

Exploring Dense Retrieval via Approximate Nearest Neighbors

Ashwin Guptha (asg8739), Geetika Bandlamudi (gb2642)

Link to the code repo: [WebSearchEnginesProject](#)

Abstract

This project addresses the limitations of sparse lexical retrieval methods, such as BM25, by leveraging dense retrieval techniques based on embeddings generated from MiniLM. The focus is on finding the nearest passages in a given set efficiently. The proposed system incorporates a multi-faceted approach, involving the transformation of the MS MARCO passage dataset into a high-dimensional space using MiniLM, fast approximate nearest neighbor algorithms (HNSW), proper evaluation methodologies, and a potential hybridization with traditional methods like BM25. The system integrates a ladder model that combines BM25 for keyword-based relevance and semantic similarity through MiniLM embeddings. This conjunctive approach aims to enhance precision by selecting passages enhanced by both methods. We have added examples to share our view and take over the performance and results of these methods.

Problem Statement

Sparse lexical retrieval methods such as BM25 miss a lot of good results due to vocabulary mismatch and related issues. One approach to overcome this issue uses query expansion. More recently, it has been proposed to instead map queries and documents into a high-dimensional space using say a BERT transformer, and to then retrieve the documents that are closest to the query in this high-dimensional space. Finding these nearest neighbors could then be done via optimized Approximate Nearest Neighbor (ANN) methods such as HNSW, DiskANN, or NSG. This project has several different parts that are needed: (1) Transforming a collection such as MSMarco and the queries into the high-dimensional space using (say) BERT, (2) a fast ANN algorithm such as HNSW, (3) possibly also combining such a dense retrieval with traditional methods such as BM25 into a hybrid method.

Dataset

MS MARCO [1] (MicroSoft MAchine Reading COmprehension) is a large-scale dataset focused on machine reading comprehension. Since its initial release, benchmarking efforts for several NLP and IR tasks have made use of this dataset—including question-answering, passage ranking, document ranking, keyphrase extraction, and conversational search. We used the passage dataset for this project. We have chosen the passage dataset to avoid the additional parsing needed for the document dataset. This provides us an opportunity to directly work with the webpage content. This passage dataset is based on the public MS MARCO dataset, although our evaluation will be quite different. Our dataset is a little over 2.9 GB with 8,841,823 entries in a TSV format with the pid, passage where pid is the passage ID.

Link to Dataset: <https://microsoft.github.io/msmarco/Datasets.html>

Overview

In the next few sections, we will discuss the various approaches we used and the performance and accuracy of those methods. We start with discussing the Embedding Generation phase, and how and why we chose MiniLM for this part. Next, we discuss the various Similarity Search algorithms like Cosine Similarity, Nearest Neighbours algorithm, and Semantic Search and judge the performance of each of these. We later delve deep into the Approximate Nearest Neighbours algorithm - HNSW and the intricate details about the working of the algorithm for our use case. Then, we propose a new ladder approach that combines the benefits of BM25 with the best algorithms seen so far. As a next step, we also discuss our attempt to enhance the quality of the results through query expansion. Finally, we discuss our take using a few examples. Just a heads up, we have used documents and passaged interchangeably throughout this report. Let's get started!

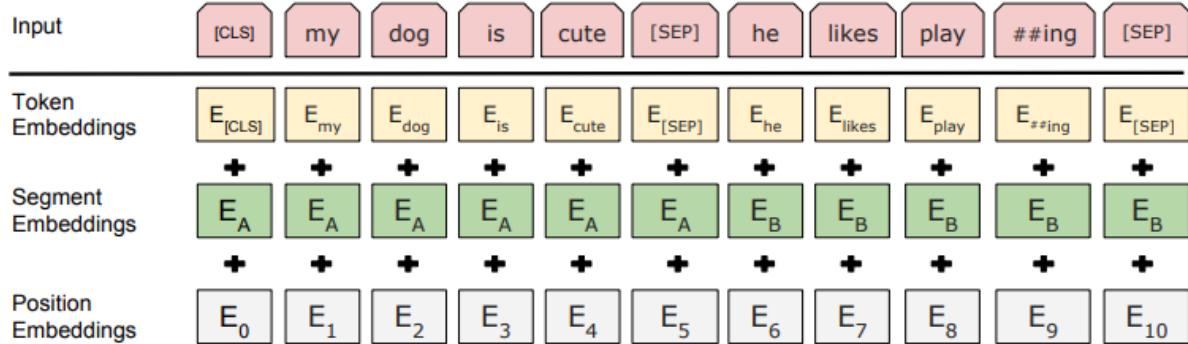
Embedding Generation

Embedding generation in machine learning and web search engines is a critical concept that encompasses the transformation of high-dimensional data into a lower-dimensional space, facilitating various tasks such as improving the efficiency and accuracy of search algorithms. In machine learning, embeddings are dense vector representations of data where the distance between vectors is indicative of the similarity between the entities they represent. The primary

goal of embeddings is to capture the essence of data in a format that is more manageable and useful for machine learning models. Embeddings in web search engines are pivotal in understanding the intent behind queries and the content of web pages. By converting words, phrases, and entire documents into vector representations, search engines can more effectively match queries with relevant results. These models understand the context and meaning of words in queries and documents, going beyond simple keyword matching.

BERT

BERT [2] (Bidirectional Encoder Representations from Transformers) is a bidirectional transformer pre-trained using a combination of masked language modeling objective and next sentence prediction on a large corpus comprising the Toronto Book Corpus and Wikipedia. BERT is built on the Transformer architecture, which relies on attention mechanisms. Unlike traditional sequence-based models that read text input sequentially, the Transformer processes the entire text at once, allowing it to capture more complex word relationships. Traditional language models were either left-to-right or right-to-left, limiting their understanding of context. BERT overcomes this by using a "masked language model" (MLM) approach, where random words in a sentence are masked out and the model is trained to predict them, considering both left and right context.



BERT tokenizes passages into words and subwords (to handle unknown words), and special tokens are added:

[CLS]: A special token placed at the beginning of the text used for classification tasks. The final hidden state corresponding to this token is used as the aggregate sequence representation for classification tasks.

[SEP]: A special token that separates segments. In this example, it separates two sentences or segments: "my dog is cute" and "he likes playing".

As shown in the Figure above, each token is converted into a vector using an embedding matrix called a token embedding. Segment embeddings are used to differentiate between different segments or sentences. In this example, E_A is used for the first sentence and E_B for the second. This allows the model to process paired sentences. Positional embeddings provide information about the token positions within the sequence. Since the Transformer does not have a recurrent structure and does not process the tokens sequentially, it requires positional encodings to understand the order of tokens. The combination of these three embeddings gives a detailed representation of each token, accounting for its meaning, its role within its sentence, and its position in the sequence. The sum of these embeddings is then passed through the Transformer's multi-layered bidirectional architecture, which processes the input in a context-aware manner, leading to highly contextualized token representations.

Why MiniLM?

MiniLM [4] is a variant of the transformer architecture, similar to BERT, but with modifications to make it more computationally efficient while maintaining strong performance in various natural language processing (NLP) tasks. The preference for MiniLM embeddings on the entire passage over a standard Word2Vec for each word in the MSMARCO passage dataset can be attributed to several key factors rooted in the inherent architectural differences. MSMARCO, a large-scale dataset designed for information retrieval and question-answering tasks, presents unique challenges that MiniLM is particularly well-equipped to handle.

Word2Vec generates embeddings based on local window-based contexts, meaning each word is represented by a single vector regardless of its usage in different contexts. This approach fails to capture the nuances of words with multiple meanings. MiniLM takes care of this and words with multiple meanings and usages are embedded differently based on context, leading to a more accurate representation of the language which in turn, is more useful for a search engine. The task in MSMARCO often involves matching user queries with relevant passages. MiniLM's ability to understand the semantic meaning of sentences as a whole allows for more effective matching of queries to passages, especially in cases where keyword overlap is minimal.

Similarity Search Algorithms

Similarity search in web search engines refers to the process of retrieving documents or items that are similar to a given query or document. We have seen the process of using traditional keyword-based search where the focus is on matching exact words or phrases in assignments 2 and 3. But here the similarity search aims to find items that are conceptually or contextually similar to the input. There are multiple ways of doing this and we have chosen a few of the approaches that we found to be common in the industry for such use cases previously.

Cosine Similarity Search

This is a metric used to measure the cosine of the angle between two vectors. In information retrieval, it is often employed to calculate the similarity between the vector representations of documents. Here, since we are working with passages, we represent the passages in a higher dimensional space using MiniLM as explained above. The search function takes the query as the input, encodes it into the higher dimension using MiniLM, and calculates the cosine similarity with each document's vector representation. The formula for cosine similarity ($\cos \theta$) between two vectors A and B is given by:

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

In the context of information retrieval using MiniLM embeddings, the vectors A and B correspond to the vector representations of the query and the document, respectively. The dot product measures the similarity in direction between the vectors, and the denominator normalizes the result based on the lengths of the vectors.

The cosine similarity ranges from -1 to 1, where 1 indicates perfect similarity, 0 indicates no similarity, and -1 indicates perfect dissimilarity. In the context of search engines, a higher cosine

similarity score between the query and a document implies a greater degree of similarity, helping the search engine rank and retrieve documents that are more contextually relevant to the user's query. We have used the sentence transformer util library to perform the cosine similarity search. It takes the query embedding and the passage corpus embeddings to find the cosine similarity. We later use the `top_k()` method to find the most similar documents by similarity score.

We noticed that the cosine similarity method is extremely slow given that the algorithm calculates the cosine angle for every pair of passages to find the most similar passages. This is very general and needs more improvement. Nowadays, this algorithm is often part of a larger and faster algorithm that uses cosine as a measure of distance between items in the higher dimensional space.

K Nearest Neighbors

K Nearest Neighbors (KNN) is a very widely applied algorithm in machine learning for various tasks, including similarity search. In our context, KNN provides a way for us to identify documents that closely match a given query. KNN transforms both the passages and queries into a higher dimensional space facilitating the comparison of their positions. This algorithm is influenced by the sparsity of the data, and in high-dimensional spaces, distances tend to become less meaningful which is called the curse of dimensionality. We wanted to know if the overall geometric arrangement of vectors in the feature space is significant for this use case.

Nearest Neighbour — Distance Measures

- Given two feature vectors with numeric values

$$A = (a_1, a_2, \dots, a_n) \text{ and } B = (b_1, b_2, \dots, b_n)$$

- Use the *distance measure*:

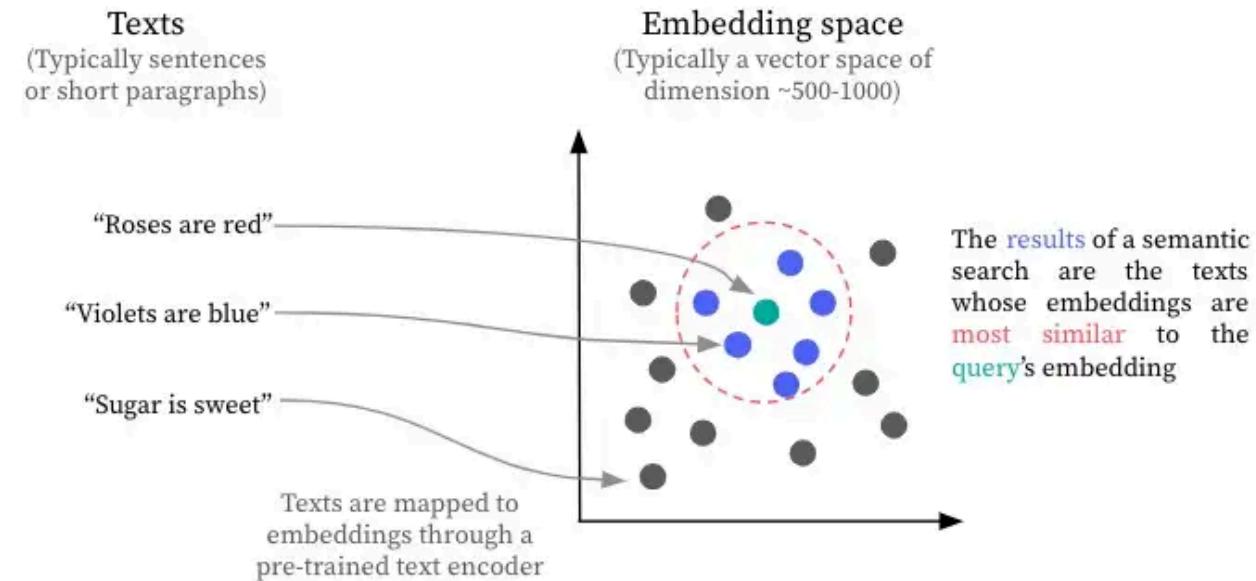
$$d = \sqrt{\sum_{i=1}^n \frac{(a_i - b_i)^2}{R_i^2}} = \sqrt{\frac{(a_1 - b_1)^2}{R_1^2} + \frac{(a_2 - b_2)^2}{R_2^2} + \dots + \frac{(a_n - b_n)^2}{R_n^2}}$$

R_i is the *range* of the i th component

Our system employs sci-kit-learn's Nearest Neighbors with the Euclidean metric to find the most similar documents in a corpus. The algorithm calculates the Euclidean distance between the query and each document, identifies the K nearest neighbors, and returns them as search results. We observed that the KNN algorithm performs just as well as the cosine similarity search. We wanted to explore faster algorithms that take the semantic information also into consideration while calculating the similarity. Naturally, the next choice was semantic search.

Semantic Search

In the process of harnessing machine understanding capabilities, we explored more about leveraging natural language understanding and semantic analysis to comprehend the meaning behind user queries and the content of documents. This approach [5] aims to enhance the precision and relevance of search results by considering the context, intent, and semantics of the information. Semantic search generates dense vector representations for both queries and documents using MiniLM just like how it is described above. These embeddings capture intricate semantic relationships, allowing for a nuanced understanding of textual data. Semantic search is especially valuable when dealing with ambiguous queries, capturing nuanced semantic relationships, and improving the overall precision of information retrieval.



As we know, the semantic search utility function from the sentence transformer library uses the cosine similarity operation. So a natural question would be as to why we have tried both

approaches if all we are interested in is the cosine similarity between the query and the documents. Cosine similarity is a mathematical metric that measures similarity between two vectors, whereas semantic search is an information retrieval approach that focuses on understanding the semantics of text for more contextually relevant results. Semantic search often involves the use of cosine similarity as part of its calculations, but it encompasses a broader set of processes beyond simple vector similarity.

Our experimentation involved comparing the performance of both approaches—direct cosine similarity and semantic search. We observed that while cosine similarity provided effective results in capturing surface-level similarities, semantic search outperformed in scenarios where a deeper understanding of the underlying meaning was crucial. This was particularly evident in instances of ambiguous queries, where semantic search demonstrated superior performance by considering the broader context and semantic relationships within the documents. For example, we searched for “purple pride” and got results related to rainbows, along with LGBTQ+-related content. The nuanced understanding facilitated by semantic search contributed to more accurate and contextually relevant passage similarity based on our dataset. However, we are still suffering from the long duration it takes to perform a search. We explore the recent research algorithms that are being brought close to developers and scientists across various domains and projects - Approximate Nearest Neighbours. Spotify, Amazon, and Microsoft have conducted various studies in this domain and more recently, ElasticSearch from Amazon has incorporated this algorithm into its search techniques. Let us use one of those algorithms for our use case.

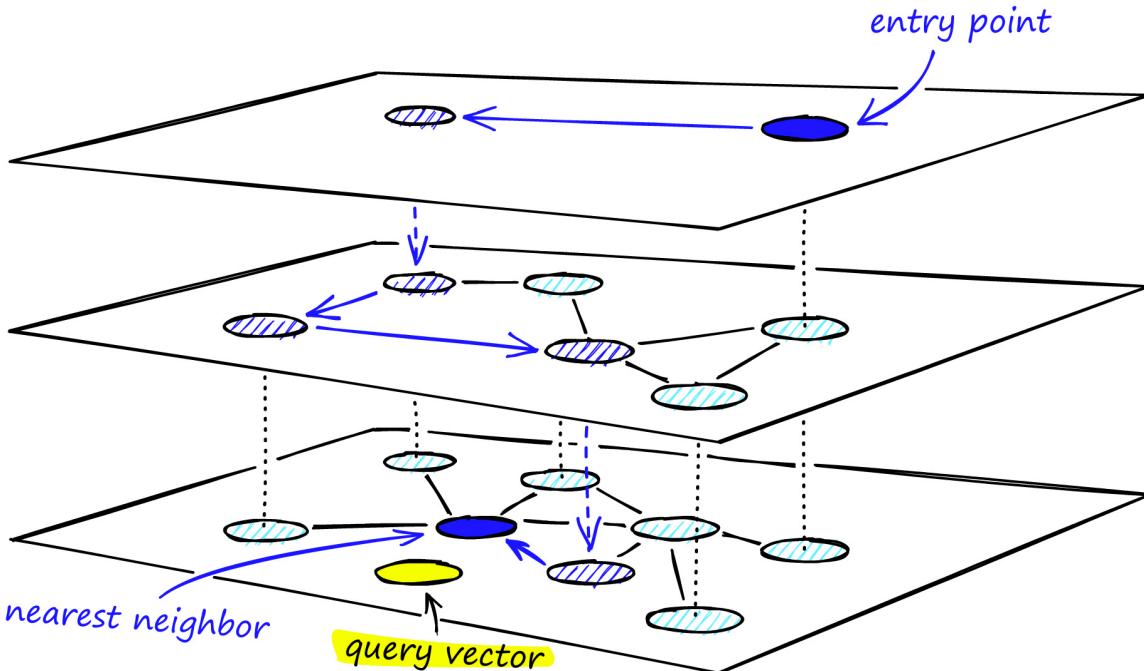
Approximate Nearest Neighbours Model

The nearest neighbor problem is a problem in the algorithms space where, given a set of points in a metric space, the goal is to construct a data structure that would efficiently identify, for any given point, its closest neighbor from the initial set without computing all distances between the points. To make things faster and use less memory, we sometimes play the “approximate” nearest neighbor game. It's like saying, “I don't need the absolute closest buddy, just someone pretty close will do.” This is perfect for our use case here because eventually, we are leaning towards finding the closest documents to a set of documents that we shortlist by other algorithms. Also, we are interested in an “approximation” to find similar passages for a given query because we're dealing with a large dataset, and computing exact distances between each document and query could be computationally expensive and memory-intensive.

HNSW

Hierarchical Navigable Small World (HNSW) is a state-of-the-art algorithm used for an approximate search of nearest neighbors. Under the hood, HNSW constructs optimized graph structures making it very different from other approaches that were discussed in previous parts of this report. Its structure represents a multi-layered graph with fewer connections on the top layers and more dense regions on the bottom layers. The search starts from the highest layer and proceeds to one level below every time the local nearest neighbor is greedily found among the layer nodes. Ultimately, the nearest neighbor on the lowest layer is the answer to the query. One of the most important and best parts of this Disk-based ANN is the time complexity which is $O(\log n)$.

One of the major reasons why we chose HNSW [6] over other methods is its extreme efficiency. When working with higher dimensional vectors, we observed that the previous methods were taking a lot of time to find the nearest neighbors and we needed a faster algorithm. Moreover, we wanted to avoid the curse of dimensionality in our use case. As we have discussed before, traditional methods struggle with the curse of dimensionality, where the distance between points becomes less meaningful as the number of dimensions increases. HNSW's hierarchical structure helps mitigate this issue. HNSW provides an approximate nearest neighbor search with guarantees. While it sacrifices exactness for efficiency, the algorithm ensures that the identified nearest neighbors are close to the true neighbors. Since in our use case, we are not going in search of "exactness", but rather in search of "correctness", we have chosen this algorithm.



One challenge we faced was loading and working with a very large corpus embedding. Hence, we split it into multiple chunks and worked with them individually. We initialized the HNSW vector space as $8M \times 384$ since we had 8M corpus embeddings each with a dimension of 384. Taking advantage of the utility functions in the HNSW library, we added the embeddings to the HNSW index while saving the index along the way. We noticed that the denser the index, the more time it took for the embeddings to be added to the index. However, the overall process took a few hours.

BM25

The user query is preprocessed before being sent as input to BM25 in the ladder method. For this version of the algorithm, conjunctive queries are being used as disjunctive or even a hybrid version will lead to passages with no context to the query to be returned. If a query of length l does not have a common passage ID between the query words in BM25, then only the most important $l-1$ words of the query are taken into account. This repeats till the BM25 algorithm is able to return a set of passage IDs. The process of selecting the most important keywords from the query will be discussed in the next section.

The BM25 algorithm represents a pivotal component in the information retrieval system. To enhance the efficiency of the retrieval process, a merge and sort operation is integral after the initial computation of BM25 scores. This operation involves the consolidation of temporary files, which contain intermediate results, and their subsequent organization. The Unix sort command serves as the cornerstone for this operation, facilitating the merger of all temporary files. It

meticulously orders the entries first by the word and subsequently by the document ID, resulting in the creation of a unified index file named merge.index.

The Unix sort utility is invoked with a specified buffer size of 1GB, a measure adopted to modulate memory consumption during the sorting phase. This parameterization ensures that the sorting remains within acceptable memory bounds, thereby optimizing system performance. In addition to its sorting capabilities, the utility also excels in deduplication; it deftly identifies and removes duplicate entries from the sorted data, ensuring the uniqueness of each line in the resultant index. The following is the formula for calculating BM25 for each document:

$$\text{score}(D, Q) = \sum_{i=1}^n \text{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)},$$

$$\text{IDF}(q_i) = \ln\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right)$$

N: Total number of documents in the corpus

n(qi): Total number of documents that contain the term q

f(qi, D): Total number of times qi occurs in the document D.

b: 0.75

k: 1.5

Avgdl: Average document length

D: Total number of words in the document D

Keyword extraction

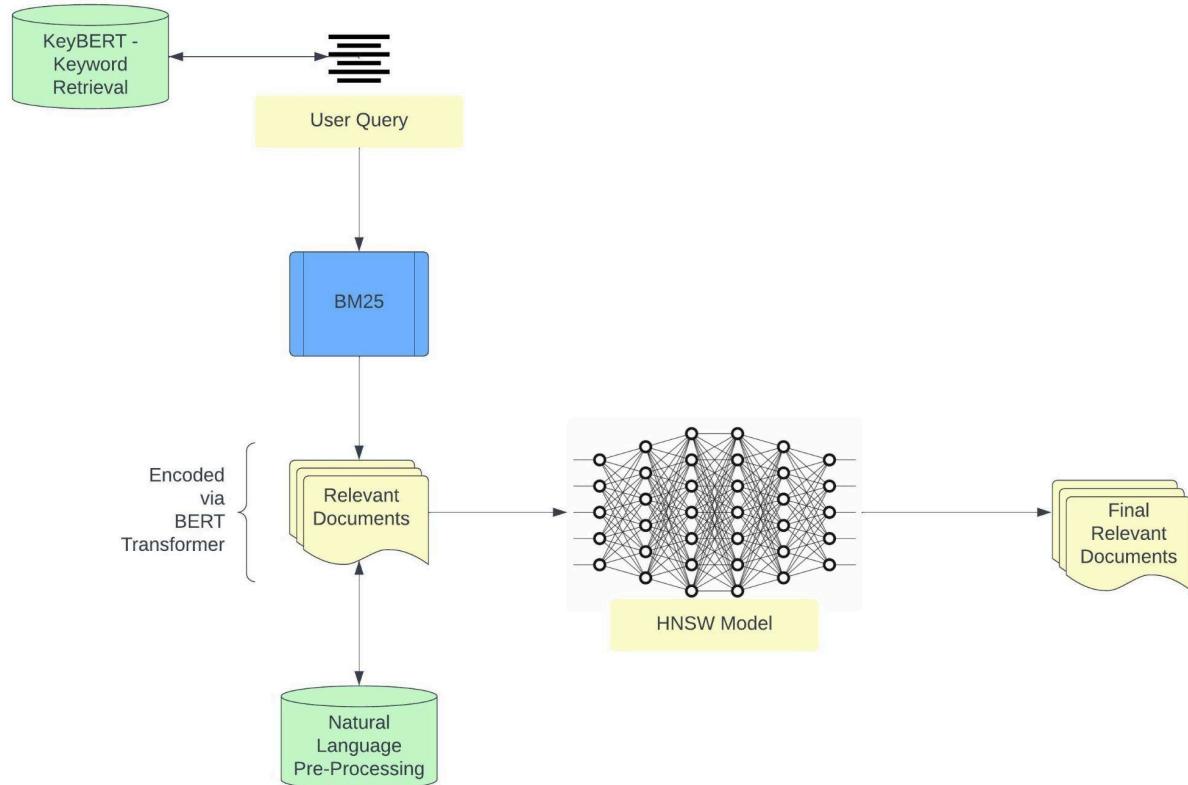
This process is used for query expansion as well as to get conjunctive query results in BM25 in case the original query cannot return any common passages.

For query expansion, a list of relevant passages is taken, usually after BM25 (in the ladder method), and combined into a single large string. This aggregation is done to create a comprehensive context for KeyBERT [9][10] to analyze, as it will look for keywords that are relevant across the entire set of passages. KeyBERT, a model that uses BERT embeddings for keyword extraction, is then utilized to extract keywords from the concatenated passages. The model is instructed to ignore the original query terms as stop words during this process, ensuring that the keywords extracted are additional terms and not those already specified by the user. Out of the extracted keywords, only the top k keywords are chosen. These are considered the most relevant or important terms related to the context provided by the list of passages. KeyBERT is effective because it uses BERT's deep understanding of context and semantics,

which allows it to capture more nuanced and meaningful keywords than methods that rely on frequency or statistical measures alone.

Ladder Method

From the above two methods, we realized that an effective search engine needs to combine the benefits of keyword-based relevancy and semantic similarity. Hence, we proposed a ladder approach that takes the input query and finds the conjunctive most relevant documents using BM25. The results of BM25 are encoded in a higher dimensional space to find the most similar passages. We have chosen the total passages returned to be 1000. A conjunctive approach is chosen to ensure that the final result set is highly relevant and satisfies both traditional keyword-based relevance (BM25) and semantic similarity (ANN) criteria. This approach aims to improve precision by requiring documents to meet criteria from both methods, reducing the likelihood of false positives and ensuring a more focused set of results. A rough architecture of this approach is shown in the figure below.



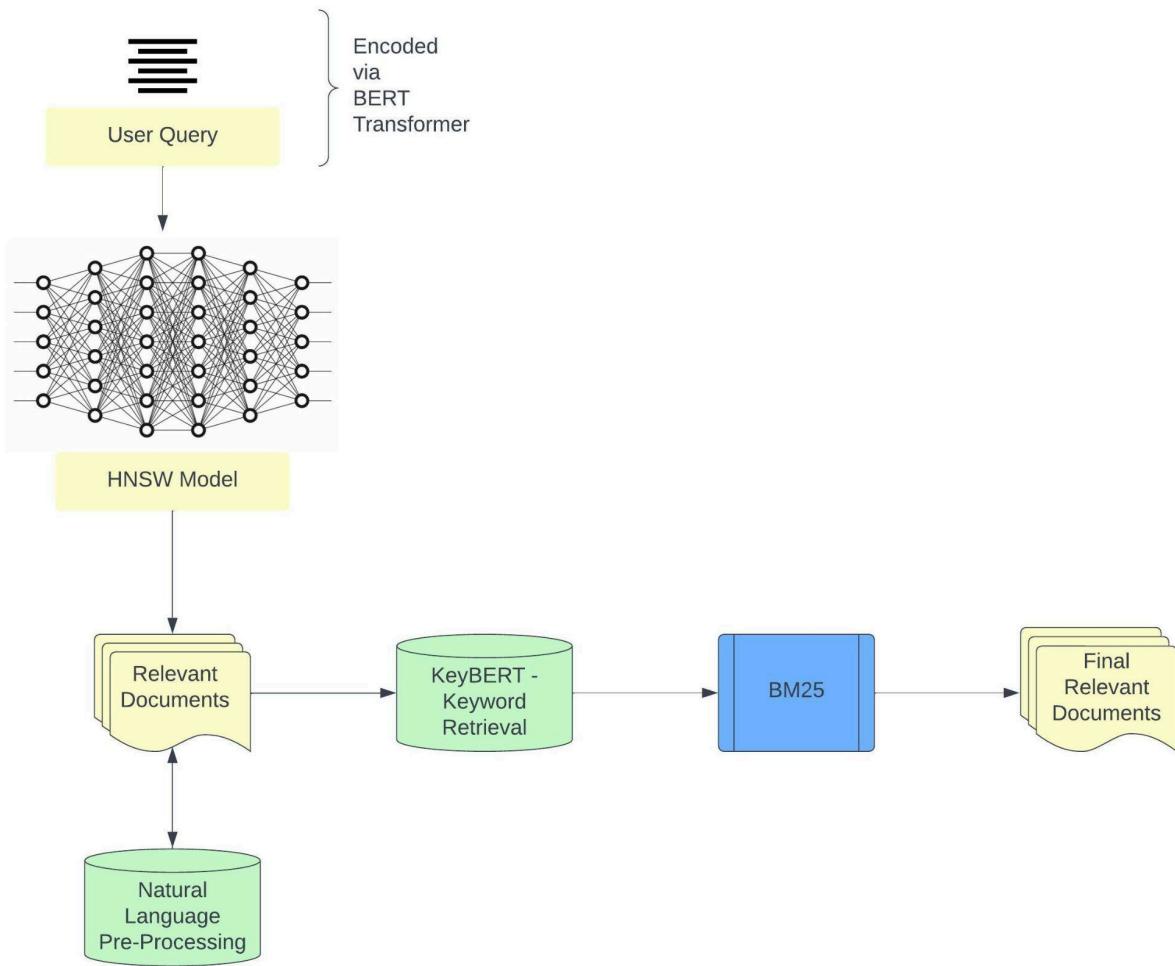
From the above diagram, the user query undergoes a transformation by the KeyBERT model.

KeyBERT is a model that likely utilizes BERT embeddings to extract keywords that are semantically relevant to the query text. This process ensures that the key concepts within the query are identified and can be used to find documents that are not just keyword matches but are also semantically relevant. The relevant documents from the BM25 process are then encoded using a MiniLM Transformer. MiniLMs is one of the fastest state-of-the-art models for natural language processing that provides deep contextual representations of text. By encoding the documents through MiniLM, the system captures the nuanced semantic relationships within the text, which goes beyond simple keyword matching. The documents are further refined through language processing. This step involves removing stopwords—words that are common and do not contribute significantly to the semantic meaning of the text—and lemmatization, which reduces words to their base or dictionary form. Once the documents have been semantically encoded and language-processed, they are input into an HNSW model. HNSW (Hierarchical Navigable Small World) is an algorithm for efficient approximate nearest neighbor search in high-dimensional spaces. The HNSW model uses the MiniLM-encoded representations of the documents to perform a similarity search, identifying documents that are semantically closest to the user query.

We observed that the Ladder model performs faster in general and better contextually than all the methods described before. This is because BM25 finds the context by identifying the documents containing the words. The HNSW finds the documents closer to the BM25 identified documents improving the search results multifold. We observed that it is better for longer queries in comparison to the Reverse Ladder method described below. We believe this can be attributed to the fact that we are identifying the common documents using text match-based approaches like BM25 which establishes the boundary for HNSW to explore comfortably. However, in shorter queries, the context is dimensionally closer and does not necessarily add any value to the ladder up.

Reverse Ladder Method

The flowchart illustrates a search engine retrieval strategy that inverts the typical order of operations seen in traditional search engines. Instead of beginning with keyword extraction and then refining the search using semantic methods, this "reverse ladder" method starts with semantic search and refines using keyword-based techniques.



The user's query is first encoded using a MiniLM transformer. This step converts the query into a vector representation that captures its semantic nuances. The encoded query is then used in an HNSW model to perform a semantic search. The HNSW model outputs a set of documents that are semantically relevant to the user's query. These documents are considered a preliminary result set, relevant based on their semantic content rather than exact keyword matching. This set of relevant documents may undergo further natural language pre-processing, which could include tasks like stopword removal, and lemmatization. This step cleans and standardizes the text, preparing it for more precise keyword analysis. Now, KeyBERT is used for keyword retrieval. It analyzes the pre-processed text of the relevant documents to extract key terms and phrases. This keyword extraction is informed by the semantic context of the documents, as KeyBERT uses BERT's embeddings to identify keywords that are representative of the main topics or themes in the text. With keywords from KeyBERT, a subsequent search or ranking is performed using the BM25 algorithm. This algorithm ranks the previously selected documents based on how well they match the extracted keywords. BM25 considers term frequency and document length to provide a score for each document's

relevance to the keywords. This final set is presented to the user as the search results.

Here, we observed that the context is established by HNSW first. It works well with shorter queries because the context is established and in the index, the similar documents stay close together. Later, the keyword extraction along with BM25 finds the intersection of the passages among all query terms. We chose 5 as a starting value for the keyword extraction. This requires more experimentation and finding the optimal value based on the query is a research area that can be explored.

Vocabulary Feedback (Pseudo - Relevance Feedback)

Vocabulary feedback in web search engine design refers to a mechanism or technique used to enhance the quality of search results and improve the user experience by incorporating user interactions and feedback into the search process. Vocabulary feedback systems generate context-aware query suggestions. When a user enters a query, the system will offer suggestions for refining or expanding the query based on similar passages. These suggestions help users better express their intent. Pseudo-relevance feedback is a technique that aims to improve the relevance of search results by analyzing the initially retrieved documents, identifying relevant terms, and using them to expand the user's query [13][14]. Our system returns a set of documents for the input query through BM25. The most important terms and relevant terms from these documents are chosen and appended to the initial query to get a modified query. This modified query serves as an input to the HNSW neural network model and returns the final set of documents which will be returned to the user. Pseudo-relevance comes into play as the system assumes that the top-k documents returned by BM25 are all relevant without the user explicitly deciding.

As an example, Let's say a user is searching for information about "sustainable energy sources" using our web search engine system.

User Query: "Sustainable energy sources"

- The system retrieves an initial set of documents using the BM25 ranking algorithm based on the original query.
- Among the initial set of documents, it identifies common terms like "wind power," "solar

energy," and "geothermal energy" as relevant terms.

- It creates a modified query by adding these relevant terms to the original query:
"sustainable energy sources"

Modified Query: "Sustainable energy sources wind power solar energy geothermal energy"

The modified query is then used as input to the HNSW model and returns a final set of documents that are expected to be highly relevant to the user's query

Results

The results comprise the outputs of our pseudo-relevance feedback algorithm as well as a comparison of the output passages for the BM25 + semantic search algorithm, HNSW algorithm, BM25+HNSW ladder method, and the HNSW+BM25 reverse ladder method.

Relevance Feedback:

```
Given Query :: java
Post Pseudo Relevance Feedback Query::: java compilers programming compiler javaworld
```

```
Given Query :: java coffee
Post Pseudo Relevance Feedback Query::: coffee coffeemaker coffees brew brewing
```

```
Given Query :: java code
Post Pseudo Relevance Feedback Query::: java programming programmers programmer computing
```

```
Given Query :: java indonesia
Post Pseudo Relevance Feedback Query::: rabies indonesia jakarta indonesian papua
```

```
Given Query :: java script
Post Pseudo Relevance Feedback Query::: jscript javascript java scripts openscript
```

```
Given Query :: purple pride
Post Pseudo Relevance Feedback Query::: transgender flags lgbt pink colors
```

Queries regarding terms such as "java" encompass diverse meanings in various contexts, and a simplistic application of BM25 fails to adequately capture the nuanced nature of the input queries. Employing the approach elucidated in the pseudo-relevance feedback section, we observe how KeyBERT generates keywords based on the pertinent documents associated with queries containing such terms.

As illustrated in the aforementioned screenshots:

Query 1: "java"

The results predominantly pertain to computer science, indicative of the fact that the predominant theme across documents related to "java" predominantly involves the programming language.

Query 2: "java coffee"

The outcomes are exclusively associated with coffee and brewing, given the presence of the term "coffee" in the query. Consequently, the query becomes more focused on the interpretation of "java" in the context of coffee, aligning with the query's intent.

A comparable pattern is discernible in the remaining three queries containing the term "java," wherein the returned keywords are aligned with the contextual relevance of the term within the respective user queries.

The next segment contains a comparison of the output of the best four methods for returning the most relevant passages. There are four different types of queries we are testing it with.

1. Definition Queries: Queries whose purpose is to ask and get answers to direct questions, usually start with "what is" or "meaning of <word>".
2. Statement Queries: These queries are direct statements that will not necessarily have a question word in them. These queries usually ask for a piece of information and are usually in the form of phrases.
3. Detailed Query: These queries are usually much longer than the average query size. A detailed query refers to a comprehensive and specific request for information or data. It involves providing thorough and precise parameters or criteria to obtain highly specific and relevant results. This type of query typically seeks in-depth information, requiring a detailed examination or analysis to fulfill the user's needs.
4. Multi-Language Queries: Multi-language queries involve the use of multiple languages in a search or information retrieval process. This type of query is designed to accommodate users who may be proficient in or prefer different languages.

Definition Queries:

Query used: "what is ar balance"

Regarding the time required to retrieve results, a consistent pattern emerges across all queries moving forward. The swiftest passage retrieval is achieved by the BM25+HNSW method,

followed by HNSW in isolation, the BM25+HNSW Reverse Ladder Method, and finally the slowest performance is observed with the BM25+Semantic Search approach. It is worth noting that the reverse ladder method exhibits a marginally inferior performance compared to the standard ladder method. This discrepancy arises from the potential necessity for additional iterations of the BM25 algorithm when the initial query fails to yield any relevant passages.

In terms of the relevance of retrieved documents for a given query, all methods demonstrate commendable performance by returning documents that align well with the validation set. Moreover, it is noteworthy that the majority of the most pertinent documents are prominently positioned within the top 10 to 20 results. However, it is important to emphasize that the ordering of these results may differ among the methods, but their overall relevance remains consistently high.

BM25 + Semantic Search:

```

bm25+algo.ipynb × bm25+hnsw.ipynb × working.ipynb hnsw+bm25+reverse-ladder.ipynb hnsw.ipynb
bm25+algo.ipynb > M+Semantic Search > ⚙ import time
+ Code + Markdown | ▶ Run All ⏪ Restart ⏴ Clear All Outputs ⏴ Variables ⏴ Outline ...
Python 3.10.9

[15]: 
import time
query = "what is ar balance"
start = time.time()
result = semantic_search(query)
end = time.time()

print("Query :: ", query)
print("Took ", end-start, "s for the query")
for pid in result:
    print(str(pid)+" : ", d.fetchPassageContent(pid))
    print()

37.6s
...
Query :: what is ar balance
Took 36.294156074523926 s for the query
3923586 : If your company issues credit to its customers, you should be aware of accounts receivable insurance. Also referred to as credit insurance, accounts receivable insurance protects companies in the event that a customer fails to make its debt payments. By understanding how accounts receivable insurance works, you can choose whether this product is right for your business.

3923588 : What is trade credit insurance? Credit insurance is a product designed to protect what is most likely your company's largest asset, its accounts receivable. It covers credit losses caused by insolvency (bankruptcy), slow payment and, in the case of international transactions, it covers political risk as well. back to the top...

6126208 : Sub ledger is the subset of General ledger in the accounting terms. The relation between sub ledger to general ledger is many to one. i.e. there can be multiple sub ledger accounts linked to same general ledger account.

1616257 : Understand your result. The result of 40 indicates that the average accounts receivable collection period is 40 days. This means that the business owner can expect a credit sale to be paid by the customer within 40 days. This can help them plan for how much cash they need to have on hand for expenses and bills.

4636595 : The AICPA's founding defined accountancy as a profession characterized by educational requirements, professional standards, a code of professional ethics, and alignment with the public interest.

7354020 : The Accounting Principles Board (APB) is the former authoritative body of the American Institute of Certified Public Accountants (AICPA).

364132 : Upcoming Events. 1 New York - Financial Modeling Boot Camp - Aug. 23rd-25th, 2017. 2 Webinar: Private Equity Case - Direct Lending -- 5pm ET, 8/24/17. 3 Rewind - PE Case - Acquiring Minority Interest of Hedge Fund + Q&A 12am ET, 8/25/2017. 4 Rewind - Panel Discussion - How to Choose MBA Programs + Q&A 12am ET, 8/26/2017.

```

HNSW Only:

```

hnsw.ipynb > user_query = "what is ar balance"
+ Code + Markdown | Run All | Restart | Clear All Outputs | Variables | Outline ... Python 3.10.9
[1] user_query = "what is ar balance"
search_result = search_hnsw(user_query)
search_result['labels']

for pid in search_result['labels']:
    print(str(pid)+": ", d.fetchPassageContent(pid))
    print()

[1]: 0.3s
...
2770991 : Failure to address the issue of credit balances poses a number of risks for the provider, including the following: Lack of compliance with government payers. Failure to timely and accurately process all credit balances can result in significant fines, withholding of money and even imprisonment.

1351779 : The Balance® Bean herbicide will include an active ingredient (IFT) that inhibits the 4-hydroxyphenylpyruvate dioxygenase (HPPD) enzyme in plants. This enzyme is key to the production of pigments and the development of chloroplasts in plant tissues.

1351780 : Definition of balance. 1 : an instrument for weighing: such as : a beam that is supported freely in the center and has two pans of equal weight suspended from its ends; : a device that uses the elasticity of a spiral spring for measuring weight or force. 2 : a means of judging or deciding the balance of a free election.

364132 : Upcoming Events. 1 New York - Financial Modeling Boot Camp - Aug. 23rd-25th, 2017. 2 Webinar: Private Equity Case - Direct Lending -- 5pm ET, 8/24/17. 3 Rewind - PE Case - Acquiring Minority Interest of Hedge Fund + Q&A -- 12am ET, 8/25/2017. 4 Rewind - Panel Discussion - How to Choose MBA Programs + Q&A -- 12am ET, 8/26/2017.

962230 : If a third party processes your payroll, the company will most likely charge a monthly processing fee. Although you can post this as a separate journal entry, it may be simpler to include payroll fees in your payroll journal entry.

2651150 : 1 A display in a pilot's helmet that allows the pilot to, in effect, see through the aircraft. 2 Mobile marketing involving product information displayed over that product or its location. 3 Video games with digital elements blended into the user's environment.

```

BM25 + HNSW Ladder Method:

```

bm25+hnsw.ipynb > result = hnsw("what is ar balance")
+ Code + Markdown | Run All | Restart | Clear All Outputs | Variables | Outline ... Python 3.10.9
[1] result = hnsw("what is ar balance")
for pid in result:
    print(str(pid)+": ", d.fetchPassageContent(pid))
    print()

[1]: 0.2s
...
what is ar balance in semantic_search()
Cleaning up query :: ar balance
huggingface/tokenizers: The current process just got forked, after parallelism has already been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
    - Avoid using `tokenizers` before the fork if possible
    - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | false)
1291320 : What is the difference between a trial balance and a balance sheet? A trial balance is an internal report that will remain in the accounting department. It is a listing of all of the accounts in the general ledger and their balances. However, the debit balances are entered in one column and the credit balances are entered in another column. Each column is then summed to prove that the total of the debit balances is equal to the total of the credit balances.

2201928 : Rose has its origins in the Latin language, and it is used largely in English and French. The name evolved as a short form of compound Germanic names, introduced to England by the Normans in the 11th century in forms such as Rohese (Medieval) and Roese (Medieval).

2515316 : Answer Selected Answer: A trial balance Question 12 1 out of 1 points The primary purpose of the trial balance is to Answer Selected Answer: prove the equality of the debit and credit amounts after posting. Question 13 1 out of 1 points Which of the following statements is not true? Answer Selected Answer: Expenses increase

2528736 : A trial balance may balance even when each of the following occurs except when a. a transaction is not journalized. b. a journal entry is posted twice. c. incorrect accounts are used in journalizing. d. a transposition error is made. 134. A list of accounts and their balances at a given time is called a(n) a. journal. b. posting. c. trial balance. d.

1291323 : Purpose of a Trial Balance. 1 Trial Balance acts as the first step in the preparation of financial statements. 2 Trial balance ensures that for every debit entry recorded, a corresponding credit entry has been recorded in the books in accordance with the double entry concept of accounting.

```

HNSW + BM25 Reverse Ladder Method:

```
import time
start = time.time()
query = "what is ar balance"
result = hnsw(query, 100)
end = time.time()
print("\n\n\nQuery ::", query)
print("Took", end-start, " seconds to search")
print("Results")
for pid in result:
    print(str(pid)+" : ", d.fetchPassageContent(pid))
    print()

✓ 7.3s
```

what is ar balance in semantic_search()
[2770991, 1351779, 1351780, 364132, 962230, 2651150, 939873, 1678023, 939876, 939874, 1063155, 567301, 1774934, 2651144, 947878, 94005, 290746, 1702255, 960579, 939871, 538377, 2144936, 1:
New Query:: accountancy balances attribute balance regulated
huggingface/tokenizers: The current process just got forked, after parallelism has already been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
- Avoid using `tokenizers` before the fork if possible
- Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | false)
[3923586, 6105564, 5250974, 974059, 5321345, 1002002, 3660166, 820930, 820929, 3392795, 5608244, 5321343, 2373741, 364132, 6871197, 2108688, 2792812, 7278709, 2821255, 67480331, 1663742, 6:

Query :: what is ar balance
Took 7.377305269241333 seconds to search
Results

3923586 : If your company issues credit to its customers, you should be aware of accounts receivable insurance. Also referred to as credit insurance, accounts receivable insurance protec:
6105564 : The creditor may be able to charge late fees or interest if the amount is not paid by the due date. Booking a receivable is accomplished by a simple accounting transaction; how:
5250974 : Cash is an asset account with a normal ____balance. Capital is an owner's equity account with a normal ____balance. debit. When cash is paid for supplies, the supplies acco:
974059 : A transaction report is data submitted to us which contains information relating to a transaction. We use the reports to detect and investigate suspected market abuse. They may i:
5321345 : b A decrease in the accounts payable and allowance accounts. c An increase in the purchase return and allowance account. d An increase in the sales return and allowance account:
1002002 : It should also be noted that this article only focuses on a few types of gases. As a more comprehensive reference, ANSI/AWS A5.32/A5.32M Specifcation for Welding Shielding:
3660166 : As a conservator, guardian or personal representative in a supervised estate, you will be required to file inventories and accounts with the Court. Good recordkeeping will ena:

Statement Query

Query used: “abraham lincoln birthday”

In the context of time efficiency, a comparable pattern to the previously discussed trends persists. Both Semantic search and the BM25+HNSW Ladder method exhibit a tendency to present the foremost query result as marginally less relevant in comparison to the remaining results. Conversely, the other two methods consistently yield notably high similarity scores with the query, resulting in a collection of top passages that exhibit a commendable level of relevance.

BM25 + Semantic Search

The screenshot shows a Jupyter Notebook interface with the title "BM25 + Semantic Search". The top bar includes tabs for "bm25+algo.ipynb", "bm25+hnsw.ipynb", "working.ipynb", "hnsw+bm25+reverse-ladder.ipynb", and "hnsw.ipynb". The toolbar has options like "Code", "Markdown", "Run All", "Restart", "Clear All Outputs", "Variables", "Outline", and "Python 3.10.9". The code cell [11] contains:

```

    return final_passages
[11]   ✓ 0.0s

```

The code in cell [18] is:

```

import time
query = "abraham lincoln birthday"
start = time.time()
result = semantic_search(query)
end = time.time()

print("Query :: ", query)
print("Took ", end-start, "s for the query")

for pid in result:
    print(str(pid)+" : ", d.fetchPassageContent(pid))
    print()
[18]   ✓ 43.2s

```

The output in cell [18] shows several results, starting with:

```

Query :: abraham lincoln birthday
Took 42.96865701675415 s for the query
4735301 : About the Sciatic Nerve in the Low Back. The sciatic nerve starts in your low back, which is called your lumbar spine. The nerve roots are at the L4 and L5 vertebrae (the 'L' means lumbar, and the numbers indicate the level of the vertebrae where it is in your back). The sciatic nerve also travels through your pelvic region (sacrum).ciatica is defined as a severe pain in a leg along the course of the sciatic nerve. The pain is felt in the back of the leg running from the buttock down the back of the thigh into the calf and foot. 1 The pain may begin abruptly or gradually, and is characterized by a sharp, shooting, or electric shock-like quality.

2318260 : February 12, 1809 - Abraham Lincoln is born in a one-room log cabin on Nolin Creek in Kentucky. 1811 - In spring, the Lincoln family moves to a 230-acre farm on Knob Creek ten miles from Sinking Spring. 1812 - A brother, Thomas, is born but dies in infancy. 1815 - Young Abraham attends a log school house.

2998870 : Many Republican Party organizations hold Lincoln's Birthday celebrations because Lincoln was the first Republican president. Lincoln's Birthday is a legal, public holiday in some U.S. states, observed on the anniversary of Abraham Lincoln's birth on February 12, 1809 In Hodgenville,Kentucky. Connecticut, Illinois, Indiana, Ohio, Missouri, and New York observe the holiday.

7121967 : Lincoln's Birthday 2019 is a day that celebrates the birthday of Abraham Lincoln and serves as a day to honour as the nation's 16th president who known as the Great Emancipator. History of Lincoln's Birthday The history of Lincoln's Birthday 2019 dates back to February 12, 1809, when Abraham Lincoln was born in Hardin County, Kentucky.

7121971 : A few states even moved their observances of Washington's Birthday, Lincoln's Birthday, and Presidents' Day to November or December in order to lengthen the Thanksgiving and Christmas holiday periods without creating additional paid holidays.

```

HNSW Only

The screenshot shows a Jupyter Notebook interface with the title "HNSW Only". The top bar includes tabs for "bm25+algo.ipynb", "bm25+hnsw.ipynb", "working.ipynb", "hnsw+bm25+reverse-ladder.ipynb", and "hnsw.ipynb". The toolbar has options like "Code", "Markdown", "Run All", "Restart", "Clear All Outputs", "Variables", "Outline", and "Python 3.10.9". The code cell [10] contains:

```

user_query = "abraham lincoln birthday"
search_result = search_hnsw(user_query)
search_result['labels']

for pid in search_result['labels']:
    print(str(pid)+" : ", d.fetchPassageContent(pid))
    print()
[10]   ✓ 0.3s

```

The output in cell [10] shows:

```

2318260 : February 12, 1809 - Abraham Lincoln is born in a one-room log cabin on Nolin Creek in Kentucky. 1811 - In spring, the Lincoln family moves to a 230-acre farm on Knob Creek ten miles from Sinking Spring. 1812 - A brother, Thomas, is born but dies in infancy. 1815 - Young Abraham attends a log school house.

2998870 : Many Republican Party organizations hold Lincoln's Birthday celebrations because Lincoln was the first Republican president. Lincoln's Birthday is a legal, public holiday in some U.S. states, observed on the anniversary of Abraham Lincoln's birth on February 12, 1809 In Hodgenville,Kentucky. Connecticut, Illinois, Indiana, Ohio, Missouri, and New York observe the holiday.

2998868 : Lincoln's Birthday is an American holiday observed only by a select few states. The holiday is meant to celebrate the anniversary of the former president's birthday, February 12th, 1809. Connecticut, New York, Indiana, Illinois, Missouri, Arizona, and California all choose to celebrate this holiday. New Jersey used to celebrate Lincoln's birthday until it enacted what is called The Public Employee Pension and Benefits Reform Act of 2008.

```

BM25 + HNSW

```

bm25+hnsw.ipynb ×
bm25+hnsw.ipynb > M+KeyBERT - Query Expansion and Keyword Generation > empty cell
+ Code + Markdown | ▶ Run All ⚡ Restart ⌂ Clear All Outputs | ⓘ Variables ⓘ Outline ...
Python 3.10.9

▶ result = hnsw("abraham lincoln birthday")
for pid in result:
    print(str(pid)+" : ", d.fetchPassageContent(pid))
    print()

[12] ✓ 0.2s
...
abraham lincoln birthday in semantic_search()
Cleaning up query :: abraham lincoln birthday
huggingface/tokenizers: The current process just got forked, after parallelism has already been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
- Avoid using `tokenizers` before the fork if possible
- Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | false)
2824158 : Types of Sciatica Doctor. Specialists who concentrate on back and leg pain therapy come from a variety of backgrounds. Some are hands-on care providers, such as chiropractors, massage therapists and physical therapists. Others are diagnostic physicians, mostly coming from either an orthopedic or neurology background.

1750880 : My feet sometimes cramp when I curl my toes down. It hurts a lot, and I can break out of it by stretching my toes out again. I don't think you have anything to worry about. For the past few months I've noticed my middle #3 toe (#1 being the big toe), on my left foot seems to be growing apart from # 2 toe.

524762 : Overview: 1 Back pain is common. In the U.S., over 3 million visits to the emergency department are made annually for back pain symptoms. 2 Back pain can be acute, subacute, or chronic. It most often occurs in the lower part of the back. 3 With proper self-care, most acute cases resolve within 4 – 6 weeks.

1979476 : In regard to throwbacks, the Pomeranian's ancestors were much larger Spitz dogs. One in a great while, a gene from many generations can appear, which produces a Pomeranian that is double or triple the size of today's modern Pom. Leo, 15.5 pounds at 1 year, 8 months.

813825 : Causes of Sciatica. Sciatica is caused by inflammation or compression of the sciatic nerve or nerve roots in the lower spine. It may also be caused by damage to nerve roots. Sciatica is not a disease in itself and its causes are various. *Compression of a nerve is often referred to as a pinched nerve.. A pinched nerve in the lower spine causes sciatica. The most common cause of sciatica is a herniated disc in the lumbar spine (lower back) that puts pressure on the sciatic nerve or a nerve root. A herniated disc can occur suddenly and cause acute pain or it may develop gradually.

```

HNSW + BM25 Reverse Ladder Method

```

bm25+algo.ipynb bm25+hnsw.ipynb working.ipynb hnsw+bm25+reverse-ladder.ipynb • hnsw.ipynb
hnsw+bm25+reverse-ladder.ipynb > M+KeyBERT - Query Expansion and Keyword Generation > import time
+ Code + Markdown | ▶ Run All ⚡ Restart ⌂ Clear All Outputs | ⓘ Variables ⓘ Outline ...
Python 3.10.9

▶ import time
start = time.time()
query = "abraham lincoln birthday"
result = hnsw(query, 100)
end = time.time()
print("\n\nQuery ::", query)
print("Took", end-start, " seconds to search")
print("Results\n")
for pid in result:
    print(str(pid)+" : ", d.fetchPassageContent(pid))
    print()

[18] ✓ 0.9s
...
abraham lincoln birthday in semantic_search()
[2318260, 2998870, 2998868, 2998869, 2998867, 1791423, 2998872, 2998874, 842393, 2363935, 2285922, 251956, 1791416, 2998875, 157508, 2998873, 2365343, 1183048, 157505, 2365346, 2998866, 2551308, 157510, 2363930, 2318257, 2285923, 2379108, 2583854, 2332950, 157509, 2318261, 2224061, 775625, 38059, 2295589, 2379113, 2579069, 2438128, 2551310, 2285926, 2714064, 1263063, 2879615, 2714062, 1091226, 2379813, 2363928, 2569055, 1355978, 2749938, 2358106, 2879617, 1791419, 38616, 2721219, 255087, 946409, 946410, 2081614, 65012, 2690223, 157506, 1263058, 2591436, 2894307, 412809, 2140392, 2379110, 2379111, 842397, 1180565, 251954, 2464148, 2736551, 2332947, 1183054, 2267851, 1651992, 2633175, 1412349, 921201, 251957, 38612, 2270344, 2509013, 53404, 2797143, 218379, 2318259, 2767058, 2379114, 157511, 251955, 251952, 2920870, 1815404, 1183049, 1943547]
New Query:: lincoln 1809 lincoln 1865 1860s
[4735301, 2318260, 2998870, 7121971, 7121967, 7121968, 2081611, 2998874, 3623560, 2998875, 2998873, 4735307, 4735304, 5949549, 2998871, 3358169, 2998868, 2363935, 6003393, 4735310, 7121949, 1998069, 45030]

Query :: abraham lincoln birthday
Took 0.9932799339294434 seconds to search
Results

2318260 : February 12, 1809 – Abraham Lincoln is born in a one-room log cabin on Nolin Creek in Kentucky. 1811 – In spring, the Lincoln family moves to a 230-acre farm on Knob Creek ten miles from Sinking Spring. 1812 – A brother, Thomas, is born but dies in infancy. 1815 – Young Abraham attends a log school house.

2998870 : Many Republican Party organizations hold Lincoln's Birthday celebrations because Lincoln was the first Republican president. Lincoln's Birthday is a legal, public holiday in some U.S. states, observed on the anniversary of Abraham Lincoln's birth on February 12, 1809. In Hodgenville, Kentucky, Connecticut, Illinois, Indiana, Ohio, Missouri, and New York observe the holiday.

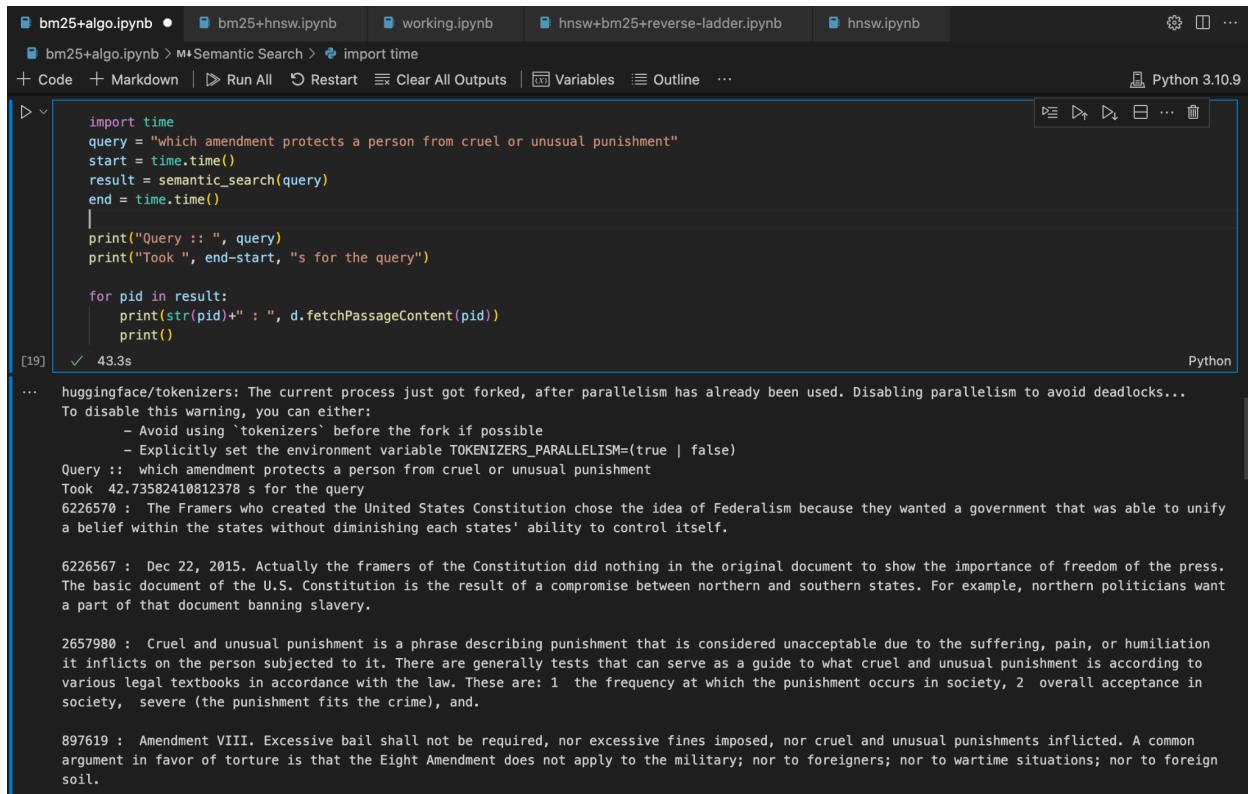
```

Long Queries

Query used: “which amendment protects a person from cruel or unusual punishment”

When it comes to longer queries, the reverse ladder method doesn't seem to do as well as the others. This is because of the limitations of finding conjunctive passages using BM25. This is usually taken care of in the ladder method as there is another search happening after BM25, and hence it performs better. But even with this setback, the reverse ladder method does get relevant documents in its top 20 results.

BM25 + Semantic Search



The screenshot shows a Jupyter Notebook interface with several tabs at the top: 'bm25+algo.ipynb' (selected), 'bm25+hnsw.ipynb', 'working.ipynb', 'hnsw+bm25+reverse-ladder.ipynb', and 'hnsw.ipynb'. Below the tabs, there's a toolbar with icons for Code, Markdown, Run All, Restart, Clear All Outputs, Variables, Outline, and Help. The Python version is listed as 'Python 3.10.9'. A code cell [19] contains the following Python code:

```
import time
query = "which amendment protects a person from cruel or unusual punishment"
start = time.time()
result = semantic_search(query)
end = time.time()

print("Query :: ", query)
print("Took ", end-start, "s for the query")

for pid in result:
    print(str(pid)+" : ", d.fetchPassageContent(pid))
    print()
```

The output cell [19] shows the execution time and the results of the search. It includes a warning about parallelism and lists several historical and legal references related to the query.

```
[19]: ✓ 43.3s
...
huggingface/tokenizers: The current process just got forked, after parallelism has already been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
  - Avoid using `tokenizers` before the fork if possible
  - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | false)
Query :: which amendment protects a person from cruel or unusual punishment
Took 42.73582410812378 s for the query
6226570 : The Framers who created the United States Constitution chose the idea of Federalism because they wanted a government that was able to unify a belief within the states without diminishing each states' ability to control itself.

6226567 : Dec 22, 2015. Actually the framers of the Constitution did nothing in the original document to show the importance of freedom of the press. The basic document of the U.S. Constitution is the result of a compromise between northern and southern states. For example, northern politicians want a part of that document banning slavery.

2657980 : Cruel and unusual punishment is a phrase describing punishment that is considered unacceptable due to the suffering, pain, or humiliation it inflicts on the person subjected to it. There are generally tests that can serve as a guide to what cruel and unusual punishment is according to various legal textbooks in accordance with the law. These are: 1 the frequency at which the punishment occurs in society, 2 overall acceptance in society, severe (the punishment fits the crime), and.

897619 : Amendment VIII. Excessive bail shall not be required, nor excessive fines imposed, nor cruel and unusual punishments inflicted. A common argument in favor of torture is that the Eighth Amendment does not apply to the military; nor to foreigners; nor to wartime situations; nor to foreign soil.
```

HNSW Only

A screenshot of a Jupyter Notebook interface titled "HNSW Only". The code cell contains Python code to search for "which amendment protects a person from cruel or unusual punishment" using the HNSW index. The output shows several legal citations and their descriptions:

```
user_query = "which amendment protects a person from cruel or unusual punishment"
search_result = search_hnsw(user_query)
search_result['labels']

for pid in search_result['labels']:
    print(str(pid)+" : ", d.fetchPassageContent(pid))
    print()
```

[11] 0.3s

... 643663 : You can bring a claim for sexual assault even if it does not result in physical injury. One good case to read on this issue is *Schwenk v. Hartford*, 204 F.3d 1187, 1196-97 (9th Cir. 2000). Also, rape and sexual assault need not be committed by a prison guard in order to violate the Eighth Amendment. Courts have held that people who are similar to prison guards, such as supervisors in prison work programs, also violate the Constitution by assaulting prisoners.

2448103 : Corrections officials must be aware that the revitalized Eighth Amendment has implications for corrections, especially because the U.S. Supreme Court's controlling precedent, *Trop v. Dulles* (1958), endorses a flexible, evolving definition of cruel and unusual punishments. In the second case, the U.S. Court of Appeals for the 6th Circuit found an Eighth Amendment violation in Michigan's policy of terminating inmate visits as a means of punishing substance abuse violations (*Bazetta v. McGinnis*, 2002).

2318980 : AMENDMENT 8&C PUNISHMENT FOR CRIME 1579 80 Woodson v. North Carolina, 428 U.S. 280 (1976); *Roberts v. Louisiana*, 428 U.S. 325 (1976). Justices Stewart, Powell, and Stevens composed the plurality, and Justices Brennan and Marshall concurred on the basis of their own views of the death penalty. 428 U.S. at 305, 306, 336.

188617 : [36] Now, merely inadvertent failure to adequately diagnose[e] or treat the plaintiff's medical condition might be sufficient to make the defendant liable in a medical malpractice or negligence action. However, such an inadvertent failure is not sufficiently reckless to establish a claim under the Eighth Amendment.

2515926 : The Constitution of the Marshall Islands, in the sixth section of its Bill of Rights (Article 2), prohibits cruel and unusual punishment, which it defines as: the death penalty; torture; inhuman and degrading treatment; and excessive fines or deprivations. United States

897619 : Amendment VIII. Excessive bail shall not be required, nor excessive fines imposed, nor cruel and unusual punishments inflicted. A common argument in favor of torture is that the Eighth Amendment does not apply to the military; nor to foreigners; nor to wartime situations; nor to foreign soil.

BM25 + HNSW Ladder Method

A screenshot of a Jupyter Notebook interface titled "BM25 + HNSW Ladder Method". The code cell contains Python code to search for "which amendment protects a person from cruel or unusual punishment" using the BM25+HNSW ladder method. The output shows several legal citations and their descriptions, similar to the HNSW only results but with some additional context about parallelism issues:

```
result = hnsw("which amendment protects a person from cruel or unusual punishment")
for pid in result:
    print(str(pid)+" : ", d.fetchPassageContent(pid))
    print()
```

[13] 1.4s

... which amendment protects a person from cruel or unusual punishment in semantic_search()
Cleaning up query :: amendment protects person cruel unusual punishment
huggingface/tokenizers: The current process just got forked, after parallelism has already been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
- Avoid using `tokenizers` before the fork if possible
- Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | false)
huggingface/tokenizers: The current process just got forked, after parallelism has already been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
- Avoid using `tokenizers` before the fork if possible
- Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | false)

2095589 : The Eighth Amendment to the United States Constitution prohibits the federal government from imposing excessive bail, excessive fines, or cruel and unusual punishment. The VIII Amendment (Eighth Amendment) to the Constitution of Pakistan, was short-time amendment to the Constitution of Pakistan, which was passed by the Majlis-e-Shoo & ra, in the absence of elected Parliament of Pakistan, in 1985.

2448100 : Definition of inhibit written for English Language Learners from the Merriam-Webster Learner's Dictionary with audio pronunciations, usage examples, and count/noncount noun labels. Learner's Dictionary mobile search

2515922 : the right to protection against Cruel and unusual punishments 169 sense to the inmate's lawyers: He died mentally. . . . No man should have to go to the chair twice. The voice of humanity and justice cries out against such an outrage. . . . [I]s this an experiment in modern forms of torture. . . . Is the state

2095593 : The VIII Amendment (Eighth Amendment) to the Constitution of Pakistan, was short-time amendment to the Constitution of Pakistan, which was passed by the Majlis-e-Shoo & ra, in the absence of elected Parliament of Pakistan, in 1985. The VIII Amendment (Eighth Amendment) to the Constitution of Pakistan, was short-time amendment to the Constitution of Pakistan, which was passed by the Majlis-e-Shoo & ra, in the absence of elected Parliament of Pakistan, in 1985.

HNSW + BM25 Reverse Ladder Method

The screenshot shows a Jupyter Notebook interface with several tabs at the top: 'bm25+algo.ipynb', 'bm25+hnsw.ipynb', 'working.ipynb', 'hnsw+bm25+reverse-ladder.ipynb' (which is the active tab), and 'hnsw.ipynb'. Below the tabs, there's a toolbar with icons for 'Code', 'Markdown', 'Run All', 'Restart', 'Clear All Outputs', 'Variables', 'Outline', and a Python version indicator 'Python 3.10.9'. The main area contains a code cell and its output. The code cell contains:import time
Execute Cell (^Enter) me.time()
query = "which amendment protects a person from cruel or unusual punishment"
result = hnsw(query, 100)
end = time.time()
print("\n\nQuery ::", query)
print("Took", end-start, " seconds to search")
print(["Results\n"])[0]
for pid in result:
 print(str(pid)+": ", d.fetchPassageContent(pid))
 print()

```
[22]    ✓ 6.9s
```

The output shows the execution time and the results of the search. The results are displayed in two parts: a warning about parallelism and the actual search results. The search results are as follows:

```
... which amendment protects a person from cruel or unusual punishment in semantic_search()  
New Query:: incarceration sentencing punishments correctional prisons  
huggingface/tokenizers: The current process just got forked, after parallelism has already been used. Disabling parallelism to avoid deadlocks...  
To disable this warning, you can either:  
- Avoid using `tokenizers` before the fork if possible  
- Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | false)  
huggingface/tokenizers: The current process just got forked, after parallelism has already been used. Disabling parallelism to avoid deadlocks...  
To disable this warning, you can either:  
- Avoid using `tokenizers` before the fork if possible  
- Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | false)
```

Query :: which amendment protects a person from cruel or unusual punishment
Took 6.951469898223877 seconds to search
Results

1168453 : One of the most important rights of an individual formally charged with a serious crime is the right to a jury trial. This right is guaranteed in Article III of the Constitution and by the Sixth Amendment.

2095590 : The Eighth Amendment to the U.S. Constitution reads: Excessive bail shall not be required, nor excessive fines imposed, nor cruel and unusual punishments inflicted. The Eighth Amendment to the U.S. Constitution, ratified in 1791, has three provisions. The cruel and unusual punishments clause restricts the severity of punishments that state and federal governments may impose upon persons who have been convicted of a criminal offense. Courts are given wide latitude under the Excessive Fines Clause of the Eighth Amendment. Fines imposed by a trial court judge or magistrate will not be overturned on appeal unless the judge or magistrate abused his or her discretion in assessing them (United States v. Hyppolite, 65 F.3d 1151 [4th Cir.]

Other Languages in English

Query Employed: "panthera leo"

In this specific instance, the reverse ladder method exhibits notably superior performance when compared to the other methods. This can be attributed to a specific workflow inherent in the reverse ladder method, wherein queries of a Latin nature, such as the definition of a word, initially traverse the HNSW model. Subsequently, the HNSW model generates a set of keywords derived from the query, which are then seamlessly integrated into the query itself before being processed by the BM25 algorithm. This strategic approach enables the BM25 algorithm to effectively search for matches related to words such as "panthera" or "leo," which, in the absence of this preprocessing step, might have a heightened likelihood of being overlooked when seeking nearest neighbors rather than direct textual matches. Consequently, when dealing with queries of this nature, the reverse ladder method consistently yields optimal results.

BM25 + Semantic Search

The screenshot shows a Jupyter Notebook interface with the title "BM25 + Semantic Search". The code cell contains Python code to perform a search for "panthera leo" using BM25 and semantic search, and then prints the results. The output shows several search results, including detailed classifications and descriptions for various lion subspecies and historical mentions.

```

import time
query = "panthera leo"
start = time.time()
result = semantic_search(query)
end = time.time()

print("Query :: ", query)
print("Took ", end-start, "s for the query")

for pid in result:
    print(str(pid)+" : ", d.fetchPassageContent(pid))
    print()

[20]  ✓  41.0s
Python

... huggingface/tokenizers: The current process just got forked, after parallelism has already been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
    - Avoid using `tokenizers` before the fork if possible
    - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | false)
Query :: panthera leo
Took 40.57480001449585 s for the query
6141792 : Classification. Lions belong to the genus Panthera which contains well known animals such as the tiger, leopard, and jaguar. Within the genus Panthera, the lion is further classified as the species Panthera leo. The full scientific classification is as follows: 1 Kingdom: Animalia. 2 Phylum: Chordata. 3 Class: Mammalia. 4 Order: Carnivora. 5 Family: Felidae. 6 Genus: Panthera. Within the genus Panthera, the lion is further classified as the species Panthera leo. The full scientific classification is as follows: 1 Kingdom: Animalia. 2 Phylum: Chordata. 3 Class: Mammalia. 4 Order: Carnivora. 5 Family: Felidae. 6 Genus: Panthera.

6141794 : Two subspecies are currently recognised: the African lion (Panthera leo leo) and the Asiatic lion (Panthera leo persica). The Asiatic lion is slightly smaller than its African cousin, and has a shorter, thinner mane and a fold of skin running the length of the belly that is rare in African lions. Two subspecies are currently recognised: the African lion (Panthera leo leo) and the Asiatic lion (Panthera leo persica). The Asiatic lion is slightly smaller than its African cousin, and has a shorter, thinner mane and a fold of skin running the length of the belly that is rare in African lions.

7303959 : Meaning: Lions; leopards; snow leopards; jaguars; tigers; cheetahs; saber-toothed tigers. Classified under: Nouns denoting animals. Synonyms: genus Panthera; Panthera. Hypernyms (Panthera is a kind of...): mammal genus (a genus of mammals) Meronyms (members of Panthera): leopard; Panthera pardus (large feline of African and Asian forests usually having a tawny coat with black spots) ounce; Panthera uncia; snow leopard (large feline of upland central Asia having long thick whitish fur)

7303958 : Pantera is the name of a Roman soldier buried in Germany believed to be the father of Jesus Christ, ancient Hebrew text point to this possibility, a statue of Pantera from ancient Roman times was discovered in Germany at his burial site , he lived to be 62.

```

HSNW

The screenshot shows a Jupyter Notebook interface with the title "HSNW". The code cell contains Python code to perform a search for "panthera leo" using the HNSW index, and then prints the results. The output shows search results, including a detailed description of the lion's mane and its role in intimidation.

```

user_query = "panthera leo"
search_result = search_hnsw(user_query)
search_result['labels']

for pid in search_result['labels']:
    print(str(pid)+" : ", d.fetchPassageContent(pid))
    print()

[12]  ✓  0.5s
Python

... 1061359 : The mane of the adult male lion, unique among cats, is one of the most distinctive characteristics of the species. It may provide an excellent intimidation display; aiding the lion during confrontations with other lions.

2393144 : Many lions in eastern Africa are also being lost to canine distemper. The last wild lion in Africa north of the Sahara was shot in Morocco in 1920. Where do lions live outside Africa? Lions were once far more widespread than they are today. In historic times they were found throughout much of southern Eurasia (map of the lion's former range). Anciently, they were also present in Greece and the Balkan Peninsula. For example, they are mentioned repeatedly in the Iliad.

1074235 : Within the genus Panthera, the lion is further classified as the species Panthera leo. The full scientific classification is as follows: 1 Kingdom: Animalia. 2 Phylum: Chordata. 3 Class: Mammalia. 4 Order: Carnivora. 5 Family: Felidae. 6 Genus: Panthera. 7 Species: Panthera leo. Within the genus Panthera, the lion is further classified as the species Panthera leo. The full scientific classification is as follows: 1 Kingdom: Animalia. 2 Phylum: Chordata. 3 Class: Mammalia. 4 Order: Carnivora. 5 Family: Felidae. 6 Genus: Panthera. 7 Species: Panthera leo.

```

BM25 + HNSW Ladder approach

```

bm25+hnsw.ipynb ×
bm25+hnsw.ipynb > M+KeyBERT - Query Expansion and Keyword Generation > empty cell
+ Code + Markdown | ▶ Run All ⌘ Restart ⌘ Clear All Outputs | 🐞 Variables ⌂ Outline ...
Python 3.10.9

result = hnsw("panthera lion")
for pid in result:
    print(str(pid)+" : ", d.fetchPassageContent(pid))
print()

[14] ✓ 0.3s
...
panthera lion in semantic_search()
Cleaning up query :: panthera lion
huggingface/tokenizers: The current process just got forked, after parallelism has already been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
- Avoid using `tokenizers` before the fork if possible
- Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | false)
343230 : Extraordinary sightings of the Timbavati White Lions at Kings Camp have been enjoyed by all. It looks like the Kubasa Pride have moved in and it looks like they are here to stay. The two older Machaton lionesses, (Djuma & Sengela) had an ill-fated run-in with this pride close to the camp and came out second best.

2291465 : Lions are the dominant carnivores in their habitat and will drive away competitors or even kill them. The Lion's head and body can be up to eight feet, two inches, and tail up to three feet, five inches. Its weight can be up to 550 pounds. Lions are primarily ground-dwellers, but occasionally jump up tree branches. Most Lions will remain in the same territory all year long, however some are nomadic and follow the seasonal prey.

952710 : African Lion. Apart from the humans squeezing African lions off their lands, the King of Beasts has no natural enemies. Of the big cats, only the tiger is larger, although not by much. Adult male lions can weigh 330 to 573 pounds, their luxuriant manes making them appear even larger; females are smaller, with weights not exceeding 400 pounds. Lions live in matriarchal groups called prides that, on average, consist of four to six related lionesses and their cubs.

952709 : Female lions are smaller than males, and usually weigh around 280 pounds. Male lions are usually around 4 feet tall, and lionesses are slightly shorter. Not including their tails, male lions are typically 6 feet to 7 feet long, with females again being slightly smaller.

1518005 : 11 thoughts on African Lion vs Hyenas . 1 Female hyenas sport massive quantities of testosterone, this enables them to bitch down their own males, & even intimidate female lions at kills.. 2 You missed a large set of Hyena facts here, they have pound for pound the strongest bite force of any mammal on the planet.

343228 : ABOUT THE SHOW. The Kruger National Park is home to one of the largest pride of lions in Africa. This real life soap opera tells the story of how the 26-strong community interact. In the Kruger National Park there is a hunting force that rivals anything else in Africa: a lion pride, 26 members strong!

```

HNSW + BM25 Reverse Ladder

```

bm25+algo.ipynb bm25+hnsw.ipynb working.ipynb hnsw+bm25+reverse-ladder.ipynb ● hnsw.ipynb
hnsw+bm25+reverse-ladder.ipynb > M+KeyBERT - Query Expansion and Keyword Generation > def query_passage_similarity(query, pid):
+ Code + Markdown | ▶ Run All ⌘ Restart ⌘ Clear All Outputs | 🐞 Variables ⌂ Outline ...
New Q... Aa ab ab* 1 of 1 ↑ ↓ × 9

import time
start = time.time()
query = "panthera leo"
print("Given Query ::", query)
result = hnsw(query, 100)
end = time.time()

print("\n\n\nTook", end-start, " seconds to search")
print("Results")
for pid in result:
    print(str(pid)+" : ", d.fetchPassageContent(pid))
    print()

[64] ✓ 1.3s
...
Given Query :: panthera leo
Post Pseudo Relevance Feedback Query:: lion panther lionesses panthera felines
huggingface/tokenizers: The current process just got forked, after parallelism has already been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
- Avoid using `tokenizers` before the fork if possible
- Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | false)

Took 1.388951063156128 seconds to search
Results

6141792 : Classification. Lions belong to the genus Panthera which contains well known animals such as the tiger, leopard, and jaguar. Within the genus Panthera, the lion is further classified as the species Panthera leo. The full scientific classification is as follows: 1 Kingdom: Animalia. 2 Phylum: Chordata. 3 Class: Mammalia. 4 Order: Carnivora. 5 Family: Felidae. 6 Genus: Panthera. Within the genus Panthera, the lion is further classified as the species Panthera leo. The full scientific classification is as follows: 1 Kingdom: Animalia. 2 Phylum: Chordata. 3 Class: Mammalia. 4 Order: Carnivora. 5 Family: Felidae. 6 Genus: Panthera.

1074234 : Humans, of course, are Homo sapiens. The full classification for a lion would be: Kingdom, Animalia (animals); Phylum, Chordata (vertebrate animals); Class, Mammalia (mammals); Order, Carnivora (meat eaters); Family, Felidae (all cats); Genus, Panthera (great cats); Species, leo (lions). Scientific Classification. Classification, or taxonomy, is a system of categorizing living things. There are seven divisions in the system: (1) Kingdom; (2) Phylum or Division; (3) Class; (4) Order; (5) Family; (6) Genus; (7) Species. Kingdom is the broadest division.

```

We have experimented with using a similarity function (like diversity@k and cosine similarity) to check how similar the top passages are with the input query and our explanations are also partly based on that, but we found out that the method usually doesn't consider the context and returns very low scores for even the most relevant documents.

Showstoppers and Future Work

- While MiniLM is utilized in this project, future work could involve exploring and experimenting with other dense retrieval models such as BERT, RoBERTa, or even more advanced models that might be available at the time
- The project currently employs HNSW for approximate nearest neighbor search. Future work could focus on fine-tuning the parameters of HNSW or exploring other ANN methods (e.g., DiskANN, NSG) to further optimize and speed up the retrieval process.
- Further research and experimentation with evaluation metrics to ensure the effectiveness of the proposed system.
- Explore the prospect of using relevance feedback directly instead of pseudo-relevance feedback
- Investigate the adaptability of the system to multilingual or cross-lingual retrieval tasks, as this can have practical applications in diverse linguistic environments.

How to run the code

Open the .ipynb file for each of the algorithms and run the cells one at a time.

The repository structure is given below with the various files that are a part of this project.

```
- EmbeddingGeneration
  - data
    Contains the collection dataset. This is obtained from
    https://microsoft.github.io/msmarco/TREC-Deep-Learning-2020

  - embeddings
    Empty initially. Will be filled up by generateEmbeddings and embeddingConcatenator.
    It will host the individual embeddings for multiple chunks of passages.
    It will host the combined_corpus_embeddings file

  - generateEmbeddings.ipynb
    Dependencies: pip install SentenceTransformer

    Generates the embeddings in chunks

  - embedding Concatenator.py
    Dependencies: already taken care of by previous notebook

    This file combines the corpus_embeddings files from previous file into a single file.
    If you want to skip these embedding generation phase, download the embeddings from our drive repo:
    combined_corpus_embeddings: https://drive.google.com/file/d/1wD-RjVjj7PeiENHKEeLtKMJWB17_YtDe/view?usp=sharing
    corpus: https://drive.google.com/file/d/1ux72i05nou1BmKi2LfPebXshfm3bxIIS/view?usp=sharing
```

- Algorithms

Dependencies:

1. Make sure to unzip binIndices.zip in the same directory.
Download this from https://drive.google.com/file/d/1swYu8i8ohV_zGEIMZ1sdMMZd_ukZSiAy/view?usp=sharing
2. pip install nltk and related dependencies.
3. pip install hnswlib
4. Run g++ bm25.cpp -o bm25 -std=c++14 and generate the bm25 executable

- maps, finalMap.bin, binIndices.zip
Contain the inverted indices, and the dependent files for bm25

- bm25.cpp
Hosts the code that exposes the bm25 functionality as an executable.

- cosineSimilaritySearch.py
Algorithm for traditional cosine similarity search using all-MiniLM-L6-v2 transformed passage embeddings
Provide the query in the main function and run the file using
python3 cosineSimilaritySearch.py

- bm25+algo.ipynb
Algorithm that uses BM25 followed by Semantic Search Algorithm
Algorithm that uses BM25 followed by Nearest Neighbors Algorithm
Query expansion logic using KeyBERT

- bm25+algo_vocabularyfeedback.ipynb
Algorithm that uses KeyBERT for Query Expansion and Keyword Generation
Performs Semantic Search and Nearest Neighbour search over expanded query

- bm25+hnsw.ipynb
Ladder algorithm that uses BM25 to find the most relevant documents followed by HNSW algorithm to refine the results

- hnswConstruction.ipynb
Contains the logic to generate the HNSW index file for all the embeddings generated previously.
Make sure to perform Embedding Generation before this step.
If you want to skip this step, use this link to download the hnsw index bin file:
<https://drive.google.com/file/d/1zqZfbKm-tqDH7zdoHIuXX8oQiJfbFd4E/view?usp=sharing>

- hnsw+bm25+reverse-ladder.ipynb
Inverse Ladder method that uses HNSW to get the documents first for a query and performs a keyword search on the returned documents to formulate a new query. This is passed to BM25 to get the most relevant documents

- hnsw.ipynb
Houses the code to test just the hnsw results for a given query

- docContentManager.py
Reads the corpus from EmbeddingGeneration/embeddings/corpus
For a given pid, it returns the passage content

- diversityScore.py
Hosts the importable functions to calculate the diversity between passages returned by the various algorithms and the similarity score between a passage and the query.

References

1. Bajaj et al. (2016) in "Ms marco: A human generated machine reading comprehension dataset," published in arXiv preprint arXiv:1611.09268.
2. Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).
3. Reimers and Gurevych (2019) introduced "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks" in the Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, published by the Association for Computational Linguistics. <https://arxiv.org/abs/1908.10084>
4. Wang, Wenhui, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. "Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers." *Advances in Neural Information Processing Systems* 33 (2020): 5776-5788.
5. Kassim, Junaidah Mohamed, and Mahathir Rahmany. "Introduction to semantic search engine." In *2009 International Conference on Electrical Engineering and Informatics*, vol. 2, pp. 380-386. IEEE, 2009.
6. Liu, Jiawen, Zhen Xie, Dimitrios Nikolopoulos, and Dong Li. "{RIANN}: Real-time incremental learning with approximate nearest neighbor on mobile devices." In *2020 USENIX Conference on Operational Machine Learning (OpML 20)*. 2020.
7. Han, Jianfeng, Xuefei Guo, Runcheng Jiao, Yun Nan, Honglei Yang, Xuan Ni, Danning Zhao et al. "An Automatic Method for Delimiting Deformation Area in InSAR Based on HNSW-DBSCAN Clustering Algorithm." *Remote Sensing* 15, no. 17 (2023): 4287.
8. Issa, Bayan, Muhammed Basheer Jasser, Hui Na Chua, and Muzaffar Hamzah. "A Comparative Study on Embedding Models for Keyword Extraction Using KeyBERT Method." In *2023 IEEE 13th International Conference on System Engineering and Technology (ICSET)*, pp. 40-45. IEEE, 2023.
9. Sammet, Jill, and Ralf Krestel. "Domain-Specific Keyword Extraction using BERT." In *Proceedings of the 4th Conference on Language, Data and Knowledge*, pp. 659-665. 2023.
10. KeyBERT Github repository: <https://github.com/MaartenGr/KeyBERT>
11. Broder, Andrei Z., Peter Ciccolo, Marcus Fontoura, Evgeniy Gabrilovich, Vanja Josifovski, and Lance Riedel. "Search advertising using web relevance feedback." In *Proceedings of the 17th ACM conference on information and knowledge management*, pp. 1013-1022. 2008.

12. Ruthven, Ian, and Mounia Lalmas. "A survey on the use of relevance feedback for information access systems." *The Knowledge Engineering Review* 18, no. 2 (2003): 95-145.
13. Xu, Yang, Gareth JF Jones, and Bin Wang. "Query dependent pseudo-relevance feedback based on wikipedia." In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp. 59-66. 2009.
14. Reuben, Maor, Aviad Elyashar, and Rami Puzis. "Iterative query selection for opaque search engines with pseudo relevance feedback." *Expert Systems with Applications* 201 (2022): 117027.
15. Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to Information Retrieval. Cambridge University Press