WIKI QA

BY MOHAMED ZAKARIA, ALI AMR & SARAH HELLA

OVERVIEW

SYSTEM ARCHITECTURE

TRAINING

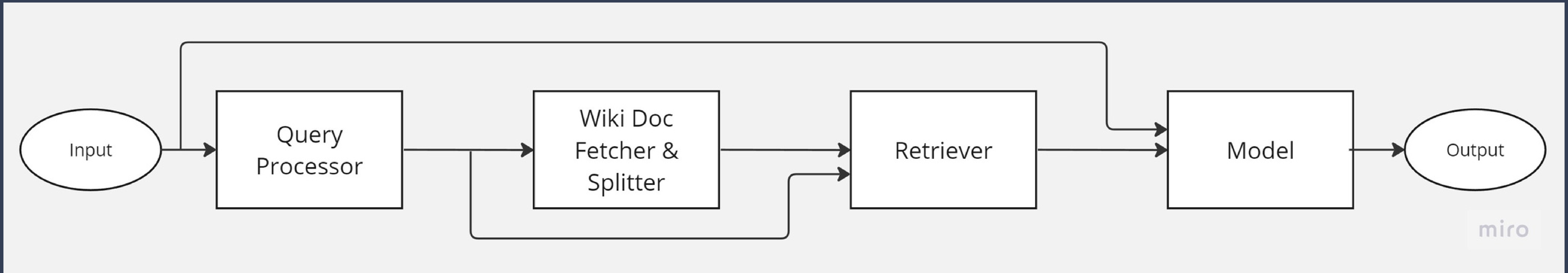
EVALUATION

EXPERIMENTS & RESULTS

LIMITATIONS

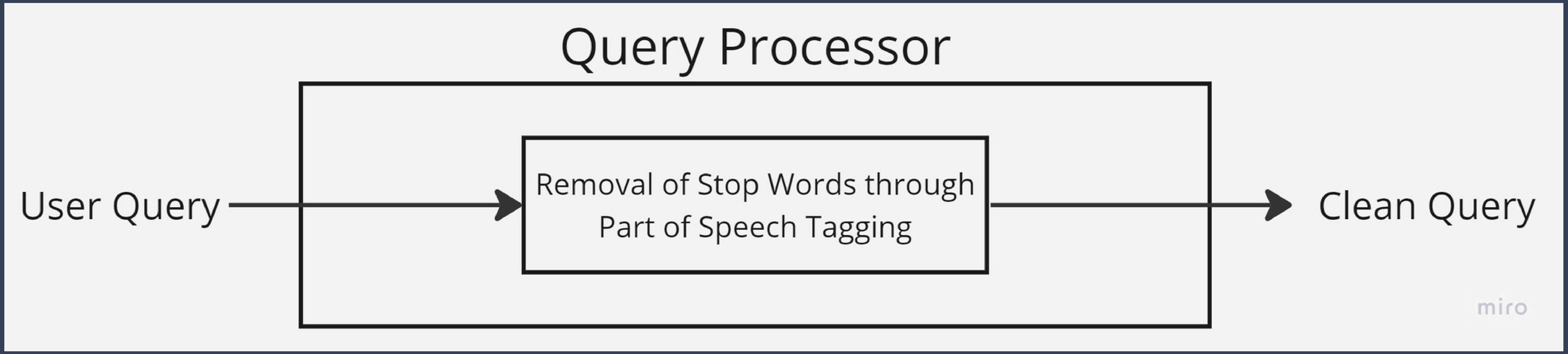
FUTURE WORK

SYSTEM ARCHITECTURE

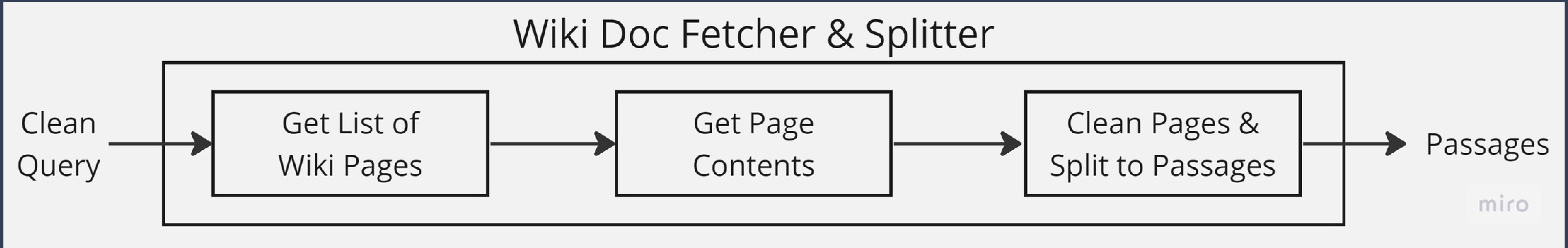


QUERY PROCESSOR

- We remove tokens with tags that are not proper noun, noun, adjective, verb or number
- Clean query is used to retrieve Wiki pages and candidate passages



WIKI DOC FETCHER & SPLITTER



Wiki Doc Fetcher & Splitter

Clean Query

Get List of Wiki Pages

Get Page Contents

Clean Pages & Split to Passages

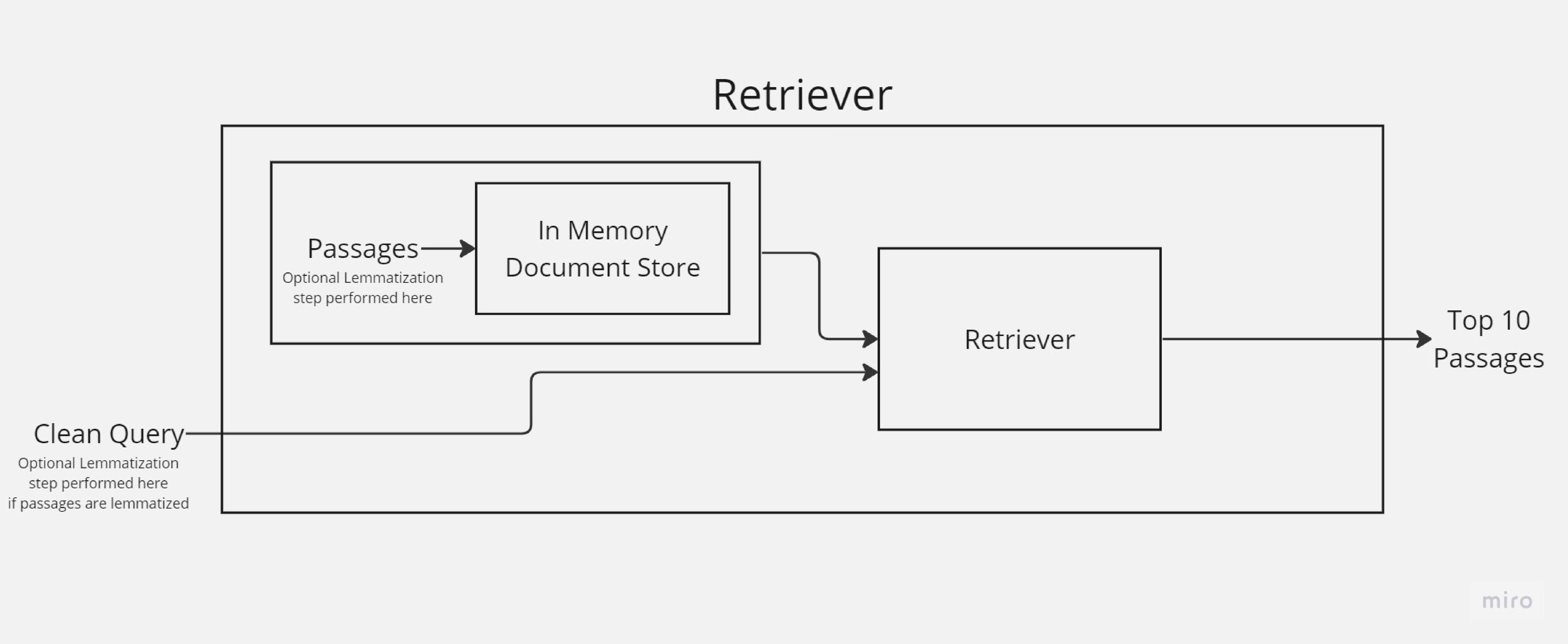
Passages

WIKI DOC FETCHER & SPLITTER

For each page the following is done

- Remove empty lines
- Remove white space
- Split into 200 Words passages while respecting sentence boundary
- 10 word Sliding window approach is used
- Removal of header & Footer
- Removal of Wikipedia hyperlinks such as "===References==="

RETRIEVER


Retriever
Passages
Optional Lemmatization
step performed here
In Memory
Document Store
Retriever
Top 10
Passages
Clean Query
Optional Lemmatization
step performed here
if passages are lemmatized
miro

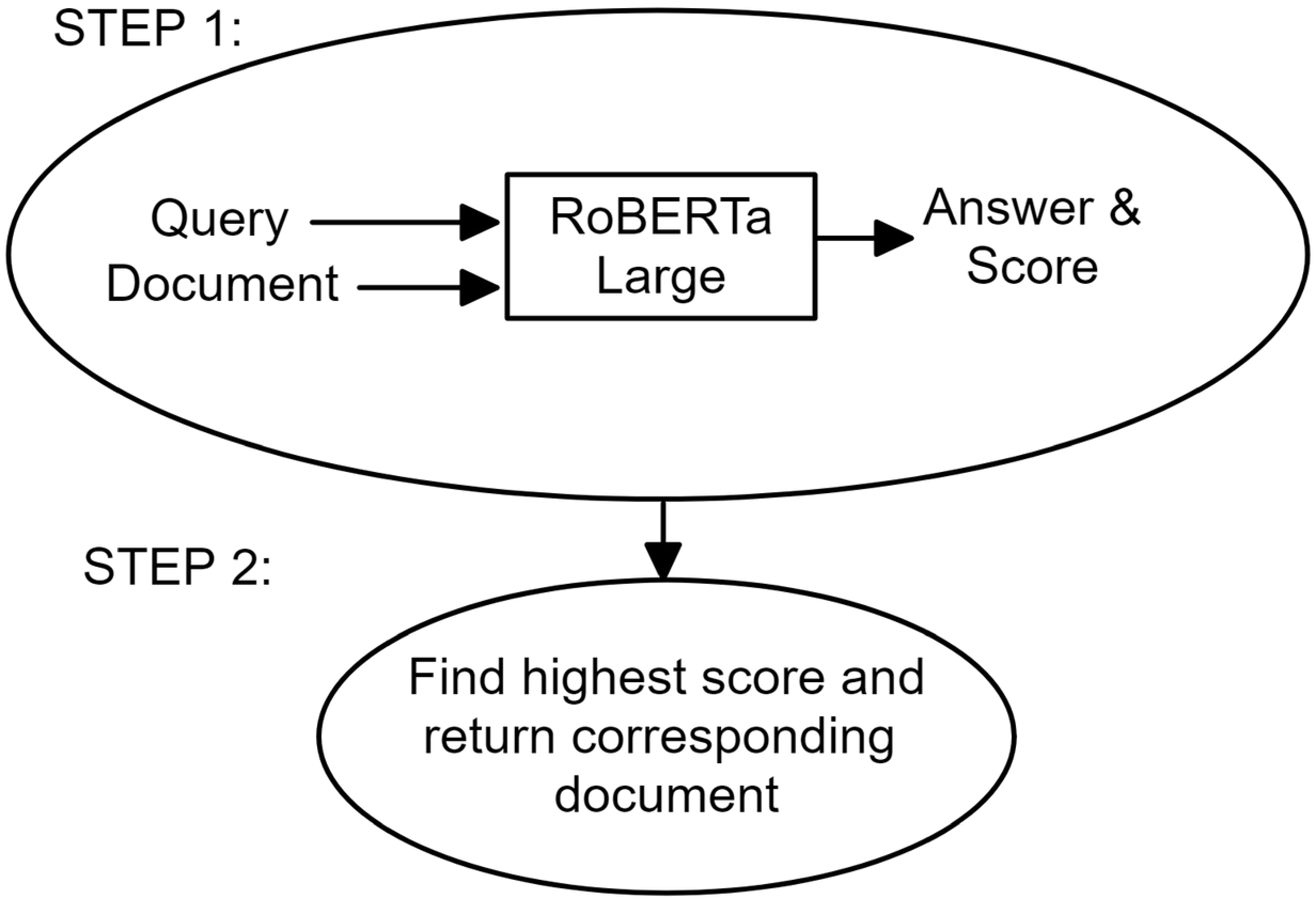
RETRIEVER

- We use the BM25 algorithm for retrieval (TF-IDF variant)
- BM25 takes into consideration:
  - The maximum effect of a term on the document
  - Relative document length

RETRIEVER

- Why not use a dense method of retrieval?
  - Frequent embedding calculation of documents (with every query)
  - Embedding calculation is slow (Passages took almost 5 mins!)
  - Very slow response time for the system
  - Suitable for systems where documents are not frequently changed

QA MODEL



STEP 1:

Query

Document

RoBERTa Large

Answer & Score

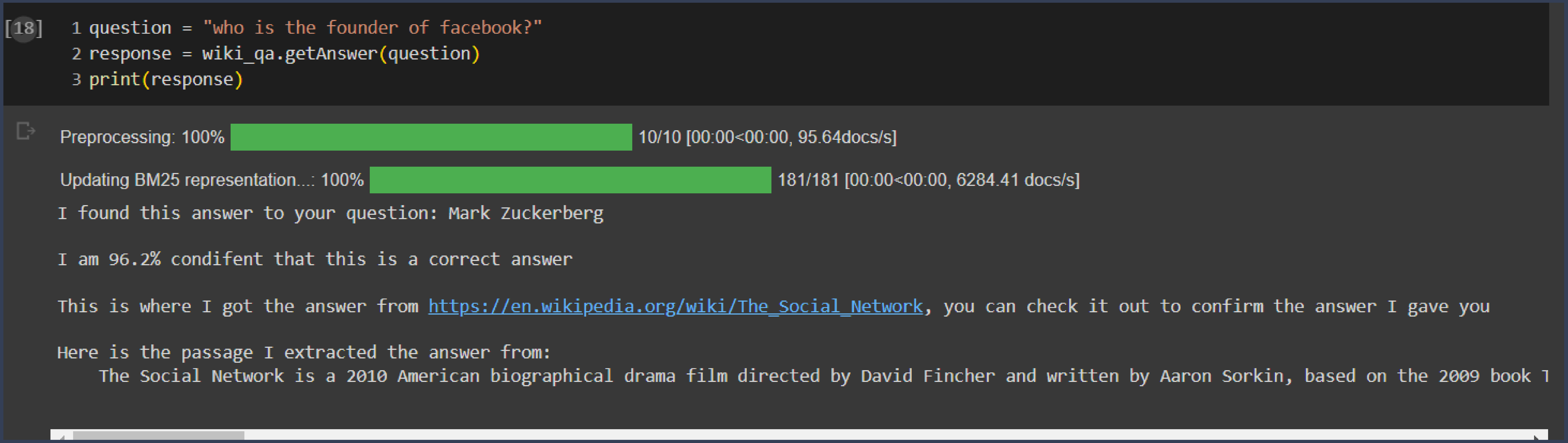
STEP 2:

Find highest score and return corresponding document

QA MODEL

- How is answer extracted?
  - Model returns start & end logits
  - Logits masked so that only context logits retain their value
  - Logits softmaxed to probabilities
  - Start & end probabilities multiplied to get probabilities for every start and end combination
  - Probabilities (answer confidence score) where start idx>=end idx are returned
  - Answer extracted from context using start and end idx pair with highest score

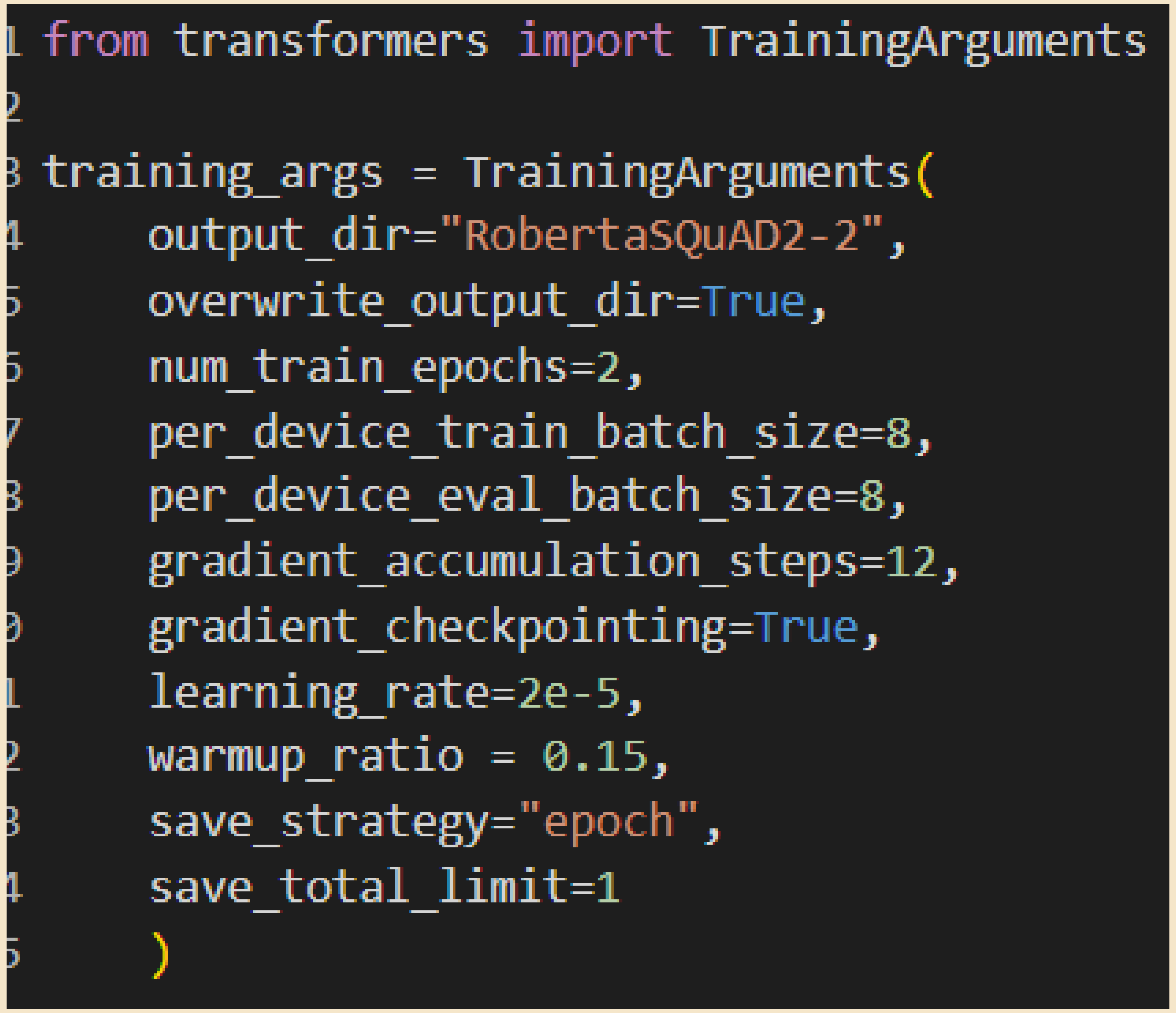
SAMPLE OUTPUT



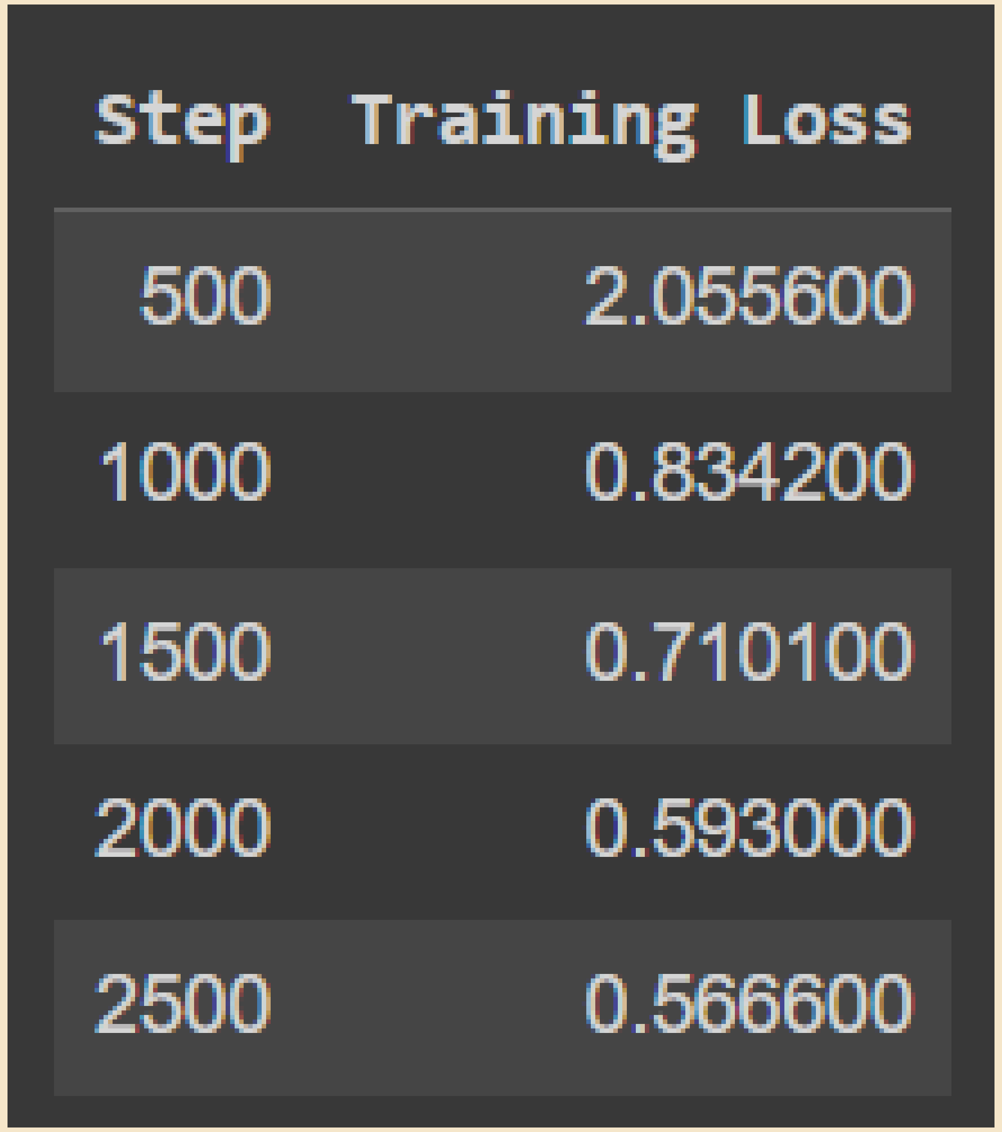
```
[18]
question = "who is the founder of facebook?"
response = wiki_qa.getAnswer(question)
print(response)
```

```
Preprocessing: 100%  10/10 [00:00<00:00, 95.64docs/s]
Updating BM25 representation...: 100%  181/181 [00:00<00:00, 6284.41 docs/s]
I found this answer to your question: Mark Zuckerberg

I am 96.2% condifent that this is a correct answer

This is where I got the answer from https://en.wikipedia.org/wiki/The_Social_Network, you can check it out to confirm the answer I gave you

Here is the passage I extracted the answer from:
    The Social Network is a 2010 American biographical drama film directed by David Fincher and written by Aaron Sorkin, based on the 2009 book T
```

TRAINING



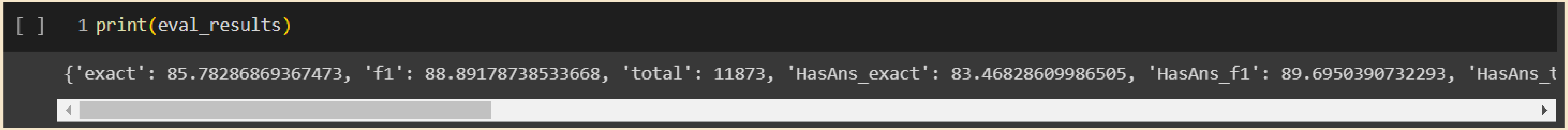
```
from transformers import TrainingArguments

training_args = TrainingArguments(
    output_dir="RobertaSQuAD2-2",
    overwrite_output_dir=True,
    num_train_epochs=2,
    per_device_train_batch_size=8,
    per_device_eval_batch_size=8,
    gradient_accumulation_steps=12,
    gradient_checkpointing=True,
    learning_rate=2e-5,
    warmup_ratio = 0.15,
    save_strategy="epoch",
    save_total_limit=1
    )
```



Step	Training Loss
500	2.055600
1000	0.834200
1500	0.710100
2000	0.593000
2500	0.566600

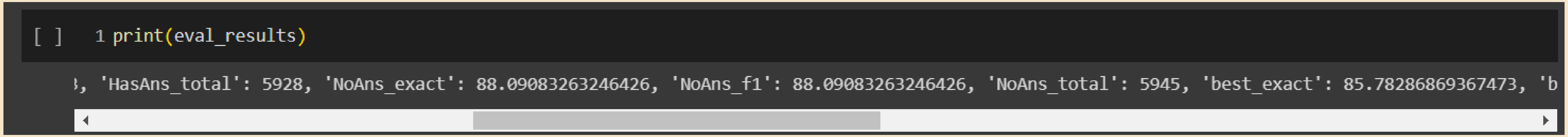
EVALUATION

- Scores are very close to those achieved in the original RoBERTa paper
  - Exact: 86.5
  - F1: 89.4



```
print(eval_results)
```

```
{'exact': 85.78286869367473, 'f1': 88.89178738533668, 'total': 11873, 'HasAns_exact': 83.46828609986505, 'HasAns_f1': 89.6950390732293, 'HasAns_t
```

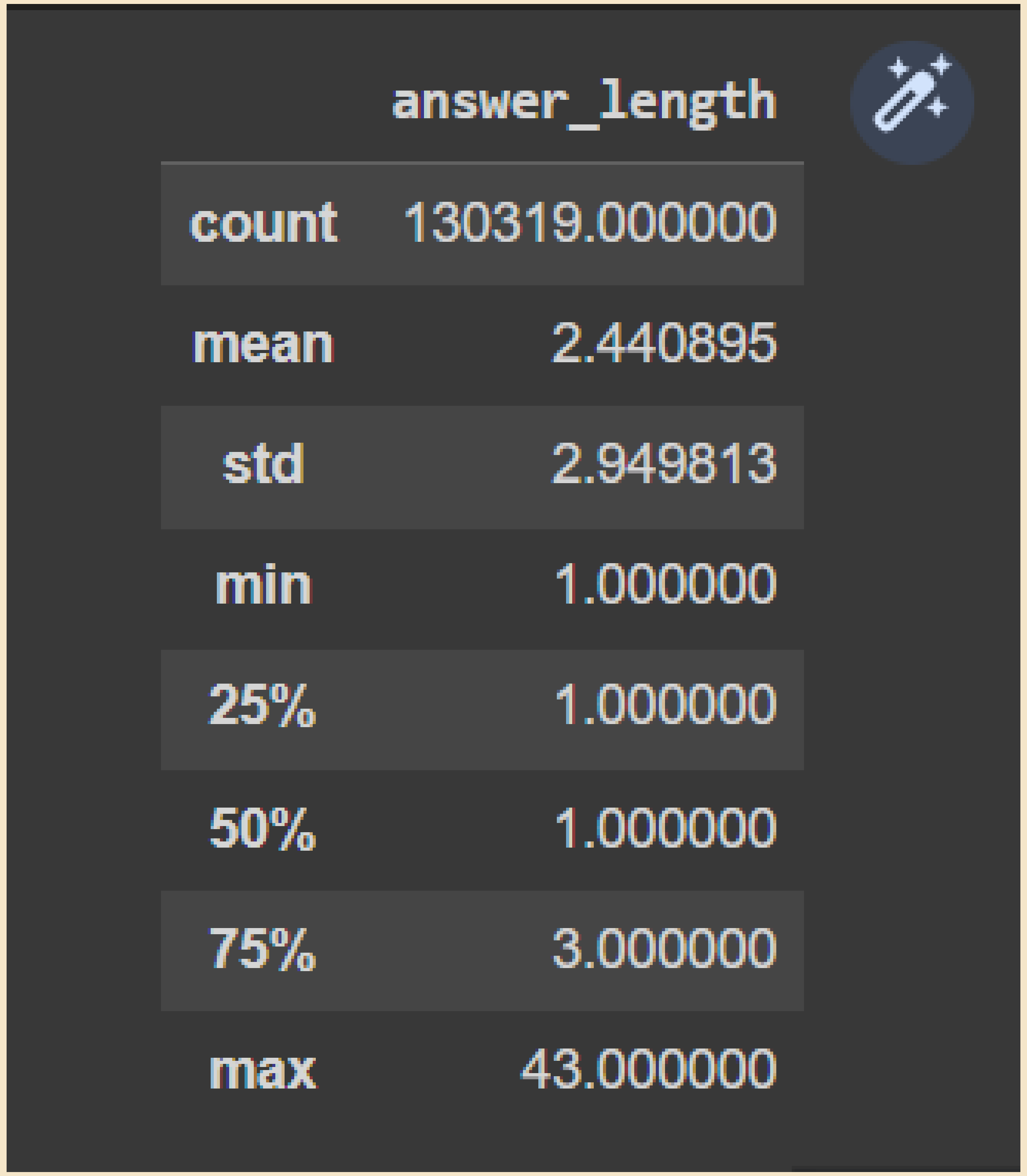


```
print(eval_results)
```

```
3, 'HasAns_total': 5928, 'NoAns_exact': 88.09083263246426, 'NoAns_f1': 88.09083263246426, 'NoAns_total': 5945, 'best_exact': 85.78286869367473, 'b
```

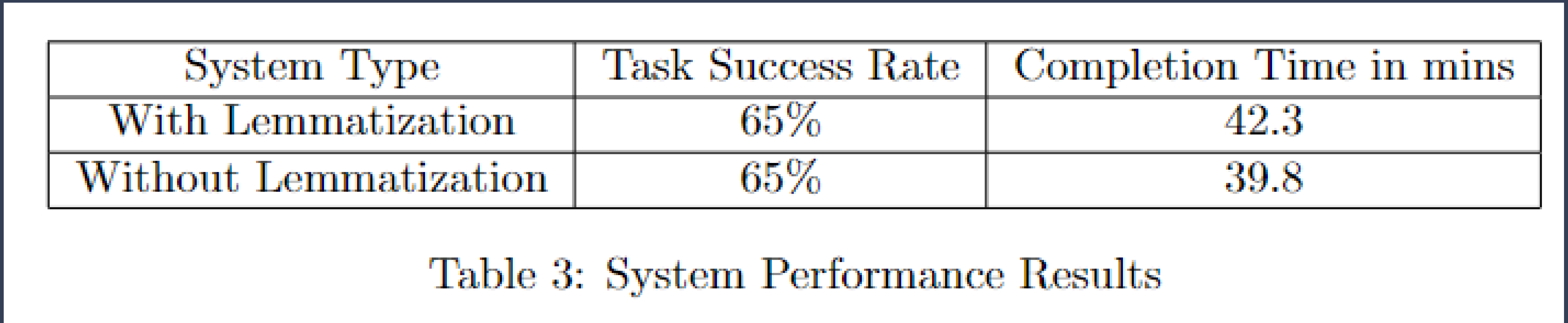
DATASET LIMITATIONS

- While dataset count may be very high
- Most answers are very short with few outliers
- Model's limited ability to extract longer answers



	answer_length
count	130319.000000
mean	2.440895
std	2.949813
min	1.000000
25%	1.000000
50%	1.000000
75%	3.000000
max	43.000000

EXPERIMENTS & RESULTS



System Type	Task Success Rate	Completion Time in mins
With Lemmatization	65%	42.3
Without Lemmatization	65%	39.8

Table 3: System Performance Results

RESULTS INTERPRETATION & SYSTEM LIMITATIONS

- While model answers most questions it seems to struggle with some
- Lemmatization doesn't appear to affect task success rate
- Difference in the questions answered using lemmatization and without
- Not all data available on Wikipedia
- Model trained on shorter answers (Quality of training data)
- Slow inference time

FUTURE WORK

- Train model on extraction of longer answers
- Extend system with multiple knowledge bases to cover a wider variety of data
- Optimization to use GPUs or TPUs to speed up inference times
- Add 2 layers of retrieval

THANK YOU

Any Questions?