NLP for Question Answering Systems

Tarushii Goel

April 2021

1 Introduction

Question-answering is a information retrieval task focused on giving the exact answer to factoid questions (as opposed to a set of relevant articles, which is what google does). For example, you could pose the question, "how many sides are in a triangle?" and the model would output "3".

2 Overview

The most successful approaches to question-answering use NLP and the reader-retriever system, so that is what we will look at.

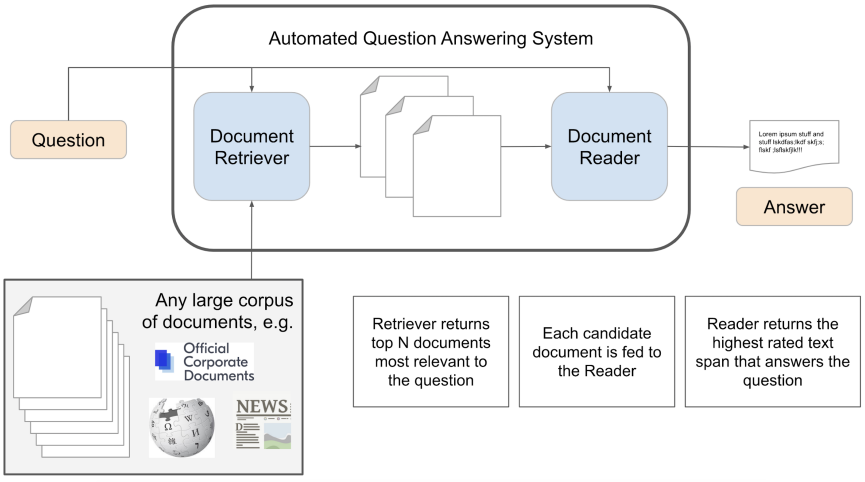

Figure 1: Reader-Retriever system [2]

3 Retriever

There are many different machine learning and non-machine learning based approaches to document retrieval. Almost all of them will focus on converting the questions and documents in the corpus into **vectors**. These vectors encode the information present in the document or question. When you want to retrieve documents for a question you would:

1. Ensure that you have precomputed a document vector for each document in the corpus of documents
2. Encode the question into a vector
3. Calculate the **similarity** between the question vector and each document vector (through some vector similarity measure, such as the dot product)
4. Return the k (you can choose any k) documents with the highest similarity score

The following sections highlight various ways you might approach encoding language into vectors.

3.1 TF-IDF

TF-IDF stands for text-frequency inverse-document-frequency, which makes sense because it computes exactly that. It treats every document as a **bag of word vector**, which just means that the ordering of the words in the document does not matter, only the frequency of the words does. [4] Here is how you would calculate it:

Definitions

tf(t, d) = text frequency of term t in document d
idf(t, D) = inverse document frequency of term t in document corpus D
freq(t, d) = raw frequency of term t in document d

$$tf(t,d) = \log(1 + freq(t,d))$$
$$idf(t,D) = \log(\frac{|D|}{|\{d \in D : t \in d\}|})$$
$$tfidf(t,d) = tf(t,d) \times idf(t,D)$$

Note the difference between capital and lowercase d. Also notice that text frequency is just the **log normalization** of the raw frequency. There are many other ways of defining text frequency, such as term frequency or double normalization, but log normalization has been found to work best.

An even more popular scoring function is BM25. It uses a very similar concept to TF-IDF, just a different and more successful formula. If you would like to learn more about it I invite you to read this (very reliable) Wikipedia article [5].

3.2 Dense Passage Retrieval

The dense passage retriever (DPR) turns to BERT-based language models instead to encode the documents and questions into vectors. The embedder mod-

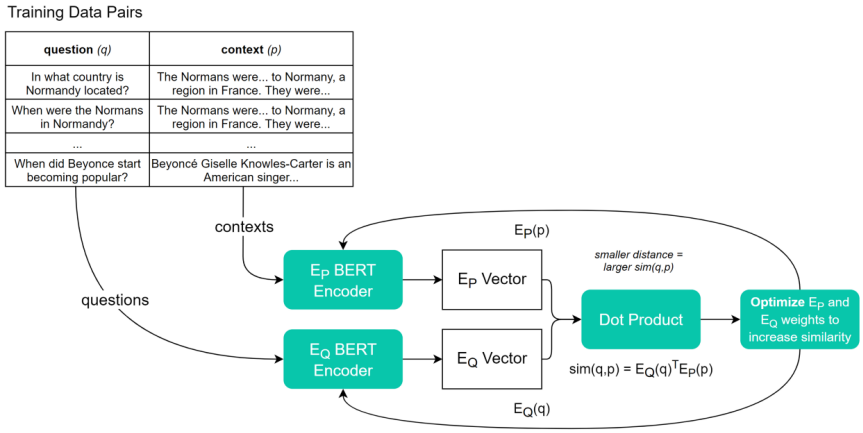

Figure 2: Dense Passage Retriever Training [3]

els are trained by passing the question along with a positive context or a negative context: positive contexts are the ones which contain the answer to the question and therefore should have a high similarity score, while negative contexts do not contain the answer and should have a low similarity score. When constructing the data, positive contexts must be hand-labeled, but negative contexts can be selected through various methods. One of those discussed in the original paper uses BM25 to select "hard" passages which do not contain the answer.

4 Reader

The reader is more like a run-of-the-mill BERT-based model, where your inputs are the documents (which are also sometimes referred to as "contexts") and you are training your model to output the answers to the questions.

4.1 BERT

BERT is a very popular architecture in NLP. It is semi-supervised and trained on two tasks: masked word prediction, and sentence order prediction.

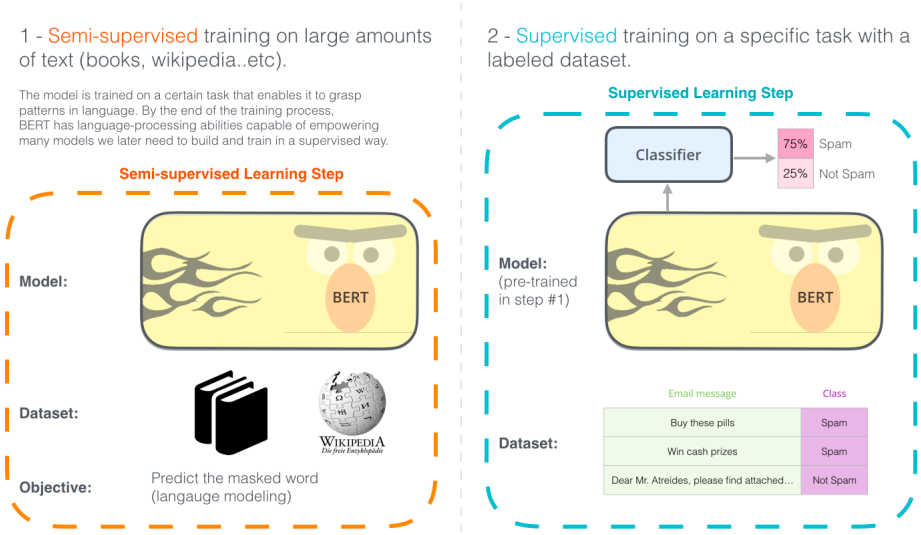

Figure 3: Usage of BERT [1]

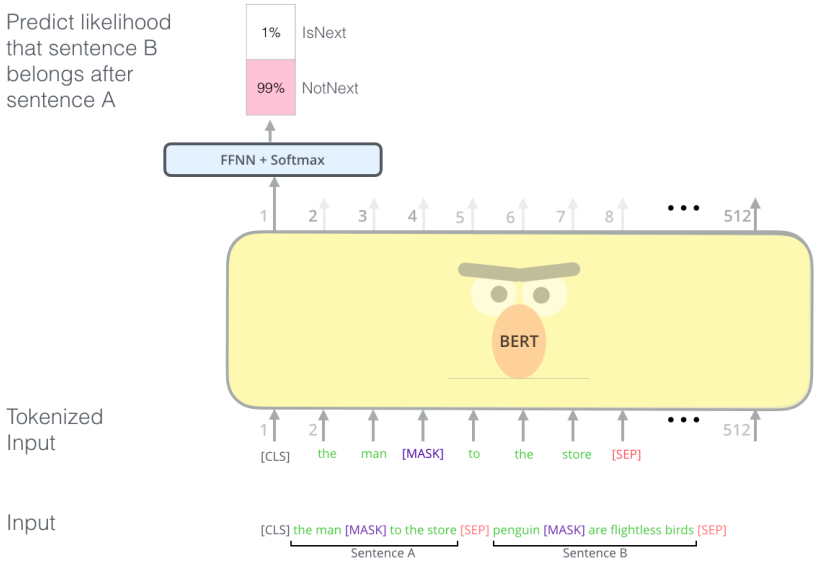

Figure 4: BERT training [1]

References

[1] Jay Alammar. The illustrated bert, elmo, and co. (how nlp cracked transfer learning). `http://jalammar.github.io/illustrated-bert/`.

[2] Melanie R. Beck. Building a qa system with bert on wikipedia. https://qa.fastforwardlabs.com/pytorch/hugging%20face/wikipedia/bert/transformers/2020/05/19/Getting_Started_with_QA.html#2.-QA-dataset:-SQuAD.

[3] James Briggs. How to create an answer from a question with dpr. https://towardsdatascience.com/how-to-create-an-answer-from-a-question-with-dpr-d76e29cc5d60.

[4] Lilian Weng. How to build an open-domain question answering system? https://lilianweng.github.io/lil-log/2020/10/29/open-domain-question-answering.html.

[5] Wikipedia. Okapi bm25. https://en.wikipedia.org/wiki/Okapi_BM25.