**Problem:**

* Specialised semantic search, i.e., fixed set of documents to search from, unlike Google which is a Generalised search engine
* Given a biomedical statement, return a set of relevant results

**Tools:**

* Jina AI
  + Jina is a cloud-native neural search platform/ecosystem. It can be deployed in containers, cloud, or on-prem servers.
  + Neural search excels with multimodal data, because it can learn to map multiple modalities – for example, text and images – to the same embedding space.
* Docarray
  + Data structure used in Jina for efficient storage and retrieval

**Data Source:**

Reactome.org

**System Design:**

* How many requests per second we need to handle?
  + Search engine must have a distributed index system (e.g. Elasticsearch)
  + System must have a Load Balancer to distribute the requests
* How many documents to search from?
* How many users are there at a time?
* Do we want to personalize? Will user be logged in?

**Questions:**

* update frequency of our vectors/models?
* Latency: how fast do we need the response? (Typically 100 ms)
* Throughput: how many search queries would come per second?
* Freshness requirements? How often we need to refresh the index

**Design choices/Assumptions:**

* We are indexing only the “summation” field since some docs don’t have this field at all
* Embedding options for documents and query
  + Avg of embedding of each word
  + Embedding of entire text (we use this currently)
* Tokenizer options for documents and query
  + simple tokenizer
  + Spacy tokenizer
  + Bert based tokenizer
  + Others (we use biomedical pretrained sentence transformer)
* How are we indexing the documents?
  + Keyword based inverted index
  + Just convert to embedding and store (we do this)
* Results need to be ranked/sorted based on relevance.
  + How are we going to score? i.e. scoring/similarity function
  + Euclidean distance, cosine similarity, etc.
  + Nearest Neighbour search
    - Brute force O(n)
    - Approximate NN search
    - K-d trees
    - Locality Sensitive Hashing
* Embedding size? We use 768 currently
* Where are we going to index the documents?
  + Jina supports Sqlite, Elasticsearch, Redis, ANNlite, etc.
* Components of Search Engine

1. Query processor
2. Indexer
3. Ranking algorithm

**Business requirements:**

* We want low computational costs
* High precision and recall

**Understanding the data fetched from source:**

* It is a json/dict object
* Has 4 Keys/Fields => Complex, Pathway, Protein, Reaction
* Each fields has 2 subfields => pathologies and nonpathologies

**Some Observations:**

* The reactome data script throws an error with the search term “amyloid beta”, hence not using the data from that.

Text

Description automatically generated

**Improvements:**

* Try finetuning the transformer rather than just using the pretrained model
* Make use of GPU to generate embeddings and for search (wherever possible)
* We need to look into speeding up the indexing process
* Prevent indexing all over again if there are no new documents, in other words, index only the delta every time
* Results to be reproducible everytime? given no change before re-running the process
* Clean the text before generating embeddings, e.g. <span> tags, special characters, etc.
* Make the endpoint accessible to the Weave team
  + Containerize the application using Docker
  + Deploy the project to JCloud

**Metrics for evaluation:**

* We can evaluate offline using ground truth created by human raters
  + Once we have it, we can use Normalised Discounted Cumulative Gain (NDCG) to measure quality of our system