QA for People Based on the Available Data

Koushik Modugu
SBID : 112609041

Abhishek Deshmukh
SBID : 112675585

Eesha Gitay
SBID : 112584867

Abstract

As part of this project we aim to leverage the data available for a person and use it to answer questions about the person. We present various approaches taken to address the problem statement and discuss the scenarios that the model performed well on and provide reasoning for the cases that the model didn't perform well on.

1. Introduction

The overall objective of the project is - given some text about a person we should be able to learn an embedding for them and be able to answer queries about the person. If we are able to learn an embedding for a person and are able to answer questions about it, we can extend this technique to solve a wide variety of other problems that tend to be business critical, such as

- Learn an embedding for users on a social platform and determine what kind of ads best targets them.
- Music streaming platforms can learn an embedding for users to study churn rates and do music taste profiling.

We used 2 different strategies to learn representations of entities, compare and contrast their performances on various tasks and evaluate them.

We scraped data about our entities from wikipedia and other news outlets and use it for training our models and to learn embeddings of entities. We plan to evaluate the performances of both the approaches by evaluating their performance on the question and answering downstream task.

This is further highlighted in the subsequent sections.

2. Data

To train the model we collected data about various famous personalities such as :

- Roger Federer
- Beyonce
- Modi
- Trump
- Angelina Jolie
- Bill Gates
- Michael Jackson
- Yoshua Bengio

We acquired data about these personalities from from various publicly available sources such

- Wikipedia,
- News articles
- Journals.

We have included Journals and interviews data too, we had an intuition that these would reveal some aspects of these personalities that isn't available on other platforms. The reason for considering Yoshua Bengio is to check how our model works when the data available for a person isn't large.

3. Approach:

Method 1 : In this approach we used the pretrained BERT model and performed a further fine tuning of person embeddings by feeding the data about the personalities. We tried two different approaches using this method, these are further discussed in the following sections.

3.1 Using pre trained BERT model:

BERT (Bidirectional Encoder Representations from Transformers), provides us vector representations with high quality language features since these were trained

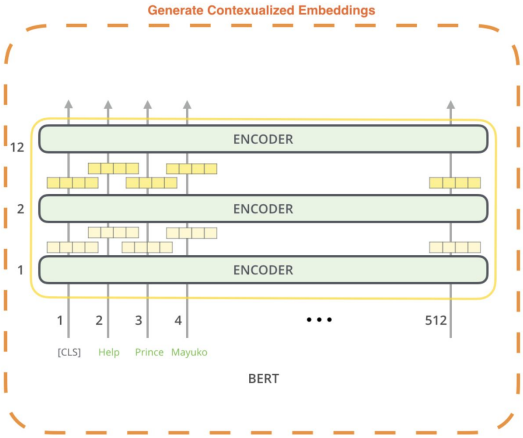
on a huge corpus. We have further performed fine tuning on our data to ensure good results.

Since BERT is a pretrained model it expects input data in a specific format, to adhere with this format we had to

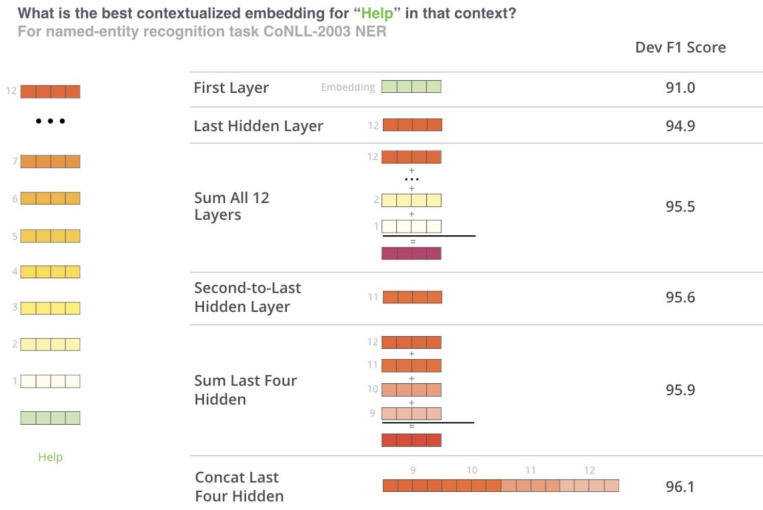
- Add special tokens in the beginning ([CLS]) and end ([SEP]) of the sentences.
- Tokenize the entire input sequence.
- Ensure that the tokens agree with the fixed vocabulary that BERT was trained on.

Once we had the input tokenized, we also added segment IDs to help BERT distinguish between sentence pairs.

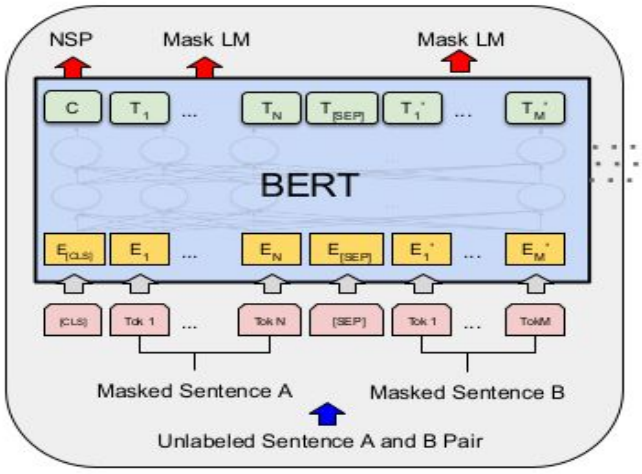
Now that we had everything in place, we fed the data to the the BERT model, which provided us with outputs of the dimensions [# layers, # batches, # tokens, # features]. After doing reshaping we were able to get back the representations for each word in the input at each hidden layer.



Since we needed single representation for each word we had to come up with a way of combining the representation at each layer in the model. To find out which combination of layers representations gives the best results we had to do a literature survey. The findings are as shown in the below figure -



Our observations are also consistent with the results in the above figure. We used the representations of the last 4 hidden layers to construct the final single representation for each entity. This was done because each of the last 4 layers capture different information about the entity. Thus to get the word representations we summed the last 4 layers. Similarly to get question vectors we passed them through the pretrained BERT model and considered the average of each vector from the final hidden layer, to produce a single vector.



The pretrained BERT model has 12 hidden layers with 768 hidden units as shown in the below model.

Once we had the representations for the entities and the question, in order to answer the questions we used spatial distance between the question and the entities.

```
[270] print(type(beyonce))
      print(len(beyonce))
      beyonce

      <class 'torch.Tensor'>
      768
      tensor([ 4.9902e-02, -4.0830e+00,  1.0270e+00, -2.8526e-01,  3.0566e+00
```

Vector representation for Beyonce

```
[269] print(type(federer))
      print(len(federer))
      federer

      <class 'torch.Tensor'>
      768
      tensor([ -2.2607,  -0.5421,   2.6228,  -0.1369,   2.7561,  -0.1194,  -1.1163,
               -0.4163,  -2.5052,  -1.4257,   1.2240,   0.0767,   3.0511,   0.5849,
```

Vector representation for Federer

```
[234] question = "Who is a tennis player ?"
      entity1 = "Federer"
      entity2 = "Beyonce"
      answer = get_answer(question,entity1,entity2)
      print(answer)

      Federer
```

```
[263] question = "who is singer ?"
      entity1 = "Federer"
      entity2 = "Beyonce"
      answer = get_answer(question,entity1,entity2)
      print(answer)

      Beyonce
```

```
[241] question = "who is from Switzerland ?"
      entity1 = "Federer"
      entity2 = "Beyonce"
      answer = get_answer(question,entity1,entity2)
      print(answer)

      Federer
```

As we can see in the above results, using spatial distance as metric to answer multiple choice questions did a decent job.

Since Federer is a tennis player, the data collected for him had a lot of references to tennis which gets encoded into his representation. Similarly data for Beyonce had a lot of references to singing which also gets encoded into her representation. Thus in the vector space Federer lies closer to tennis than Beyonce while Beyonce lies closer to Singing then Federer.

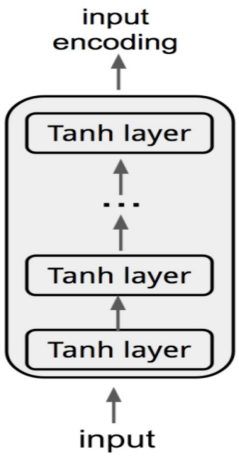
Challenges

BERT has a certain vocabulary list which comprises of WordPieces (subwords). BERT can tokenize any word not present in its vocabulary into tokens present in the vocabulary. BERT has learnt embeddings for each of these tokens by training over large corpus of data. One issue that we ran into during the embedding generation for the people is that BERT tokenized names of the people. It used embeddings already learnt for these tokens along with the context to generate embedding for the people. The attention associated with context is less than the attention associated with the word token itself. Thus, irrelevant word tokens are contributing a large portion to the embedding of the person. For example, Modi was being split into 2 subparts namely Mod and I. The learnt embeddings for Mod and I are contributing to the embedding for Modi which is bad as tokens aren't connected to person Modi. We got good results for the words that were present in BERT vocabulary and poor results for the words whose embeddings are learnt using context around the people in the data we collected.

3.2 Using Universal Sentence Encoders:

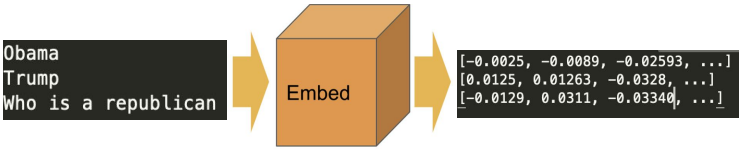
One other model that we tried for learning representations for questions and entities is the Universal sentence Encoders.

This is a Deep Averaging Network (DAN) sentence encoding model which begins by averaging together word and bigram level embeddings. Sentence embeddings are then obtained by passing the averaged representation through a feedforward deep neural network. The primary advantage of the DAN encoder is that compute time is linear in the length of the input sequence.



DAN based Universal Sentence Encoder

This universal sentence encoder provides a vector representation of 512 dimensions for any input as shown below



Encoding inputs into a 512 dimensional Vector

Using this DAN based architecture of the universal sentence encoder we generated vector representations of questions and the entities and tried answering the questions using spatial distance as the metric.

This model got the right answers almost all the time for prominent people. Few of the questions and answers results are attached below -

```
# Compute a representation for each message, showing various lengths supported.
entity1 = "Obama"
entity2 = "Beyonce"
question = "who was president"
messages = [entity1, entity2, question]
```

```
Q: who was president
A: Obama
```

```
# Compute a representation for each message, showing various lengths supported.
entity1 = "Rafael Nadal"
entity2 = "Roger Federer"
question = "who is from spain"
messages = [entity1, entity2, question]
```

```
Q: who is from spain
A: Rafael Nadal
```

```
# Compute a representation for each message, showing various lengths
entity1 = "Modi"
entity2 = "Trump"
question = "who is from India"
messages = [entity1, entity2, question]
```

```
Q: who is from India
A: Modi
```

3.3 Classification Based Model

These days companies own resources to train a single model for all their customers together. In this approach, we consider a 4 layered Neural network with a softmax applied on the last layer to classify (associate) input with one of the people or None of them. Thus, if we are learning the model for K people, the output layer would have (K+1) nodes. Extra node for None. The input is the sentence representation generated by the Universal Sentence Encoder which is a vector of size 512. The hidden layers are of size 300, 200, 100 respectively. Each hidden layer is followed by Batch Normalization and Dropout of 0.4.

The data we have gathered are broken at Period (.) to avoid generating sentence representation for long sentences. For the 8 people considered, 10,000 (1000 - 1500 per person, 70 in case of Bengio) sentences were available for training purposes. Not all sentences are well formed sentences. An extra set of 4000 sentences were added to the training set labelled None. The intuition behind this is that, if a question irrelevant to all the people is asked, the output should be None. The data associated with the None output is picked from fiction novel Moby Dick and Sally Dows. During training, when a sentence representation associated with a person is input, the output should be the Node in the Output layer associated with the person. During testing phase, the name of the person in the sentence is masked and then the sentence representation for the sentences are

pumped into the model and the node in the output layer for which the softmax value is highest is picked as the output. Cross Entropy loss is used for the backpropagation purposes.

4. Results

```
['Who is born in Switzerland ?'] federer
['Who is a singer ?'] beyonce
['Who is a dancer ?'] jackson
['Whose album was on BillBoard ?'] jackson
['Whose won Turing award ?'] bengio
['Whose is working in artificial intelligence ?'] bengio
['Whose is working in machine learning ?'] bengio
['Who is a Cricketer ?'] tendulkar
['Who belongs to India ?'] None
['Who is Rich ?'] gates
['Who served Prison Time ?'] jackson
['Who is in Software Industry ?'] gates
['Who is not a singer ?'] jackson
['Who is a Computer Scientist ?'] gates
['Who is Scientist ?'] dicaprio
['Who is into the Research ?'] gates
['Who sponsored the brand MRF ?'] tendulkar
['Who does farming ?'] None
['Who owns a vineyard ?'] None
['Who is famous ?'] dicaprio
['Who is Corrupted ?'] None
['Who is an alcoholic ?'] None
['Who grew up in Namibia ?'] federer
['Who owns the military ?'] dicaprio
['Who is She ?'] beyonce
['Who is Woman ?'] None
['Who is Female ?'] None
['Who plays Tennis ?'] federer
['Who is a Movie Star ?'] dicaprio
['Who is divorced ?'] jackson
['Who is dead ?'] None
['Who plays video games ?'] dicaprio
['Who played in Rawalpindi ?'] tendulkar
['Who is born in Pakistan ?'] tendulkar
['Who won Grand Slams ?'] federer
['Who is called Master Blaster ?'] tendulkar
['Who is member of Rajya Sabha ?'] tendulkar
['Who worked with Scorsese ?'] dicaprio
['Who worked with Director Scorsese ?'] dicaprio
['Who worked with Movie Director Scorsese ?'] dicaprio
['Who is an enviromentalist ?'] dicaprio
```

5. Analysis of Results:

The model has an accuracy of 85-88% in correct classification of sentences during validation. Also, The model answered all the questions very well. We are specifically more impressed with results to a good number of the questions whose answer is captured in just 1 or 2 instances of training data sentences. The model did well on questions associated with Benjio for whom we had very limited data. Also, when irrelevant are asked, model did the right job by picking None as the output rather than picking one of the people which is good. The model failed to capture the structure and semantics of the questions and worked like BOW model for some questions. We expected the sentence representations generated for the questions to handle this for us. For instance, When asked, “Who is she ?”, model correctly classified it as Beyonce (referred a lot as she in data) but failed to do that when “Who is female ?” is asked. This specific scenario is addressed when Sentence encodings are generated using pretrained-BERT but overall results suffered. This issue can probably be addressed by fine-tuning the BERT, which we didn’t do.

6. Other Models Considered

1. *We have thought of one other approach that doesn’t require any entity embedding learning*. Pick a pre trained QA model fine tuned on top of BERT. Now, of the 2 sentences input to BERT, the first sentence would be the question. The second sentence would be segments of the text associated with the person about whom we are trying to answer questions. The text associated with the person would be pumped into the model as the 2nd sentence segment after segment based on sentence length limits. Ideally, the model would output Start/End span with high probability for one of the text segments and in case of other segments as the answer isn’t present in them, the start/end span pointers would point to CLS token (1st output token). Of all possible such iterations run for a person, we shall output the result as the one that generated the highest probability for the start/end span.

2. We’ve considered various other architectures that require significant amount of data to train. Named Entity Recognition datasets with entities masked can be used for sentence representations and the entity that is masked as the output that should be generated. This scenario is close to that of ours, but again, the data we found isn’t sufficient to train the model.

7. Code

Drive Link to the code : https://drive.google.com/drive/folders/1D84SlOyzKxeMRzaP3qX-SotBc8LR_ZuK?usp=sharing

8. References

- https://towardsdatascience.com/understanding-entity-embeddings-and-its-application-69e37ae1501d
- https://blog.insightdatascience.com/entity2vec-dad368c5b830
- BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova arXiv:1810.04805
- https://medium.com/huggingface/universal-word-sentence-embeddings-ce48ddc8fc3a
- https://mccormickml.com/2019/05/14/BERT-word-embeddings-tutorial/
- https://www.dlology.com/blog/keras-meets-universal-sentence-encoder-transfer-learning-for-text-data/