Questions

1) Which techniques you have used while cleaning the data if you have cleaned it?

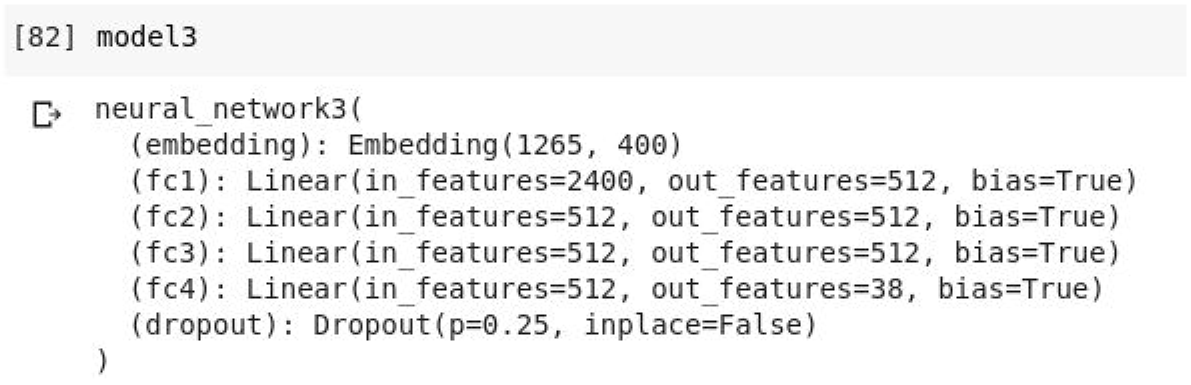
- At first, I’ve removed all one-character words in samples at the input data.
- I’ve removed any Non-English words.
- I’ve removed places and cities from input vocabulary as I believed that they shouldn’t contribute to determining a job function.

2) How does your model work?

I have used a model based on deep neural network

- The model gets data as numeric vectors produced by some preprocessing functions which map every word in the input text to its integer value.
- then we pad input vectors with zeros to form a constant Pre-defined length which will be the input size to our network. at first, I've set that input size to 10 to cover the longest input samples in the dataset, then I found that reducing it to 6 has led the model to perform better.
- After that, input get forward passed through network layers of the model to result a final vector of length 38
- After that, the input gets forward passed through network layers of the model to result in a final vector of length 38 which is the total number of classes(job functions) we trying to predict. those logits(output score values) to each class then passed through sigmoid function to produce an independent probability for each class (from 0 to 1) of being a true output for that input .
- then we choose a threshold probability value to consider any class should exist in output or not. I've set that threshold to 0.35
  Note: reducing that threshold would lead to catching more classes that may increase the recall score but also will drop the precision score a little bit.

Model architecture in details:



```
[82] model3

neural_network3(
  (embedding): Embedding(1265, 400)
  (fc1): Linear(in_features=2400, out_features=512, bias=True)
  (fc2): Linear(in_features=512, out_features=512, bias=True)
  (fc3): Linear(in_features=512, out_features=512, bias=True)
  (fc4): Linear(in_features=512, out_features=38, bias=True)
  (dropout): Dropout(p=0.25, inplace=False)
)
```

3) Why have you chosen this approach?

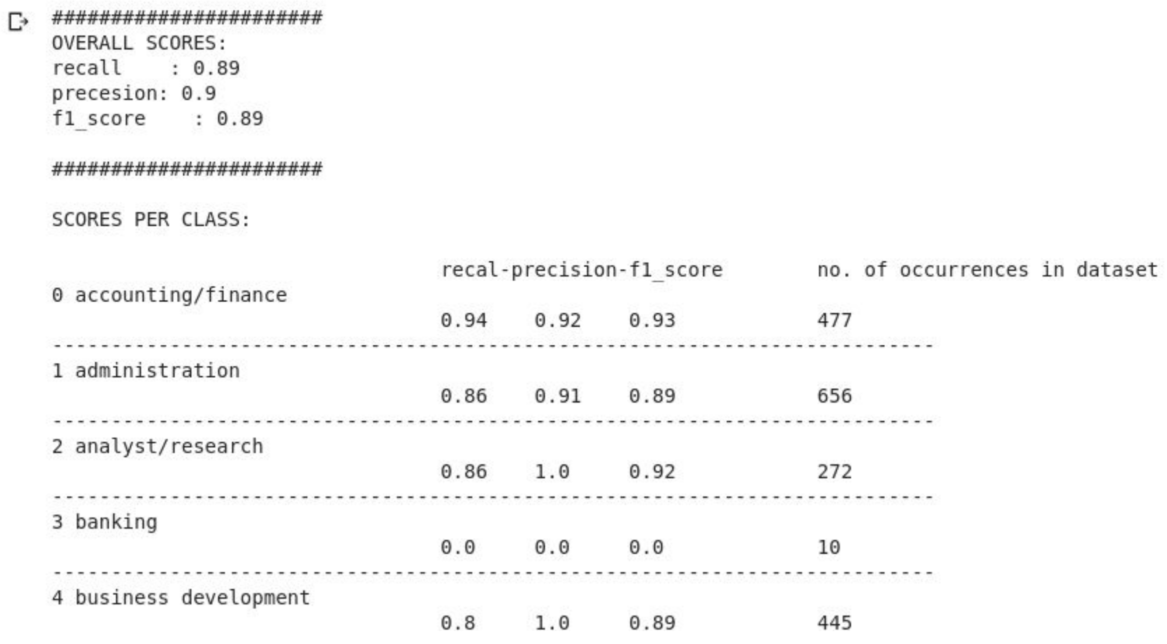
Deep learning methods are proving very good at text classification, achieving state-of-the-art results on a suite of standard academic benchmark problems.
Also, I think deep learning is the most preferable approach for handling Unstructured data like text data. especially, when it comes to multi-class multi-label problems.

4) How can you extend the model to have better performance?

- capturing relations between words is a key point in text classification tasks. embedding models is a powerful way to achieve this.
  when I 've built the first model for this problem it was a simple multi-layer perceptrons network. it didn't do very well (achieve just around 0.5 recall score). so, I decided to add an embedding layer at the beginning. and the score increased to around 0.9 easily.
- so, as a better extension. we can use a pre-trained embedding model with pre-trained weights like facebook Fasttext model which was trained on millions of words and sub-words which helps in recognizing new words and handling user input typos.
  also, those pre-trained models can be used for identifying and representing different languages words. meaning that we can build a multi-language job recommendation model.
- We can also increase the size of our dataset by replicating samples with a shuffled word order which may also help to reduce over-fitting to training data as the model won't come to memorizing the usual order of specific input words but instead, it will learn the contribution of each word.

5) How do you evaluate your model? (i.e. accuracy, F1 score, Recall)

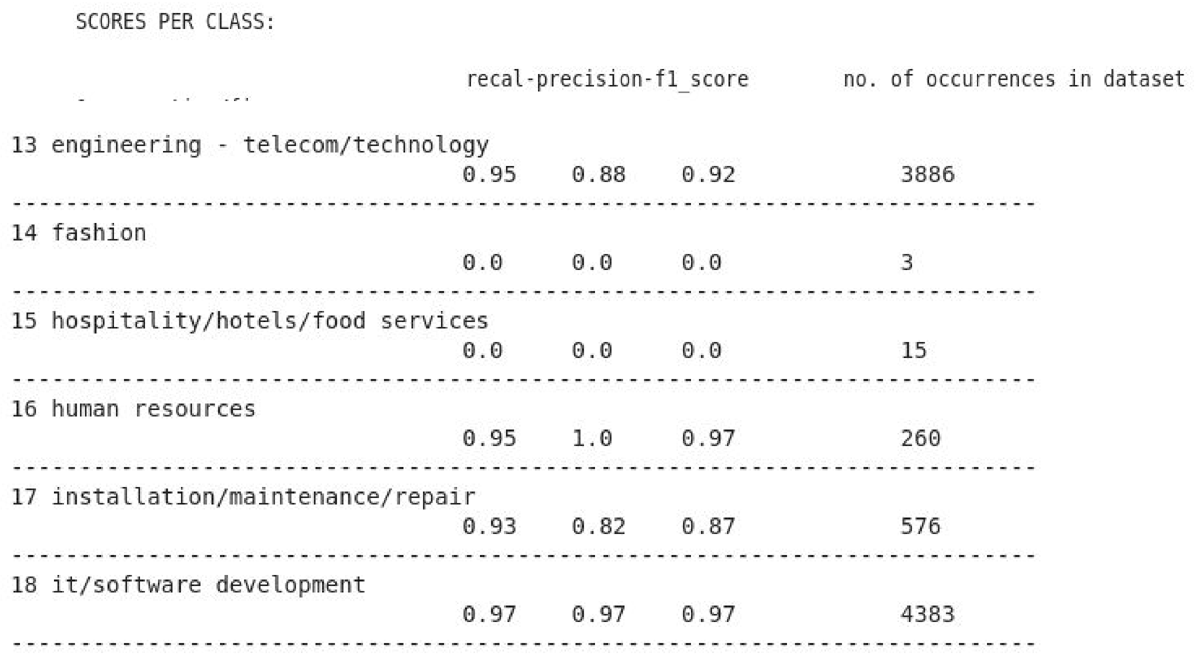
I've used recall, precision, F1_score metrics for evaluating the model.



```
#######################
OVERALL SCORES:
recall    : 0.89
precesion: 0.9
f1_score    : 0.89

#######################

SCORES PER CLASS:

                                 recal-precision-f1_score        no. of occurrences in dataset
0 accounting/finance
                                 0.94    0.92    0.93            477
---------------------------------------------------------------------------
1 administration
                                 0.86    0.91    0.89            656
---------------------------------------------------------------------------
2 analyst/research
                                 0.86    1.0     0.92            272
---------------------------------------------------------------------------
3 banking
                                 0.0     0.0     0.0             10
---------------------------------------------------------------------------
4 business development
                                 0.8     1.0     0.89            445
```

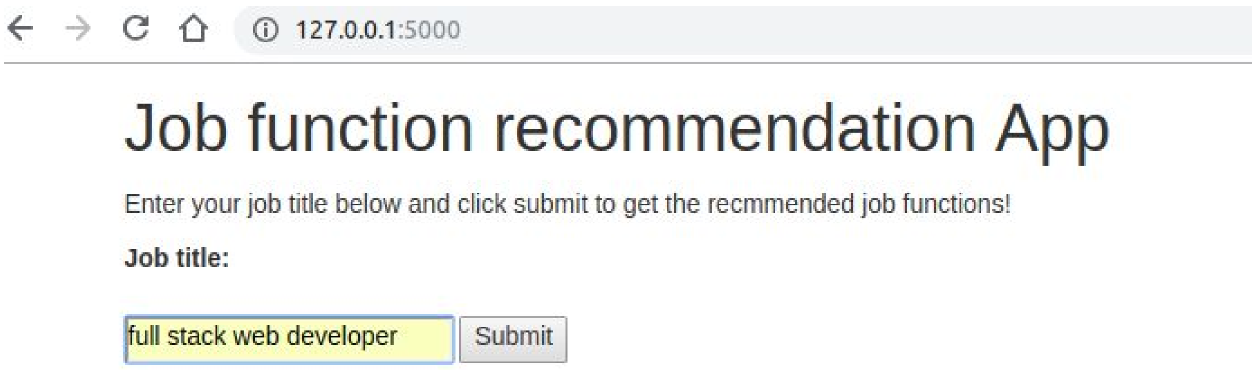
6) What are the limitations of your methodology or where does your approach fail?

- it was obvious that the model does fail with job functions that didn't appear that much in the dataset.
  we could easily increase model performance by oversampling those low-frequency classes as it is clear the huge difference between their scores and the classes that have enough samples(shown in the figure below).



```
      SCORES PER CLASS:

                                   recal-precision-f1_score       no. of occurrences in dataset

 13 engineering - telecom/technology
                                   0.95    0.88    0.92            3886
 ----------------------------------------------------------------------------
 14 fashion
                                   0.0     0.0     0.0             3
 ----------------------------------------------------------------------------
 15 hospitality/hotels/food services
                                   0.0     0.0     0.0             15
 ----------------------------------------------------------------------------
 16 human resources
                                   0.95    1.0     0.97            260
 ----------------------------------------------------------------------------
 17 installation/maintenance/repair
                                   0.93    0.82    0.87            576
 ----------------------------------------------------------------------------
 18 it/software development
                                   0.97    0.97    0.97            4383
 ----------------------------------------------------------------------------
```

API screenshots

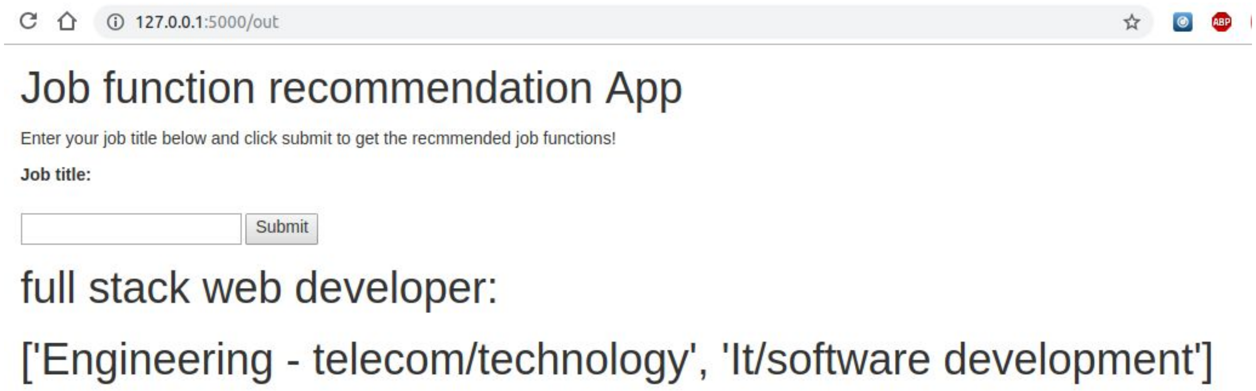

127.0.0.1:5000

Job function recommendation App

Enter your job title below and click submit to get the recmmended job functions!

Job title:

full stack web developer


127.0.0.1:5000/out

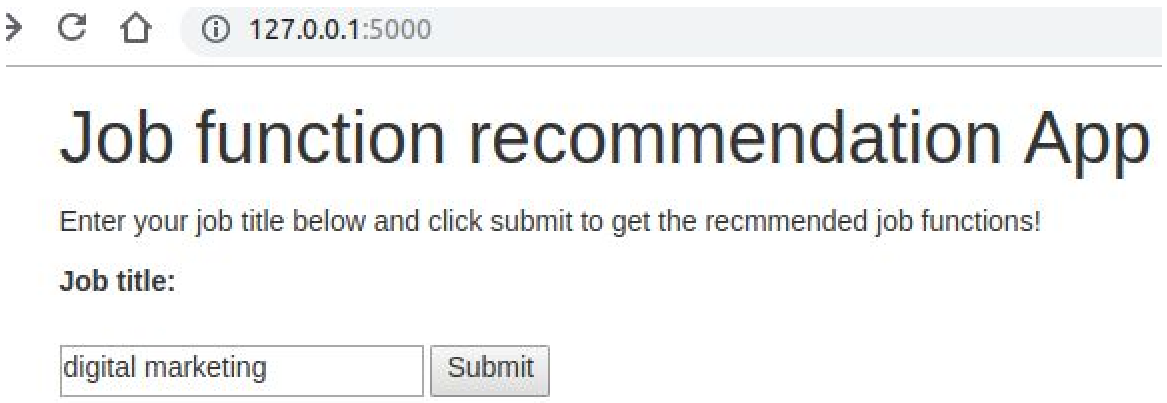
Job function recommendation App

Enter your job title below and click submit to get the recmmended job functions!

Job title:

full stack web developer:

['Engineering - telecom/technology', 'It/software development']

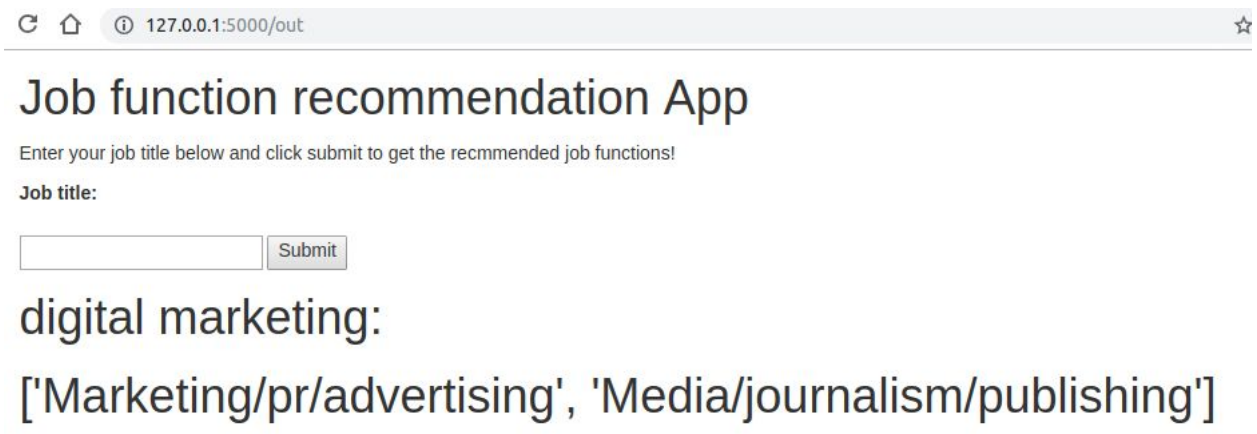

127.0.0.1:5000

Job function recommendation App

Enter your job title below and click submit to get the recmmended job functions!

Job title:

digital marketing



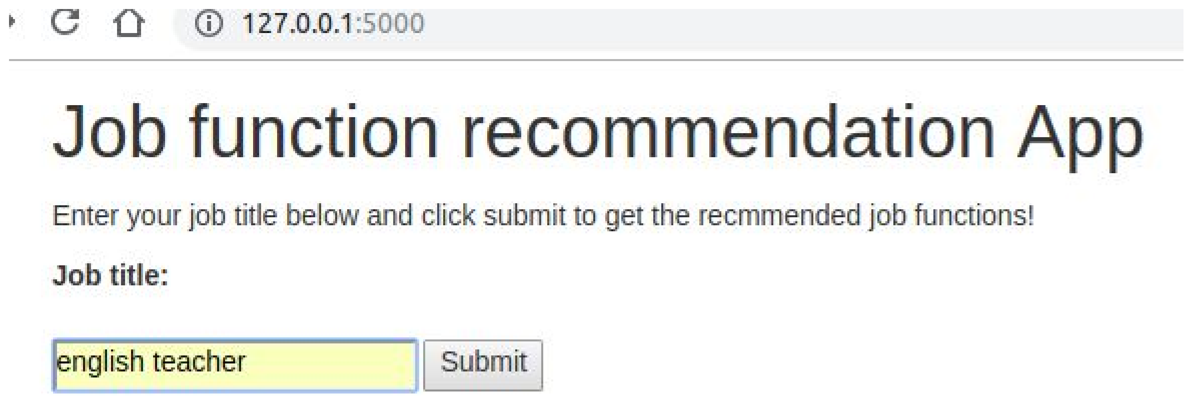
Job function recommendation App

Enter your job title below and click submit to get the recmmended job functions!

Job title:

digital marketing:

['Marketing/pr/advertising', 'Media/journalism/publishing']

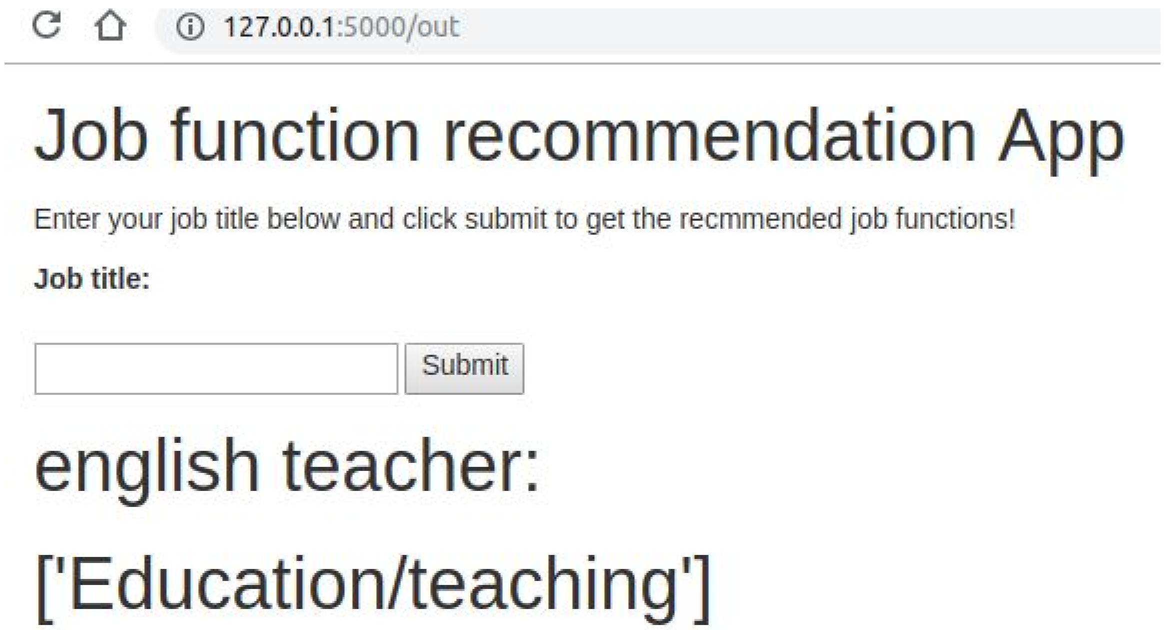


Job function recommendation App

Enter your job title below and click submit to get the recmmended job functions!

Job title:

english teacher



Job function recommendation App

Enter your job title below and click submit to get the recmmended job functions!

Job title:

english teacher:

['Education/teaching']