Recommendation System – Hopscotch Forum

Group : ML-Team 5, Subgroup 3(Recommendation System)

(a) Hopscotch Forum

The Hopscotch forum is a space for people to discuss the Hopscotch app and language. People can ask questions, get help and learn about coding with Hopscotch. They can also participate in the language evolution by reporting bugs, suggesting new features and chiming in on the planned roadmap. Lastly, it's a place where people can discuss topics related to Hopscotch such as projects they are working on, tutorials they've created, coding projects they've created around Hopscotch in other languages, or ideas for collaborations.

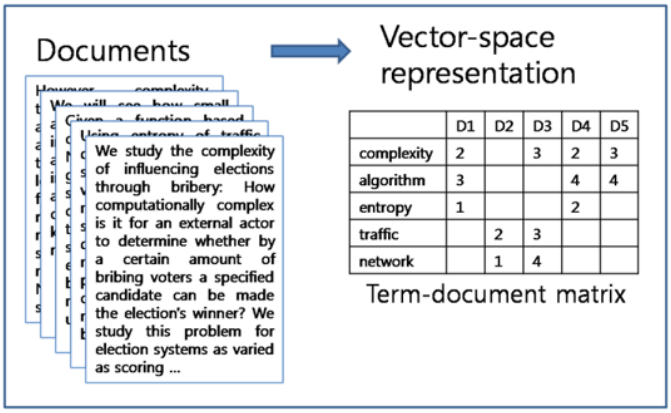
(b) TF-IDF

TF-IDF is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It consists of two terms namely "Term Frequency" and "Inverse Document Frequency".

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$tf_{i,j}$ = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

The first term (TF) essentially rewards the word for how common it is in that document. The second term (IDF) penalizes the word for how common it is among all documents. For instance, articles like "the" are common and prevalent in almost all documents and therefore, does not contribute much to the meaning. This is taken care by IDF and on the contrary, words like "spaceship" is not common among documents and therefore does not get penalized. This concept is used to vectorize text in documents for further required computation.

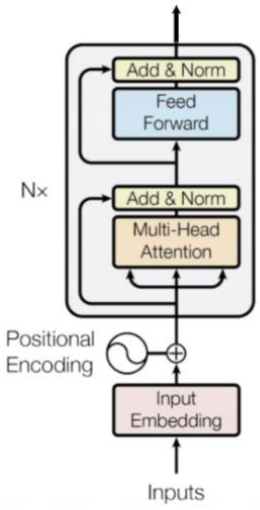


	D1	D2	D3	D4	D5
complexity	2		3	2	3
algorithm	3			4	4
entropy	1			2	
traffic		2	3		
network		1	4		

As we can see from the above figure, TF-IDF score is computed for each pair in the term document matrix. From this filled out matrix, we can extract our embedding for a document by just looking at the corresponding column. The dimension of embedding for a document of text would ideally be the number of unique words in all the documents combined.

(c) BERT

Since English words cannot directly be comprehended by machines, we need to input them as vectors in a certain dimension that also contains some essence of the word and this is called Embedding. Word vectors are essentially a certain type of word embedding that could be obtained by training a basic feed-forward neural network or a RNN on a simple task such as n-gram or skip gram task thereby encouraging the network to project the high dimensional one-hot vector encodings to a lower dimensional space and also making sure the compressed encoding aids in solving the task. The obtained word vectors capture the essence and meaning of the word and this could be observed by comparing the distance between similar words in the embedding dimensional space to that of completely different words which have no correlation.



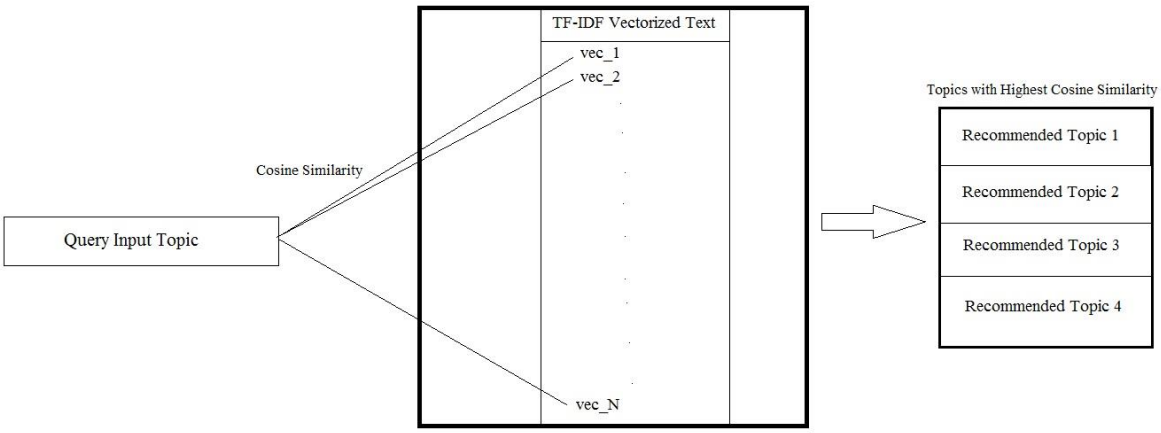
BERT makes use of Transformer, an attention mechanism that learns contextual relations between words (or sub-words) in a text. In its vanilla form, Transformer includes two separate mechanisms — an encoder that reads the text input and a decoder that produces a prediction for the task. Since BERT's goal is to generate a language model, only the encoder mechanism is necessary. As opposed to directional models, which read the text input sequentially (left-right or right-left), the transformer encoder reads the entire sequence of words at once. Therefore, it is considered bidirectional, though it would be more accurate to say that its non-directional. This characteristic allows the model to learn the context of a word based on all of its surroundings.

(d) Recommendation

Recommendation is done using three approaches. The first one is using TF-IDF vectorized text, second one is using BERT embeddings and the third approach is a combination of the two approaches.

(i) Using TF-IDF

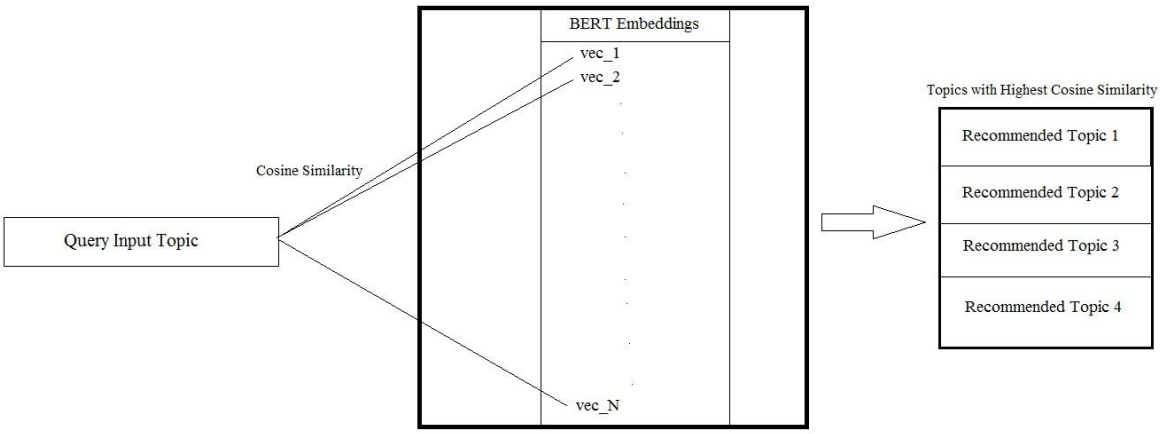
As discussed above, the concept of TF-IDF is used to vectorize text data from the forum. First, we create a dataset with each topic text represented by its corresponding vector. When a new topic query is given, it is first transformed through the same procedure and once its vector is obtained, cosine similarity is computed between the input vector and all the existing vectors in the dataset.



Once the cosine similarities are obtained, we simply sort them in descending order and output the top 5 or 10 topics that have the maximum similarity with the input query. The flow diagram above depicts the entire flow of recommendation using TF-IDF.

(ii) Using BERT

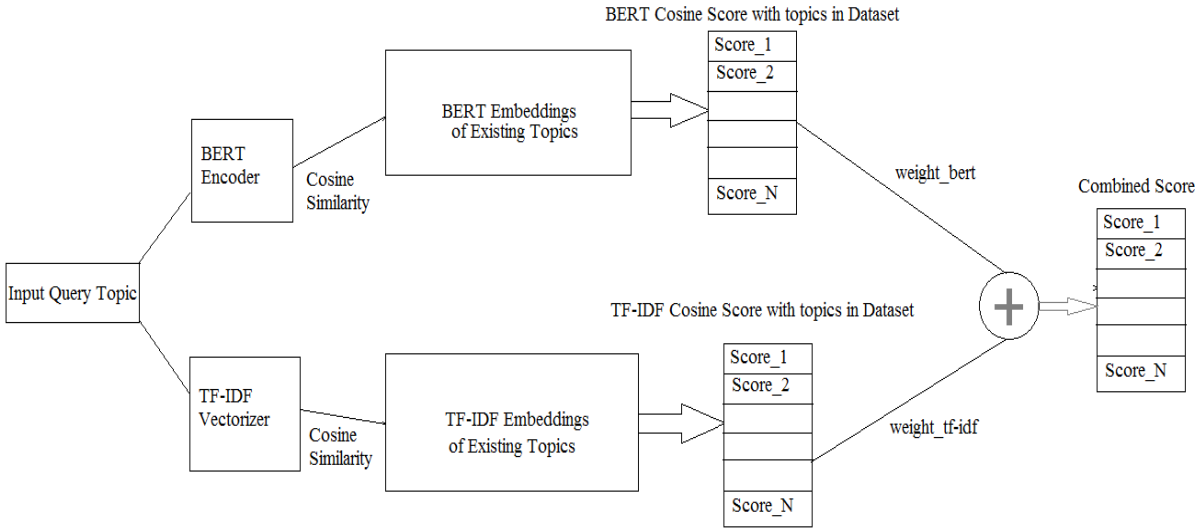
The same procedure is followed except that instead of TF-IDF vectorized text, we use the BERT embeddings obtained from the pre-trained encoder.



The difference between the approaches lies in what features does the vectorization captures. TF-IDF embeddings rely upon the word frequencies and not about the meaning of the topic post. On the contrary, the BERT embeddings capture the semantic structure of the topic instead of fixating on a word. As expected the results of the recommendation depict the same behaviour as well.

(iii) Combining TF-IDF and BERT

The basic intuition for attempting a combination approach was to take advantage of both semantic structure captured by BERT and statistical frequency based information captured by TF-IDF. A recommendation system that takes both the factors into account would ideally perform better than the individual approaches.



At first, the input query topic is transformed using both techniques separately and the corresponding cosine similarities are computed with all the examples in the dataset. The

two approaches would separately yield corresponding cosine similarities. Instead of considering either one of them, we computed a weighted average of them giving a new score in the assumption that the new score would capture both semantic and structural similarities.

(e) Evaluation

Recommendations by definition are hard to objectively evaluate and they require manual feedback for improving the quality of recommendations. However, we have chosen Jaccard similarity to compare the three approaches if not objectively evaluate them. Jaccard similarity between two text sentences is the ratio of the number of common words between the two and the total number of unique words in both of them. For an input query, we had 10 recommendations and the evaluation was based on the average jaccard similarity for all 10 recommendations.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

(f) Results

The results showed that the Jaccard similarity was biased towards TF-IDF than towards BERT. The reason for this lies in the similarity of the definitions of TF-IDF and jaccard measure. Both of them consider the words in the sentence and not the meaning of them. The interesting yet expected observation was that the combination approach almost always performed better than the two individual approaches. Below are some examples of the recommendation output.

TF-IDF Recommendation

```
Input is

['https://forum.gethopscotch.com//t/variables-not-setting-properly/55261', 'i need help with variables not setting to the correct value/being


Displaying 10 - Recommended topics

Recommendation - 1
Similarity is : tensor(0.2447, dtype=torch.float64)

['https://forum.gethopscotch.com//t/how-to-do-basic-variables-hwc-topic-remake/34959', 'Once again, this is from November and hasn't been read

Recommendation - 2
Similarity is : tensor(0.2391, dtype=torch.float64)

['https://forum.gethopscotch.com//t/how-to-make-clones-do-something-individually/38723', 'So I'm trying to have a project where the clones wil

Recommendation - 3
Similarity is : tensor(0.2257, dtype=torch.float64)

['https://forum.gethopscotch.com//t/how-to-learn-basic-values/27435', '#Hi, I'm @Petrichor.\n##I am going to teach you some cool things to do
```

BERT Recommendation

```
Input is

['https://forum.gethopscotch.com//t/variables-not-setting-properly/55261', 'i need help with variables not setting to the correct value/being


Displaying 10 - Recommended topics

Recommendation - 1
Similarity is : tensor(0.8140)

['https://forum.gethopscotch.com//t/chocoraspberrys-test-topic/42186', 'I'm  testing making topics with this so please ignore this']

Recommendation - 2
Similarity is : tensor(0.8048)

['https://forum.gethopscotch.com//t/when-bumps-doesn-t-seem-to-work-closed/47478', 'I'm making a game, and "when bumps" doesn't work for some

Recommendation - 3
Similarity is : tensor(0.8003)

['https://forum.gethopscotch.com//t/is-this-a-bug-or-is-my-code-wrong/36162', 'Is this a bug, or is there something wrong with the code?The c
```

TF-IDF and BERT Recommendation

```
Input is

['https://forum.gethopscotch.com//t/variables-not-setting-properly/55261', 'i need help with variables not setting to the correct value/b


Displaying 10 - Recommended topics

Recommendation - 1
Similarity is : tensor(0.4883, dtype=torch.float64)

['https://forum.gethopscotch.com//t/is-this-a-bug-or-is-my-code-wrong/36162', 'Is this a bug, or is there something wrong with the code?T

Recommendation - 2
Similarity is : tensor(0.4231, dtype=torch.float64)

['https://forum.gethopscotch.com//t/custom-rules-sometimes-change-behaviour-of-when-blocks/46696', '(NOTE: I don't know that much about t

Recommendation - 3
Similarity is : tensor(0.4204, dtype=torch.float64)

['https://forum.gethopscotch.com//t/um-i-cant-see-the-drawing-topic/35435', "When I click on my notification for it, I can't see it....
```