Information Retrieval and Web Search Project (Winter 2023) - Group 5

Topic : AthleticAlyze: A Twitter Sports Search Engine

Team Members:

1.Gayatri Bhosale (*gbhos002*)
2.Manoj Nagarajappa (*mnaga024*)
3.Puneet Singhania (*psing088*)
4.Sanjana Senthilkumar (*ssent013*)
5.Susmita Purushothaman (*spuru001*)

Data Source: **Twitter**

Collaboration details:

All team members took part in deciding the general project workflow to implement in this part. Manoj Nagarajappa worked on utilizing the PyTorch to create embeddings for each passage in the data and store them in a FAISS index. Gayatri Bhosale and Puneet Singhania worked on developing the BERT indexer and retriever. Sanjana Senthilkumar and Susmita Purushothaman calculated BERT's indexing time and contrasted it with Lucene's indexing time. Puneet Singhania worked on creating the flask web application that accepts a query and an index type preference and outputs the top-k results. Manoj Nagarajappa and Gayatri Bhosale worked on adding more features to the web application like beautifying the homepage and adding tables for the output results. Sanjana Senthilkumar worked on showing the results on a map. Susmita Purushothaman worked on contrasting the Lucene and BERT rankings for quality and providing instances for the same.

We all then created the web-application capable of being installed on a web server. We tested the code end-to-end. All of the members worked on integrating the system's various parts and resolving any problems or issues that came up while the project was being developed. All team members worked on the preparation of the report, describing the architecture, the specifics of how BERT was applied, documenting the overall system design, preparation of the presentation and video.

Overview of the BERT indexer system and Web application for Query Interface:

The BERT indexer system and the web application for query interface are built to deliver efficient search results for a specific query entered by the user. We are providing two choices to the user to create and display indexes for the same query entered - Lucene indexer and BERT indexer.

For our search engine, we focused on tweets related to sports. We chose python as our scripting language and utilized Jupyter Notebook for our execution. We had used Tweepy to crawl the tweets and used the same dataset collected earlier for this phase of the project. We had collected

520MB (more than 500MB) of data. Since the tweet URLS were missing in the dataset for Part A, we have now also added tweet URLs for Part B. For each passage of the supplied data, the indexer is in charge of producing dense embeddings using the BERT implementation. Afterwards, a FAISS index is utilized to store the embeddings. This is done so that retrieval can be done during query time. To maintain compatibility with the BERT paradigm, each passage is tokenized using the BERT tokenizer and trimmed to 512 tokens. The FAISS package is highly tuned for similarity search on dense vectors so we used it to build the index. As a part of the web interface, the users can enter a query and then pick the index through using the query interface (Lucene index or the BERT index). The query is routed through the BERT model to create a dense embedding. We save those embeddings. Afterwards, we search the FAISS index and retrieve the most pertinent passages for the BERT index. The top-k passages are returned to the user after the passages are ranked by their cosine similarity to the query embedding.

We developed a user interface for the web application utilizing the Flask framework. The output is a results page that lists the top k passages, their relevance score, and a snippet of the text. Also, for Lucene indexer, we had several geotagged tweets which we represented on a map for easy visualization.

Our indexer system, utilizing the power of pyLucene and BERT along with a seamless web application integrating the map, offer a strong tool for efficient search of big text corpora, enabling users to identify pertinent data and insights.

<u>Architecture:</u>

There are two fundamental parts of our indexer system and web application. Those are the querying component and the indexing component.

For the BERT indexer, the indexing component uses a pre-trained BERT model(using distilroberta) to create dense embeddings from the input data, the input is ingested in batches to create tokens, then encode the collection/batch of sentences, tokenize the embeddings using the BERT tokenizer for the batch, and then uses FAISS to index those embeddings. The indexed embeddings are stored in a file for retrieval at a later time.

A user query and an index selection (PyLucene or BERT) are passed to the querying component. The component retrieves the pertinent embeddings from the FAISS index and ranks them according to cosine similarity if the user selects BERT. The user receives the top-10 results.

The Flask-based Web application offers a user-friendly interface for querying the indexed data. An input textbox, a radio button to choose the index are the major parts of the user interface. The results are shown in a table. The Google Maps API is used to display the results on a map if the indexed data includes coordinates.

In general, our architecture offers a versatile and effective method for indexing and querying enormous amounts of text data utilizing both sparse and dense representations.

Details of how BERT was used:

We generated embeddings using a BERT-based model. The model used for this job was "all-distilroberta-v1". Utilizing the Hugging Face Transformers library, the model was loaded. The FAISS library was utilized for a quick and effective search through vast collections of embeddings. It indexed the embeddings produced by BERT. The embeddings have already been indexed using FAISS, as the index used in the code was loaded from a pre-existing index file.

The same BERT model is used to convert the user's search query into an embedding when it is entered. Afterwards, using FAISS, this embedding is utilized to search through the indexed embeddings. The top 10 most similar embeddings, which correspond to the most comparable tweets in the collection, are returned by the search. Lastly, for each of the top 10 tweets, the tweet text and matching distance score (similarity score) are printed in the UI.

To explain in more detail, We find related documents based on a query using Faiss. This is a library for similarity search and clustering of dense vectors.The transformers library for BERT and faiss for similarity search were imported. The Hugging Face transformers library's BERT model and tokenizer are then loaded, and a Faiss index is created for similarity searching. The script then compiles a list of distinct tweets using data from a csv file. The BERT model is then used to process tweets in batches, turning each tweet into an embedding vector. The Faiss index is then updated to include the embeddings for similarity search. Also, the script has a method for turning a query into an embedding vector, which is then used to search the Faiss index for documents that are comparable to the query. The top 10 tweets that are the most comparable to the query are then printed.

Explain how you use the BERT index to do the ranking:

We generate an index of BERT embeddings for a collection of tweets using the FAISS package. The embeddings were produced using the "all-distilroberta-v1" BERT model. The tokenizer "all-distilroberta-v1" was also used to tokenize the input tweets. The final hidden state output from the model is used to create the BERT embeddings. The user inputs a search term to look for pertinent tweets based on the query. The same BERT model and tokenizer are used to convert the search query into a BERT embedding. Afterwards, to locate the documents closest to the search, the cosine similarity between the query embedding and the embeddings of all the indexed pages is calculated. The documents that are deemed to be the most similar to the query are those with the highest cosine similarity scores.

We order the obtained documents based on their cosine similarity scores. The highest-scoring documents are deemed to be the most pertinent to the query and are sorted as such.

We divide the input document(tweets) into passages. The batch size parameter determines the length of each passage. The batched tweets list is used to process the tweets in batches after they have initially been read from a csv file. The amount of tweets in each batch is batch size, and a transformer model from the Hugging Face library is used to encode each tweet. A FAISS index is created as a result, and it is then written to disk along with the embeddings. Up till all tweets have been processed, the procedure continues.

The input query is afterwards encoded using the same transformer model and used to search the Faiss index for the passages with the highest similarity. The passages are then located and provided as the search's output. As a result, by encoding the input documents as embeddings and indexing them using a nearest neighbor search technique, the input documents are efficiently divided into passages.

Utilization of the Lucene index to return the results:

The purpose of the Lucene indexer is to store and retrieve information related to tweets. To find tweets based on a specified term or hashtag, this method utilizes the Lucene index. The index tweets method reads a CSV file and, using a TextField for each field, builds an index of the tweets, including the tweet text and any associated hashtags. A query string and field name are passed to the search tweets method, which employs a QueryParser to produce a query object for the chosen field. The results are returned with the accompanying tweet and hashtag content once the index has been searched using the query using the IndexSearcher.

Depending on the kind of field being searched, either the StandardAnalyzer or the KeywordAnalyzer should be used with the QueryParser. For the tweet text box, the StandardAnalyzer is utilized since it analyzes the text by tokenizing, deleting stop words, and stemming, all of which can enhance the quality of search results. The hashtag field uses the KeywordAnalyzer since it does not tokenize or stem the text and instead considers it as a single token.

The MultiFieldQueryParser can be given a list of field names so that it can configure the QueryParser to search over several fields. Using the IndexSearcher and the query object the parser constructed, you can then search the index.

Runtime of the BERT index construction:

Below is a performance analysis of BERT indexing, that is, the *run time of the BERT index* creation process. A graph with *run time on the y axis* and *number of documents on the x axis*.

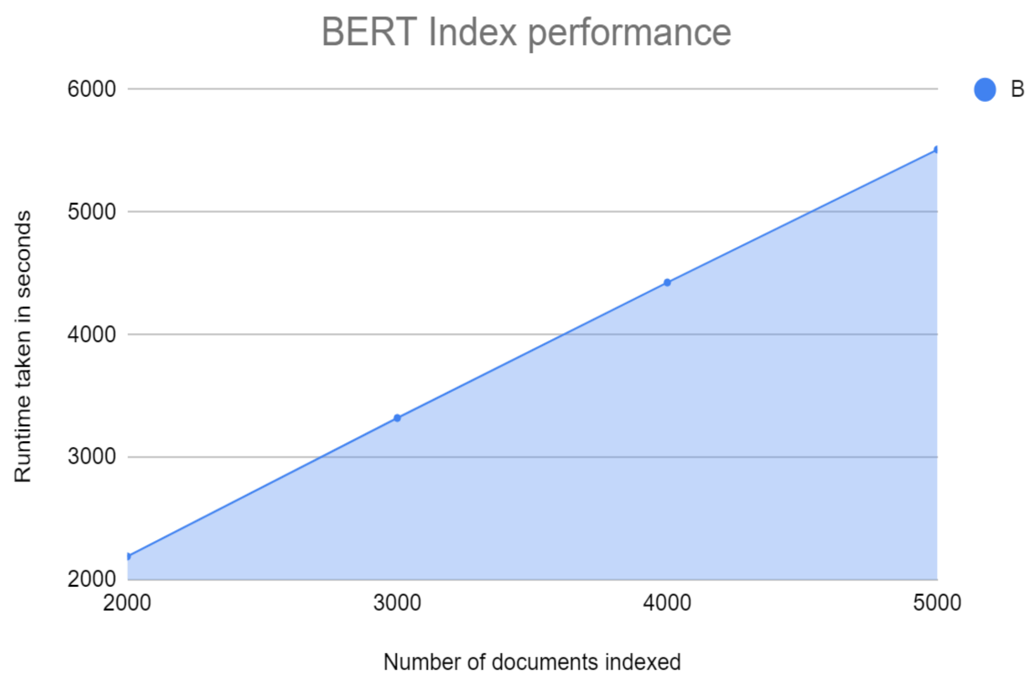


Fig.1. BERT Index performance

Below is a performance analysis of Lucene indexing, that is, the *run time of the Lucene index* creation process. A graph with *run time on the y axis* and *number of documents on the x axis*.

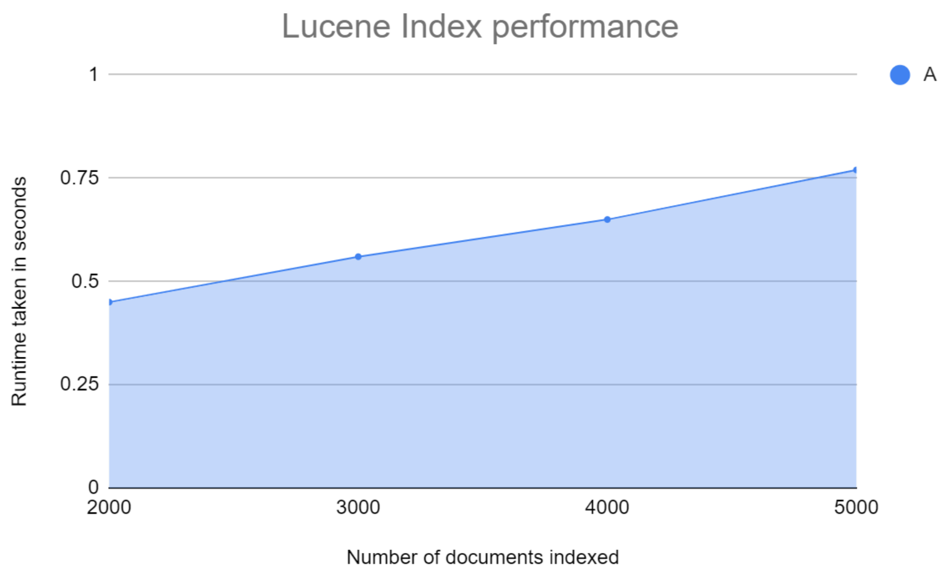


Fig.2. Lucene Index performance

<u>Discussion explaining the differences between the run times:</u>

The BERT indexer and the Lucene indexer have very different run times which has significant differences. By transforming text input into high-dimensional vectors that may be utilized for similarity search, the BERT indexer performs similarity searches. In this method, text data is run through a pre-trained BERT model, and the resulting embeddings are used to create an index. The procedure requires a lot of work, and the run durations grow considerably as more records are indexed. The provided run times show that BERT indexing for 5000 records takes approximately 1.5 hours or 5500 seconds.

However, the Apache Lucene search engine library is the foundation upon which the Lucene indexer is built. Tokenizing the input text into words, eliminating stop words, stemming the words to their root form, and creating an index of the resulting terms are all steps in the Lucene indexing process. The provided run times show that this procedure is much faster than BERT indexing. The Lucene indexer takes just 0.77 seconds to index 5000 records.

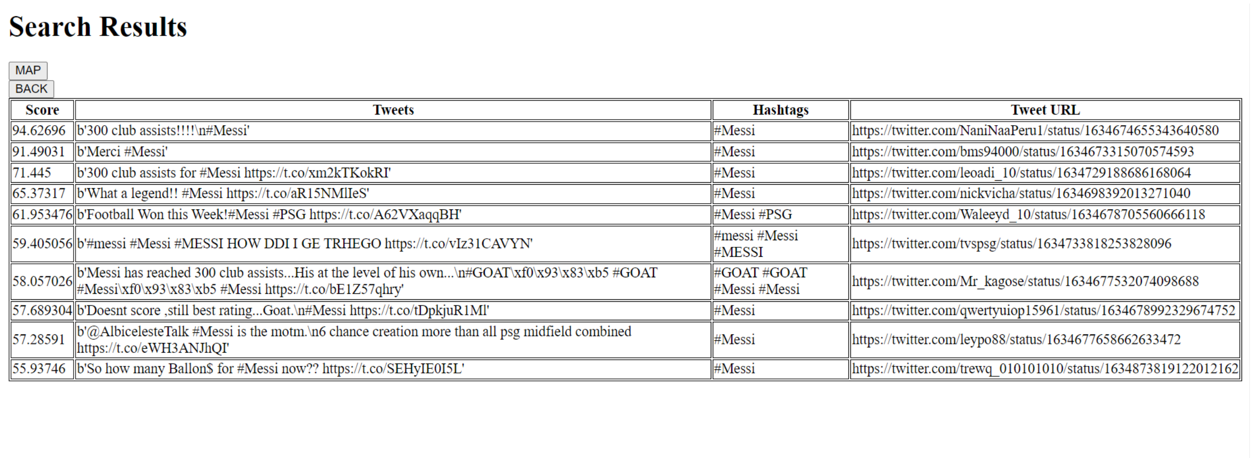
The variations in indexing methods between Lucene and BERT are to blame for the variations in run times. While producing high-dimensional embeddings for BERT indexing is more difficult and computationally intensive, doing so for Lucene indexing only requires tokenizing and stemming the input text. Moreover, BERT indexing uses a deep learning model that has already been trained, which can delay the processing of large amounts of data. Lucene indexing, on the other hand, is a lighter, more effective procedure that is tailored for search applications.

<u>Comparing the quality of the rankings of Lucene and BERT:</u>

We have observed that the quality of rankings of Lucene and BERT for Sports Twitter data depends on the specific objectives of this project and the nature of the data being analyzed. However, to compare the strengths and weaknesses of Lucene and BERT for Sports Twitter Search Engine, we evaluated the results based on the following example queries:

The following are the search results for the **QUERY: messi**

BERT INDEXER RESULTS:

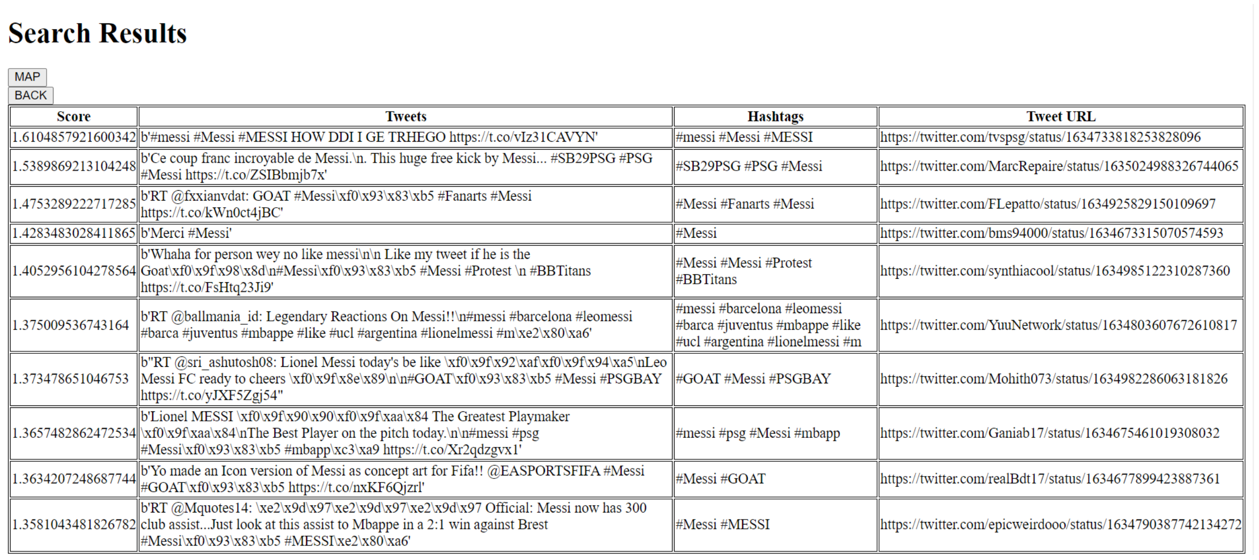

Search Results

MAP
BACK

Score	Tweets	Hashtags	Tweet URL
94.62696	b'300 club assists!!!!\n#Messi'	#Messi	https://twitter.com/NaniNaaPeru1/status/1634674655343640580
91.49031	b'Merci #Messi'	#Messi	https://twitter.com/bms94000/status/1634673315070574593
71.445	b'300 club assists for #Messi https://t.co/xm2kTKokRI'	#Messi	https://twitter.com/leoadi_10/status/1634729188686168064
65.37317	b'What a legend!! #Messi https://t.co/aR15NMlIeS'	#Messi	https://twitter.com/nickvicha/status/1634698392013271040
61.953476	b'Football Won this Week!#Messi #PSG https://t.co/A62VXaqqBH'	#Messi #PSG	https://twitter.com/Waleeyd_10/status/1634678705560666118
59.405056	b'#messi #Messi #MESSI HOW DDI I GE TRHEGO https://t.co/vIz31CAVYN'	#messi #Messi #MESSI	https://twitter.com/tvspsg/status/1634733818253828096
58.057026	b'Messi has reached 300 club assists...His at the level of his own...\n#GOAT\xf0\x93\x83\xb5 #GOAT #Messi\xf0\x93\x83\xb5 #Messi https://t.co/bE1Z57qhry'	#GOAT #GOAT #Messi #Messi	https://twitter.com/Mr_kagose/status/1634677532074098688
57.689304	b'Doesnt score ,still best rating...Goat.\n#Messi https://t.co/tDpkjuR1Ml'	#Messi	https://twitter.com/qwertyuiop15961/status/1634678992329674752
57.28591	b'@AlbicelesteTalk #Messi is the motm.\n6 chance creation more than all psg midfield combined https://t.co/eWH3ANJhQI'	#Messi	https://twitter.com/leypo88/status/1634677658662633472
55.93746	b'So how many Ballon$ for #Messi now?? https://t.co/SEHyIE0I5L'	#Messi	https://twitter.com/trewq_010101010/status/1634873819122012162

Fig.3. Search results for a specific short query using BERT indexer

PYLUCENE INDEXER RESULTS:


Search Results

MAP
BACK

Score	Tweets	Hashtags	Tweet URL
1.6104857921600342	b'#messi #Messi #MESSI HOW DDI I GE TRHEGO https://t.co/vIz31CAVYN'	#messi #Messi #MESSI	https://twitter.com/tvspsg/status/1634733818253828096
1.5389869213104248	b'Ce coup franc incroyable de Messi.\n. This huge free kick by Messi... #SB29PSG #PSG #Messi https://t.co/ZSIBbmjb7x'	#SB29PSG #PSG #Messi	https://twitter.com/MarcRepaire/status/1635024988326744065
1.4753289222717285	b'RT @fxxianvdat: GOAT #Messi\xf0\x93\x83\xb5 #Fanarts #Messi https://t.co/kWn0ct4jBC'	#Messi #Fanarts #Messi	https://twitter.com/FLepatto/status/1634925829150109697
1.4283483028411865	b'Merci #Messi'	#Messi	https://twitter.com/bms94000/status/1634673315070574593
1.4052956104278564	b'Whaha for person wey no like messi\n\n Like my tweet if he is the Goat\xf0\x9f\x98\x8d\n#Messi\xf0\x93\x83\xb5 #Messi #Protest \n #BBTitans https://t.co/FsHtq23Ji9'	#Messi #Messi #Protest #BBTitans	https://twitter.com/synthiacool/status/1634985122310287360
1.375009536743164	b'RT @ballmania_id: Legendary Reactions On Messi!!\n#messi #barcelona #leomessi #barca #juventus #mbappe #like #ucl #argentina #lionelmessi #m\xe2\x80\xa6'	#messi #barcelona #leomessi #barca #juventus #mbappe #like #ucl #argentina #lionelmessi #m	https://twitter.com/YuuNetwork/status/1634803607672610817
1.373478651046753	b"RT @sri_ashutosh08: Lionel Messi today's be like \xf0\x9f\x92\xaf\xf0\x9f\x94\xa5\nLeo Messi FC ready to cheers \xf0\x9f\x8e\x89\n\n#GOAT\xf0\x93\x83\xb5 #Messi #PSGBAY https://t.co/yJXF5Zgj54"	#GOAT #Messi #PSGBAY	https://twitter.com/Mohith073/status/1634982286063181826
1.3657482862472534	b'Lionel MESSI \xf0\x9f\x90\x90\xf0\x9f\xaa\x84 The Greatest Playmaker \xf0\x9f\xaa\x84\nThe Best Player on the pitch today.\n\n#messi #psg #Messi\xf0\x93\x83\xb5 #mbapp\xc3\xa9 https://t.co/Xr2qdzgvx1'	#messi #psg #Messi #mbapp	https://twitter.com/Ganiab17/status/1634675461019308032
1.3634207248687744	b'Yo made an Icon version of Messi as concept art for Fifa!! @EASPORTSFIFA #Messi #GOAT\xf0\x93\x83\xb5 https://t.co/nxKF6Qjzrl'	#Messi #GOAT	https://twitter.com/realBdt17/status/1634677899423887361
1.3581043481826782	b'RT @Mquotes14: \xe2\x9d\x97\xe2\x9d\x97\xe2\x9d\x97 Official: Messi now has 300 club assist...Just look at this assist to Mbappe in a 2:1 win against Brest #Messi\xf0\x93\x83\xb5 #MESSI\xe2\x80\xa6'	#Messi #MESSI	https://twitter.com/epicweirdooo/status/1634790387742134272

Fig.4. Search results for a specific short query using Lucene indexer

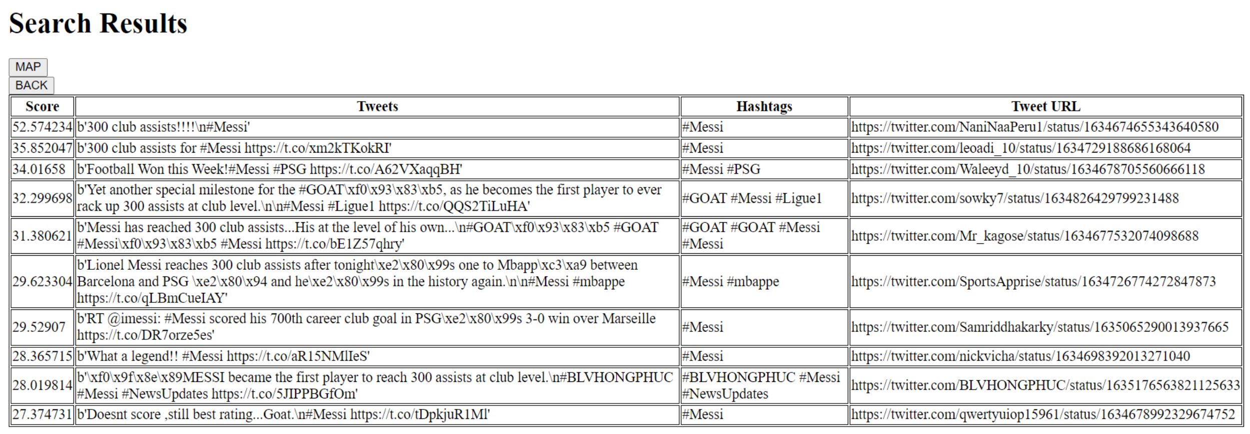
This example query shows that Lucene is doing better than BERT since it is an exact match query. So, if a user searches for a specific term or phrase related to sports, like "Messi", Lucene can accurately retrieve documents that contain that exact term or phrase.

For BERT, it struggles to retrieve the exact term or phrase if it has not seen that specific sequence of words during training.

Also, if a user types a keyword-based queries, like "NBA playoffs", Lucene accurately retrieves relevant documents that contain those keywords. But BERT struggles to accurately rank documents based on specific keywords, as it relies on the semantic meaning of words and phrases.

Now, the below shows the results for the **QUERY: Wow, what a week for football! Messi reaching 300 club assists is just further proof of his GOAT status. And let's not forget about the PSG win and the incredible team effort that made it happen. Congrats to #Messi on another milestone and to #PSG won an unforgettable victory! #football #teamwork #champions**

BERT INDEXER RESULTS:

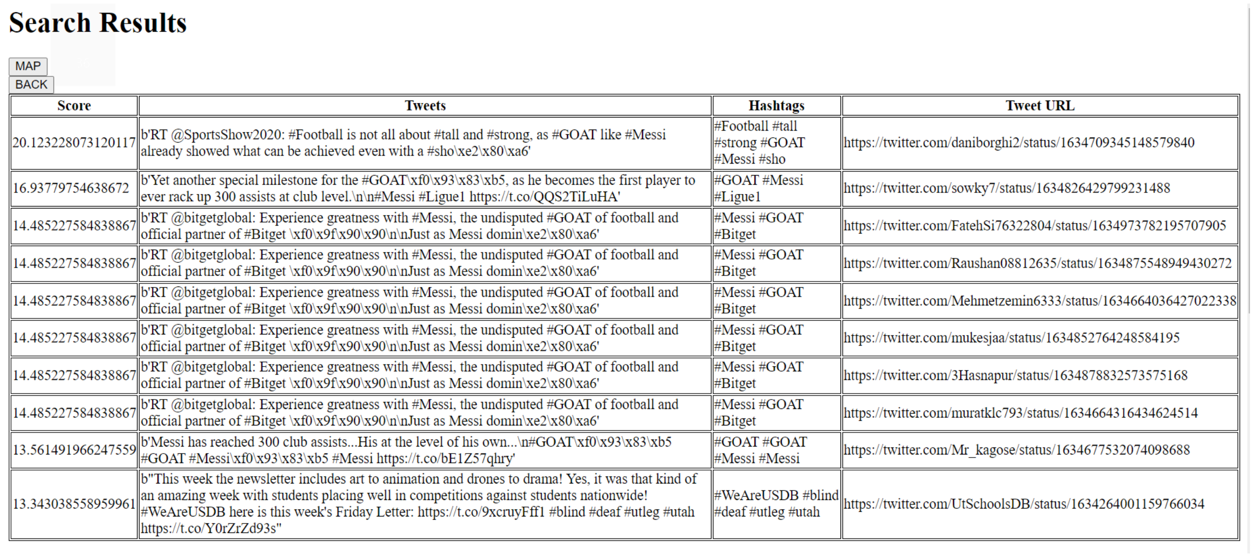


Search Results

Score	Tweets	Hashtags	Tweet URL
52.574234	b'300 club assists!!!!\n#Messi'	#Messi	https://twitter.com/NaniNaaPeru1/status/1634674655343640580
35.852047	b'300 club assists for #Messi https://t.co/xm2kTKokRI'	#Messi	https://twitter.com/leoadi_10/status/1634729188686168064
34.01658	b'Football Won this Week!#Messi #PSG https://t.co/A62VXaqqBH'	#Messi #PSG	https://twitter.com/Waleeyd_10/status/1634678705560666118
32.299698	b'Yet another special milestone for the #GOAT\xf0\x93\x83\xb5, as he becomes the first player to ever rack up 300 assists at club level.\n\n#Messi #Ligue1 https://t.co/QQS2TiLuHA'	#GOAT #Messi #Ligue1	https://twitter.com/sowky7/status/1634826429799231488
31.380621	b'Messi has reached 300 club assists...His at the level of his own...\n#GOAT\xf0\x93\x83\xb5 #GOAT #Messi\xf0\x93\x83\xb5 #Messi https://t.co/bE1Z57qhry'	#GOAT #GOAT #Messi #Messi	https://twitter.com/Mr_kagose/status/1634677532074098688
29.623304	b'Lionel Messi reaches 300 club assists after tonight\xe2\x80\x99s one to Mbapp\xc3\xa9 between Barcelona and PSG \xe2\x80\x94 and he\xe2\x80\x99s in the history again.\n\n#Messi #mbappe https://t.co/qLBmCueIAY'	#Messi #mbappe	https://twitter.com/SportsApprise/status/1634726774272847873
29.52907	b'RT @imessi: #Messi scored his 700th career club goal in PSG\xe2\x80\x99s 3-0 win over Marseille https://t.co/DR7orze5es'	#Messi	https://twitter.com/Samriddhakarky/status/1635065290013937665
28.365715	b'What a legend!! #Messi https://t.co/aR15NMlIeS'	#Messi	https://twitter.com/nickvicha/status/1634698392013271040
28.019814	b'\xf0\x9f\x8e\x89MESSI became the first player to reach 300 assists at club level.\n#BLVHONGPHUC #Messi #NewsUpdates https://t.co/5JIPPBGfOm'	#BLVHONGPHUC #Messi #NewsUpdates	https://twitter.com/BLVHONGPHUC/status/1635176563821125633
27.374731	b'Doesnt score ,still best rating...Goat.\n#Messi https://t.co/tDpkjuR1Ml'	#Messi	https://twitter.com/qwertyuiop15961/status/1634678992329674752

Fig.5. Search results for a specific long query using BERT indexer

PYLUCENE INDEXER RESULTS:



Search Results

Score	Tweets	Hashtags	Tweet URL
20.123228073120117	b'RT @SportsShow2020: #Football is not all about #tall and #strong, as #GOAT like #Messi already showed what can be achieved even with a #sho\xe2\x80\xa6'	#Football #tall #strong #GOAT #Messi #sho	https://twitter.com/daniborghi2/status/1634709345148579840
16.93779754638672	b'Yet another special milestone for the #GOAT\xf0\x93\x83\xb5, as he becomes the first player to ever rack up 300 assists at club level.\n\n#Messi #Ligue1 https://t.co/QQS2TiLuHA'	#GOAT #Messi #Ligue1	https://twitter.com/sowky7/status/1634826429799231488
14.485227584838867	b'RT @bitgetglobal: Experience greatness with #Messi, the undisputed #GOAT of football and official partner of #Bitget \xf0\x9f\x90\x90\n\nJust as Messi domin\xe2\x80\xa6'	#Messi #GOAT #Bitget	https://twitter.com/FatehSi76322804/status/1634973782195707905
14.485227584838867	b'RT @bitgetglobal: Experience greatness with #Messi, the undisputed #GOAT of football and official partner of #Bitget \xf0\x9f\x90\x90\n\nJust as Messi domin\xe2\x80\xa6'	#Messi #GOAT #Bitget	https://twitter.com/Raushan08812635/status/1634875548949430272
14.485227584838867	b'RT @bitgetglobal: Experience greatness with #Messi, the undisputed #GOAT of football and official partner of #Bitget \xf0\x9f\x90\x90\n\nJust as Messi domin\xe2\x80\xa6'	#Messi #GOAT #Bitget	https://twitter.com/Mehmetzemin6333/status/1634664036427022338
14.485227584838867	b'RT @bitgetglobal: Experience greatness with #Messi, the undisputed #GOAT of football and official partner of #Bitget \xf0\x9f\x90\x90\n\nJust as Messi domin\xe2\x80\xa6'	#Messi #GOAT #Bitget	https://twitter.com/mukesjaa/status/1634852764248584195
14.485227584838867	b'RT @bitgetglobal: Experience greatness with #Messi, the undisputed #GOAT of football and official partner of #Bitget \xf0\x9f\x90\x90\n\nJust as Messi domin\xe2\x80\xa6'	#Messi #GOAT #Bitget	https://twitter.com/3Hasnapur/status/1634878832573575168
14.485227584838867	b'RT @bitgetglobal: Experience greatness with #Messi, the undisputed #GOAT of football and official partner of #Bitget \xf0\x9f\x90\x90\n\nJust as Messi domin\xe2\x80\xa6'	#Messi #GOAT #Bitget	https://twitter.com/muratklc793/status/1634664316434624514
13.561491966247559	b'Messi has reached 300 club assists...His at the level of his own...\n#GOAT\xf0\x93\x83\xb5 #GOAT #Messi\xf0\x93\x83\xb5 #Messi https://t.co/bE1Z57qhry'	#GOAT #GOAT #Messi #Messi	https://twitter.com/Mr_kagose/status/1634677532074098688
13.343038558959961	b"This week the newsletter includes art to animation and drones to drama! Yes, it was that kind of an amazing week with students placing well in competitions against students nationwide! #WeAreUSDB here is this week's Friday Letter: https://t.co/9xcruyFff1 #blind #deaf #utleg #utah https://t.co/Y0rZrZd93s"	#WeAreUSDB #blind #deaf #utleg #utah	https://twitter.com/UtSchoolsDB/status/1634264001159766034

Fig.6. Search results for a specific long query using Lucene indexer

Here, the query is long and is semantically similar, i.e. it has multiple interpretations or meanings. So, BERT accurately retrieves relevant documents that are related to this, even if the documents do not contain the exact phrase “Messi is GOAT".

Whereas, Lucene only matches Messi and GOAT and gives result tweets which have only those keywords. BERT will also perform better than Lucene for natural language queries like “Who won FIFA23”. Note: In Lucene output, we can see duplicate tweets because some tweets have been retweeted.

To summarize, Lucene is better suited for exact phrase and keyword-based queries, while BERT is better suited for semantic similarity and natural language queries which are long queries..

Limitations of the system:

1)Dependency on the Pretrained Model : To accomplish indexing, the BERT indexer uses a model that has already been trained. The model may not be optimized for certain use cases or language peculiar to a given area despite having been trained on a substantial amount of data.

2)Difficulty with Long Documents : Long passages or documents may be difficult for BERT indexer to process because it is only built to handle input sequences that are up to a particular length. Its use in situations requiring the indexing of lengthy papers or passages may be constrained as a result.

3)Limited Vocabulary : Due to its small vocabulary, the BERT indexer may have trouble handling uncommon or out-of-vocabulary words (OOV). As a result, documents containing such words may be inaccurately indexed and retrieved

4)Computational Cost : Large volumes of text data must be processed and indexed using the BERT indexer, which is computationally costly and uses a lot of resources. The indexing procedure can be time-consuming, and significant storage space may be needed.

Obstacles and Solutions:

1)Overfitting :

Obstacle : If the training data are not correctly regularized, the BERT indexer may overfit. When testing or producing new data, this may lead to subpar performance.

Solution : Regularization strategies like dropout, weight decay, or early halting can be utilized to avoid overfitting. These methods can enhance the model's ability to generalize by preventing it from memorizing the training data.

2)Scalability :

Obstacle : Due to its computational complexity and resource needs, BERT indexer may not be scalable to very big datasets or high-traffic applications.

Solution : One way to scale the BERT indexer is to use distributed training approaches like data parallelism or model parallelism on large datasets or in high-traffic applications. To lessen the computational load, this includes training the model over numerous processors or nodes.

3)Tuning Hyperparameters :

Obstacle : Due to its computational complexity and resource needs, BERT indexer may not be scalable to very big datasets or high-traffic applications.

Solution : One way to scale the BERT indexer is to use distributed training approaches like data parallelism or model parallelism on large datasets or in high-traffic applications. To lessen the computational load, this includes training the model over numerous processors or nodes.

4)Training Data Availability :

Obstacle : To function at its best, BERT indexer needs a significant amount of training data. Finding a sufficient amount of training data, however, may be difficult in some circumstances, such as for particular languages or in specialized topics.

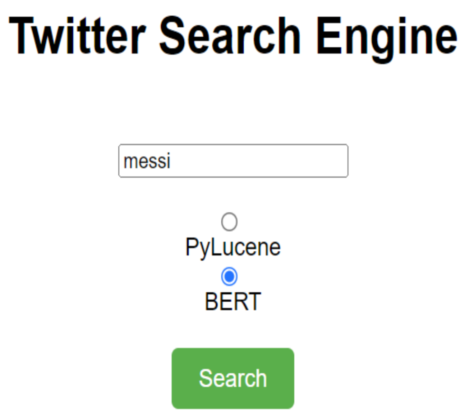
Solution : Using transfer learning could be one approach to overcoming the problem of the lack of training data. Using a smaller domain-specific dataset, this entails optimizing a pre-trained BERT model to produce better results.

Instructions on how to deploy the system:

1. Download the zip file in the server and extract.
2. Run the app.py file which is inside the Flask folder using the command `python3 app.py`
3. Run the ngrok on port 8888 (ngrok file is present in the zip file).
4. Open the link that you get from ngrok in the browser to view the app.

Screenshots showing the system in action:

1. Query: messi



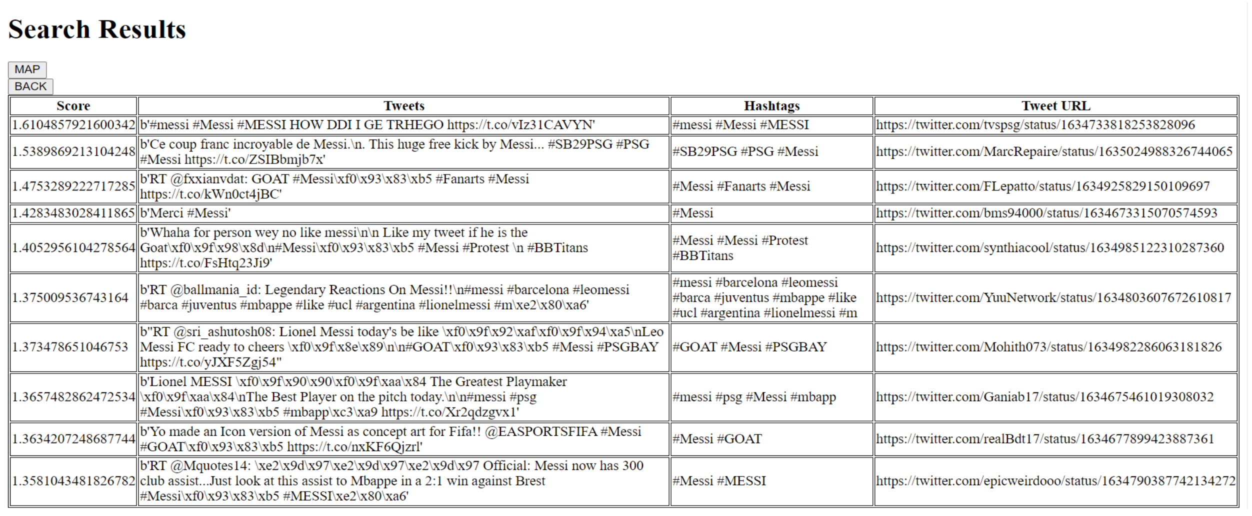
Twitter Search Engine

messi

PyLucene [✓ BERT]

Search

Fig.7. Home Page of the web application



Search Results

MAP
BACK

Score	Tweets	Hashtags	Tweet URL
1.6104857921600342	b'#messi #Messi #MESSI HOW DDI I GE TRHEGO https://t.co/vIz31CAVYN'	#messi #Messi #MESSI	https://twitter.com/tvspsg/status/1634733818253828096
1.5389869213104248	b'Ce coup franc incroyable de Messi.\n. This huge free kick by Messi... #SB29PSG #PSG #Messi https://t.co/ZSIBbmjb7x'	#SB29PSG #PSG #Messi	https://twitter.com/MarcRepaire/status/1635024988326744065
1.4753289222717285	b'RT @fxxianvdat: GOAT #Messi\xf0\x93\x83\xb5 #Fanarts #Messi https://t.co/kWn0ct4jBC'	#Messi #Fanarts #Messi	https://twitter.com/FLepatto/status/1634925829150109697
1.4283483028411865	b'Merci #Messi'	#Messi	https://twitter.com/bms94000/status/1634673315070574593
1.4052956104278564	b'Whaha for person wey no like messi\n\n Like my tweet if he is the Goat\xf0\x9f\x98\x8d\n#Messi\xf0\x93\x83\xb5 #Messi #Protest \n #BBTitans https://t.co/FsHtq23Ji9'	#Messi #Messi #Protest #BBTitans	https://twitter.com/synthiacool/status/1634985122310287360
1.375009536743164	b'RT @ballmania_id: Legendary Reactions On Messi!!\n#messi #barcelona #leomessi #barca #juventus #mbappe #like #ucl #argentina #lionelmessi #m\xe2\x80\xa6'	#messi #barcelona #leomessi #barca #juventus #mbappe #like #ucl #argentina #lionelmessi #m	https://twitter.com/YuuNetwork/status/1634803607672610817
1.373478651046753	b"RT @sri_ashutosh08: Lionel Messi today's be like \xf0\x9f\x92\xaf\xf0\x9f\x94\xa5\nLeo Messi FC ready to cheers \xf0\x9f\x8e\x89\n\n#GOAT\xf0\x93\x83\xb5 #Messi #PSGBAY https://t.co/yJXF5Zgj54"	#GOAT #Messi #PSGBAY	https://twitter.com/Mohith073/status/1634982286063181826
1.3657482862472534	b'Lionel MESSI \xf0\x9f\x90\x90\xf0\x9f\xaa\x84 The Greatest Playmaker \xf0\x9f\xaa\x84\nThe Best Player on the pitch today.\n\n#messi #psg #Messi\xf0\x93\x83\xb5 #mbapp\xc3\xa9 https://t.co/Xr2qdzgvx1'	#messi #psg #Messi #mbapp	https://twitter.com/Ganiab17/status/1634675461019308032
1.3634207248687744	b'Yo made an Icon version of Messi as concept art for Fifa!! @EASPORTSFIFA #Messi #GOAT\xf0\x93\x83\xb5 https://t.co/nxKF6Qjzrl'	#Messi #GOAT	https://twitter.com/realBdt17/status/1634677899423887361
1.3581043481826782	b'RT @Mquotes14: \xe2\x9d\x97\xe2\x9d\x97\xe2\x9d\x97 Official: Messi now has 300 club assist...Just look at this assist to Mbappe in a 2:1 win against Brest #Messi\xf0\x93\x83\xb5 #MESSI\xe2\x80\xa6'	#Messi #MESSI	https://twitter.com/epicweirdooo/status/1634790387742134272

Fig.8. Search results for a specific query using Lucene indexer

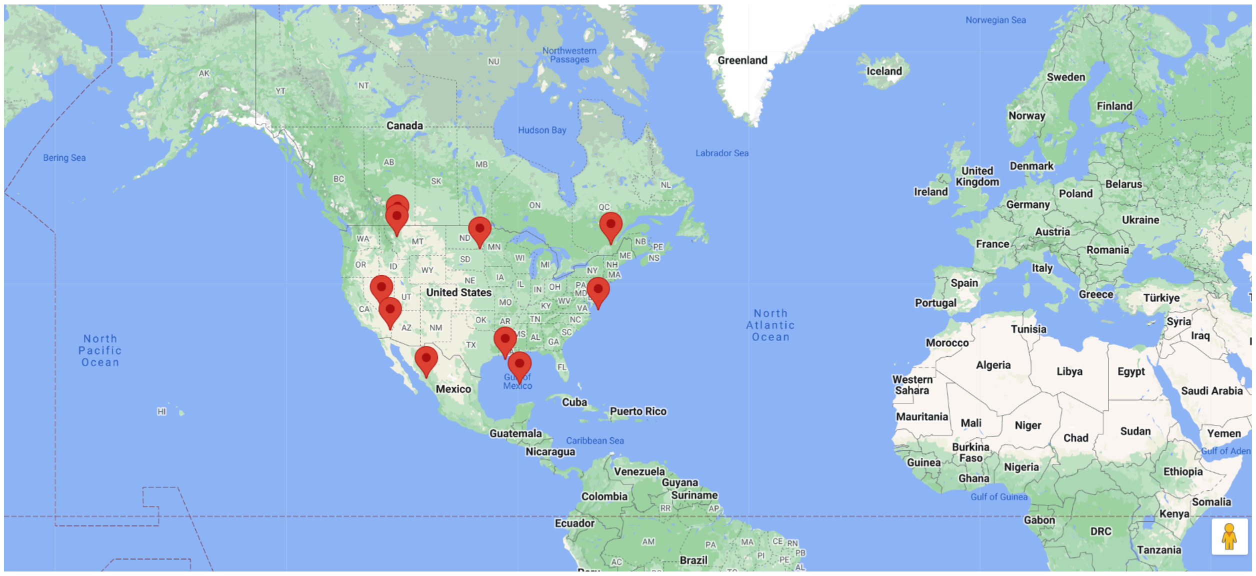
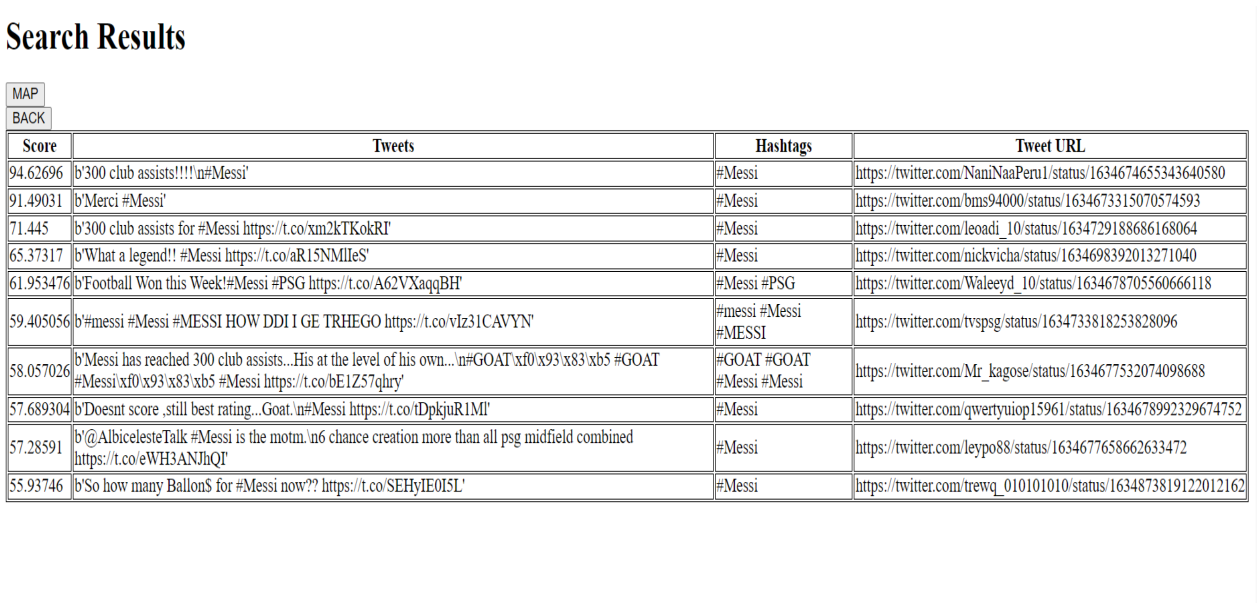


Fig.9. Map representation of geotagged tweets generated using Lucene indexer



Search Results

MAP

BACK

Score	Tweets	Hashtags	Tweet URL
94.62696	b'300 club assists!!!!\n#Messi'	#Messi	https://twitter.com/NaniNaaPeru1/status/1634674655343640580
91.49031	b'Merci #Messi'	#Messi	https://twitter.com/bms94000/status/1634673315070574593
71.445	b'300 club assists for #Messi https://t.co/xm2kTKokRI'	#Messi	https://twitter.com/leoadi_10/status/1634729188686168064
65.37317	b'What a legend!! #Messi https://t.co/aR15NMlIeS'	#Messi	https://twitter.com/nickvicha/status/1634698392013271040
61.953476	b'Football Won this Week!#Messi #PSG https://t.co/A62VXaqqBH'	#Messi #PSG	https://twitter.com/Waleeyd_10/status/1634678705560666118
59.405056	b'#messi #Messi #MESSI HOW DDI I GE TRHEGO https://t.co/vIz31CAVYN'	#messi #Messi #MESSI	https://twitter.com/tvspsg/status/1634733818253828096
58.057026	b'Messi has reached 300 club assists...His at the level of his own...\n#GOAT\xf0\x93\x83\xb5 #GOAT #Messi\xf0\x93\x83\xb5 #Messi https://t.co/bE1Z57qhry'	#GOAT #GOAT #Messi #Messi	https://twitter.com/Mr_kagose/status/1634677532074098688
57.689304	b'Doesnt score ,still best rating...Goat.\n#Messi https://t.co/tDpkjuR1Ml'	#Messi	https://twitter.com/qwertyuiop15961/status/1634678992329674752
57.28591	b'@AlbicelesteTalk #Messi is the motm.\n6 chance creation more than all psg midfield combined https://t.co/eWH3ANJhQI'	#Messi	https://twitter.com/leypo88/status/1634677658662633472
55.93746	b'So how many Ballon$ for #Messi now?? https://t.co/SEHyIE0I5L'	#Messi	https://twitter.com/trewq_010101010/status/1634873819122012162

Fig.10. Search results for a specific query using BERT indexer

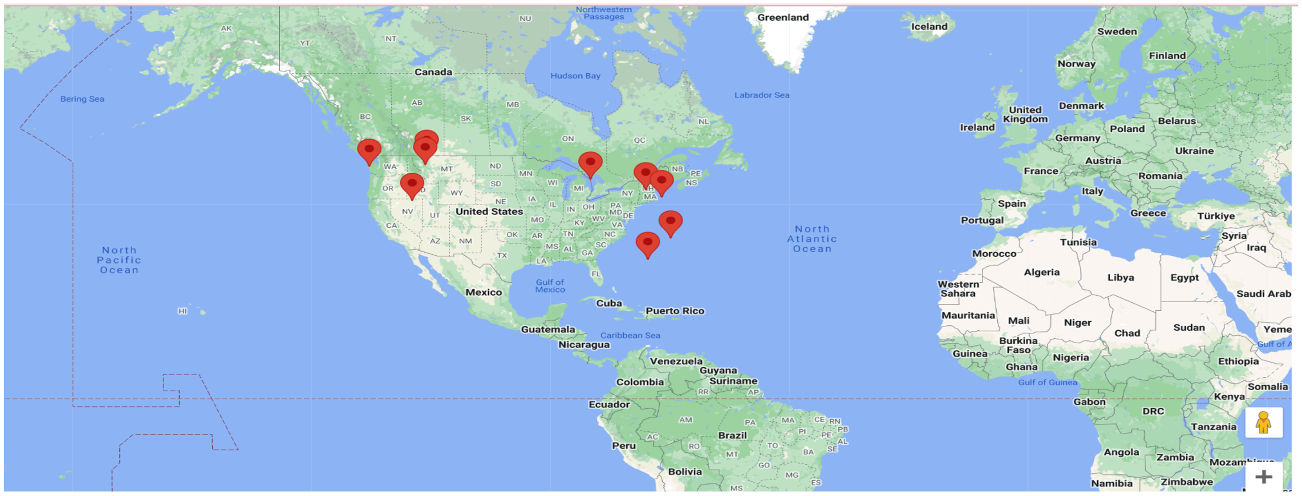


Fig11. Map representation of geotagged tweets generated using BERT indexer