Methodology Description

Alessandro Ardenghi

March 2024

1 Track1 (Discrete)

In this track, I decided to use TFIDF as representation (with TFIDFvectorizer from Scikit-Learn), with the specific set of parameters I found doing a search over the many different parameter combinations.
To compute the most similar questions to the ones in the test, I got the TFIDF representations and then used the cosine similarity to get the most similar to each test prompt. Regarding the preprocessing, I have manually implemented a tokenization function to correctly separate all words and punctuation, and then I fed the tokenized data (hence no lemmatization or stemming) into the model.

2 Track2 (Static)

In the second track, I chose to use a Word2Vec model pretrained by Google (which I downloaded using: import gensim.downloader as api, path = api.load(”word2vec-google-news-300”, return_path=True) and for each sentence I embedded all the words using the model (and discarded the ones which were not in the model's vocabulary), and then I took the mean of the vectors coordinate-wise. Once again, in this track I tokenized the data with a function I implemented, and then I fed this data to the model.
Again, to get the question similarity I used the cosine similarity.

3 Track3

In this last track, I used a Sentence Transformer pretrained model called $all-mpnet-base-v2$ on the sentences without any preprocessing, obtained the embeddings of each sentence and then simply computed the cosine similarity between each sentence embedding and returned the question from the training and dev set with higher similarity.