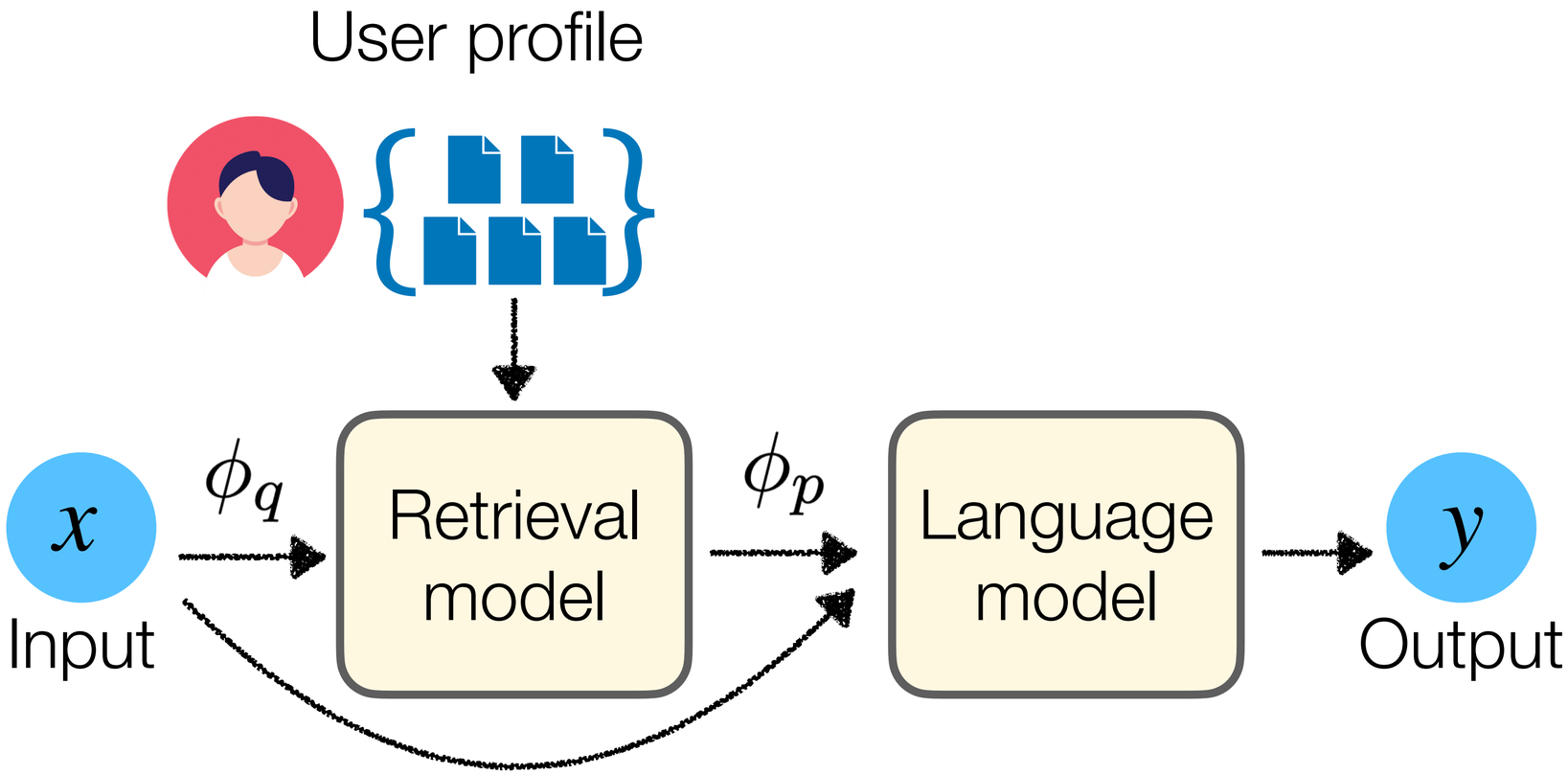
数据按时间划分
训练集←🕐→验证/测试集



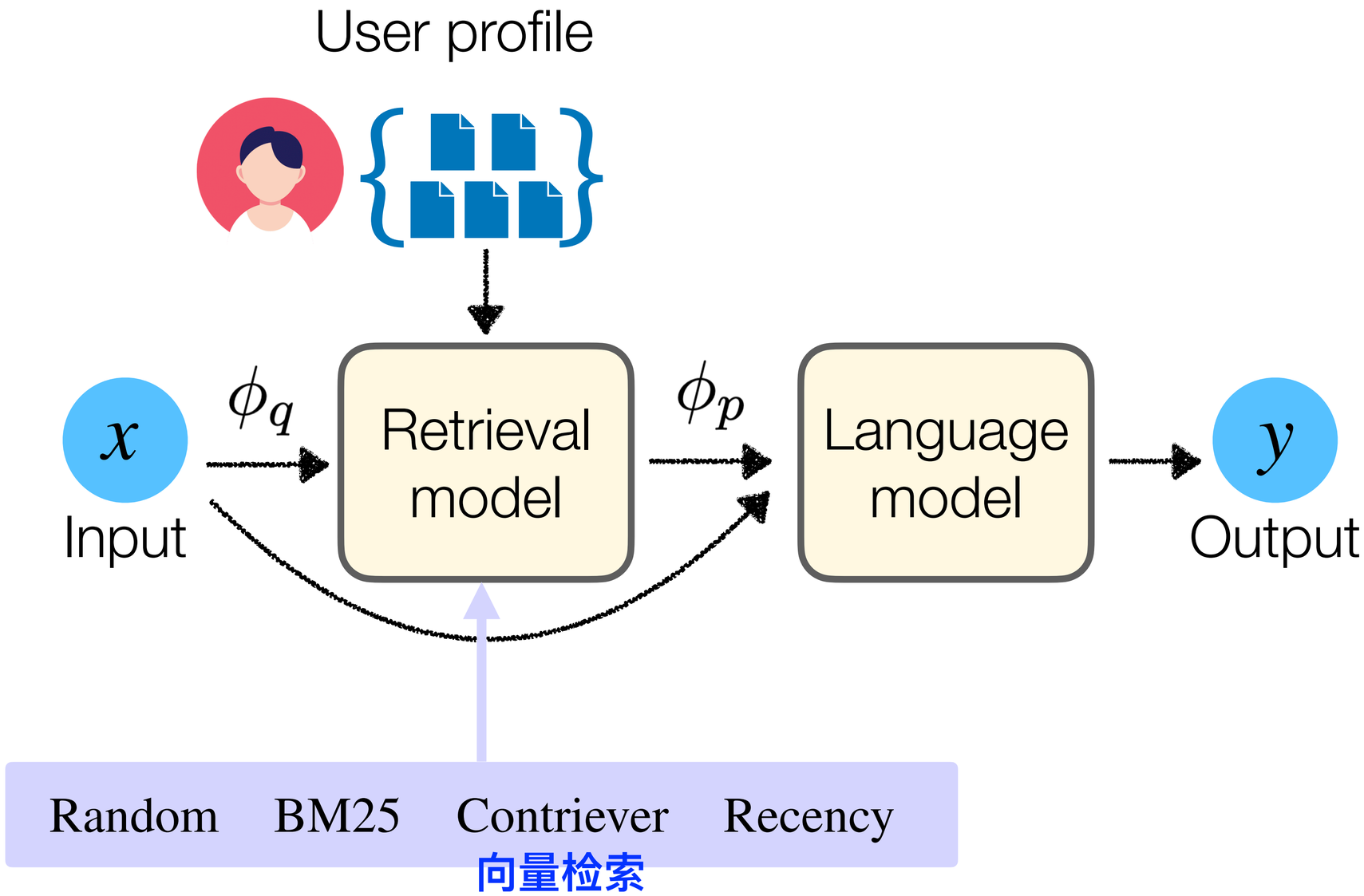
LaMP Benchmark: Prompt

Task	Per Profile Entry Prompt (PPEP)	Aggregated Input Prompt(AIP)
LaMP-1: Citation Ident.	"P_i[title]"	add_to_paper_title(concat([PPEP(P_1), ..., PPEP(P_n)], ", and "), [INPUT])
LaMP-2: News Cat.	the category for the article: "P_i[text]" is "P_i[category]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-3: Product Rat.	P_i[score] is the score for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-4: News Headline	"P_i[title]" is the title for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-5: Scholarly Title	"P_i[title]" is the title for "P_i[abstract]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). Following the given patterns [INPUT]
LaMP-6: Email Subject	"P_i[title]" is the title for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-7: Tweet Para.	"P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and ") are written by a person. Following the given patterns [INPUT]

Retrieval-augmented personalizing LLM



Retrieval-augmented personalizing LLM



LaMP Results

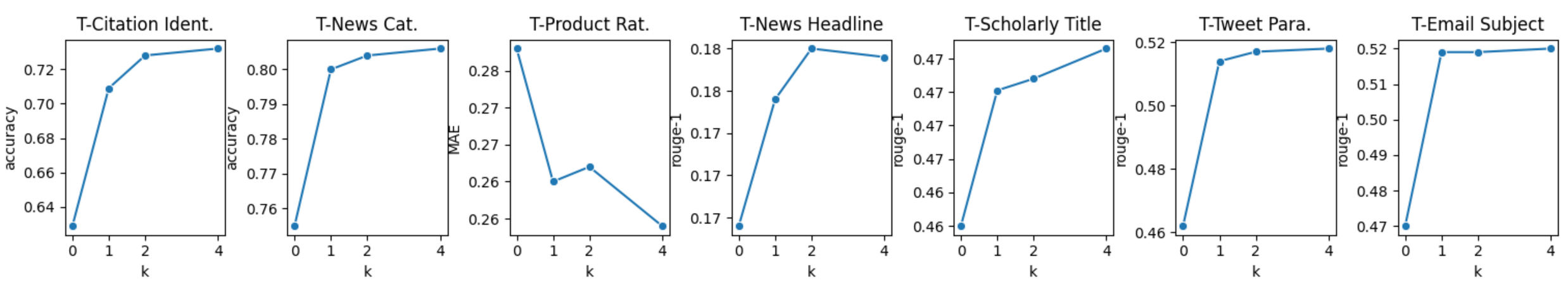
		FlanT5-base (fine-tuned)					
		Non-Personalized	Untuned profile, $k = 1$				Tuned profile
Dataset	Metric		Random	BM25	Contriever	Recency	
LaMP-1T: Personalized Citation Identification	Accuracy	0.628	0.657	0.682	0.688	0.691	**0.714**
LaMP-2T: Personalized News Categorization	Accuracy	0.762	0.794	0.783	**0.815**	0.800	0.806
	F1	0.574	0.634	0.613	0.656	0.645	**0.659**
LaMP-3T: Personalized Product Rating	MAE	0.280	0.279	0.278	0.281	0.279	**0.266**
	RMSE	0.615	0.612	0.614	0.606	0.608	**0.598**
LaMP-4T: Personalized News Headline Generation	ROUGE-1	0.159	0.169	0.171	0.176	0.173	**0.177**
	ROUGE-L	0.145	0.155	0.157	0.162	0.158	**0.162**
LaMP-5T: Personalized Scholarly Title Generation	ROUGE-1	0.462	0.460	0.471	0.472	0.466	**0.479**
	ROUGE-L	0.416	0.414	0.423	0.426	0.420	**0.431**
LaMP-6T: Personalized Email Subject Generation	ROUGE-1	0.479	0.525	0.537	0.545	0.532	**0.547**
	ROUGE-L	0.463	0.507	0.522	0.530	0.518	**0.533**
LaMP-7T: Personalized Tweet Paraphrasing	ROUGE-1	0.462	0.505	0.508	0.505	0.503	**0.516**
	ROUGE-L	0.416	0.456	0.457	0.455	0.453	**0.465**

LaMP Results

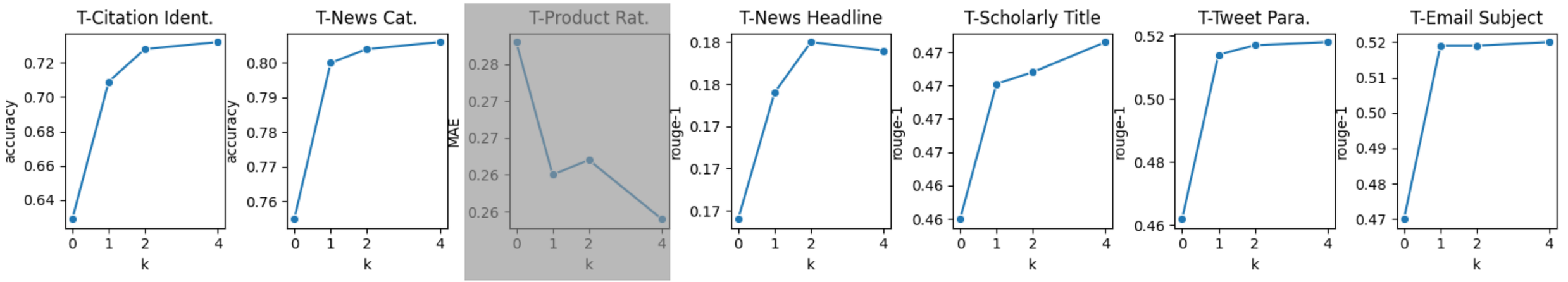
		FlanT5-base (fine-tuned)					
		Non-Personalized	Untuned profile, $k = 1$				Tuned profile
Dataset	Metric		Random	BM25	Contriever	Recency	
LaMP-1T: Personalized Citation Identification	Accuracy	0.628	0.657	0.682	0.688	0.691	**0.714**
LaMP-2T: Personalized News Categorization	Accuracy	0.762	0.794	0.783	**0.815**	0.800	0.806
	F1	0.574	0.634	0.613	0.656	0.645	**0.659**
LaMP-3T: Personalized Product Rating	MAE	0.280	0.279	0.278	0.281	0.279	**0.266**
	RMSE	0.615	0.612	0.614	0.606	0.608	**0.598**
LaMP-4T: Personalized News Headline Generation	ROUGE-1	0.159	0.169	0.171	0.176	0.173	**0.177**
	ROUGE-L	0.145	0.155	0.157	0.162	0.158	**0.162**
LaMP-5T: Personalized Scholarly Title Generation	ROUGE-1	0.462	0.460	0.471	0.472	0.466	**0.479**
	ROUGE-L	0.416	0.414	0.423	0.426	0.420	**0.431**
LaMP-6T: Personalized Email Subject Generation	ROUGE-1	0.479	0.525	0.537	0.545	0.532	**0.547**
	ROUGE-L	0.463	0.507	0.522	0.530	0.518	**0.533**
LaMP-7T: Personalized Tweet Paraphrasing	ROUGE-1	0.462	0.505	0.508	0.505	0.503	**0.516**
	ROUGE-L	0.416	0.456	0.457	0.455	0.453	**0.465**

Retrieval 🏆 7, Contriever 🎖️

LaMP Results: Tuned



LaMP Results: Tuned



Input Layer: Conclusion

	通用性	推理效率	隐私保护	存储成本	特点
🎯Input					
🎯hidden					
🎯output					

Input Layer: Conclusion

	通用性	推理效率	隐私保护	存储成本	特点
🎯Input	**适用API** **最大长度**	**一次检索** **样本需要编码**	**检索返回明文**	**向量+文本**	**Top-K位置敏感**
🎯hidden					
🎯output					

Input Layer: Conclusion

	通用性	推理效率	隐私保护	存储成本	特点
🎯Input	🥈 适用API 最大长度	🥉 一次检索 样本需要编码	🥉 检索返回明文	🏅 向量+文本	Top-K位置敏感
🎯hidden					
🎯output					

Hidden Layer: Attention

DeepMind

Improving language models by retrieving from trillions of tokens

Sebastian Borgeaud[†], Arthur Mensch[†], Jordan Hoffmann[†], Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones,

Hidden Layer: Attention

DeepMind

Improving language models by retrieving from trillions of tokens

Sebastian Borgeaud[†], Arthur Mensch[†], Jordan Hoffmann[†], Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones,

What to retrieve?

- **Chunks** ✓
- Tokens
- Others

How to use retrieval?

- Input layer
- **Intermediate layers** ✓
- Output layer

When to retrieve?

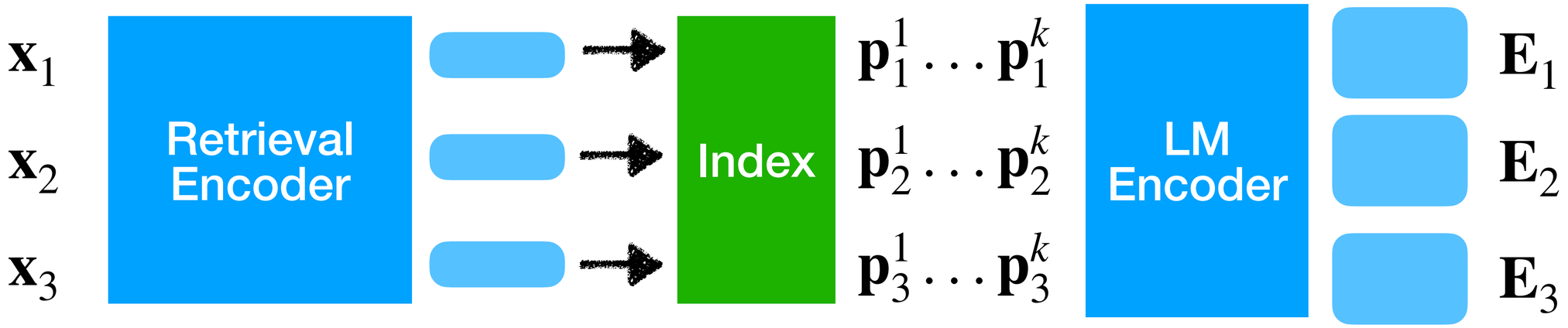
- Once
- **Every *n* tokens (*n*>1)** ✓
- Every token

RETRO

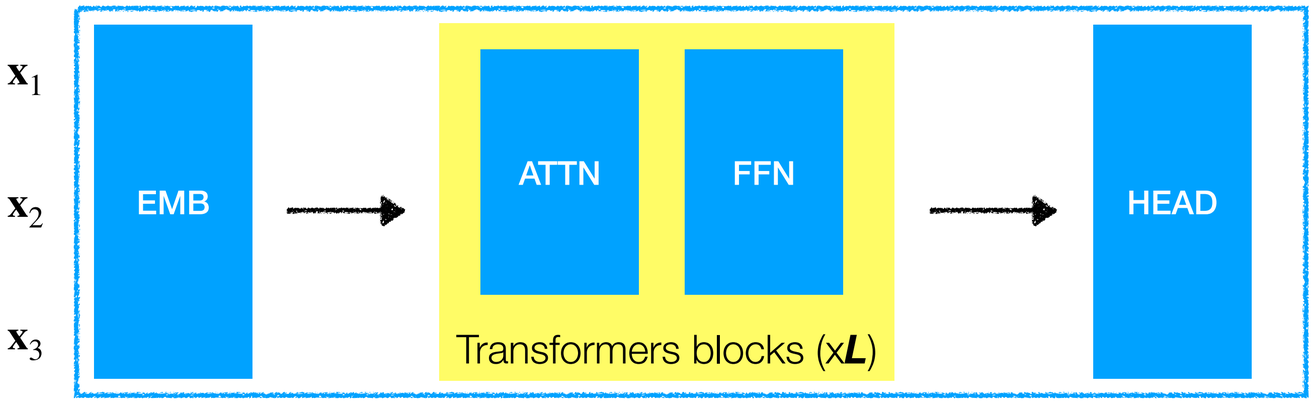
$\boldsymbol{x}$ = World Cup 2022 was / the last with 32 teams, / before the increase to

$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$

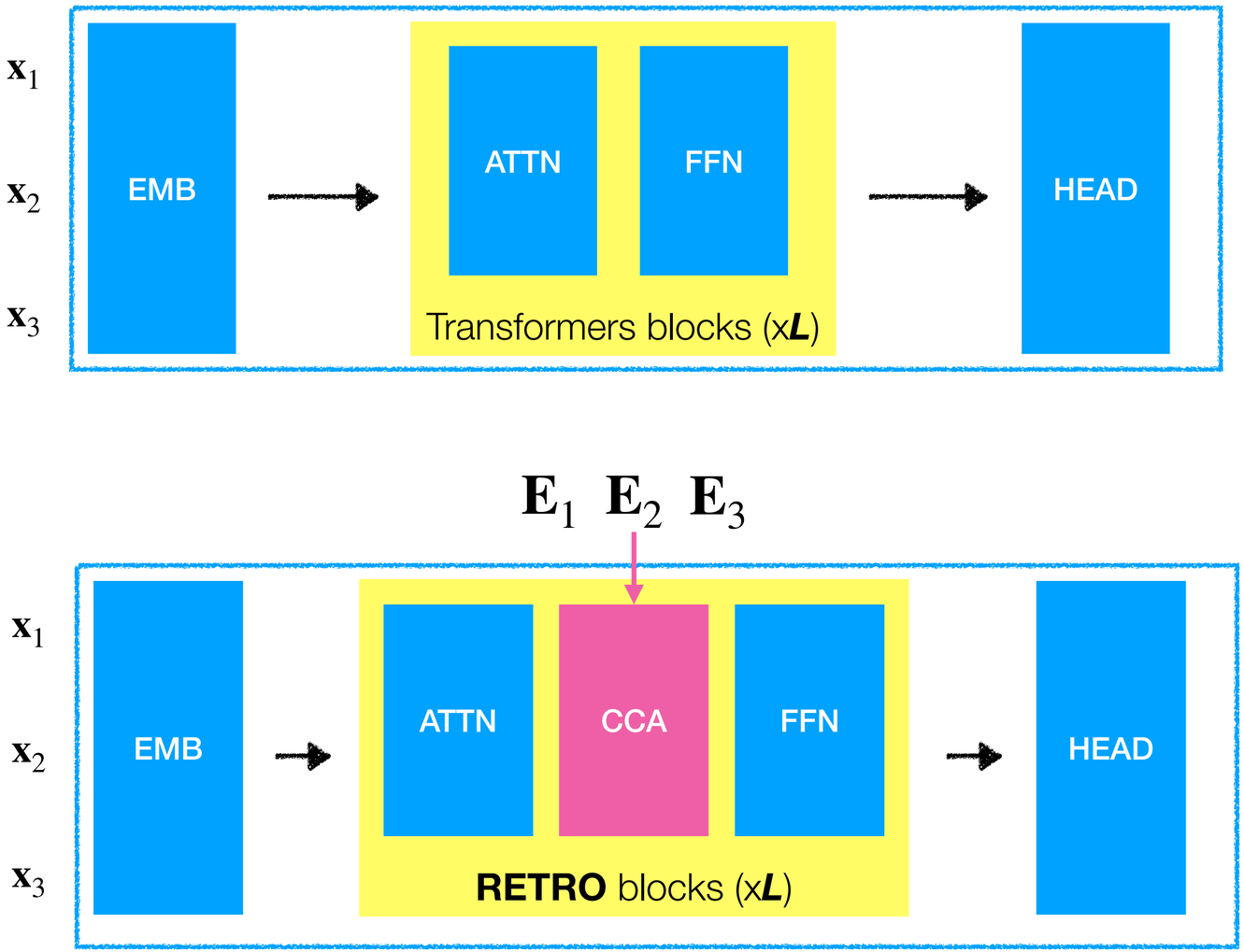
(k chunks of text per split)



RETRO

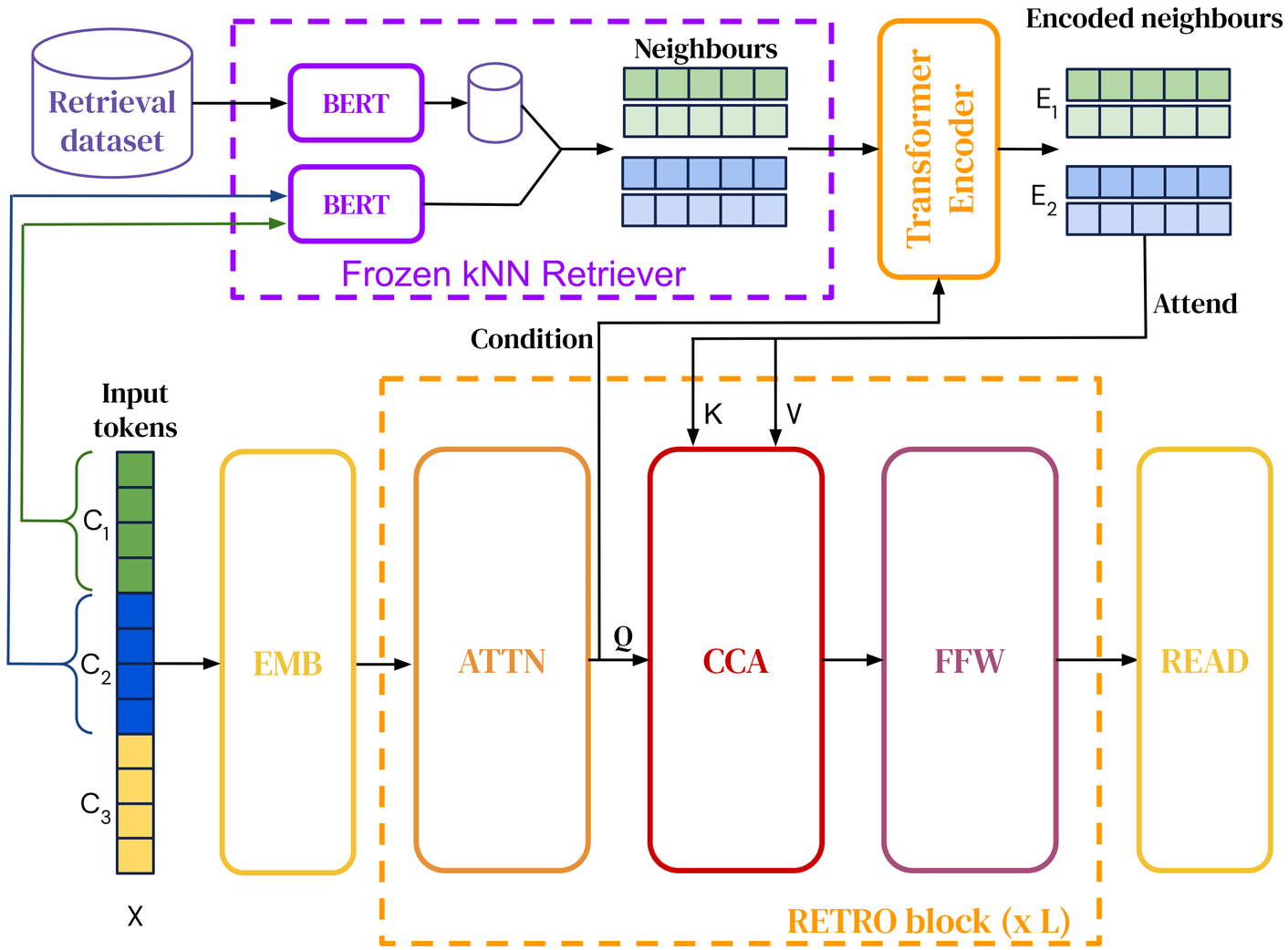

$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$
EMB
ATTN
FFN
Transformers blocks (x*L*)
HEAD

RETRO

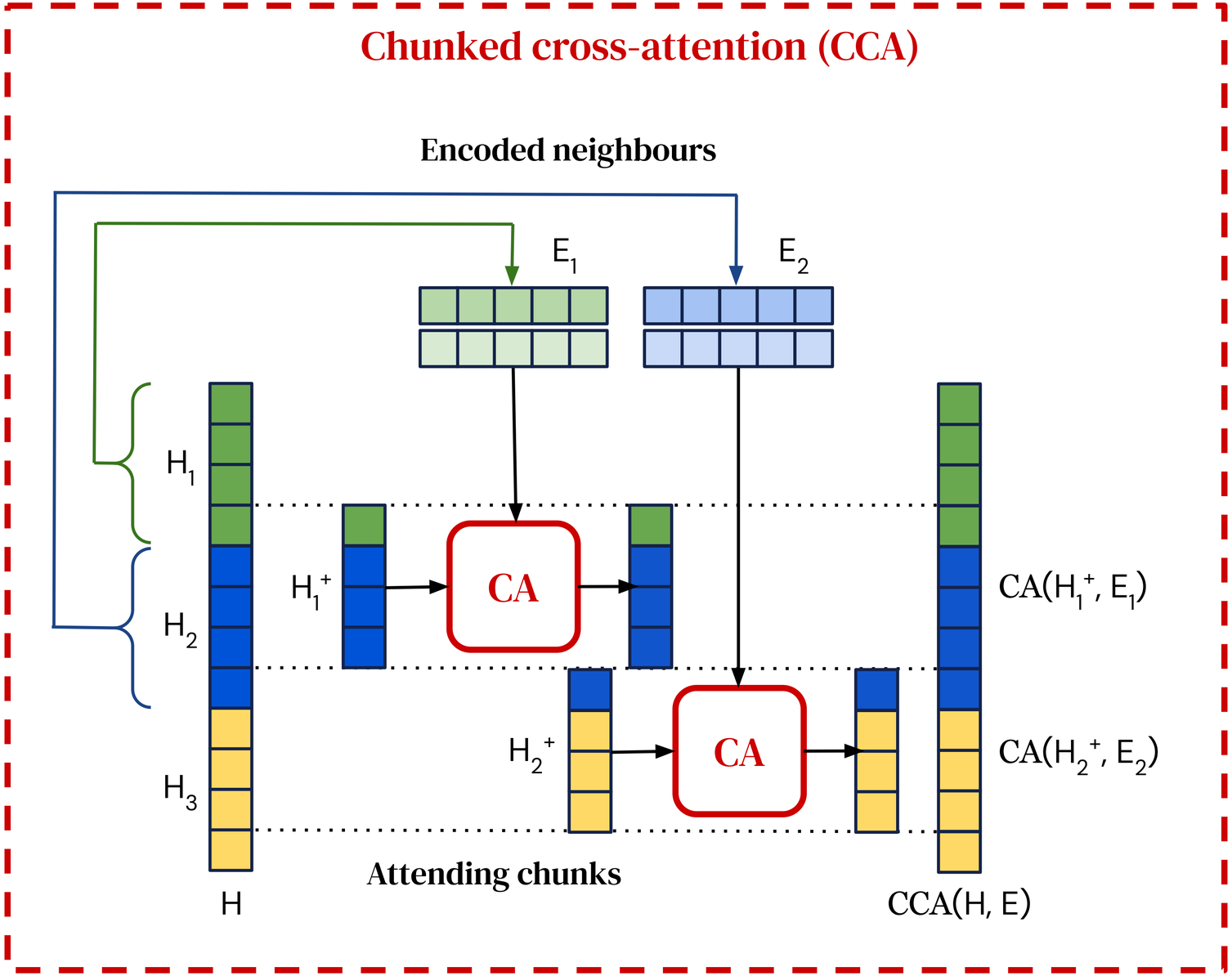


Chunked Cross Attention (CCA)

RETRO


Retrieval dataset
BERT
BERT
Frozen kNN Retriever
Neighbours
Transformer Encoder
Encoded neighbours
E1
E2
Attend
Condition
Input tokens
C1
C2
C3
X
EMB
ATTN
Q
K
V
CCA
FFW
READ
RETRO block (x L)

RETRO

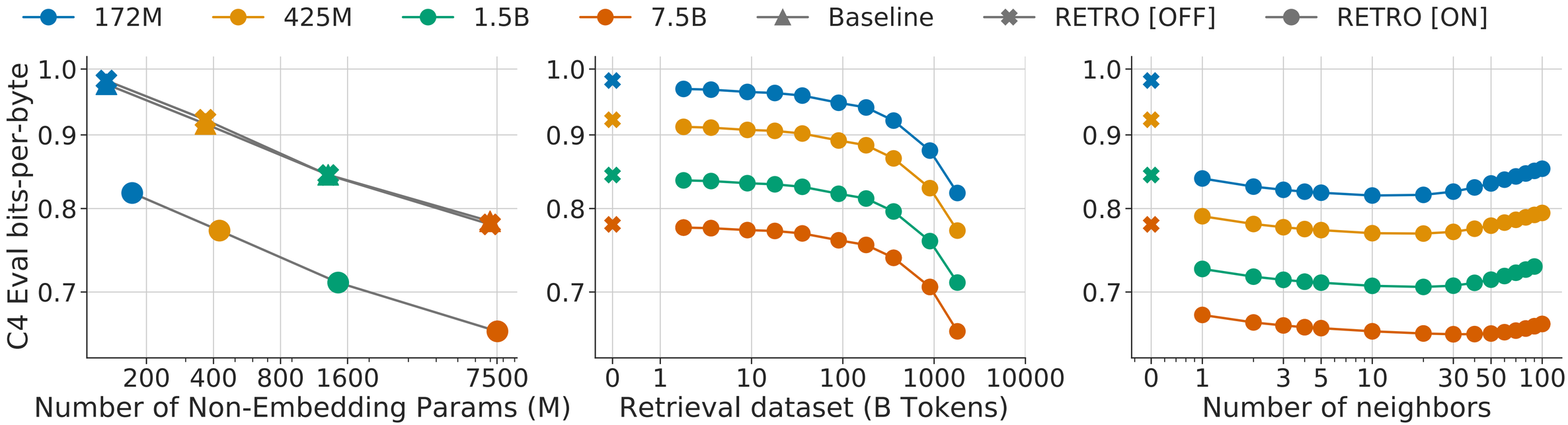

Chunked cross-attention (CCA)
Encoded neighbours
E1
E2
H1
H2
H3
H1+
CA
CA(H1+, E1)
H2+
CA
CA(H2+, E2)
Attending chunks
H
CCA(H, E)

RETRO: C4 Results

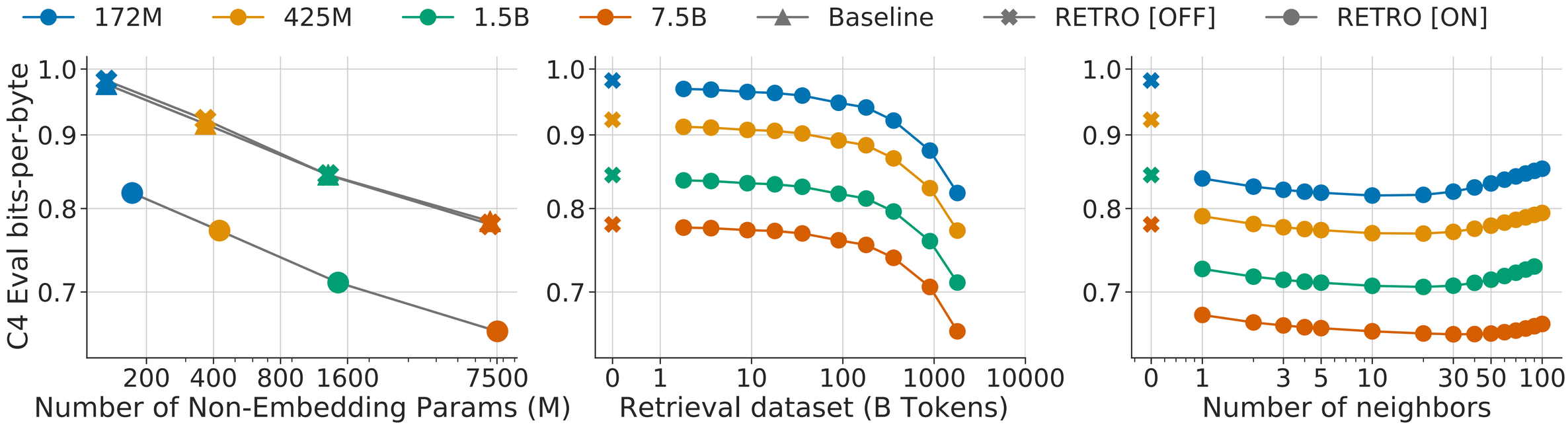
Perplexity: The lower the better

Model	Retrieval Set	#Database tokens	#Database keys	Valid	Test
Adaptive Inputs (Baevski and Auli, 2019)	-	-	-	17.96	18.65
SPALM (Yogatama et al., 2021)	Wikipedia	3B	3B	17.20	17.60
*k*NN-LM (Khandelwal et al., 2020)	Wikipedia	3B	3B	16.06	16.12
Megatron (Shoeybi et al., 2019)	-	-	-	-	10.81
Baseline transformer (ours)	-	-	-	21.53	22.96
*k*NN-LM (ours)	Wikipedia	4B	4B	18.52	19.54
RETRO	Wikipedia	4B	0.06B	18.46	18.97
RETRO	C4	174B	2.9B	12.87	10.23
RETRO	MassiveText (1%)	18B	0.8B	18.92	20.33
RETRO	MassiveText (10%)	179B	4B	13.54	14.95
RETRO	MassiveText (100%)	1792B	28B	**3.21**	**3.92**

RETRO: C4 Results

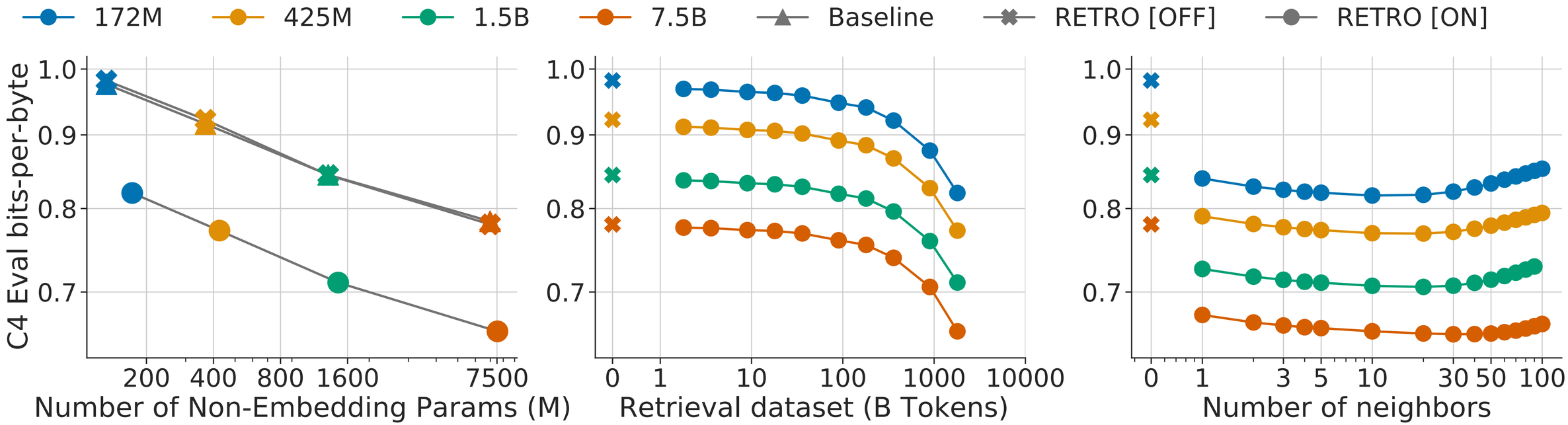


RETRO: C4 Results



Retrieval=Params.

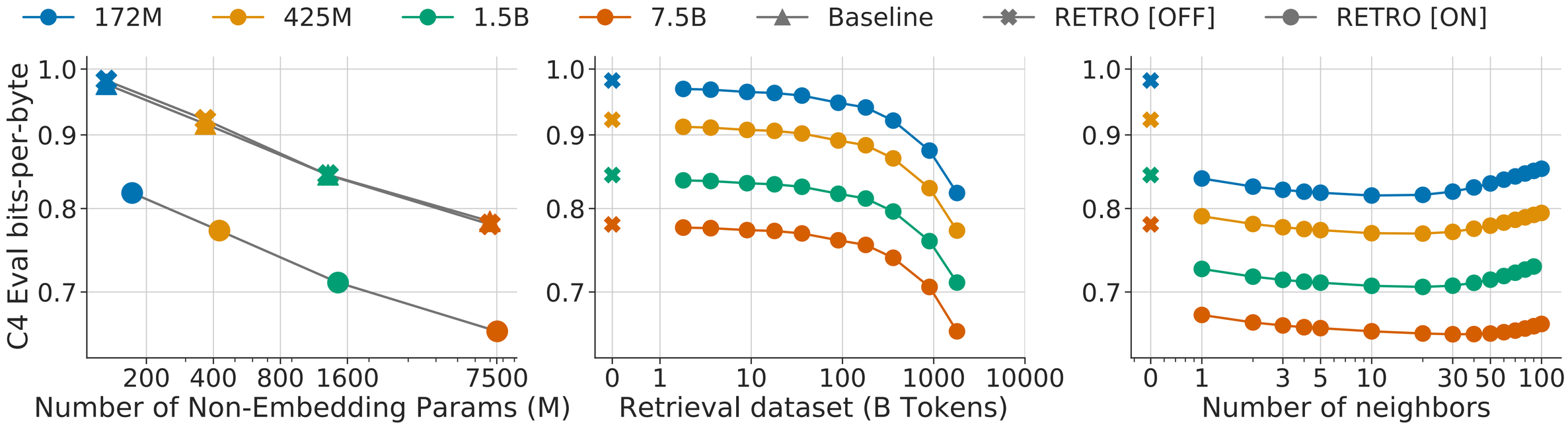
RETRO: C4 Results



Retrieval=Params.

Retrieval Set 📈

RETRO: C4 Results



Retrieval=Params.

Retrieval Set 📈

#Retrieval 📈📉

RETRO Simplified

Task	Per Profile Entry Prompt (PPEP)	Aggregated Input Prompt(AIP)
LaMP-1: Citation Ident.	"P_i[title]"	add_to_paper_title(concat([PPEP(P_1), ..., PPEP(P_n)], ", and "), [INPUT])
LaMP-2: News Cat.	the category for the article: "P_i[text]" is "P_i[category]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-3: Product Rat.	P_i[score] is the score for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-4: News Headline	"P_i[title]" is the title for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-5: Scholarly Title	"P_i[title]" is the title for "P_i[abstract]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). Following the given patterns [INPUT]
LaMP-6: Email Subject	"P_i[title]" is the title for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-7: Tweet Para.	"P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and ") are written by a person. Following the given patterns [INPUT]

🎯Input

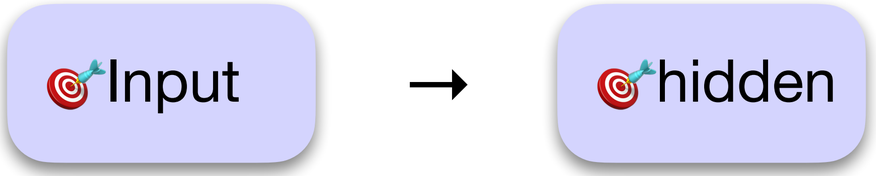
RETRO Simplified

Task	Per Profile Entry Prompt (PPEP)	Aggregated Input Prompt(AIP)
LaMP-1: Citation Ident.	"P_i[title]"	add_to_paper_title(concat([PPEP(P_1), ..., PPEP(P_n)], ", and "), [INPUT])
LaMP-2: News Cat.	the category for the article: "P_i[text]" is "P_i[category]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-3: Product Rat.	P_i[score] is the score for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and [illegible]NPUT]
LaMP-4: News Headline	"P_i[title]" is the title for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-5: Scholarly Title	"P_i[title]" is the title for "P_i[abstract]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). Following the given patterns [INPUT]
LaMP-6: Email Subject	"P_i[title]" is the title for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-7: Tweet Para.	"P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and ") are written by a person. Following the given patterns [INPUT]

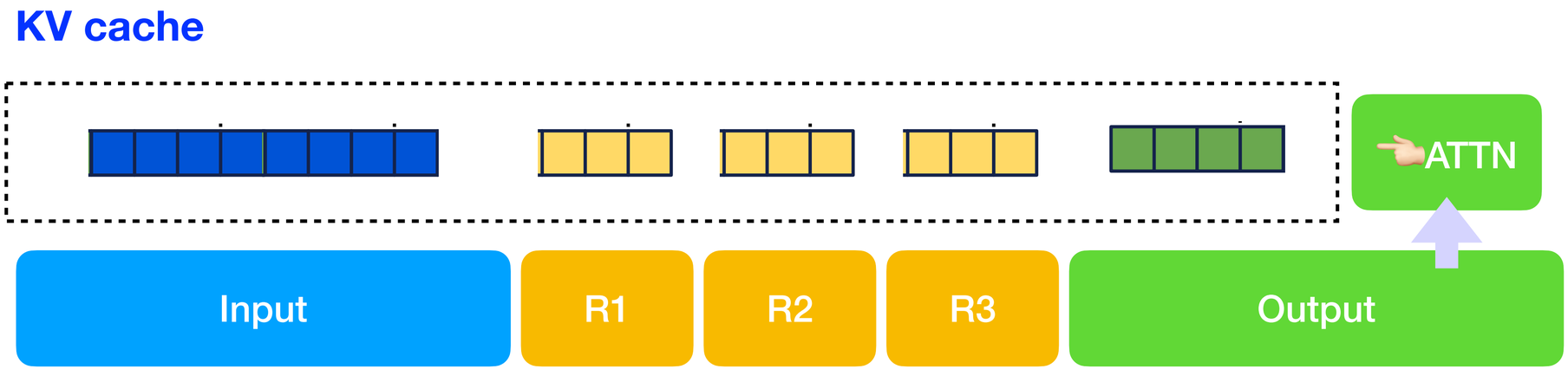
🎯Input

RETRO Simplified

Task	Per Profile Entry Prompt (PPEP)	Aggregated Input Prompt(AIP)
LaMP-1: Citation Ident.	"P_i[title]"	add_to_paper_title(concat([PPEP(P_1), ..., PPEP(P_n)], ", and "), [INPUT])
LaMP-2: News Cat.	the category for the article: "P_i[text]" is "P_i[category]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-3: Product Rat.	P_i[score] is the score for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and [illegible]NPUT]
LaMP-4: News Headline	"P_i[title]" is the title for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-5: Scholarly Title	"P_i[title]" is the title for "P_i[abstract]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). Following the given patterns [INPUT]
LaMP-6: Email Subject	"P_i[title]" is the title for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
LaMP-7: Tweet Para.	"P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and ") are written by a person. Following the given patterns [INPUT]

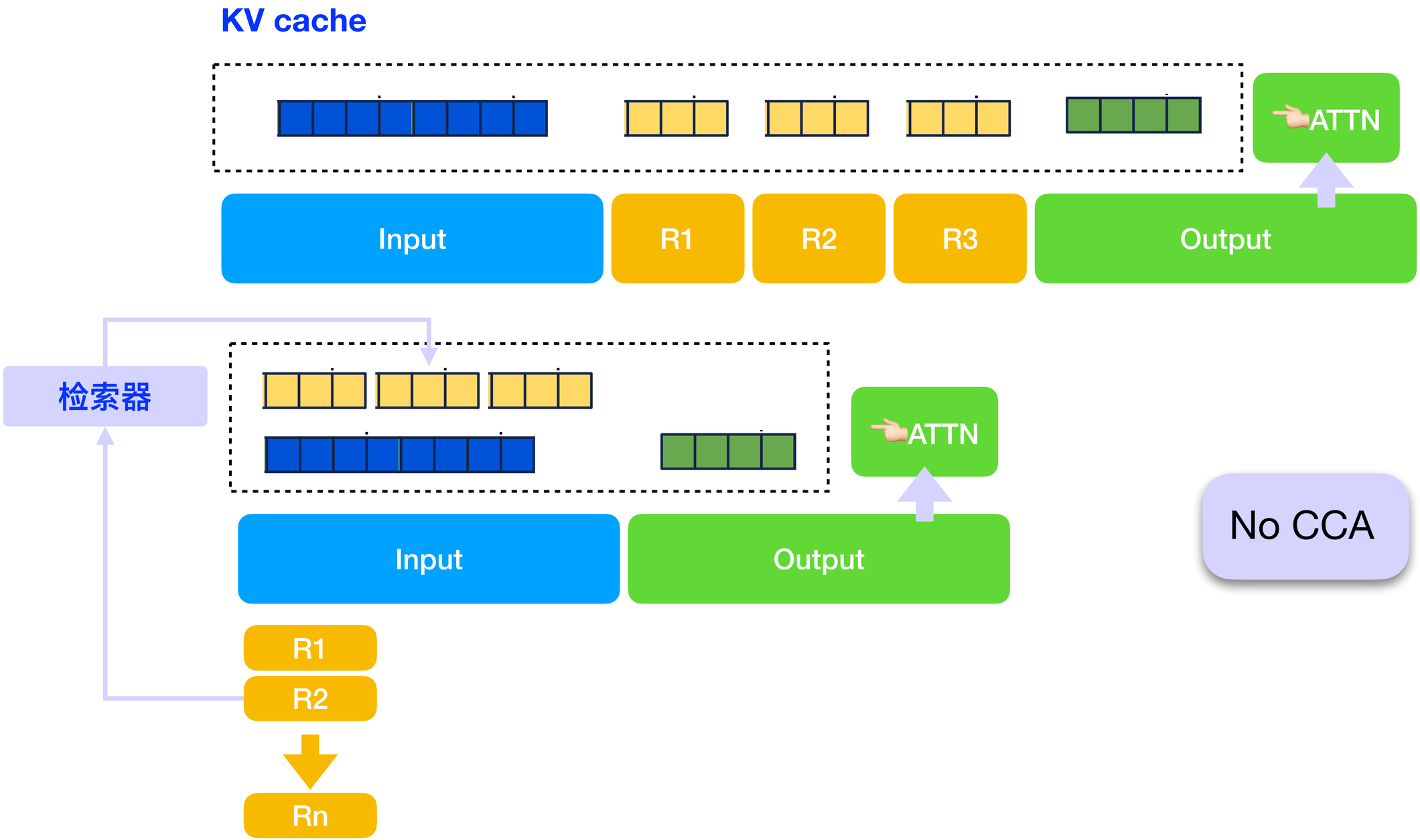


RETRO Simplified



No CCA

RETRO Simplified



Our Results: 1U-task

Our LaMP

- NoP = 0.654 ↔ 0.652
- PIO = 0.670 ↔ 0.690
- IPO = 0.656
- (Pv)IO = 0.713

Our Results: 1U-task

Our LaMP

- NoP = 0.654 ↔ 0.652
- PIO = 0.670 ↔ 0.690
- IPO = 0.656
- (Pv)IO = 0.713

- Full results
- n_tok > max_length
- 隐私

Hidden Layer: Conclusion

	通用性	推理效率	隐私保护	存储成本	特点
🎯Input	🥈 适用API 最大长度	🥉 一次检索 样本需再编码	🥉 检索返回明文	🏅 向量+文本	Top-K位置敏感
🎯hidden					
🎯output					

Hidden Layer: Conclusion

	通用性	推理效率	隐私保护	存储成本	特点
🎯Input	🥈 适用API 最大长度	🥉 一次检索 样本需再编码	🥉 检索返回明文	🥇 向量+文本	Top-K位置敏感
🎯hidden	无长度约束 不适用API	样本无再编码 逐词/层/头检索	返回Topk向量	逐词/层/头向量	PEFT
🎯output					

Hidden Layer: Conclusion

	通用性	推理效率	隐私保护	存储成本	特点
🎯Input	🥈 **适用API** **最大长度**	🥉 **一次检索** **样本需再编码**	🥉 **检索返回明文**	🏅 **向量+文本**	**Top-K位置敏感**
🎯hidden	🥈 **无长度约束** **不适用API**	🥈 **样本无再编码** **逐词/层/头检索**	🏅 **返回Topk向量**	🏅 **逐词/层/头向量**	**PEFT**
🎯output					

Output Layer: Copy Generation

ICLR 2023

COPY IS ALL YOU NEED

Tian Lan[♢,♡,*] **Deng Cai**[♢,*,†] **Yan Wang**[♢,†] **Heyan Huang**[♡] **Xian-Ling Mao**[♡]
[♢]Tencent AI Lab
[♡]School of Computer Science and Technology, Beijing Institute of Technology

Output Layer: Copy Generation

ICLR 2023

COPY IS ALL YOU NEED

Tian Lan[♢,♡,*] **Deng Cai**[♢,*,†] **Yan Wang**[♢,†] **Heyan Huang**[♡] **Xian-Ling Mao**[♡]
[♢]Tencent AI Lab
[♡]School of Computer Science and Technology, Beijing Institute of Technology

What to retrieval

- Chunk
- Tokens
- **Phrase**

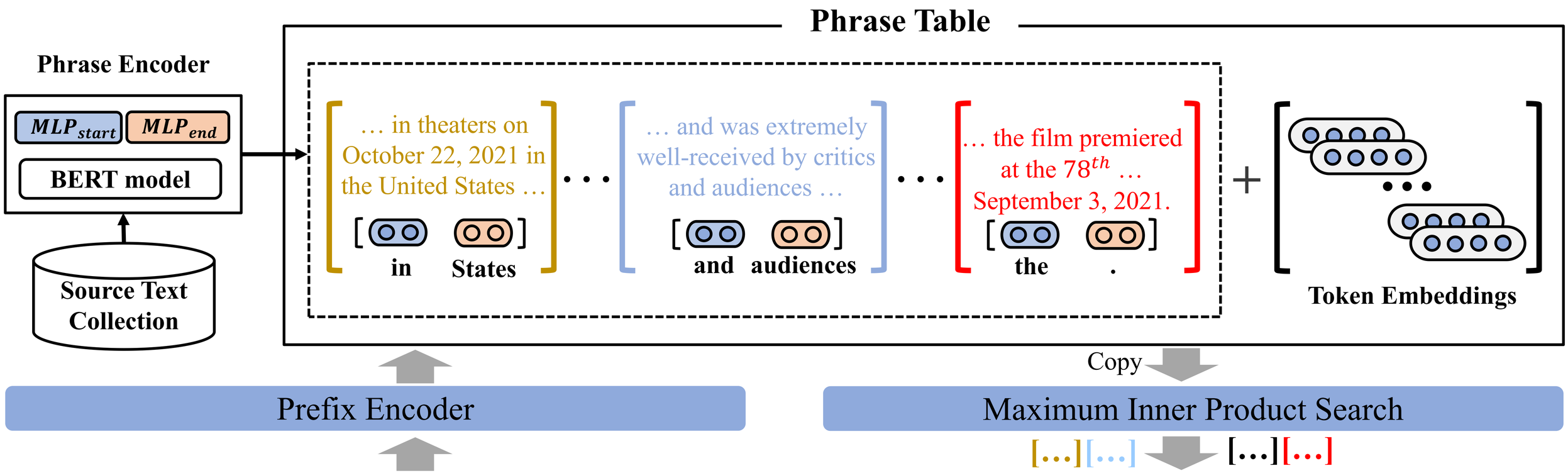
How to use retrieval

- Input layer
- Intermediate layers
- **Output layer**

When to retrieval

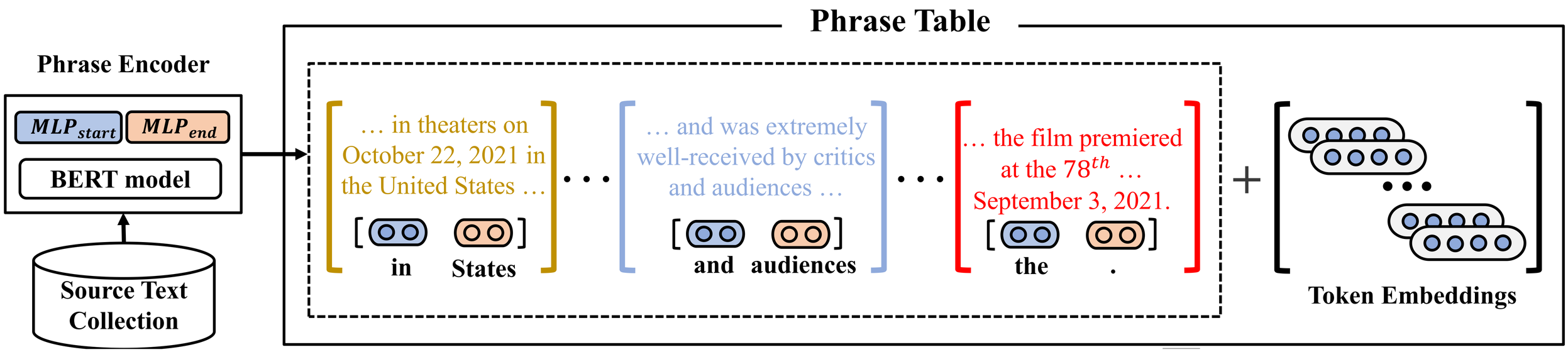
- Once
- Every n tokens (n>1)
- Every token
- **Adaptive**

Method

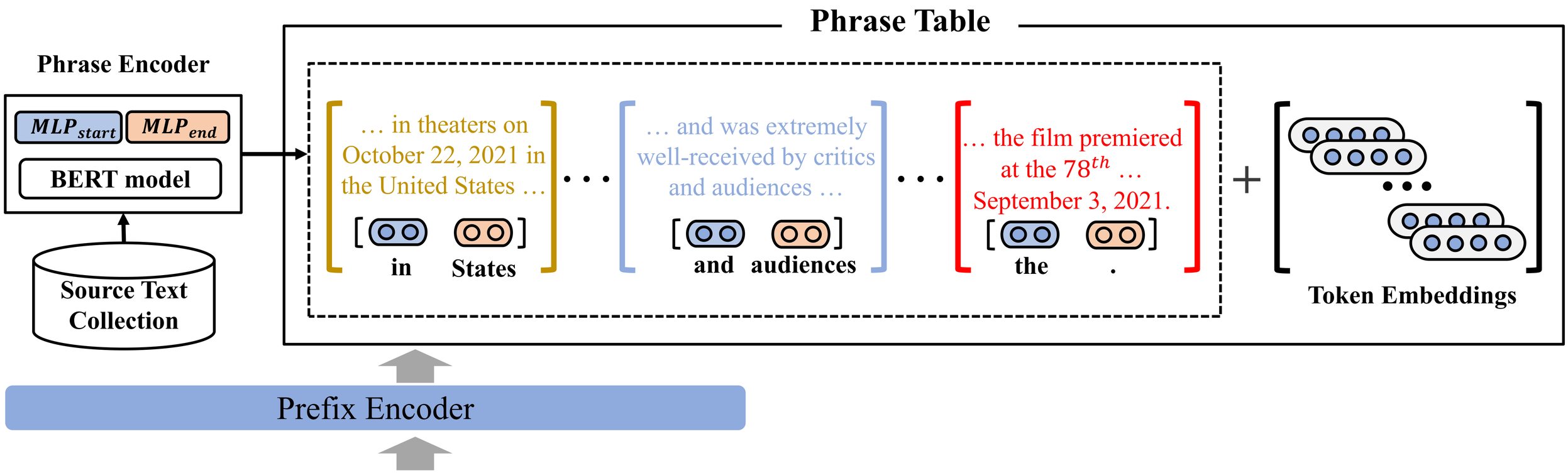


The Dune film was released [in theaters on October 22, 2021 in the United States] [and was extremely well-received by critics and audiences] **[Before] [that] [,]** [the film premiered at the 78^{th} International Film Festival on September 3, 2021.]

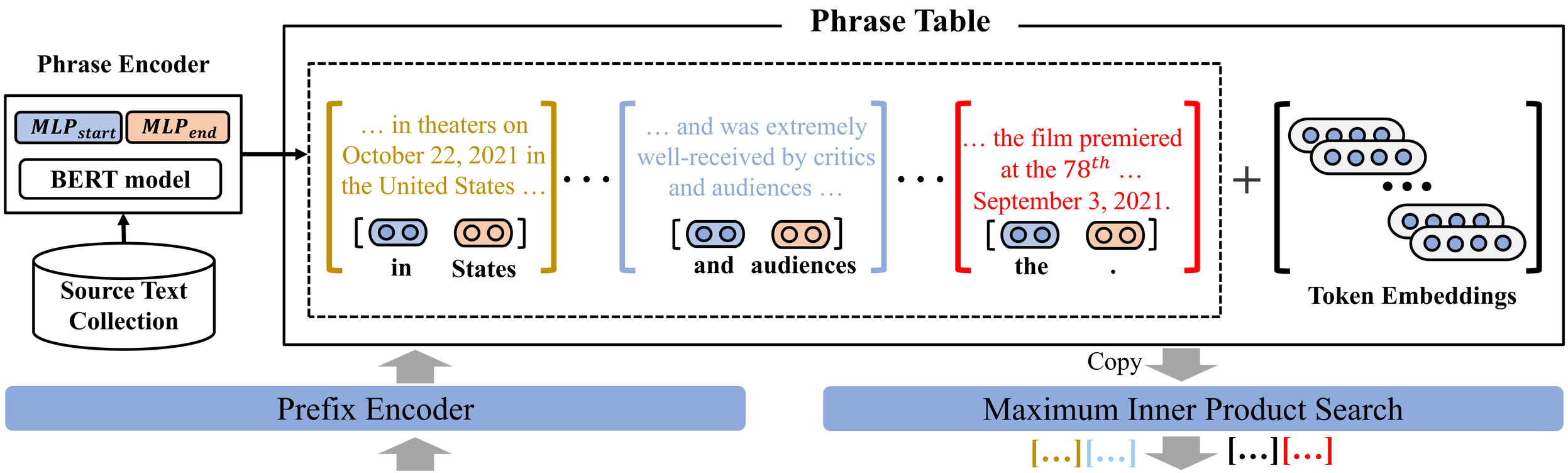
Method


Phrase Table
Phrase Encoder
MLP_{start}
MLP_{end}
BERT model
Source Text Collection
… in theaters on October 22, 2021 in the United States …
in
States
… and was extremely well-received by critics and audiences …
and
audiences
… the film premiered at the 78^{th} … September 3, 2021.
the
.
Token Embeddings

Method


Phrase Table
Phrase Encoder
MLP_{start}
MLP_{end}
BERT model
Source Text Collection
… in theaters on October 22, 2021 in the United States …
in
States
… and was extremely well-received by critics and audiences …
and
audiences
… the film premiered at the 78^{th} … September 3, 2021.
the
.
Token Embeddings
Prefix Encoder
The Dune film was released

Method



The Dune film was released [in theaters on October 22, 2021 in the United States]

Finetuned Phrase Encoder

Phrase Table → LM Vocabulary

Method

- phrase怎么获得？ 传统分词算法→最大前向算法
- 动态词表size？50257 + 950942（k=1024）
- 训练目标？🤔

Method

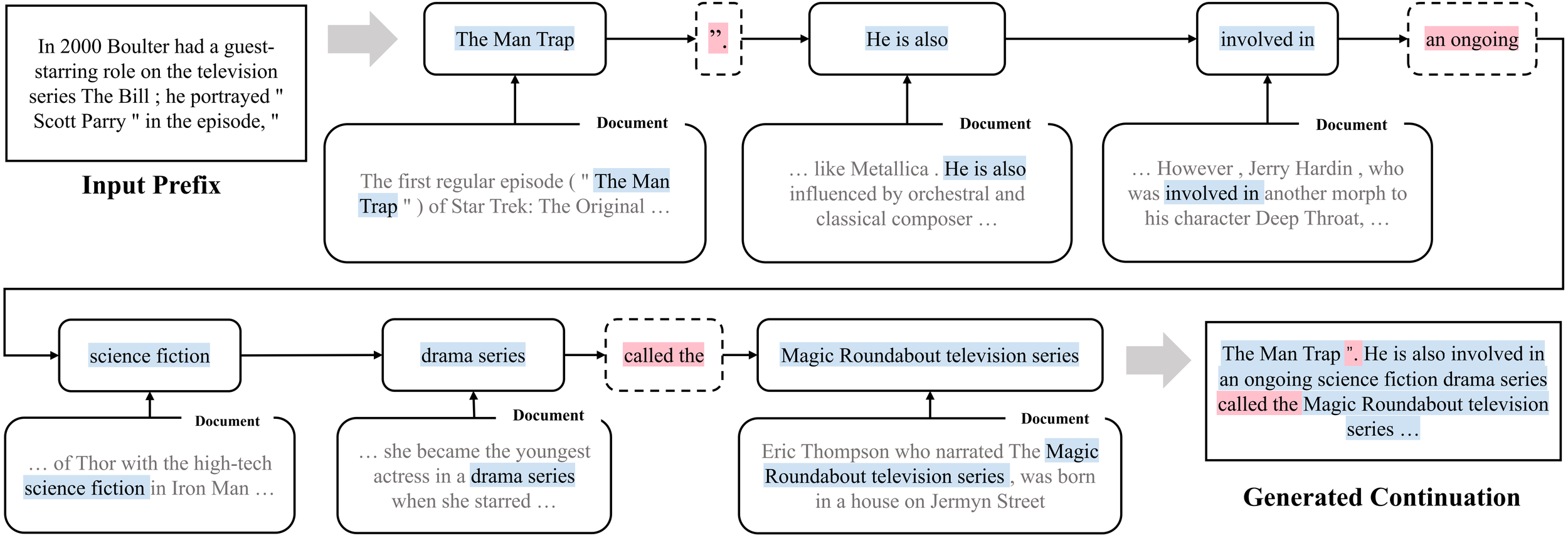
- phrase怎么获得？ 传统分词算法→最大前向算法
- 动态词表size？50257 + 950942（k=1024）
- 训练目标？🤔

$$\mathcal{L}_p = -\frac{1}{n}\sum_{k=1}^{n}\log\frac{\exp(q_k \cdot p_k)}{\sum_{p\in\mathcal{P}_k}\exp(q_k \cdot p_p) + \sum_{w\in V}\exp(q_k \cdot v_w)}$$

$$\mathcal{L}_t = -\frac{1}{m}\sum_{i=1}^{m}\log\frac{\exp(q_i, v_{D_i})}{\sum_{w\in V}\exp(q_i, v_w)}$$

$$\mathcal{L} = \mathcal{L}_p + \mathcal{L}_t$$

Method


In 2000 Boulter had a guest-starring role on the television series The Bill ; he portrayed " Scott Parry " in the episode, "
Input Prefix
The Man Trap
".
He is also
involved in
an ongoing
Document
The first regular episode (" The Man Trap ") of Star Trek: The Original …
Document
… like Metallica . He is also influenced by orchestral and classical composer …
Document
… However , Jerry Hardin , who was involved in another morph to his character Deep Throat, …
science fiction
drama series
called the
Magic Roundabout television series
Document
… of Thor with the high-tech science fiction in Iron Man …
Document
… she became the youngest actress in a drama series when she starred …
Document
Eric Thompson who narrated The Magic Roundabout television series , was born in a house on Jermyn Street
The Man Trap ". He is also involved in an ongoing science fiction drama series called the Magic Roundabout television series …
Generated Continuation

WikiText103 Results

Model	Decoding	MAUVE↑	Rep-2↓	Rep-3↓	Rep-4 ↓	Diversity↑	Latency (s)↓
Transformer	greedy	19.87	43.56	38.55	35.5	22.37	1.32
	nucleus	23.43	5.10	1.33	0.50	93.22	1.48
kNN-LM	greedy	19.92	43.79	38.76	35.69	22.13	10.36
	nucleus	22.50	**3.33**	**0.69**	**0.21**	**95.8**	10.42
RETRO	greedy	21.19	44.65	39.63	36.6	21.19	4.39
	nucleus	22.86	6.21	1.93	0.86	91.19	4.51
CoG	greedy	26.01	28.14	23.80	21.40	43.03	**1.29**
	nucleus	**26.14**	7.31	2.66	1.28	89.07	1.54

WikiText103 Results

Model	Decoding	MAUVE↑	Rep-2↓	Rep-3↓	Rep-4 ↓	Diversity↑	Latency (s)↓
Transformer	greedy	19.87	43.56	38.55	35.5	22.37	1.32
	nucleus	23.43	5.10	1.33	0.50	93.22	1.48
kNN-LM	greedy	19.92	43.79	38.76	35.69	22.13	10.36
	nucleus	22.50	**3.33**	**0.69**	**0.21**	**95.8**	10.42
RETRO	greedy	21.19	44.65	39.63	36.6	21.19	4.39
	nucleus	22.86	6.21	1.93	0.86	91.19	4.51
CoG	greedy	26.01	28.14	23.80	21.40	43.03	**1.29**
	nucleus	**26.14**	7.31	2.66	1.28	89.07	1.54

CoG🏆MAUVE or greedy dec.

WikiText103 Results

Model	Decoding	MAUVE↑	Rep-2↓	Rep-3↓	Rep-4 ↓	Diversity↑	Latency (s)↓
Transformer	greedy	19.87	43.56	38.55	35.5	22.37	1.32
	nucleus	23.43	5.10	1.33	0.50	93.22	1.48
kNN-LM	greedy	19.92	43.79	38.76	35.69	22.13	10.36
	nucleus	22.50	**3.33**	**0.69**	**0.21**	**95.8**	10.42
RETRO	greedy	21.19	44.65	39.63	36.6	21.19	4.39
	nucleus	22.86	6.21	1.93	0.86	91.19	4.51
CoG	greedy	26.01	28.14	23.80	21.40	43.03	**1.29**
	nucleus	**26.14**	7.31	2.66	1.28	89.07	1.54

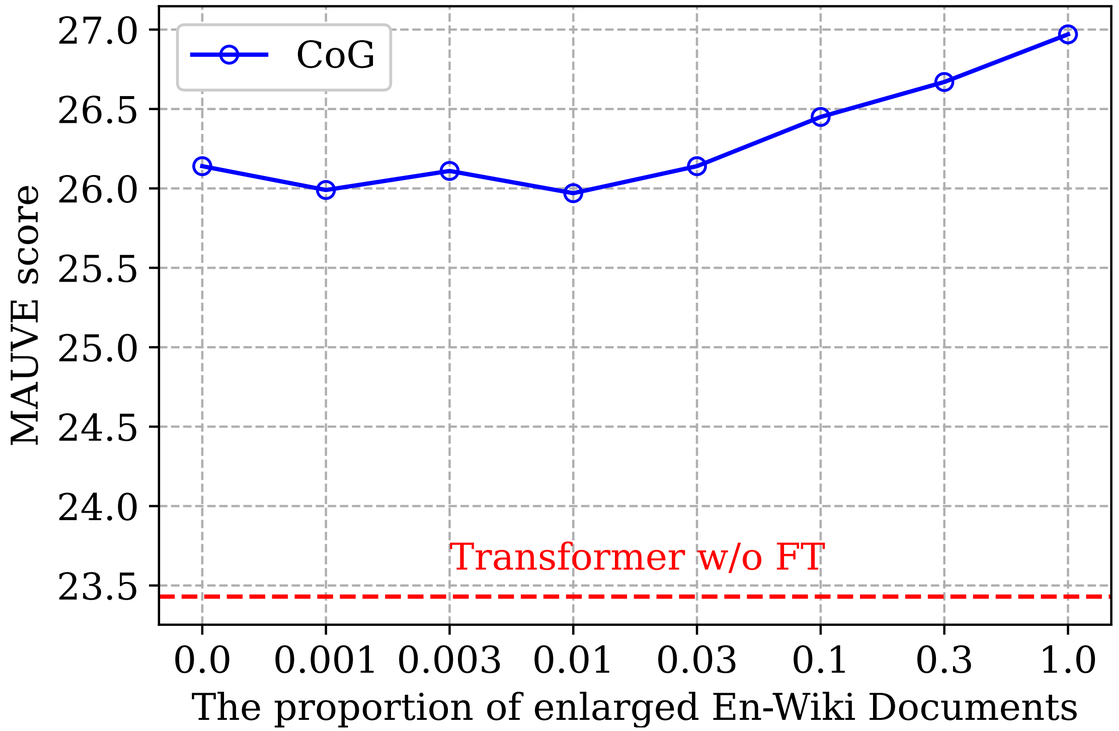
CoG🏆MAUVE or greedy dec.

$$\mathcal{L}_p = -\frac{1}{n}\sum_{k=1}^{n}\log\frac{\exp(q_k \cdot p_k)}{\sum_{p\in\mathcal{P}_k}\exp(q_k \cdot p_p) + \sum_{w\in V}\exp(q_k \cdot v_w)}$$

$$\mathcal{L}_t = -\frac{1}{m}\sum_{i=1}^{m}\log\frac{\exp(q_i, v_{D_i})}{\sum_{w\in V}\exp(q_i, v_w)}$$

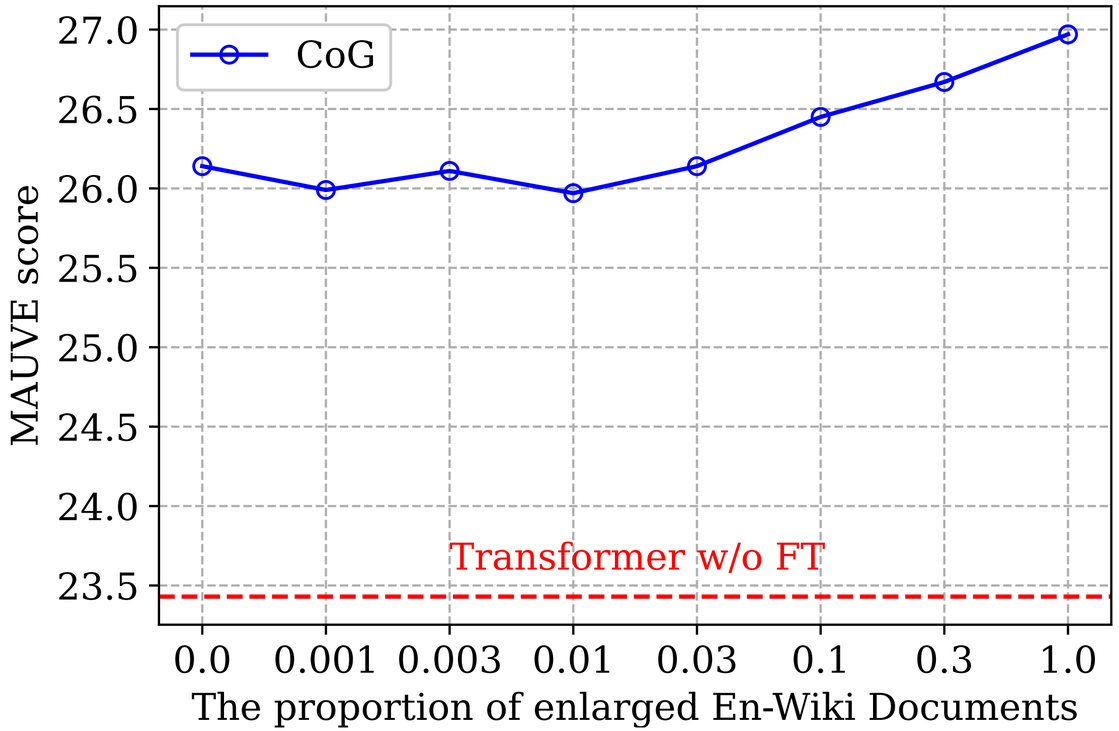
Law-MT Results

Model	Decoding	MAUVE ↑	Diversity ↑
Transformer w/o FT	greedy	19.87	22.37
	nucleus	23.43	93.22
Transformer w/ FT	greedy	20.21	19.62
	nucleus	21.31	92.92
kNN-LM	greedy	23.21	20.33
	nucleus	23.39	**96.37**
RETRO	greedy	19.75	21.15
	nucleus	22.87	91.09
CoG	greedy	24.68	40.45
	nucleus	**26.97**	90.00



Law-MT Results

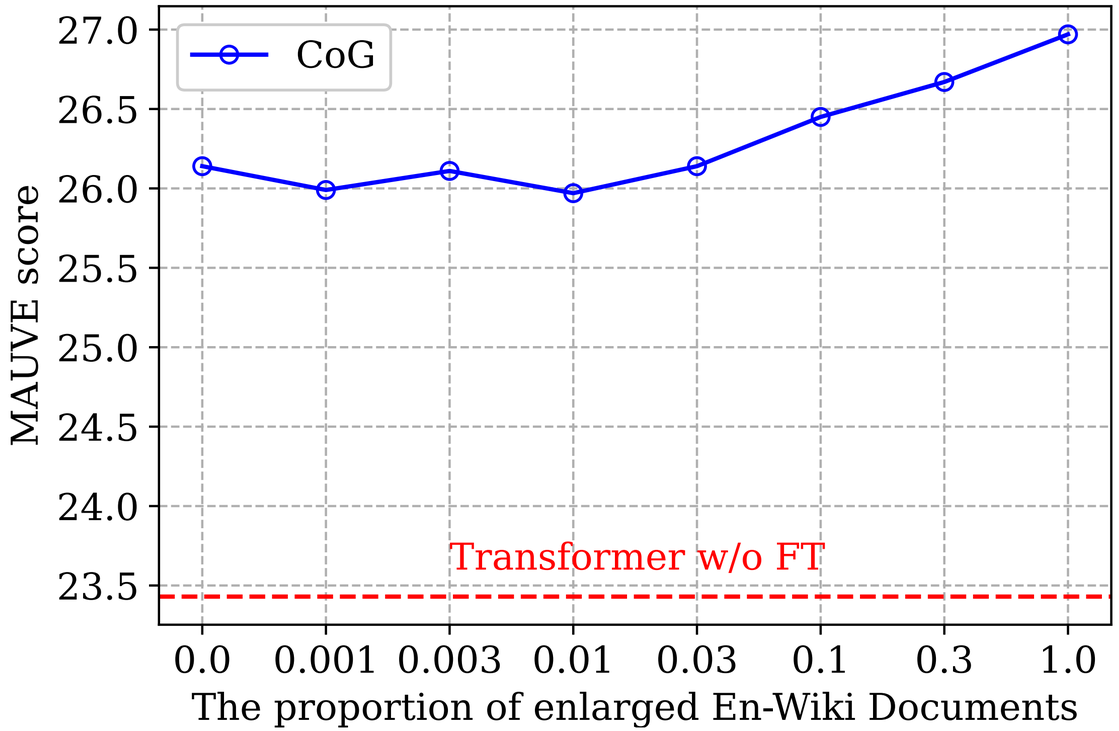
Model	Decoding	MAUVE ↑	Diversity ↑
Transformer w/o FT	greedy	19.87	22.37
	nucleus	23.43	93.22
Transformer w/ FT	greedy	20.21	19.62
	nucleus	21.31	92.92
kNN-LM	greedy	23.21	20.33
	nucleus	23.39	**96.37**
RETRO	greedy	19.75	21.15
	nucleus	22.87	91.09
CoG	greedy	24.68	40.45
	nucleus	**26.97**	90.00



#phrase 📈

Law-MT Results

Model	Decoding	MAUVE ↑	Diversity ↑
Transformer w/o FT	greedy	19.87	22.37
	nucleus	23.43	93.22
Transformer w/ FT	greedy	20.21	19.62
	nucleus	21.31	92.92
kNN-LM	greedy	23.21	20.33
	nucleus	23.39	**96.37**
RETRO	greedy	19.75	21.15
	nucleus	22.87	91.09
CoG	greedy	24.68	40.45
	nucleus	**26.97**	90.00



🤦‍♀️BERT ↔️ GPT

Humen Evaluation

Comparison	Better	No Prefer.	Worse
CoG vs. Transformer	**48%**	24%	28%

Comparison	Better	No Prefer.	Worse
CoG vs. Transformer w/ FT	**52%**	12%	36%

Inference Speed

Model	Decoding	MAUVE↑	Rep-2↓	Rep-3↓	Rep-4 ↓	Diversity↑	Latency (s)↓
Transformer	greedy	19.87	43.56	38.55	35.5	22.37	1.32
	nucleus	23.43	5.10	1.33	0.50	93.22	1.48
kNN-LM	greedy	19.92	43.79	38.76	35.69	22.13	10.36
	nucleus	22.50	**3.33**	**0.69**	**0.21**	**95.8**	10.42
RETRO	greedy	21.19	44.65	39.63	36.6	21.19	4.39
	nucleus	22.86	6.21	1.93	0.86	91.19	4.51
CoG	greedy	26.01	28.14	23.80	21.40	43.03	**1.29**
	nucleus	**26.14**	7.31	2.66	1.28	89.07	1.54

Inference Speed

Model	Decoding	MAUVE↑	Rep-2↓	Rep-3↓	Rep-4 ↓	Diversity↑	Latency (s)↓
Transformer	greedy	19.87	43.56	38.55	35.5	22.37	1.32
	nucleus	23.43	5.10	1.33	0.50	93.22	1.48
kNN-LM	greedy	19.92	43.79	38.76	35.69	22.13	10.36
	nucleus	22.50	**3.33**	**0.69**	**0.21**	**95.8**	10.42
RETRO	greedy	21.19	44.65	39.63	36.6	21.19	4.39
	nucleus	22.86	6.21	1.93	0.86	91.19	4.51
CoG	greedy	26.01	28.14	23.80	21.40	43.03	**1.29**
	nucleus	**26.14**	7.31	2.66	1.28	89.07	1.54

Output Layer: Conclusion

	通用性	推理效率	隐私保护	存储成本	特点
🎯Input	🥈 **适用API** **最大长度**	🥉 **一次检索** **样本需再编码**	🥉 **检索返回明文**	🏅 **向量+文本**	**Top-K位置敏感**
🎯hidden	🥈 **无长度约束** **不适用API**	🥈 **样本无再编码** **逐词/层/头检索**	🏅 **返回Topk向量**	🥈 **逐词/层/头向量**	**PEFT**
🎯output					

Output Layer: Conclusion

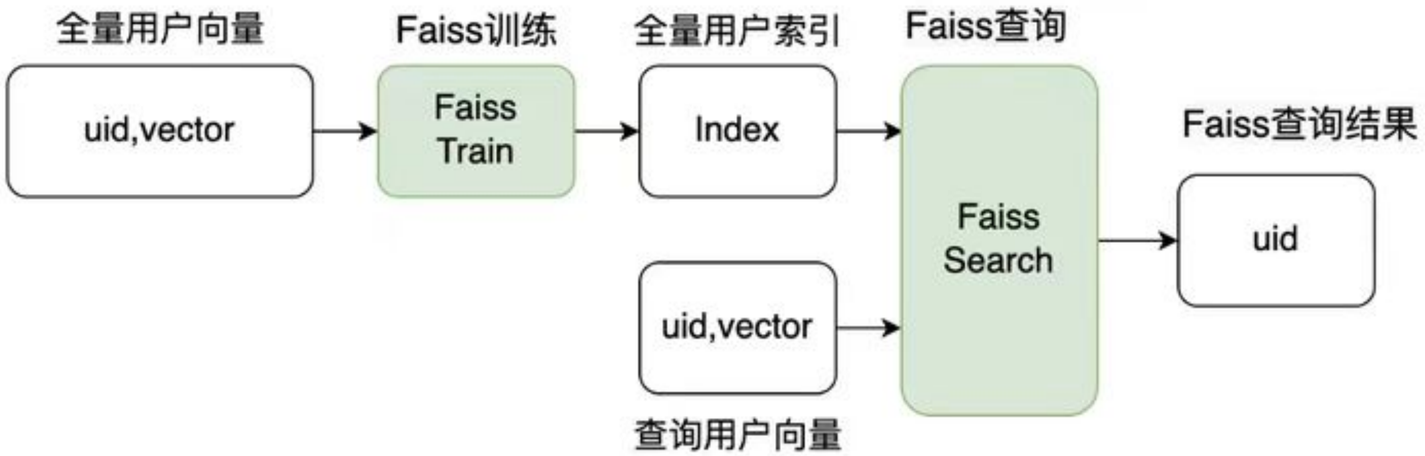
	通用性	推理效率	隐私保护	存储成本	特点
🎯Input	🥈 **适用API** 最大长度	🥉 **一次检索** 样本需再编码	🥉 检索返回明文	🏅 **向量+文本**	Top-K位置敏感
🎯hidden	🥈 **无长度约束** 不适用API	🥈 **样本无再编码** 逐词/层/头检索	🏅 **返回Topk向量**	🥈 逐词/层/头向量	PEFT
🎯output	**只需输出层**	**样本无再编码** **缩短长度**	**返回Topk向量**	**向量+文本**	拷贝→引用/解释

Output Layer: Conclusion

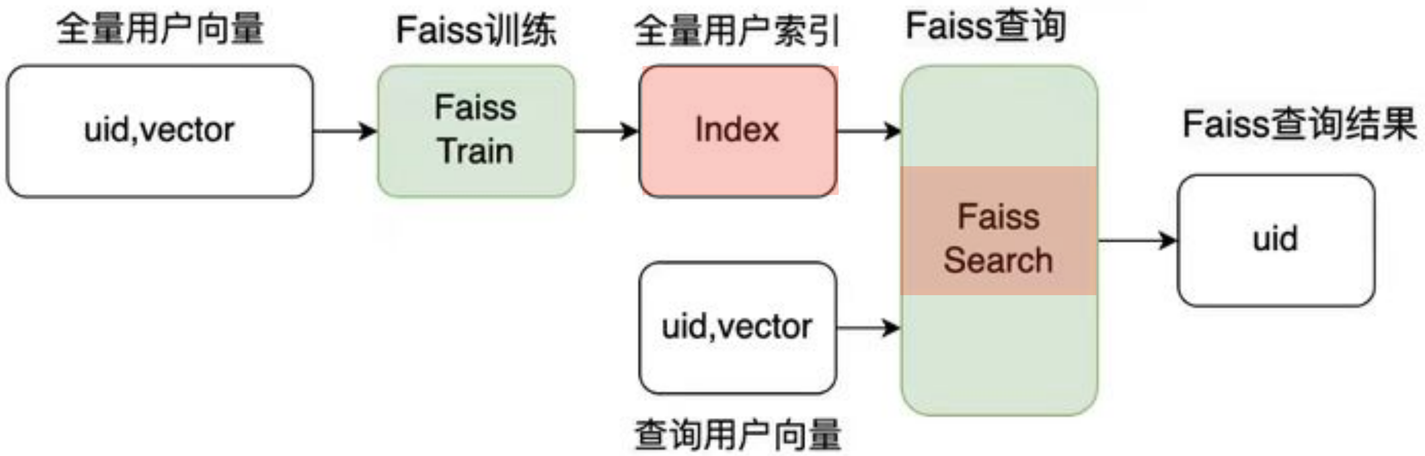
	通用性	推理效率	隐私保护	存储成本	特点
🎯Input	🥈 **适用API** 最大长度	🥉 **一次检索** 样本需再编码	🥉 检索返回明文	🥇 **向量+文本**	Top-K位置敏感
🎯hidden	🥈 **无长度约束** 不适用API	🥈 **样本无再编码** 逐词/层/头检索	🥇 **返回Topk向量**	🥈 逐词/层/头向量	PEFT
🎯output	🥇 **只需输出层**	🥇 **样本无再编码** **缩短长度**	🥇 **返回Topk向量**	🥇 **向量+文本**	拷贝→引用/解释

Q & A

Faiss

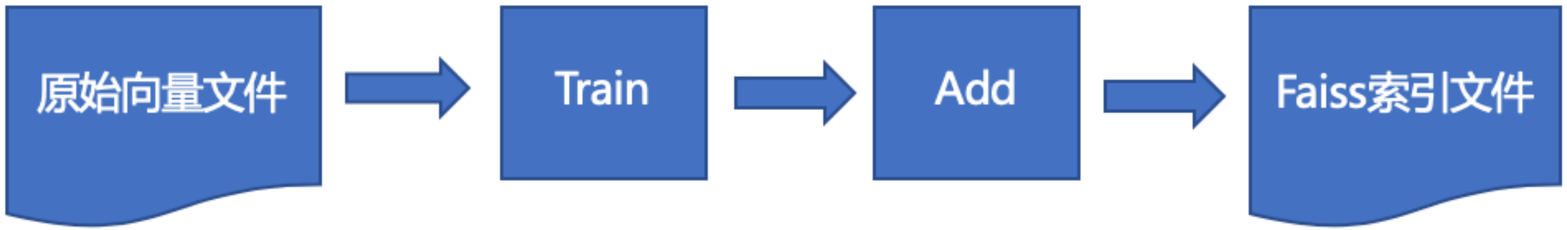

全量用户向量
uid,vector
Faiss训练
Faiss
Train
全量用户索引
Index
Faiss查询
Faiss
Search
Faiss查询结果
uid
uid,vector
查询用户向量

Faiss

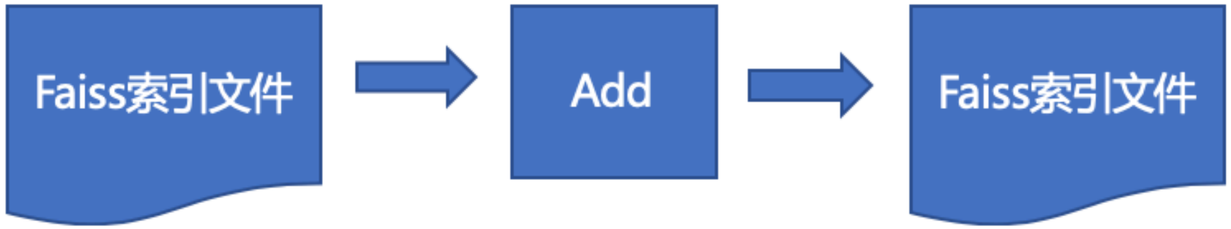

全量用户向量
uid,vector
Faiss训练
Faiss
Train
全量用户索引
Index
Faiss查询
Faiss
Search
Faiss查询结果
uid
uid,vector
查询用户向量

Faiss

全量构建索引：



增量构建索引：



Search: Dist

原生支持

- INNER_PRODUCT
- L1, L2, …, Lp
- Canberra
- BrayCurtis
- JensenShannon

可转换

- Cosine similarity
- Mahalanobis

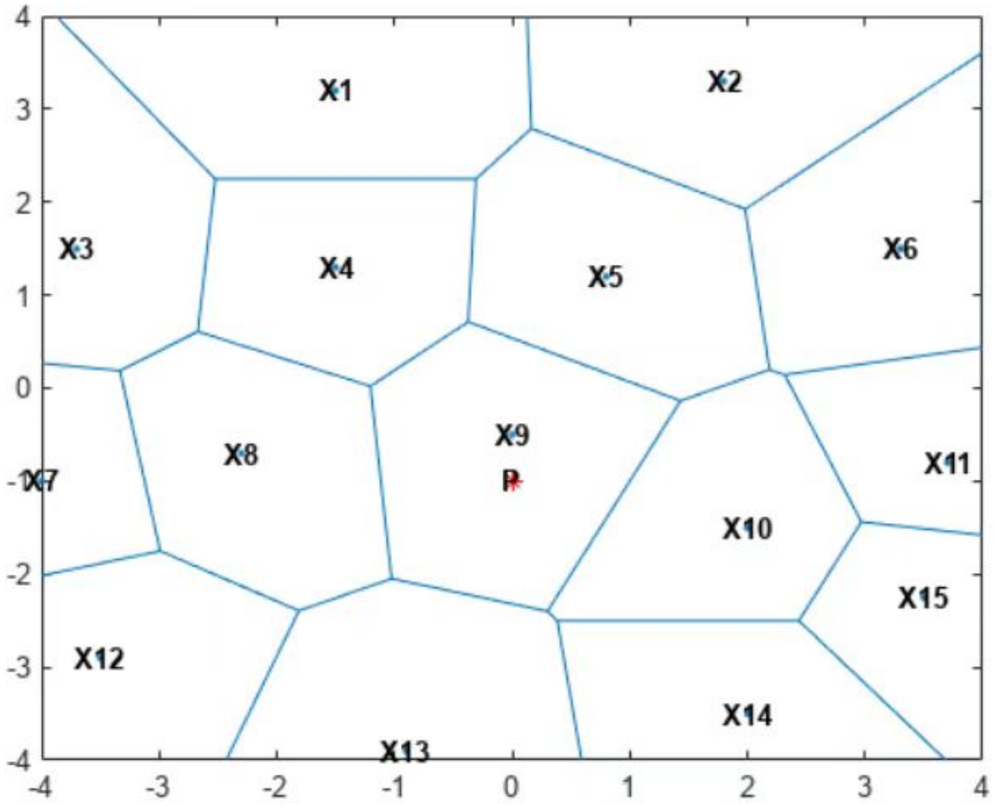
Faiss: 暴力查询IndexFlatL2

Faiss: 查询优化IndexIVFFlat

🎯聚类→减少查询数量（c+n/c, nprobe）

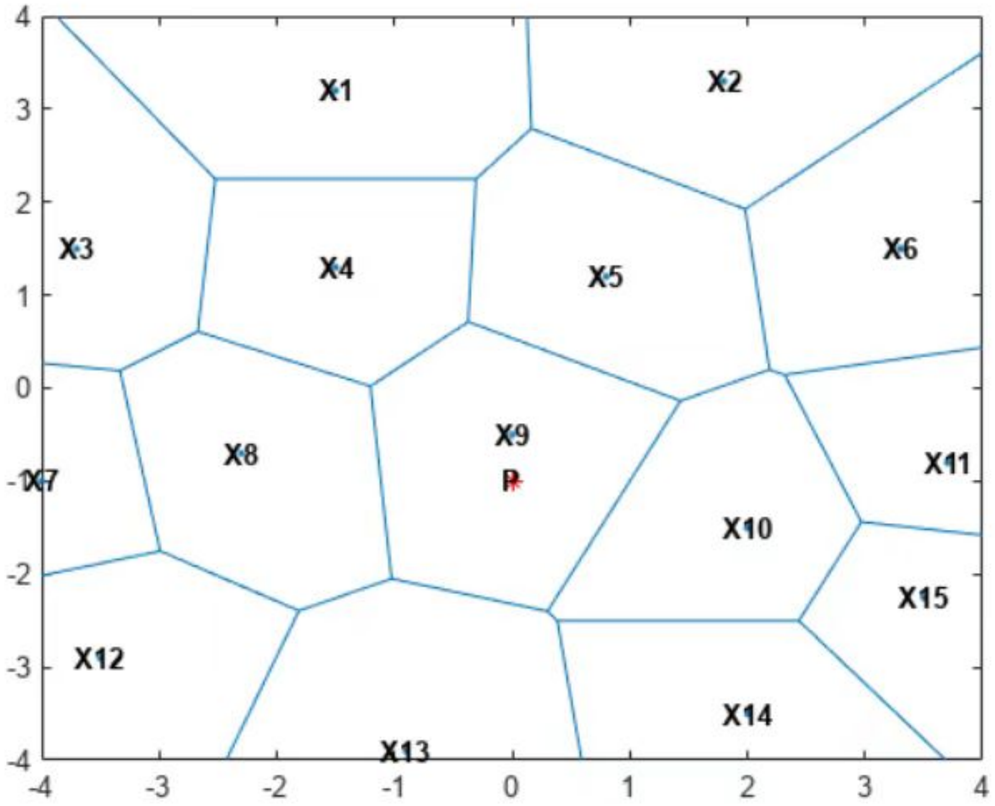
Faiss: 查询优化IndexIVFFlat

🎯聚类→减少查询数量（c+n/c, nprobe）



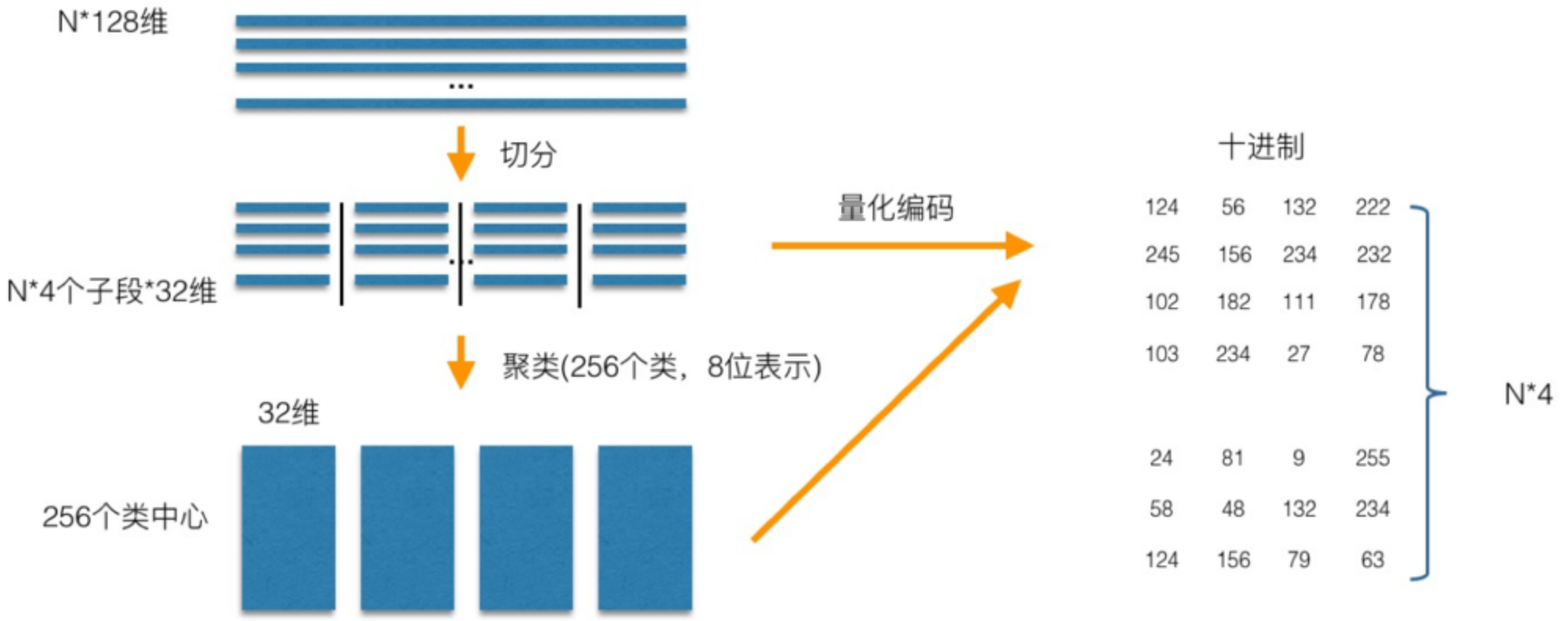
Faiss: 查询优化IndexIVFFlat

🎯聚类→减少查询数量（c+n/c, nprobe）



[4*sqrt(N), 16*sqrt(N)] → [30K, 256K]

Faiss: 存储/计算优化IndexPQ



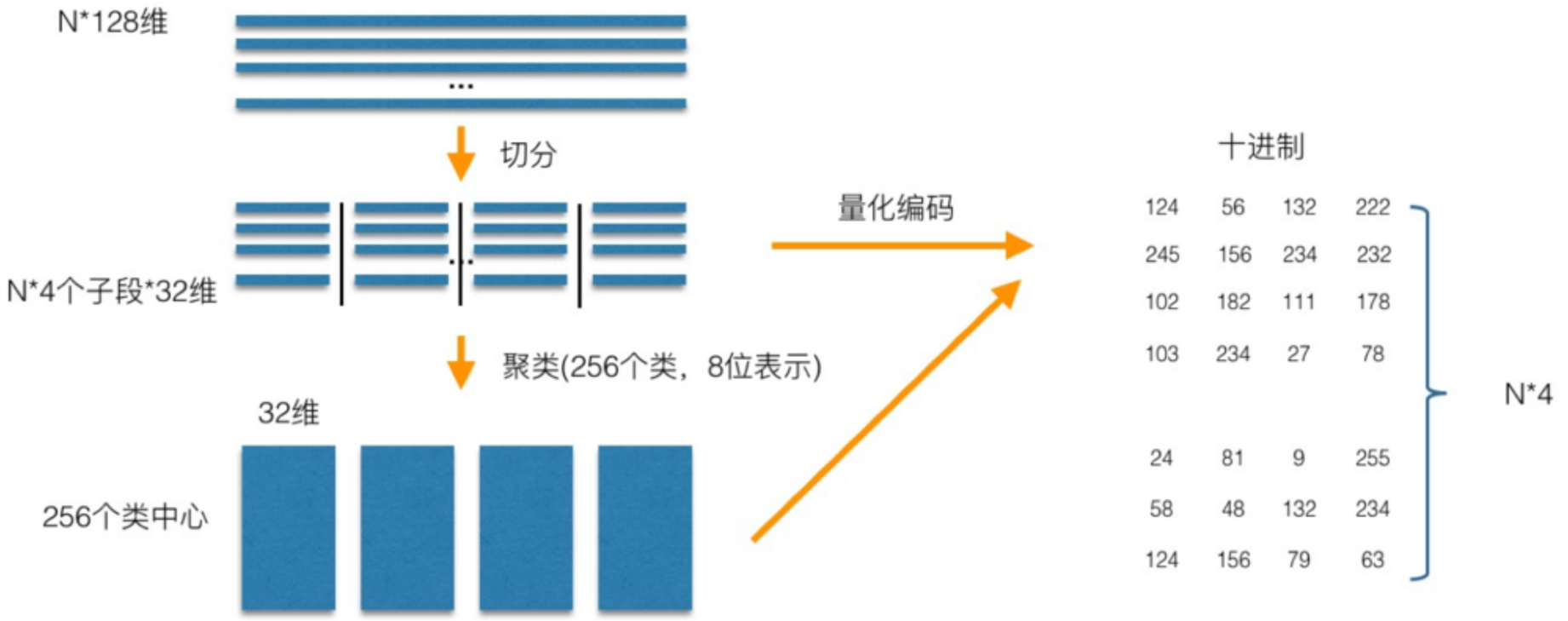
PQ

4x256次128维向量距离计算；

暴力

N次128维向量距离计算。

Faiss: 存储/计算优化IndexPQ



PQ

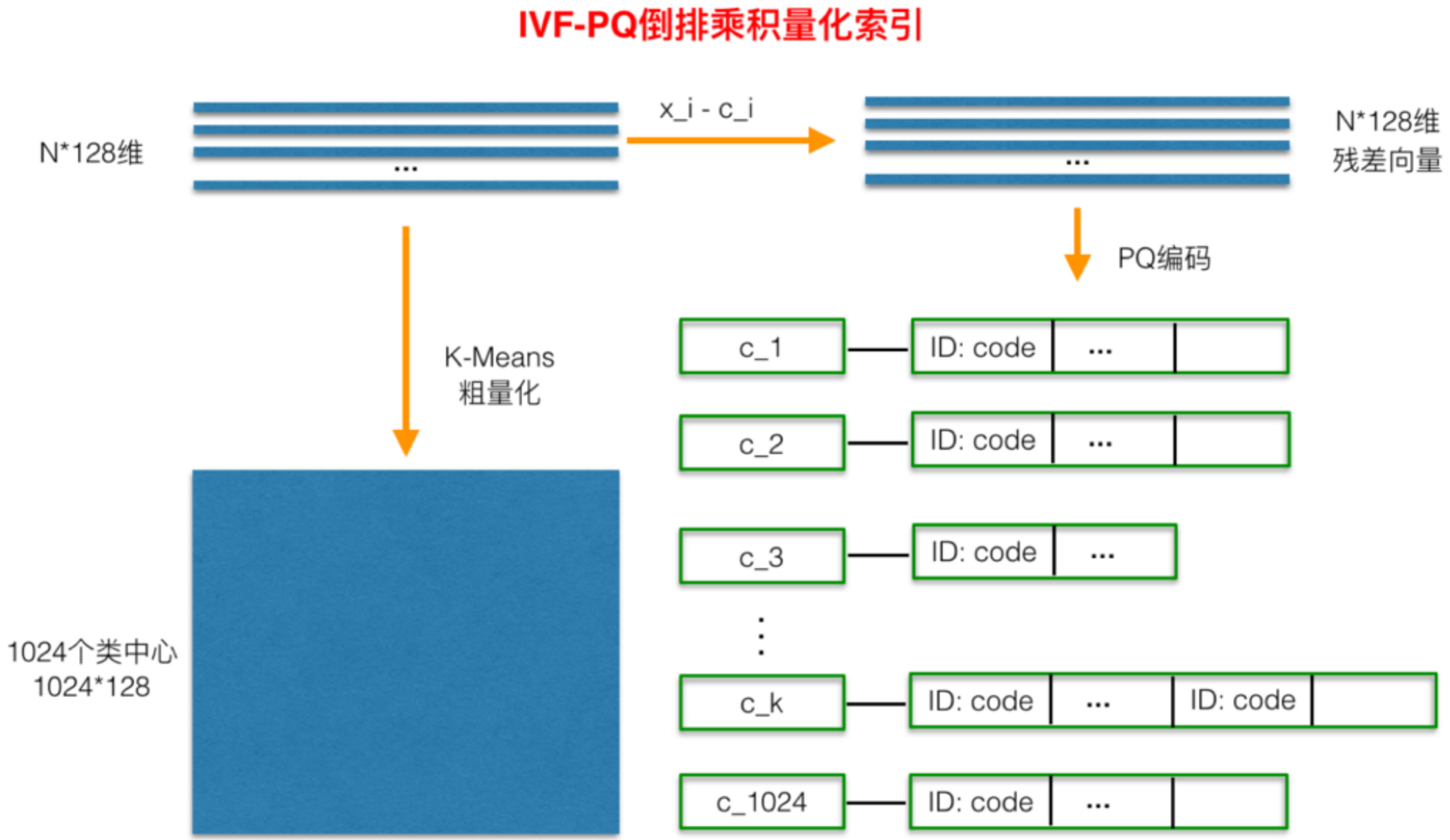
4x256次128维向量距离计算；

暴力

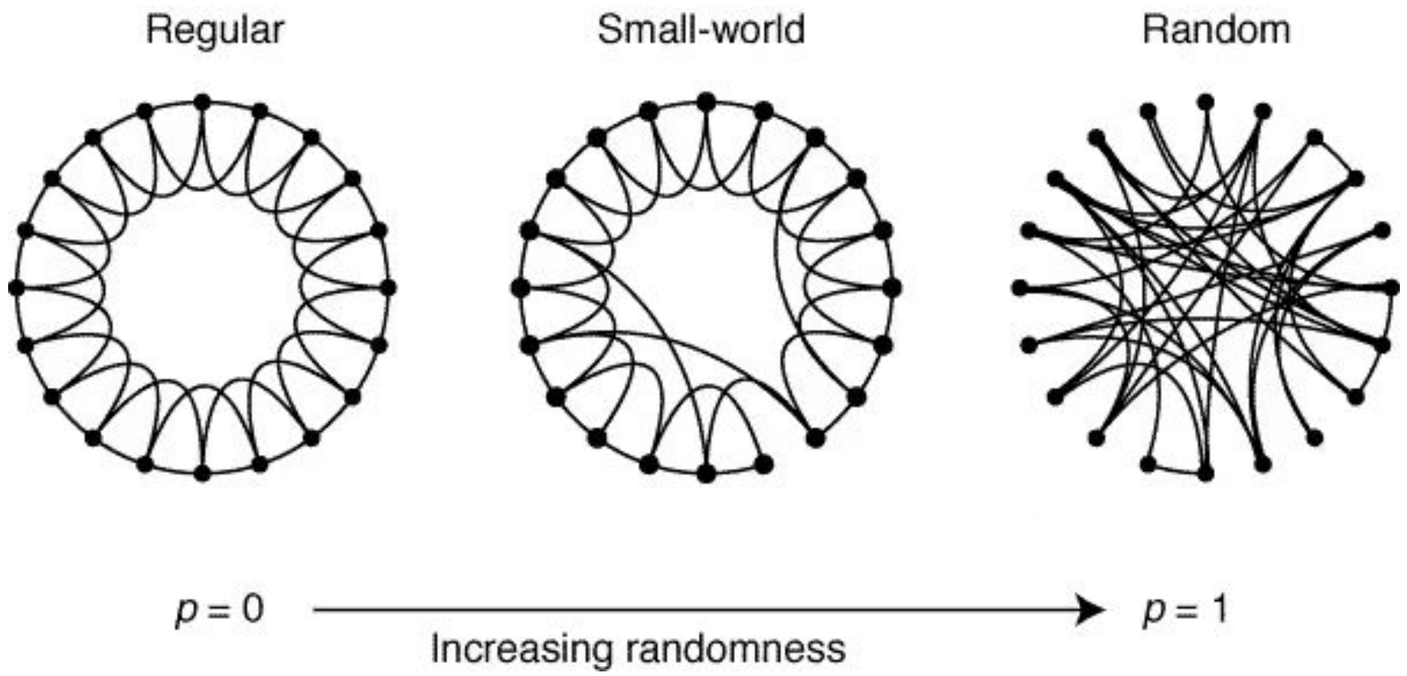
N次128维向量距离计算。

M = 64

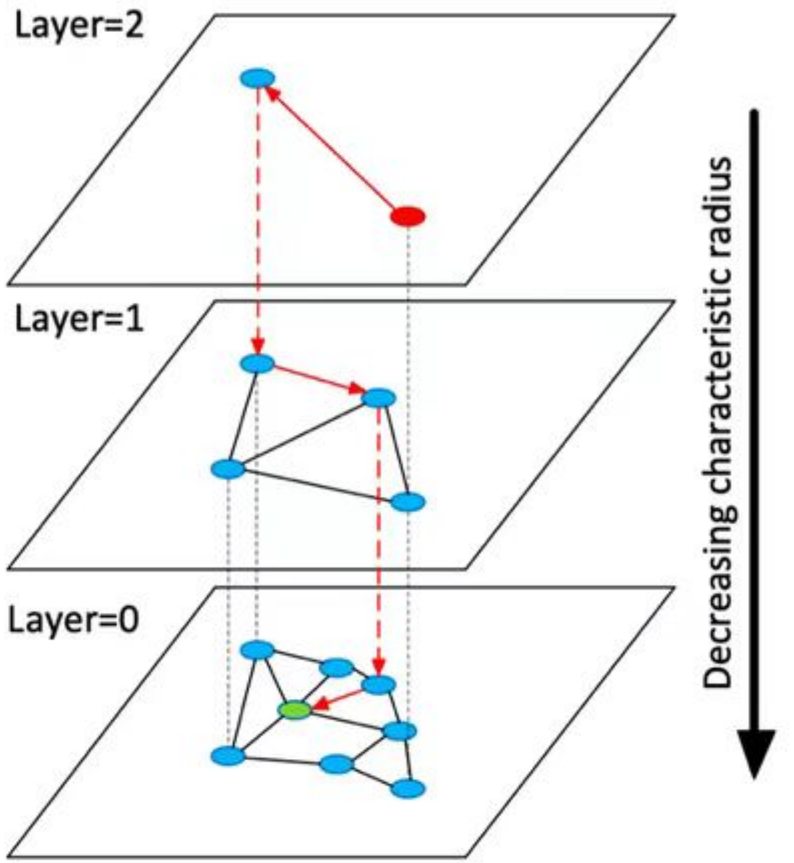
Faiss:存询优化IndexIVFPQ



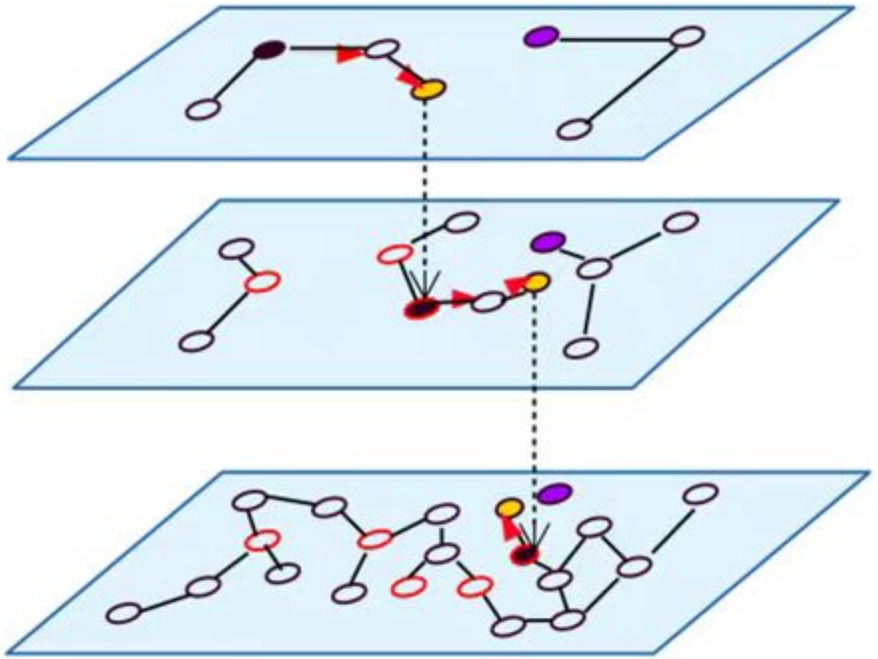
Faiss: 图索引IndexHNSWFlat



Faiss: 图索引IndexHNSWFlat



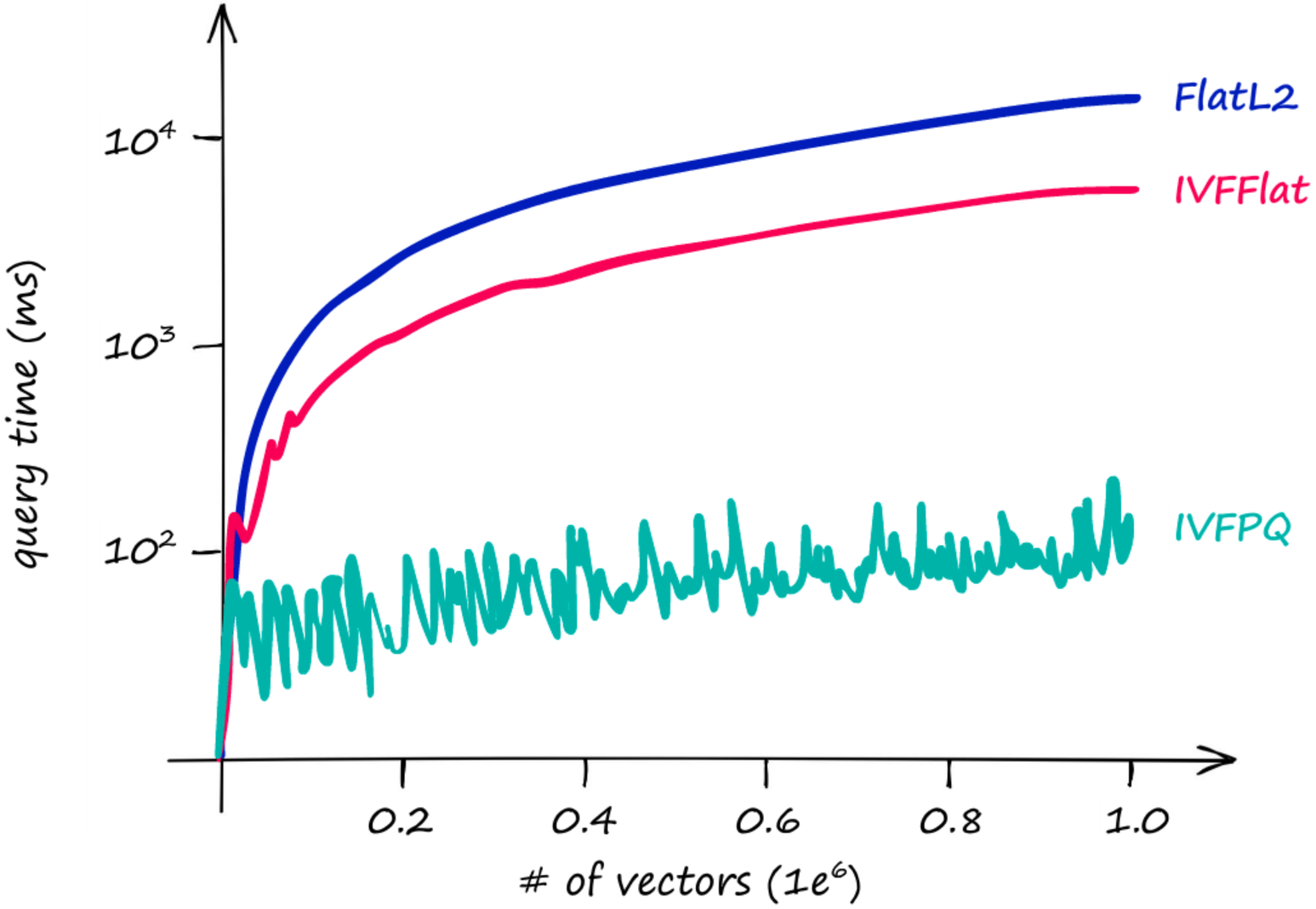
Faiss: 图索引IndexHNSWFlat



Faiss: PQ vs HNSW

特性	OPQ	HNSW
内存占用	小	大
召回率	较高	高
数据动态增删	灵活	不易

Faiss:存询优化IndexIVFPQ@GPU



1e5 query

Faiss:存询优化IndexIVFPQ@M1CPU

Index	Memory (MB)	Query Time (ms)	Recall	Notes
Flat	~500	18	1.0	适用于查询时间要求不高的小数据集
IVF	~520	1 - 9	0.7 - 0.95	一个扩展性比较高的方案
LSH	20 - 600	1.7 - 30	0.4 - 0.85	对于低维向量最好的方案
HNSW	600 - 1600	0.6 - 2.1	0.5 - 0.95	适用于对于精度和速度要求高的场景，但是费内存

d=128，N=1M，k=10

Retrieval Cost

Retrieval Cost

Retrieval n vectors → Add m tokens

Retrieval Cost

Retrieval n vectors → Add m tokens

params	dimension	n heads	n layers
6.7B	4096	32	32
13.0B	5120	40	40
32.5B	6656	52	60
65.2B	8192	64	80

Retrieval Cost: Llama-13B

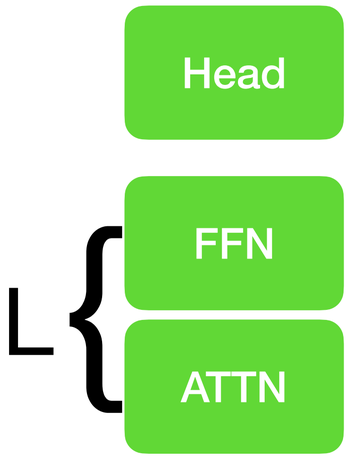
Retrieval n vectors → Add m tokens

Input = 1024, add 1 token

Retrieval Cost: Llama-13B

Retrieval n vectors → Add m tokens

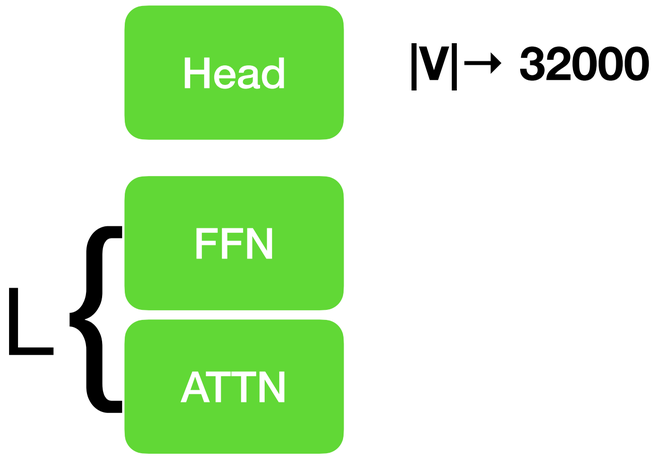
Input = 1024, add 1 token



Retrieval Cost: Llama-13B

Retrieval n vectors → Add m tokens

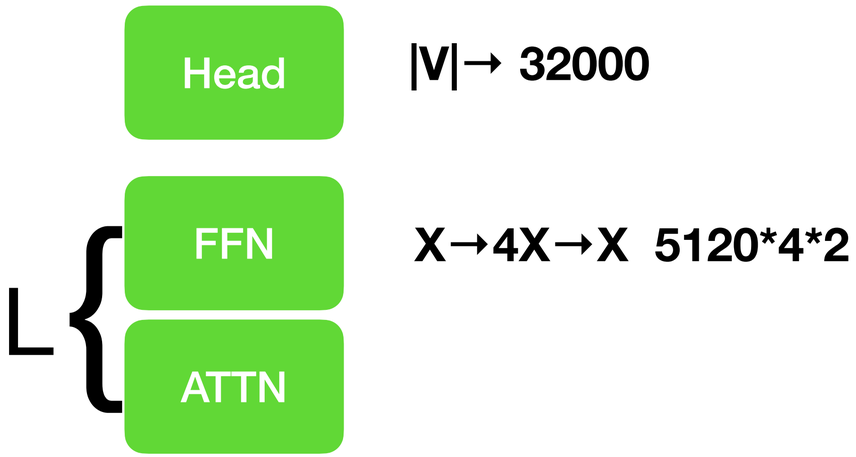
Input = 1024, add 1 token



Retrieval Cost: Llama-13B

Retrieval n vectors → Add m tokens

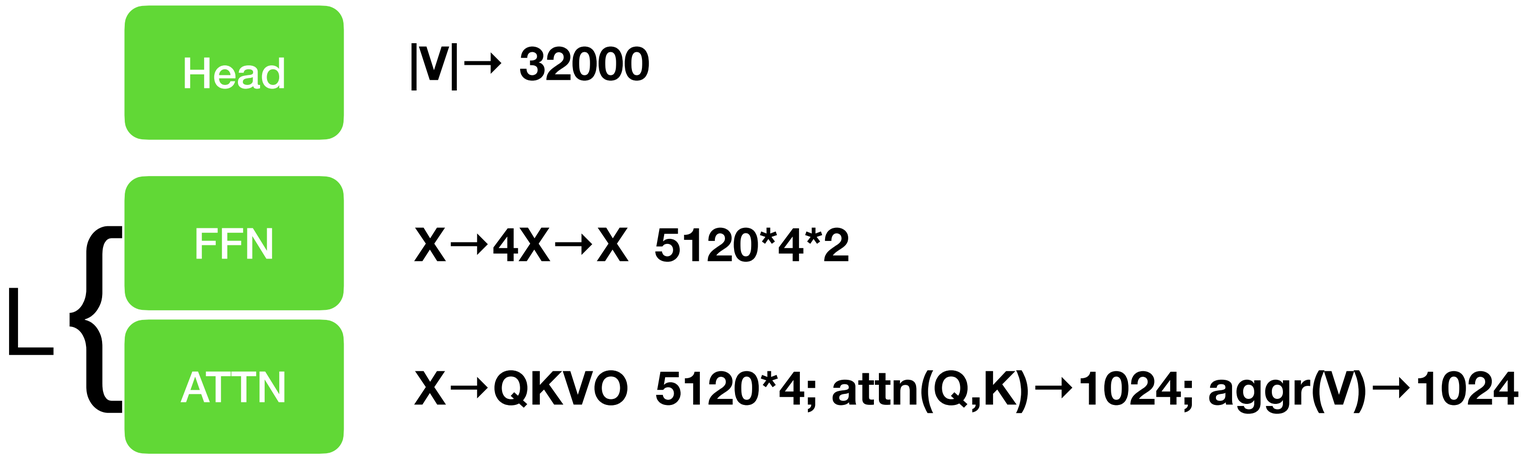
Input = 1024, add 1 token



Retrieval Cost: Llama-13B

Retrieval n vectors → Add m tokens

Input = 1024, add 1 token



Retrieval Cost: Llama-13B

Retrieval n vectors → Add m tokens

Input = 1024, add 1 token

Head — **|V|→ 32000**

L { FFN — **X→4X→X 5120*4*2**

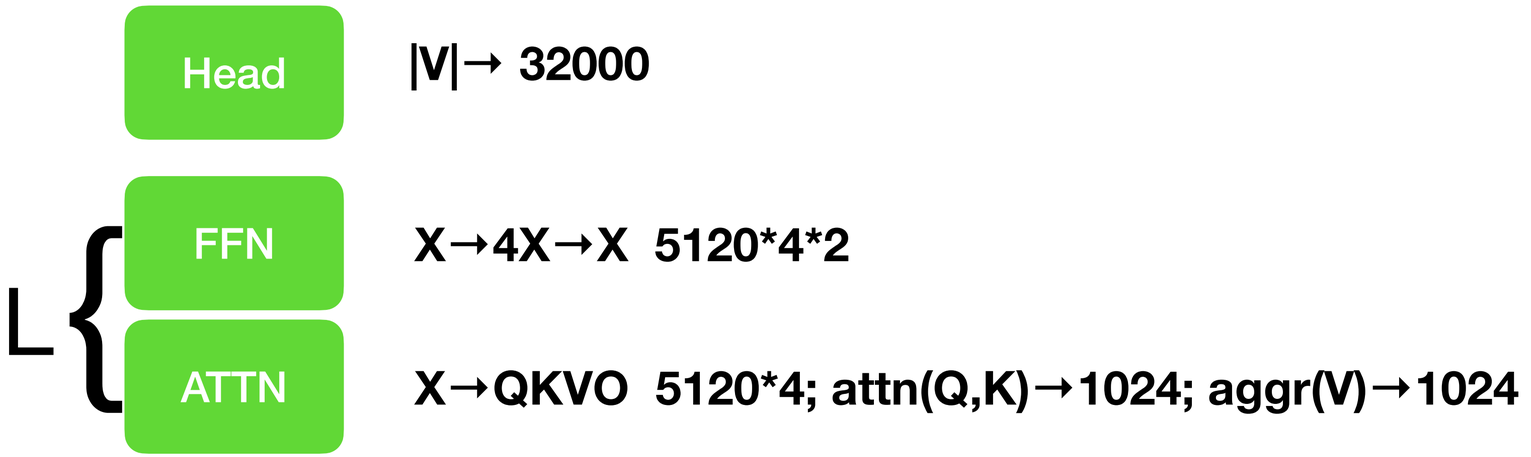
L { ATTN — **X→QKVO 5120*4; attn(Q,K)→1024; aggr(V)→1024**

> 2571520

Retrieval Cost: Llama-13B

Retrieval n vectors → Add m tokens

Input = 1024, add 1 token



> 2571520

- CoG: 动态词表size？ 50257 + 950942（k=1024）
- 检索3个token等于长度上增加1个token

Retrieval Cost: Llama-13B

Retrieval n vectors → Add m tokens

Input = 1024, add 1 token

Head

|V|→ 32000

L{

FFN

X→4X→X 5120*4*2

MoE 👉 向量近似检索

ATTN

X→QKVO 5120*4; attn(Q,K)→1024; aggr(V)→1024

> 2571520

- CoG: 动态词表size？ 50257 + 950942（k=1024）
- 检索3个token等于长度上增加1个token

Similarity

检索负例🤔　向量距离远→负例?

条件检索：C-STS

Similarity

检索负例🤔 向量距离远→负例?

条件检索：C-STS

C-STS: Conditional Semantic Textual Similarity

Ameet Deshpande[⋆1,2] **Carlos E. Jimenez**[⋆1] **Howard Chen**[1]
Vishvak Murahari[1] **Victoria Graf**[1] **Tanmay Rajpurohit**[3]
Ashwin Kalyan[2] **Danqi Chen**[1] **Karthik Narasimhan**[1]

[1]Princeton University [2]The Allen Institute for AI [3]Georgia Tech
asd@cs.princeton.edu

Similarity

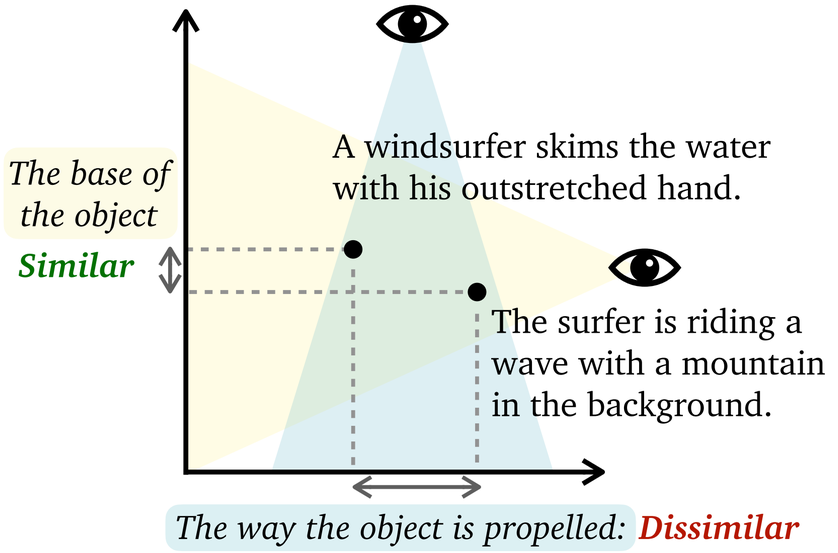
检索负例🤔 向量距离远→负例?

条件检索：C-STS

C-STS: Conditional Semantic Textual Similarity

Ameet Deshpande[⋆1,2] **Carlos E. Jimenez**[⋆1] **Howard Chen**[1]
Vishvak Murahari[1] **Victoria Graf**[1] **Tanmay Rajpurohit**[3]
Ashwin Kalyan[2] **Danqi Chen**[1] **Karthik Narasimhan**[1]

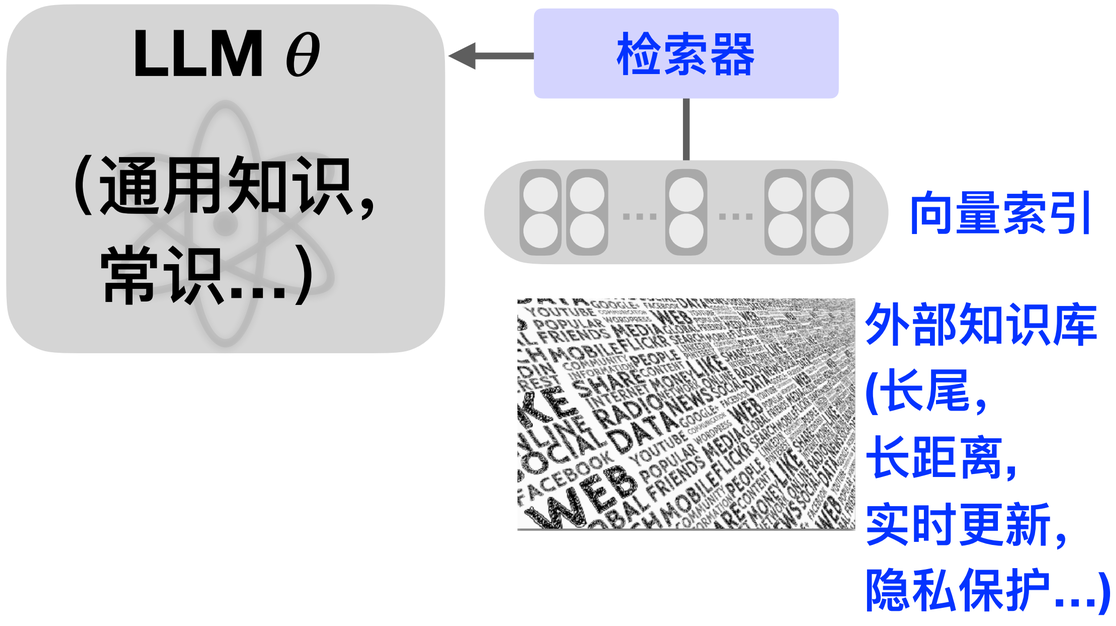
[1]Princeton University [2]The Allen Institute for AI [3]Georgia Tech
asd@cs.princeton.edu



基于向量检索增强的LLM

博士后 纪焘

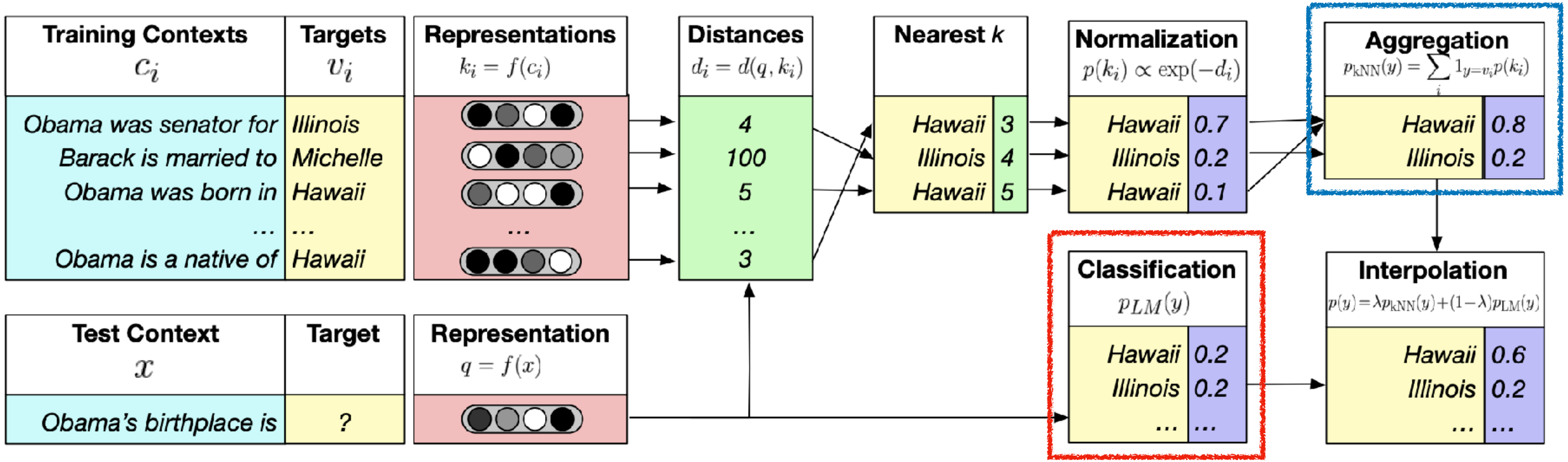
A5029
taoji@fdu.edu.cn



检索+LLM框架

1. Input
2. hidden
3. output

KNN-LM



λ: hyperparameter

$$P_{k\text{NN-LM}}(y \mid x) = (1 - \lambda) P_{\text{LM}}(y \mid x) + \lambda P_{k\text{NN}}(y \mid x)$$