南京航空航天大學
NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

BGE Landmark Embedding: A Chunking-Free Embedding Method For Retrieval Augmented Long-Context Large Language Models

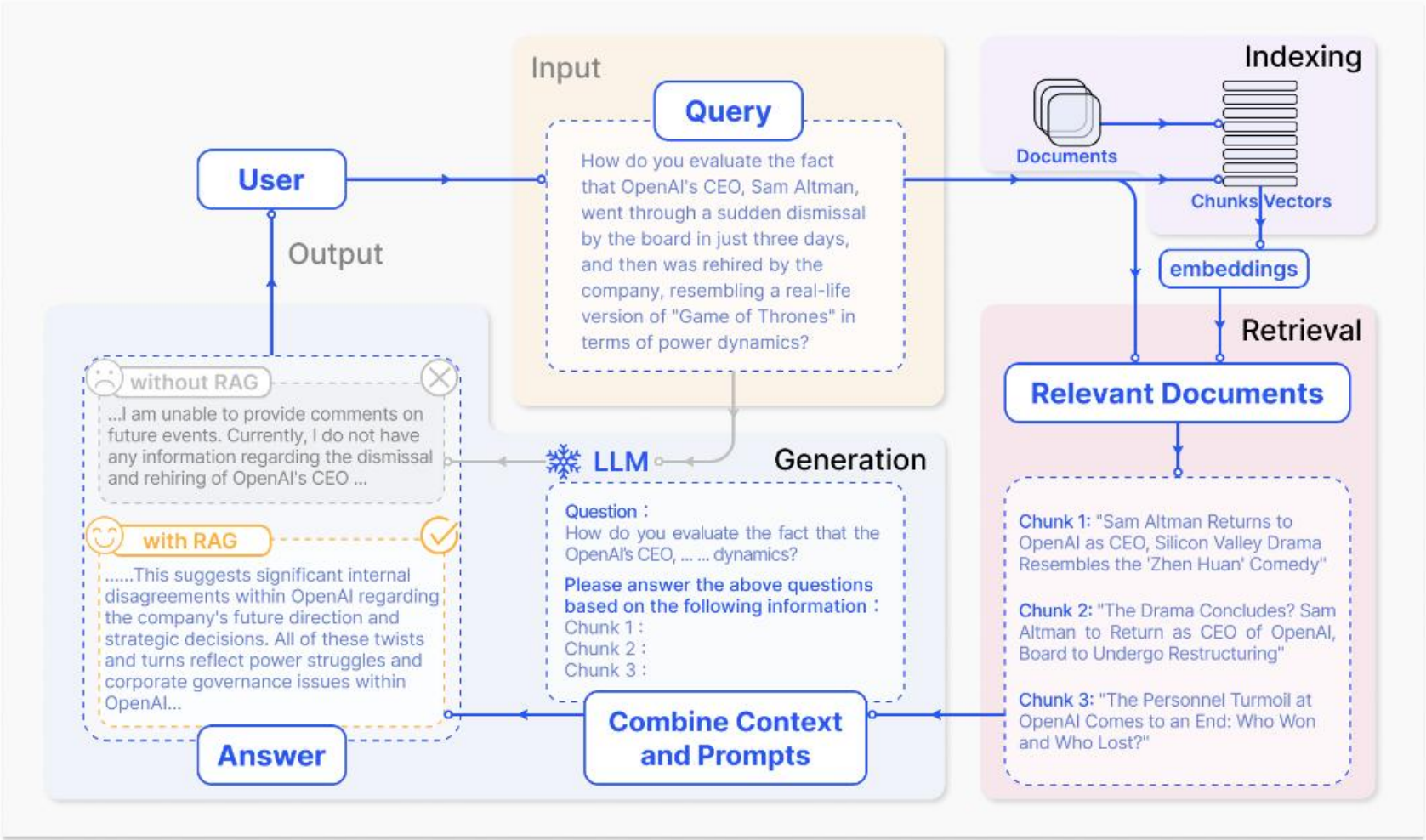
Luo Kun[1,2] **Zheng Liu**[1†] **Shitao Xiao**[1] **Kang Liu**[2]

1: Beijing Academy of Artificial Intelligence 2: Chinese Academy of Sciences

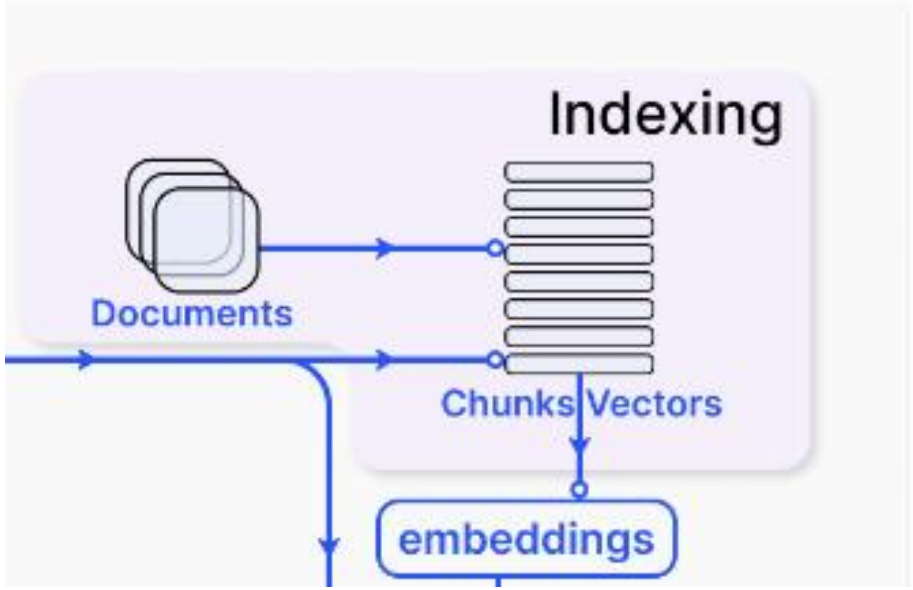
背景

大型语言模型（LLMs）在自然语言处理中扮演着重要角色，但当它们需要处理长序列输入（如问答和阅读理解任务）时会遇到上下文窗口限制的问题。

背景

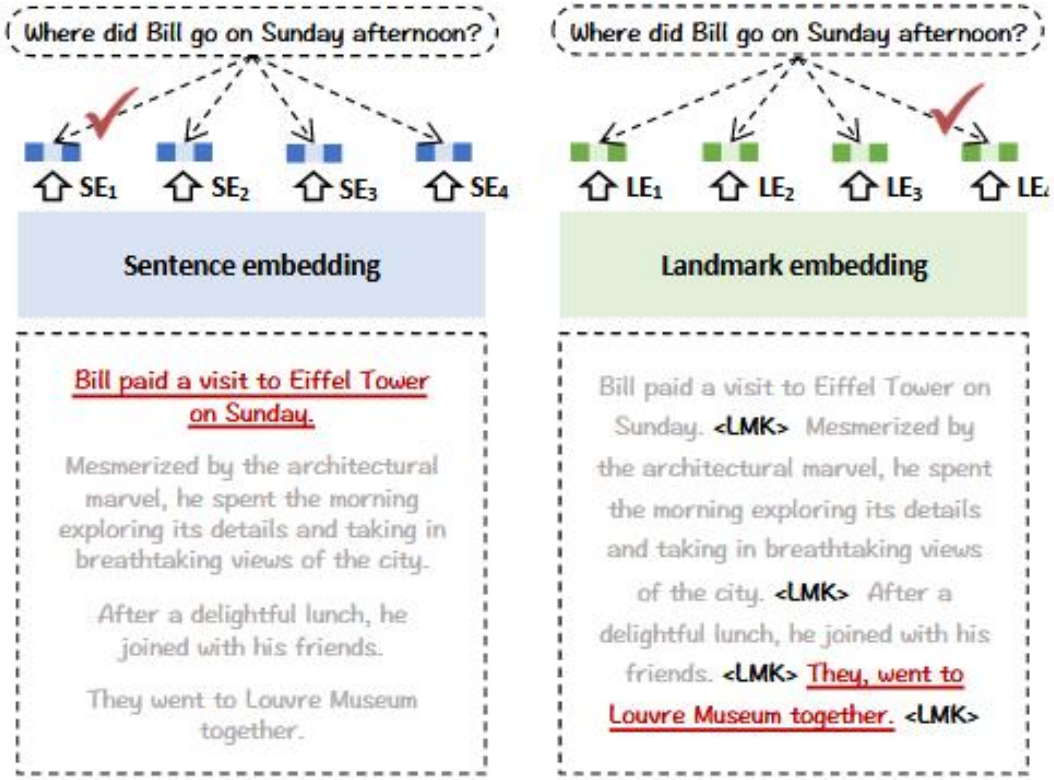

Input
Query
How do you evaluate the fact that OpenAI's CEO, Sam Altman, went through a sudden dismissal by the board in just three days, and then was rehired by the company, resembling a real-life version of "Game of Thrones" in terms of power dynamics?
User
Output
Indexing
Documents
Chunks Vectors
embeddings
Retrieval
Relevant Documents
Chunk 1: "Sam Altman Returns to OpenAI as CEO, Silicon Valley Drama Resembles the 'Zhen Huan' Comedy"
Chunk 2: "The Drama Concludes? Sam Altman to Return as CEO of OpenAI, Board to Undergo Restructuring"
Chunk 3: "The Personnel Turmoil at OpenAI Comes to an End: Who Won and Who Lost?"
LLM
Generation
Question：
How do you evaluate the fact that the OpenAI's CEO, ... dynamics?
Please answer the above questions based on the following information：
Chunk 1：
Chunk 2：
Chunk 3：
Combine Context and Prompts
without RAG
...I am unable to provide comments on future events. Currently, I do not have any information regarding the dismissal and rehiring of OpenAI's CEO ...
with RAG
.....This suggests significant internal disagreements within OpenAI regarding the company's future direction and strategic decisions. All of these twists and turns reflect power struggles and corporate governance issues within OpenAI...
Answer

背景



现有的RAG方法通常需要将长文本拆分成多个小块（chunk），然后对每个块生成嵌入表示。

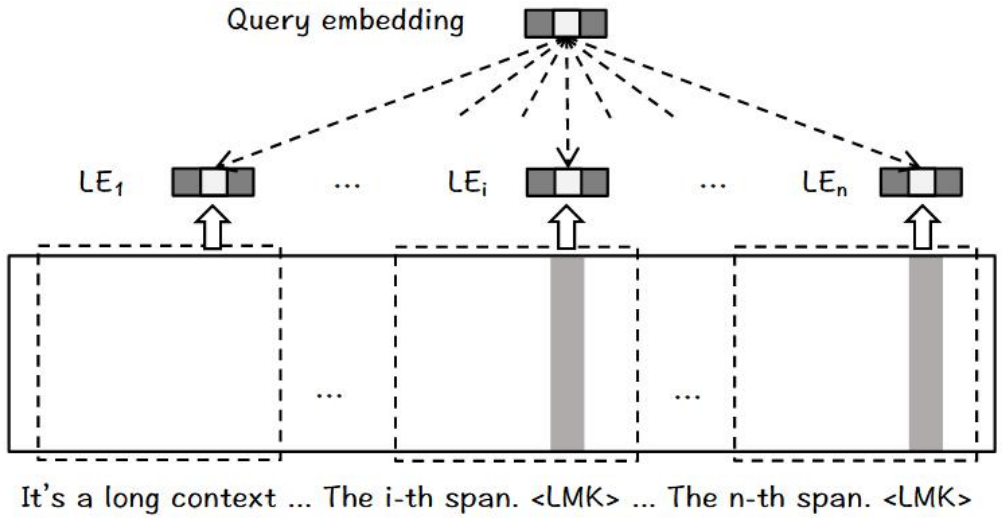
核心问题



上下文连贯性被破坏：
长文本在分块后被割裂成不连续的部分，导致上下文的完整性丧失，从而影响嵌入的质量。例如，当模型需要理解段落中的复杂关系或连续语义时，分块会使得这些语义信息被分散，导致嵌入表示的准确性降低。

信息不完整：
检索系统容易选择显著性较高的块，然而其他重要但不显著的块可能会被忽略。这样一来，模型可能无法获取完整的背景信息，影响检索的完整性。

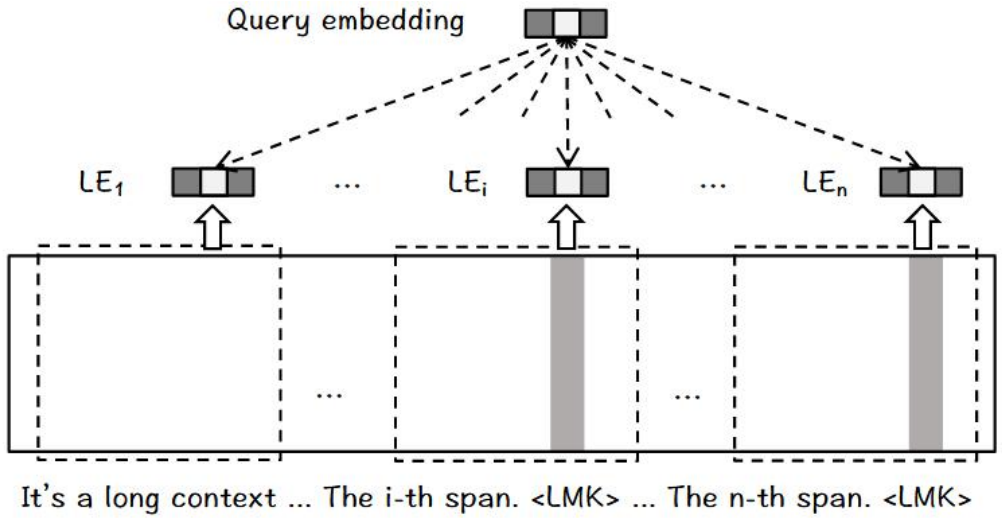
方法-无分块架构（Chunking-Free Architecture）



提出Landmark Embedding架构:
假设输入上下文由n个句子组成，它不是将输入上下文分块成不相连的片段，而是在每个句子的末尾调度一个特殊的标记，称为界标（LMK）。landmark用于捕获其对应句子的底层语义，与句子和相邻上下文联合编码,输出嵌入LE作为句子的嵌入表示

$$\mathrm{LE}_i \leftarrow \mathrm{LLM}(c1, ..., c_i; \mathrm{LMK}).embed[-1],$$
$$\mathrm{E}_q \leftarrow \mathrm{LLM}(query; \mathrm{LMK}).embed[-1].$$

方法-无分块架构（Chunking-Free Architecture）



滑动窗口：
为了处理超出大型语言模型（LLM）上下文窗口大小的长文本，Landmark Embedding采用了滑动窗口技术。通过这种方式，可以在保持上下文连贯性的同时，逐段处理长文本。

$$\mathrm{LE}_i \leftarrow \mathrm{LLM}(c_{i-l}, ..., c_i; \mathrm{LMK}).embed[-1],$$

方法-位置感知目标函数（Position-Aware Objective）

对于查询（query）和其相关句子（sentences）之间的相似性进行建模时，不仅要考虑句子本身的内容，还要考虑它们在上下文中的位置。

方法-位置感知目标函数（Position-Aware Objective）

$$min. -\sum_{q}\sum_{i\leq m}\lg\frac{\exp(\langle \mathrm{E}_q, \mathrm{LE}_{z-i}\rangle)}{\sum_{j=1\ldots n}\exp(\langle \mathrm{E}_q, \mathrm{LE}_j\rangle)}. \quad (4)$$

基础的对比学习损失函数

landmark的嵌入是通过对比学习来学习的，在这种学习中，查询及其相关句子可以通过更高的嵌入相似性来区分。

方法-位置感知目标函数（Position-Aware Objective）

在上述损失函数中，每个句子给予了同等重要性的积极标签，但它可能会倾向于最显著的句子

$$c_{z-i}\colon w_i \leftarrow \exp(-\alpha * i)$$

$$min. -\sum_{q}\sum_{i\leq m} \lg \frac{w_i * \exp(\langle \mathrm{E}_q, \mathrm{LE}_{z-i}\rangle)}{\sum_{j=1\ldots n}\exp(\langle \mathrm{E}_q, \mathrm{LE}_j\rangle)}. \quad (5)$$

我们旨在完全检索有用的信息，故引入位置权重改写损失函数

方法-多阶段学习算法（Multistage Learning Algorithm）

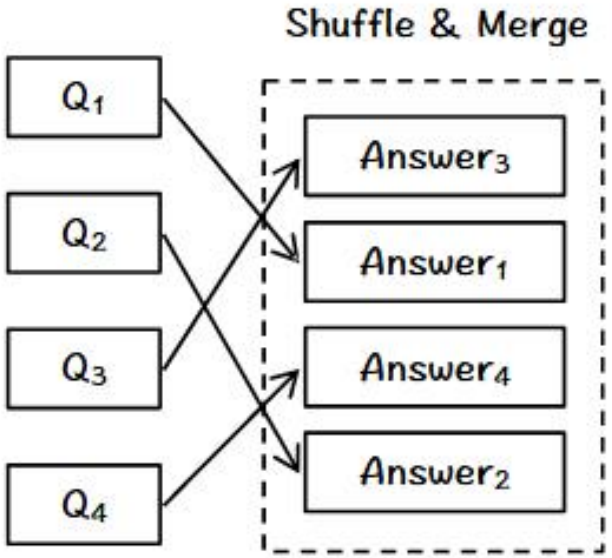
提出了一个多阶段的训练方法，利用不同的数据来源来提升Landmark Embedding的嵌入质量，使1)基本的语义区分能力、2)上下文表示能力这两种能力能够在适当的训练数据之上逐步建立：

方法-多阶段学习算法（Multistage Learning Algorithm）

1.**远程监督**：在第一阶段，使用成对的训练数据（如MS MARCO数据集）来初始化landmark嵌入模型。在这个阶段，模型被训练为基本的句子级嵌入器，通过硬负样本和批次内负样本来增强模型的性能。

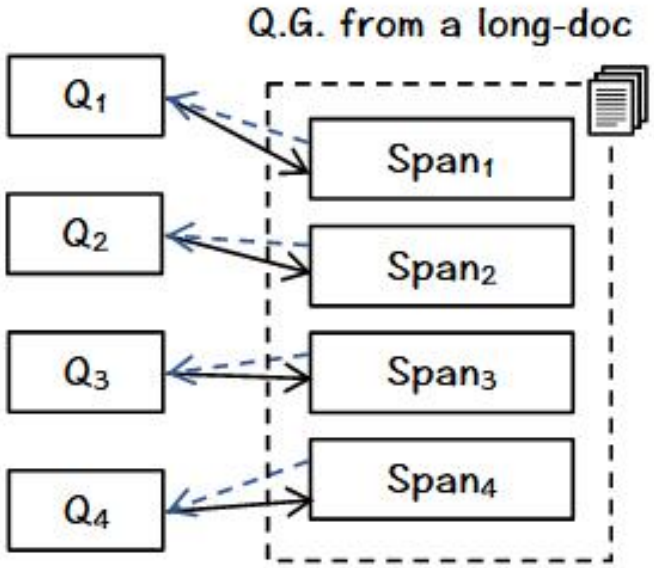
假设有一个问题：“哪一方赢得了这场战役？”，其对应的答案是“盟军赢得了这场战役。”
将答案与landmark输入到嵌入模型生成嵌入，并利用15个硬负样本和批次内的负样本进行对比学习

方法-多阶段学习算法（Multistage Learning Algorithm）



2.**弱监督**：第二阶段对成对训练数据进行修改，通过随机打乱不同查询的答案，并将它们合并为一个伪长文档，来训练模型在长上下文中生成区分性的句子嵌入。在这个阶段，模型继续使用批次内负样本进行训练。

方法-多阶段学习算法（Multistage Learning Algorithm）



3.**微调**：最后阶段使用合成数据进行微调。这些数据是通过从维基百科等真实世界长文档中随机采样文本片段，并使用ChatGPT-3.5 API生成相关问题来构建的。虽然合成数据的生成可能会产生额外的成本，并且可能与真实世界数据分布有所不同，但由于前两个阶段已经建立了基础能力，landmark嵌入在适度的微调后能够达到优越的性能。

实验设置

模型选择：作者选用了LLaMA-2-7B（上下文窗口4K）和ChatGPT-3.5-turbo（上下文窗口16K）作为实验的基准模型，分别评估它们在长文本任务中的性能。

数据集：实验在多个长上下文理解数据集上进行，包括NarrativeQA、Qasper、MultifieldQA、HotpotQA、2WikiMQA和MuSiQue等。这些数据集包含长文本的单文档和多文档问答任务，覆盖不同长度的上下文，使实验结果更具普遍性。

评价指标：采用F1得分来衡量检索增强方法的有效性。

实验

南京航空航天大學 NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

实验一：对比不同检索方法

LLM	Retriever	Unit	Len.	NQA	QASP	MFQA	HQA	2WIKI	MSQ	Avg.
Llama2-7B-chat	w/o retrieval	-	3,500	18.7	19.2	36.8	25.4	32.8	9.4	23.7
	Contriever	chunk	2,275	18.3	23.8	41.8	33.6	34.5	17.2	28.2
	OpenAI-2	chunk	2,275	20.0	25.7	40.3	34.7	34.4	17.3	28.7
	BGE-large	chunk	2,275	17.6	21.7	45.4	34.3	**36.9**	19.9	29.3
	$E5_{mistral\text{-}7b}$	chunk	2,275	**21.6**	24.1	42.2	37.6	31.4	20.7	29.6
	Contriever	sentence	2,190	16.2	26.5	44.4	33.5	33.3	17.5	28.6
	BGE-large	sentence	2,190	17.9	24.4	46.3	37.4	35.0	21.3	30.3
	$E5_{mistral\text{-}7b}$	sentence	2,190	16.5	24.0	47.3	37.6	35.4	**21.7**	30.4
	Ours	sentence	2,190	21.3	**27.7**	**47.6**	**40.2**	36.3	**21.7**	**32.5**
ChatGPT-3.5-turbo	w/o retrieval	-	15,500	**23.6**	**43.3**	52.3	51.6	37.7	26.9	39.2
	Contriever	chunk	2,275	18.3	35.6	54.3	47.0	39.5	25.2	36.6
	OpenAI-2	chunk	2,275	21.8	38.1	52.8	46.6	44.9	30.4	39.1
	BGE-large	chunk	2,275	21.9	37.2	49.1	49.5	42.2	30.4	38.4
	$E5_{mistral\text{-}7b}$	chunk	2,275	21.0	41.2	49.2	54.0	43.7	27.2	39.4
	Contriever	sentence	2,190	17.5	41.0	50.2	46.2	41.9	24.1	36.8
	BGE-large	sentence	2,190	19.8	41.2	51.3	50.5	**46.5**	29.6	39.8
	$E5_{mistral\text{-}7b}$	sentence	2,190	20.0	39.0	49.4	55.4	45.9	**31.1**	40.1
	Ours	sentence	2,190	22.3	42.7	**55.7**	**56.1**	46.2	29.5	**42.1**

实验二：消融实验

Train Objective	Retrieval	Unit	NQA	QASP	MFQA	HQA	2WIKI	MSQ	Avg.
w/o Position-aware	Surround-k	sentence	19.1	29.8	46.8	**40.2**	34.2	17.0	31.2
w/o Position-aware	Front-k	sentence	19.4	28.5	46.5	38.5	33.8	19.0	31.0
w. Position-aware	Surround-k	sentence	19.4	**29.7**	**47.9**	39.0	36.0	17.8	31.6
w. Position-aware*	Front-k	sentence	**21.3**	27.7	47.6	**40.2**	**36.3**	**21.7**	**32.5**
Stage I. only	Front-k	sentence	18.9	27.0	45.0	35.5	33.2	17.2	29.4
Stage II. only	Front-k	sentence	19.0	27.4	43.9	34.4	32.7	16.5	29.0
Stage III. only	Front-k	sentence	20.5	27.2	45.3	39.2	34.3	15.3	30.3
Stage I. + II.	Front-k	sentence	19.2	26.5	47.0	36.2	32.8	16.8	29.8
Stage II. + III.	Front-k	sentence	19.4	26.7	46.8	39.8	35.4	18.3	31.0
All three stages*	Front-k	sentence	**21.3**	**27.7**	**47.6**	**40.2**	**36.3**	**21.7**	**32.5**

Thanks