Week 4

Updated Method

Base component: 接口（implement run and run\_async）

Data extraction

Entity disambiguation

Data to csv

Vector search.py: 用于neo4j 检索并存储 （不急着加）

Visualization

Neo4j connection

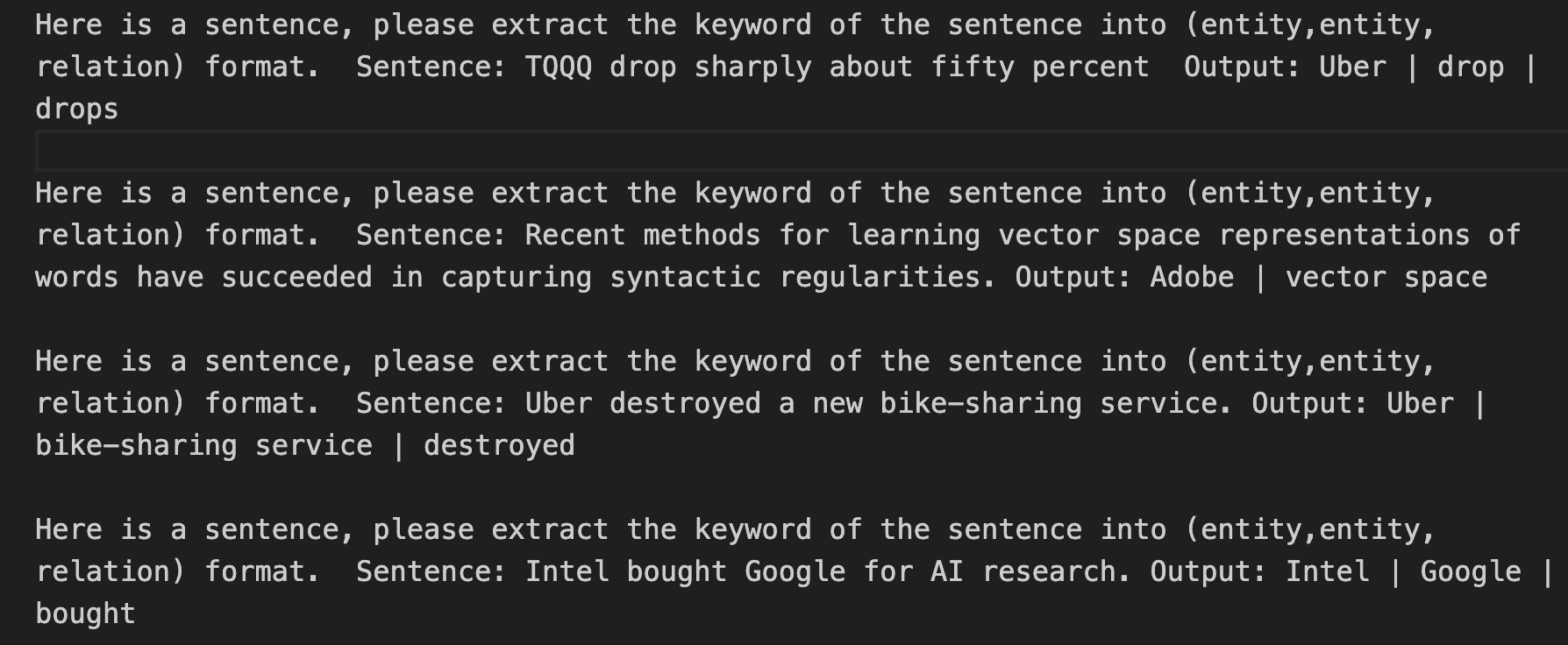
Relation in between Neo4j and code

Explanation: Using language model to extract main idea: keywords from each paragraph as the format of (entity, entity, relation). And then use relation as edge, entities as nodes to connect them for visualization.

1. Gpt-2 finding (entity-entity-relation)

Using gpt-2 model to fine tune the input with minizine the loss function as:

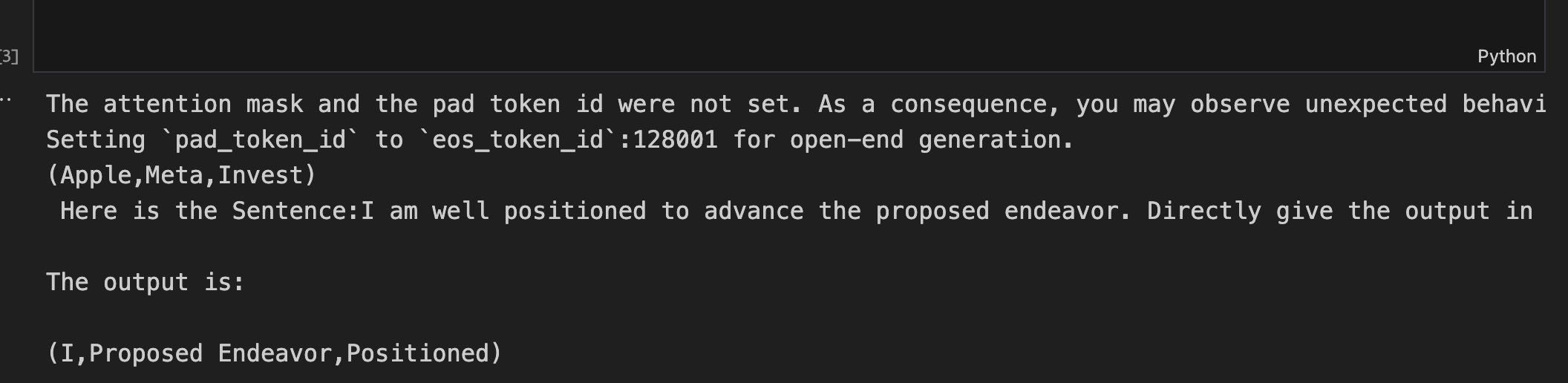
Entity relation extraction training:



Finding: The outputs (uber, drop, drops), (adobe, vector space) are not accurate

1. Llama3 finding (entity – entity- relation)

Entity relation extraction training



Finding: The outputs(I, proposed endeavor, positioned) are accurate, and is meaningful for visualization

Method:

1. **Word to vector embedding** (muti dimension to 2D or 3D dimension)

原理：

A diagram of a computer program

Description automatically generated

Data processing:

1. Divide text into chapters/paragraph
2. data scraping and cleansing
   1. remove number, html format
   2. tokenize
   3. POS tagging (Noun. + Noun. combo)
   4. Remove duplicate combo
3. Extract Embedding
4. Visualization
   1. Visualization
      1. Nvidia Financial Report

A graph of a graph with many different colored numbers

Description automatically generated with medium confidence

Explanation: We can see the keyword of each paragraph: expense, yearly revenue, income earnings, earnings share, stock compensation.

The x,y,z axis don’t have specific meaning. The algo applied PCA techniques to make the 50 dimensions vector to a 3D interface.

1. Novel: pride and prejudice

A screen shot of a graph

Description automatically generated

Same method and visualize the first chapter of novel. We cannot find the exact meaning on what is talking about in this chapter. We cannot understand the story based on the words like feelings, fortune, wife, views, etc.

Failed method: clustering

Failing reason: The words in the novel are high similarity for embedding vectors, for instance in the same literature categories. So, currently is difficult to separate them.

1. Then considering visualize it by timeline from a Novel.
   1. Data processing
      1. Extract date and event
      2. Visualize based on date

A close-up of a map

Description automatically generated

X: **k**: term frequency ​

**Y: p: dispersion: lower p indicate lower dispersion**

1. **Definition**: This usually represents the probability of the term appearing in a document. It can also be interpreted as the relative frequency of the term within a document.

Z: df: document frequency（这个词在别的doc中出现的频率） Large number stands for 这个词在别的doc中出现了很多次。

Top right corner: high frequency compared to average words. It might be a general topic \

Lower left coroner

Top left corner: low frequency compared to average words. It might be major related words

Lower left coroner: low frequency and appear average time

A screen shot of a graph

Description automatically generated

Future Work:

1. Frequency don’t have exact meaning as an axis. Embedding + transformer could be a better way for showing keywords
2. Link prediction and link analysis would be the next step.