Designing Hybrid Similarity based Search Engine Using Artificial Intelligence

Parv Gupta
CSE, Chandigarh University
Mohali, India
karanguptaneo@gmail.com

Abstract— In the 20th century it is been observed that there is drastic increase of information on web and the biggest reason for that is the availability of computation and content transferring on internet is increasing and it is not a less known fact that everybody is seeking for the most relevant information , it is been already know that since the beginning of the era of internet the search is quite a challenging problem and also a necessary problem to solve , however there are already plenty of solutions available for the search engines and they are serving us quite well but as the searching content and the users seeking for the content is increasing second by second it became necessary for us to move forward and experiment other techniques as well , best way to compare two sentences is to compare their similarities there are basically 2 types of similarities word based and semantic based similarity nowadays search engines are been built on either of similarity and in this paper we will see how can we implement a powerful search technique which leverages both the techniques , ideally semantic based similarity is a machine learning based technique which uses encoder model to generate semantic vector which are discussed in more detailed manner in the paper , another dimension that we shouldn't ignore while solving this problem is time it is important to understand that user will spend more time .

Keywords—Machine Learning , Artificial intelligence, NLP, LSTM, Search Engine, Encoder Model, Deep learning, Bert, Universal Vector Encoder

1. INTRODUCTION

In the era of technology where world is moving towards web. Digital content has become new source of knowledge gaining however with respect to digital content the complexity of solving the searching problem is also increasing. As era change, it is been observed that the innovations are not only coming from one domain but from several domains. It also became necessary to provide the best possible search results for the given query that too in an optimized manner. In content searching it is a necessary for any business to give their users provide the most relevant results. There are many deep learning architectures[1] which can be used to generate some quality search results as these models are already pre-trained on lots of text data .The pre-trained models are specifically designed to extract the semantic information from the text which could be used to compute the semantic similarity [2]. However, there is exhaustive number of techniques to compute the similarity between two points [3]. There is another very simple technique called word based similarity [4]. Another approach could be to combine the both mentioned similarity techniques. However, combining the results will not be as trivial as it sounds so we will experiment the techniques in order to get the best possible output. Another aspect for the problem is the latency where we already know the user will not even spend more than 0.5 second for the results. To tackle the latency problem we will be employing the technique called inverted indices [5].The main aim of this study is to solve the above stated problem using some machine learning techniques and software engineering approaches by using the data of the stack overflow to give relevant search results to the user. One of the major key points which should be taken into the consideration is the latency requirement for the above problem is that, the user will not spend more than half a second to get the results so the latency requirement for this problem would be less than 0.5 second. So objective of the search engine is to give most relevant results from the available answered questions to the user using the similarity based techniques in a computationally optimized

manner. So from the design stand point we will be building a Q/A search engine which on the run time takes the query question as text and give the most similar answered by using the text and the semantic similarity of the query as results to the user in less than 0.5 second.

2. RELATED REVIEW

This section will discuss about the different researches uses and comparison of previous search engines models used in internet search.

In [6], the author proposed a method for converting node to vector in a graph, they named it node2vec. The author proposed this technique to convert each of the node of the graph to a vector. The author discussed about feature learning in graph can be done using the optimization of graphs. It also explains the exploration-exploitation trade-off on different search techniques. In [7], author proposed an algorithm to traverse the data randomly in a graph network, they called it Random Walk. The author discussed about how to generate embedding from randomly created paths. The author also discussed about the useful of randomness in creating the paths of graph networks. In [8], author proposed a research about the role of matrix factorization in recommender system. The author discussed about the collaborative filtering and how it can be used for recommendation system just by shaping the problem in the optimization problem. The author states that matrix factorization plays vital role in reducing the dimensionality of the data.

In [9], the author proposed architecture of Docker, in the proposed work author discussed about its comparison with the virtual machines. In the proposed research work the author shows how Docker can be used to experiment different environment through docker images and can be contained in the container of a docker as an instance. The author also shows that how docker can be a light alternative to the virtual machines and also how docker containers can be computational efficient against virtual machines. In [10] discussed about a search engine called Elastic search, in the proposed work the author discussed about the ways of searching through a data corpus. The author also discussed about the optimizations which can be effective and useful in the search. In [11] author discussed about a technique using which we can generate the similarity between 2 sentences, named as word based similarity. In [12] author discussed about a technique which can be used to generate the similarity score between two sentences using semantics of the sentences, they named it sentence based similarity.

In [13], the author discussed about deep learning architecture using which we could generate a semantically stable embedding for the sentences. They named it as Encoder model , The author proposed this algorithm with feeding the sentences as a different time steps sequentially. In [14], the author discussed about a encoding technique called word2vec. The author states that technique could be used to generate semantically stable embedding from a single word. In [15], the author proposed and discussed some techniques on computation of similarities which is ideally proposed for the recommender system. The author also discussed about the advantages and disadvantages of the techniques.

In [16] the author discussed about the different aspects of exploratory data analysis. In the proposed work, the author mentioned different methods to visualize and interpret from the plots and graphs. In [17] the author discussed about the preprocessing techniques which can be useful for the natural language data. In the proposed work the author pointed out different preprocessing methodologies for different types of data.

3. METHODOLOGY USED

3.1. Dataset Used

Dataset has been collected from stack overflow data dump [18]. The dataset has been further divided into more than 200 categories out of that four categories is been used for the further analysis of this paper that is Computer Science, math, data science, artificial intelligence. Each category contains the questions and answers related to their name. Each category has a single XML file contains following attributes

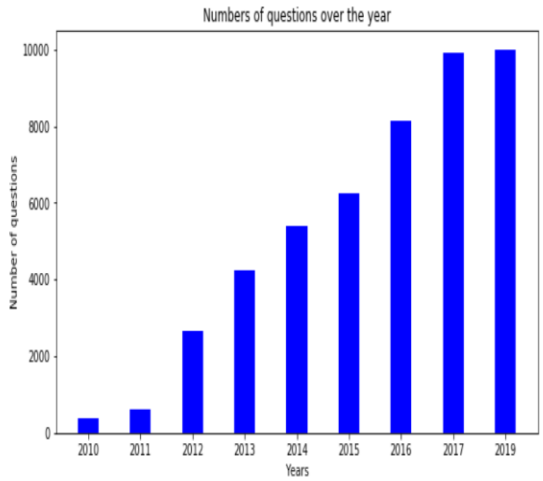
- Id : Feature contains the unique values of each posts.
- PostTypeId: It shows the type of post (in case of answers PostTypeId is 2 in case for questions it is 1)
- ParentId: It contains the values which shows what is the ParentId of the Post (For answers ParentId is the Id of its Question)
- Body: It contains the text it can be Question or Answer.
- Title: It contains the Title of the Post
- AnswersCount: It contains the values showing how many number of Answers Are there.
- Tags: It contains the Tags of the Post.
- Category: Shows Which Category it belongs to.
- CreationDate: Shows the Date and Time of the Post Creation

Out of the following attributes only some attributes are of use for building our models i.e., Body, Title.

3.2. Exploratory Data Analysis

This section is about the analysis which is done on the data to get the useful insights which can be helpful for the search engine problem. Firstly data must be downloaded from stack overflow data dump. As our data is in the XML files so we have to make sure that our data takes shape of necessary format. Exploratory data analysis needs be applied on our data so that the insights of the data could be fetched in order to build robust search engine.

Elastic Search is employed which is containerized in a docker Instance. A data dictionary is used which contains each key as unique id for the question and values as a title and answers for the fast retrieval of the questions using ids, so as we will be combining the title, question and answer and then this data will be fed to the elastic search which is containerized in docker. Exploratory data analysis is a crucial step before building a solution for a business problem. So some plots and graphs are plotted in order understand our problem better. Figure 1 shows the Number of questions versus year plot. Figure 2 shows the Number of answers versus years plot.



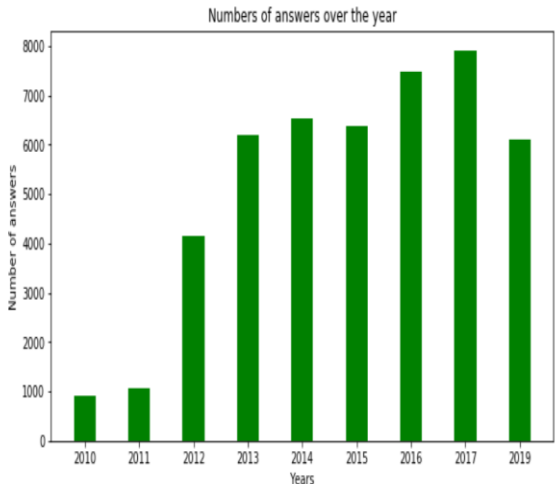
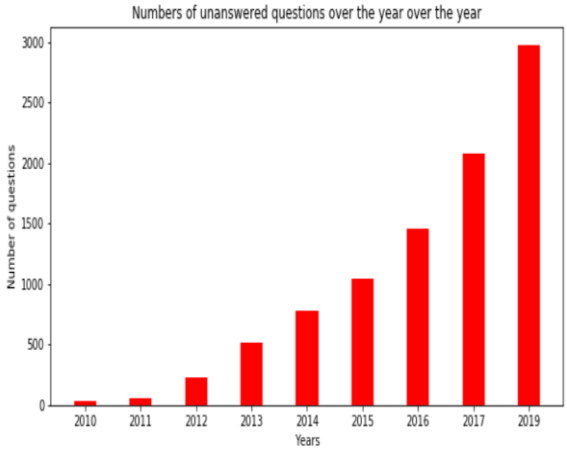


Figure 1: Numbers of Questions over the year

Figure 2: Number of answers over the years

Figure 3 shows the Number of unanswered questions vs years plot. Figure 4 shows the Number of Questions from each category plot.



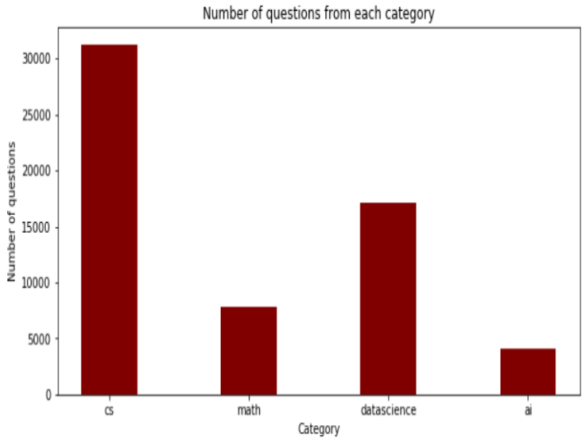


Figure 3: Number of unanswered Questions Over the Year

Figure 4: Number of Questions from each category

Figure 5 shows the Questions Word Cloud. Figure 6 Answers Word cloud.

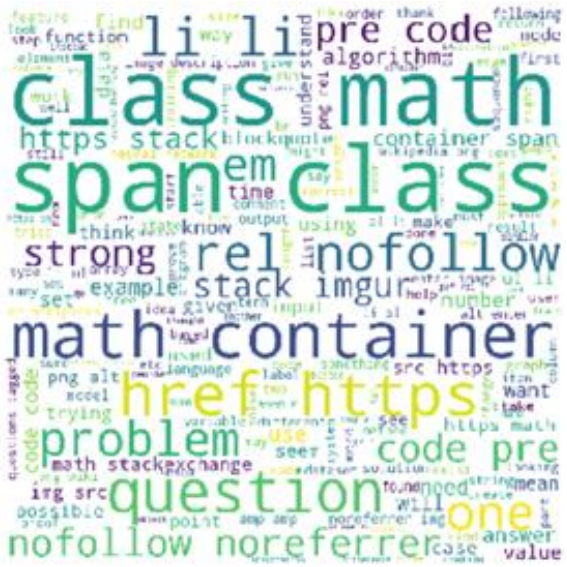


Figure 5: Questions Word Cloud

Figure 6: Answers Word cloud

3.3.Proposed Model

This section discuss about the proposed model of the search engine which is used in this paper to build search engine. Firstly a query text will get embedded through our encoder and then both our search text and embedding would be passed to elastic search container independently to get the score and then it will get added using weighted sum and on the basis of that we will select the answers having top 10 scores. The model depicted in this section includes the all the bits and pieces used and discussed in the above sections. Finalized model can inferred from Figure 7.

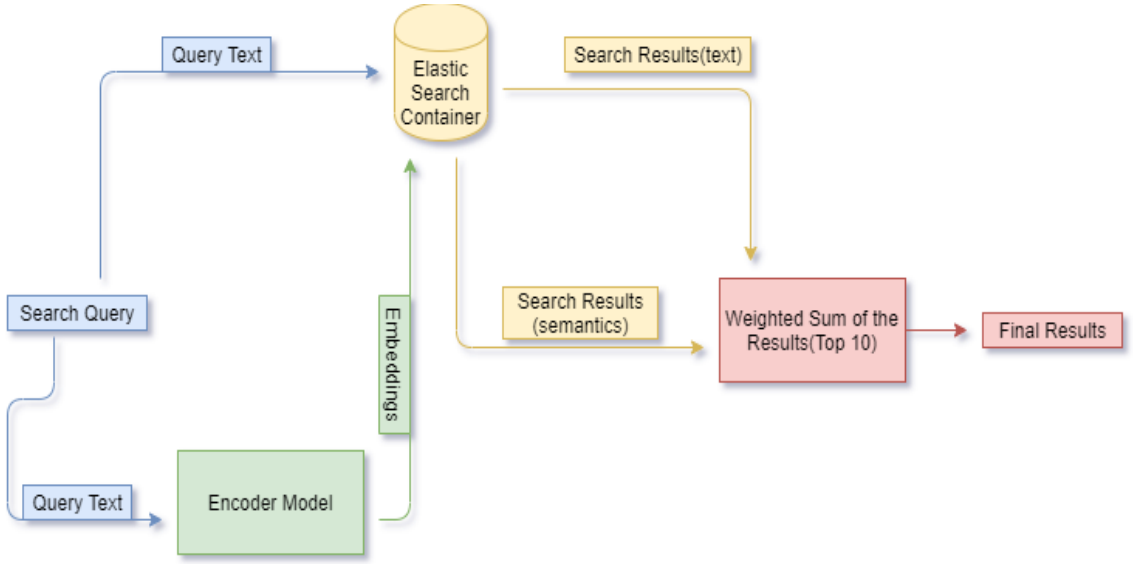


Figure 7: Hybrid search engine model

4. EXPERIMENTAL RESULT ANALYSIS

This section will discuss about the results and analysis of different models used in this paper for building the architecture

4.1. Universal Sentence Encoder Model

In this section we are using the Universal-sentence-encoder to generate our embedding. The universal sentence encoder model is the state of the art NLP model which encodes sentences into embedding. The model takes a sequence or sentence as input and returns 512-dimensional data for each query text. After feeding your data to elastic search we need to define a search function. Search function gets the text based similarity scores and semantic based similarity scores and combines them with weighted average by giving more weight to semantic similarity scores than word based similarity scores. Figure 8 defines the result from Universal Sentence Encoder Model if the question is "*what is red black trees?"*

Figure 8: Universal Vector Encoder Result

4.2.Analysis using Bi-directional Encoder Representations from Transformers

This section discusses about the results and analysis of model using Bi-directional Encoder Representations from Transformer (BERT). We are using the BERT to generate our embedding**,** BERT Is a transformer based model which encodes text into embedding. The model takes a sequence or sentence as input and returns 512-dimensional data for each query text. After feeding your data to elastic search we need to define a search function. Search function gets the text based similarity scores and semantic based similarity scores and combines them with weighted average by giving more weight to semantic similarity scores than word based similarity scores. Figure 8 defines the result from Universal Sentence Encoder Model if the question is "*what is red black trees?"*

Figure 9: BERT Encoder Result

4.3. DISCUSSION

On the Basis of the above results shown in Figure 8 and Figure 9 this has been shown that the results are very similar in terms of quality. It is been also observed that model BERT model is quite complex and requires somewhat slightly more time in order to generate the results than universal vector encoder model so it is better choice to proceed with the universal vector encoder model.

5. CONCLUSION AND FUTURE SCOPE

In this paper, universal sentence encoder and BERT technique has been used for the analysis of this hybrid similarity search engine. On the Basis of execution analysis it is been noticed that due to the complexity of BERT model the execution time for BERT in greater than the Universal Sentence encoder. The techniques which are used in this paper are latest state of the art techniques but there some methods which might be useful for the whole project. We know that we have around 1 lakh data points but the quality of the results could even increase if we feed more data to our Model. As of now the universal encoder model is trained on general dataset but not on the programming codes so the relevancy of results could increase if we train our encoder model with some programming codes data. The technique used to combine the semantic and word based similarity results are weighting average but we could look for some techniques to combine the both similarity scores.

References

[1]. Khan, A., Sohail, A., Zahoora, U., & Qureshi, A. S. (2020). A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, *53*(8), 5455-5516.

[2]. Liu, M., Lang, B., & Gu, Z. (2017). Calculating semantic similarity between academic articles using topic event and ontology. *arXiv preprint arXiv:1711.11508*.

[3]. Ontañón, S. (2020). An overview of distance and similarity functions for structured data. *Artificial Intelligence Review*, *53*(7), 5309-5351.

[4]. Farouk, M. (2019). Measuring sentences similarity: a survey. *arXiv preprint arXiv:1910.03940*.

[5]. Mahapatra, A. K., & Biswas, S. (2011). Inverted indexes: Types and techniques. *International Journal of Computer Science Issues (IJCSI)*, *8*(4), 384.

[6]. Grover, A., & Leskovec, J. (2016, August). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 855-864).

[7]. Xia, F., Liu, J., Nie, H., Fu, Y., Wan, L., & Kong, X. (2019). Random walks: A review of algorithms and applications. *IEEE Transactions on Emerging Topics in Computational Intelligence*, *4*(2), 95-107.

[8]. Girase, S., & Mukhopadhyay, D. (2015). Role of matrix factorization model in collaborative filtering algorithm: a survey. *arXiv preprint arXiv:1503.07475*.

[9]. Rad, B. B., Bhatti, H. J., & Ahmadi, M. (2017). An introduction to docker and analysis of its performance. International Journal of Computer Science and Network Security (IJCSNS), 17(3), 228.

[10]. Kalyani, D., & Mehta, D. (2017). Paper on searching and indexing using elasticsearch. *Int. J. Eng. Comput. Sci*, *6*(6), 21824-21829.

[11]. Gomaa, W. H., & Fahmy, A. A. (2013). A survey of text similarity approaches. *International Journal of Computer Applications*, *68*(13), 13-18.

[12]. Liu, Y., Sun, C. J., Lin, L., Wang, X., & Zhao, Y. (2015, October). Computing semantic text similarity using rich features. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation* (pp. 44-52).

[13]. Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*.

[14]. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[15]. Mannan, N. B., Sarwar, S. M., & Elahi, N. (2014, September). A new user similarity computation method for collaborative filtering using artificial neural network. In *International Conference on Engineering Applications of Neural Networks* (pp. 145-154). Springer, Cham.

[16]. Yu, C. H. (2010). Exploratory data analysis in the context of data mining and resampling. *International Journal of Psychological Research*, *3*(1), 9-22.

[17]. Agarwal, V. (2015). Research on data preprocessing and categorization technique for smartphone review analysis. *International Journal of Computer Applications*, *975*, 8887.

[18]. File for StackExchange, *Internet archieve*. Accessed at: https://archive.org/download/stackexchange.