ISMT S-117 Text Analytics and Natural Language Processing (2020 Summer)

Final Project Report

Resume Screening System Using RoBERTa

by Micheal Choi Kwong Chak

Table of Contents

[Introduction 3](#_Toc47732662)

[Problem Statement 3](#_Toc47732663)

[Project Objective 3](#_Toc47732664)

[Description of Dataset 3](#_Toc47732665)

[Resumes Dataset 3](#_Toc47732666)

[Job Descriptions Dataset 4](#_Toc47732667)

[Description of Methodology Used 5](#_Toc47732668)

[Building the Training / Testing Corpus 5](#_Toc47732669)

[Models Setup 6](#_Toc47732670)

[Tools Used 6](#_Toc47732671)

[Project Repository Structure 6](#_Toc47732672)

[Detailed Project Description and Findings 8](#_Toc47732673)

[1. Explore Dataset 8](#_Toc47732674)

[2. Create Labels from Job Description and Resume Datasets 13](#_Toc47732675)

[3. Use Cosine Similarity to Estimate Interview Worthiness of Resumes to Jobs 15](#_Toc47732676)

[4. Similarity Classification Using Sentence Pair Comparison with RoBERTa 16](#_Toc47732677)

[5. Model Trial on Testing Data 18](#_Toc47732678)

[Project Conclusion 19](#_Toc47732679)

[Deployment Strategy 19](#_Toc47732680)

[References 20](#_Toc47732681)

# Introduction

In the labor market, resumes and job descriptions are the documents to describe the applicants and the jobs they applied, respectively. These are natural candidates for natural language processing and enable both the HR professionals to find their best candidates for the jobs, and vice versa. With 98% of resumes that needs to be rejected (Elmers, n.d.) before a candidate worth interviewing is found, it is a great productivity improvement to build a recommendation engine for an job application tracking system.

# Problem Statement

Traditional resume parsers often focus on key word matching and sometimes supplemented with results of NER (Named Entity Recognition) (Wu, 2019), or one step forward, compare document similarities using fixed document representations like doc2vec or GloVe (Raman, 2019). However, the content of both resumes and job descriptions are 1) context sensitive, for example, two jobs named “software engineer” can mean very different things, and 2) domain knowledge required to succeed in the job are not necessarily described even in the job description. Optimally, to build a recommendation engine one needs to provide resumes labelled as suitable for interview for the job. However, as the job descriptions of the same job evolves over time, and that the goal of the recommendation engine is to save human labor, 3) such data is likely unavailable, at least for the initial stage of the system build up.

# Project Objective

The project targets to resolve problem 1) by using RoBERTa (Liu, et al., 2019) to learn the word representation under the context it is given so as to provide a better baseline for similarity calculation.

For problem 3), while resumes matching particular job descriptions are not readily available, an insight that the people holding similar positions will have their resume written in a way that would be worthwhile for an interview. With these resumes labeled as likely candidates in the training data into BERT for a particular job description, it is expected to have an improvement on the accuracy.

With the model learning other resumes and job descriptions documents, the resultant model will also contain knowledge to the domain and not just the job description in question.

# Description of Dataset

In this project we are targeting to limit both resumes and job descriptions to IT related jobs to limited the context and vocabs to be learned by the model. So I have selected the following data sets under this criterion:

## Resumes Dataset

* Resume Dataset from Avani Siddhapura (Siddhapura, 2020) <https://www.kaggle.com/avanisiddhapura27/resume-dataset>
  1. This data set has 14800 IT related resumes with past job experiences, education records, certificates and skills parsed into separate sections and this is optimal for selecting target description for particular jobs title for labelling as mentioned above

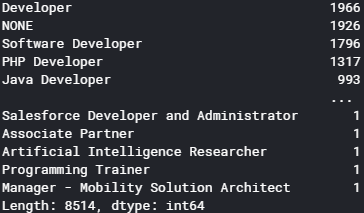


Figure 1: Counts of Job Title in Avani Resume Dataset

* 1. As this is segregated into sections, each job description entry in the past experience of the resume is relatively short with many null values which may or may not be a good thing in terms of training. It is found that by removing any rows with less than 20 tokens will generally give a reasonable job description.

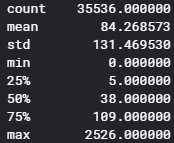


Figure 2: Distribution of # tokens for each job descriptions in Avani Resume Dataset

* 1. All resumes in this dataset is from India which could create a bias in the training but that is common to all resume datasets available, so we will have to keep this in mind on result interpretation. As more relevant data is generated from a deployed system, the model can be retrained for data that conform to what the population is like for the particular company.

## Job Descriptions Dataset

* Online Job Postings – Armenian Online Job Posts from 2004 – 2015 (Hab, 2017) <https://www.kaggle.com/madhab/jobposts>
  1. This dataset has 19000 online job posting on Armenian human resource portal, with 3759 IT jobs covering all areas.

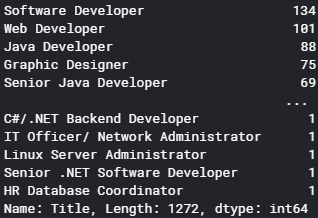


Figure 3: Job Titles in Armenian Job Posting Dataset

* 1. The data is a bit dated but 50% of all IT jobs are after 2010
  2. The average # of tokens for each post is around 300 which is suitable for BERT sentence pair classification (max 512)

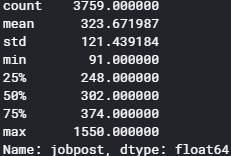


Figure 4: # Token Distribution for Armenian Job Posting Dataset

# Description of Methodology Used

## Building the Training / Testing Corpus

As we do not have a labeled corpus during the initial training of the model, similar job titles embedded using the pretrained model to calculate the embedding for similarity. Similarities over a threshold are labeled as a match for the training set. As more resume-job description matches are generated during the course of daily HR activities after the model is rolled out, a better set of data can be used to train the model.

As mentioned in the previous sections, the Avani resume data set, where each job description, written by the current holder of the position, is matched with the job title, is used to train the model, matching randomly with the job descriptions to build the training set.

The Armenia Job Description dataset also contains a lot of different fields where only selected fields like Job Description, Job Responsibilities and Qualifications are relevant to the resume suitability for interviews. So we will have these extracted instead of training with the whole job description to improve accuracy.

Currently we are only using IT related jobs for training. In a production env we can extend this into all job categories and compare the results.

## Models Setup

In this project, we are going to build two variants of the implementation using RoBERTa, including:

* Use sentence pair classification model to classify whether a candidate of a resume is worthy of an interview
* Use a pretrained RoBERTa model to generate customized embeddings for cosine similarity computation

One of the General Language Understanding Evaluation Benchmark (GLUE) tasks tested on the original BERT model is to find similarity on the STS-B corpus. While the original corpus is much shorter than job description or a resume, in essence it is still a similarity classification of two sentences so it is very suitable to deal with this kind of problems. Supposedly, the domain knowledge not really present in the particular job description is also captured in the model – so it would supposedly provide improved prediction capabilities

On the other hand, the task above should not be different from simply fine-tuning a pretrain-ed model and embed documents with the model, as all decision information should come from the embeddings. From our results, it seems the job descriptions and work experience descriptions from resumes occupies distinct parts in the vector space – so it takes an additional feed forward network to capture the pattern.

## Tools Used

In this project we are going to do both and compare on the accuracy and usability of these implementations. In either case we are going to use the Simple Transformer library (Rajapakse, 2020) <https://github.com/ThilinaRajapakse/simpletransformers#simple-transformers> which is a wrapper for the HuggingFace (Huggingface, n.d.) implementation on PyTorch.

On visualization we are going to use a mix of matplotlib and Weights and Bias to monitor training progress.

# Project Repository Structure

GitHub repo link: <https://github.com/crispin-nosidam/vec2rec-with-RoBERTa>

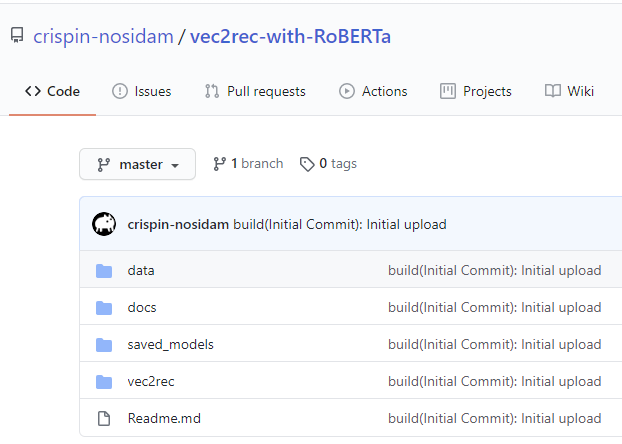


Figure 5: Project Repo Structure

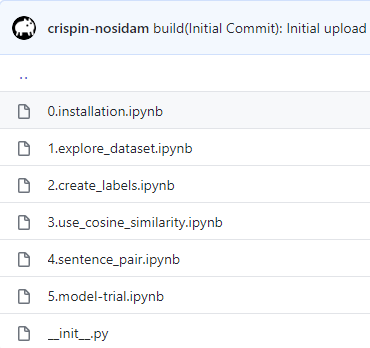


Figure 6: vec2rec Package Content

* vec2rec package: All code is in the vec2rec python package. There are 6 python notebooks:

1. Installation procedure for Colab. This is to allow the correct packages to be installed on Colab. Expected python package combination is also included.
2. Data Exploration – in addition to what is done above to explore the datasets, LDA and BERT embedding is used to discover data pattern and suitability of transformer usage.
3. Create Labels – as mentioned above, we do not have a labelled dataset between job and resume work experience to train the model with. This notebook is to build the training set using the heuristic of RoBERTa sentence-pair similarity of the **Job Title** as the label, which is joined with **Job Description** from the Job Posting and Resume for subsequent training (Job Titles are not included in training).
4. Cosine Similarity – instead of using transformer model, this code uses the RoBERTa embedding to embed the job descriptions then see how well cosine similarity perform as a predictor
5. Sentence Pair – Train RoBERTa model sentence pair with the labelled dataset we created in 2 and see how well the model performs as a predictor to the training and validation sets
6. Model Trial – run the test dataset against the model and see how well it performs.

* Other directories
  1. data – contains job and resume dataset
  2. saved\_models – contains STS-B saved model for label creation, and Good-SM saved\_model as one of the better trained models in notebook 4. As the model pickle files are too large, they are zip and split – and **needs to be recombined by tools like winzip** before they can be used again.
  3. doc – contains documentation

# Detailed Project Description and Findings

## 1. Explore Dataset

1. Firstly, the files for job description and resume are loaded by the function desc\_df which will also print a sample of the summary statistics of each column
2. Using spaCy BERT model, and then with LDA, 50-60 IT jobs and 25-30 non-IT jobs selected from job description and resumes are embedded into vectors
3. Using t-SNE to reduce the dimension into 2D, they are plotted with matplotlib with points highlighted into different colors for “developers” (**blue**), “programmers” (**red**), “administrators” (**green**) and non-IT jobs (**black**)
4. For BERT embeddings, 3 plots are arranged:
   1. Job Description data points
      * Here we can see programmers and developers are really close and IT administrators are somewhere in between
      * On the other hand, the non-IT tasks are generally clustered outside of the IT jobs, which shows good correlation in the corpus and suggesting that the transformer is a good match for the solution.

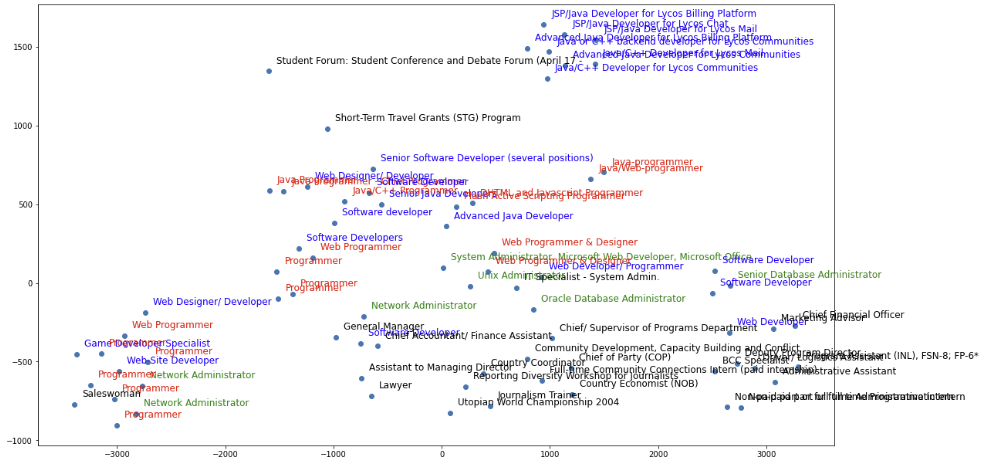


Figure 7: Job description vectors embedded with BERT

* 1. Resume job experience data points
     + Here we can still observe that developer and programmer generally go with one another with IT admins on one side
     + However, we are seeing a far less distinct segmentation for non-IT jobs.

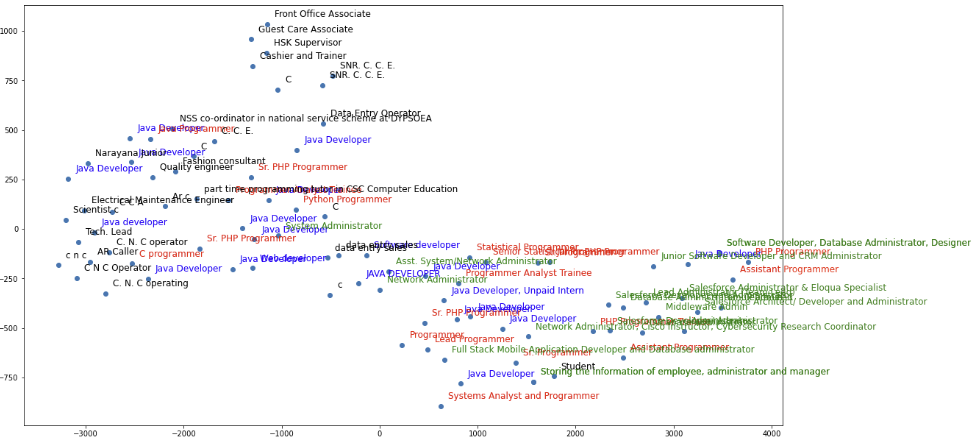


Figure 8: Resume job exp embedded with BERT

* 1. Job and Resume data points combined
     + Here are putting the two graphs together with jobs as circle markers (**o**) and resumes as triangles (**∆**).
     + We can observe that while these two sets are descripting similar job natures, their wording are not exactly similar and thus the embeddings are clearly separated into two groups.
     + Later in the project illustration we can see that this would present problems if we use simple cosine similarities as our predictor for worthiness for interview of resumes.

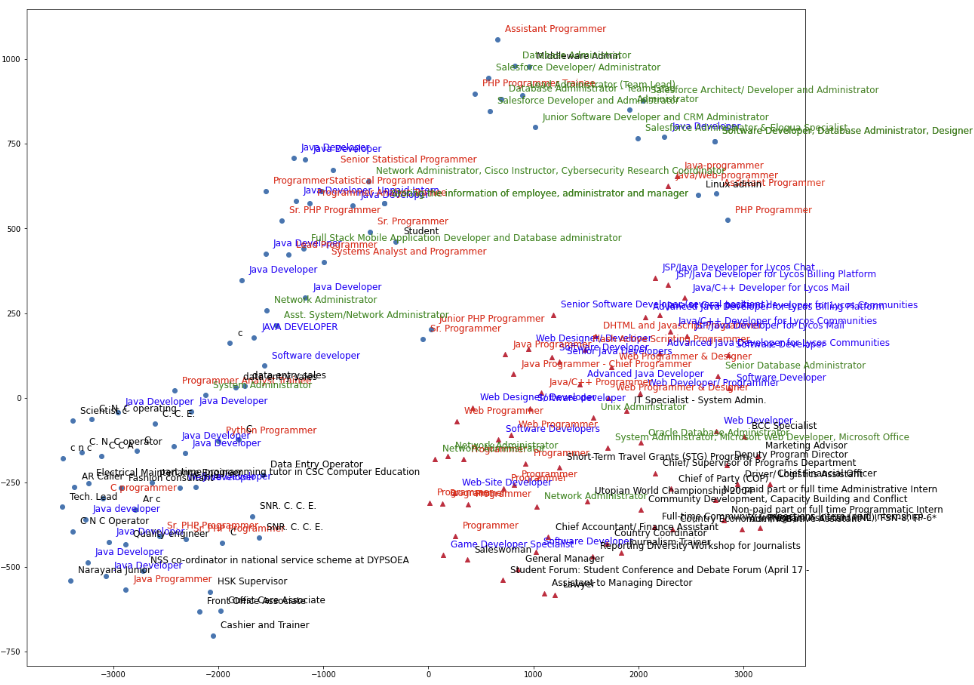


Figure 9: Job Desc and Resume Combined, Embedded with BERT

1. For LDA embeddings
   1. While the countvectorizer is initialized with the full set of data only 1000 records are chosen to allow quick completion of this exploration.
   2. Here we choose a relatively large n\_components (100) in hopes to capture the variety of jobs. As shown in the below only a handle will be considered relevant in the dim-reduced representation. For example:

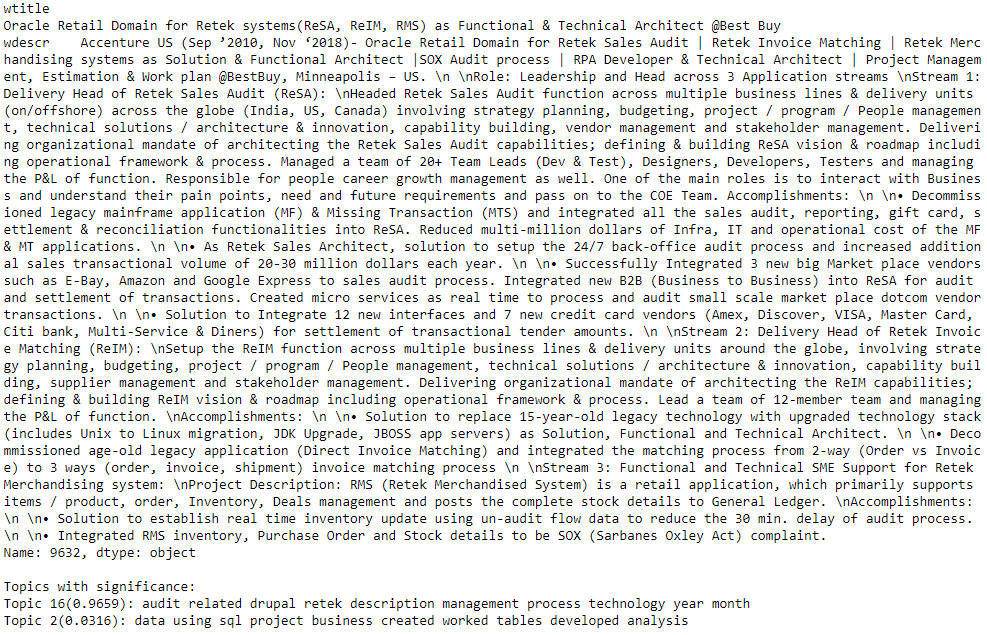


Figure 10: Sample Resume Job Exp with LDA Topics with significance > 0.0002

* + - We can see that words in the first topics like Audit, Retek, which has a significance of 0.97, quite neatly summarized the job.
  1. On the other hand, it does not perform as good for embeddings. as the results are clearly inferior to BERT embeddings, here we do not combine both data points into a single mode but are trained separately, as below:
     + Resume Job Experience data points:
       1. Here we can see that, while the developers and programmers still group together, with the IT admins grouped aside, the distinction is far less clear cut. The non-IT jobs are spread across different types
       2. This is also consistent with our embeddings with BERT.

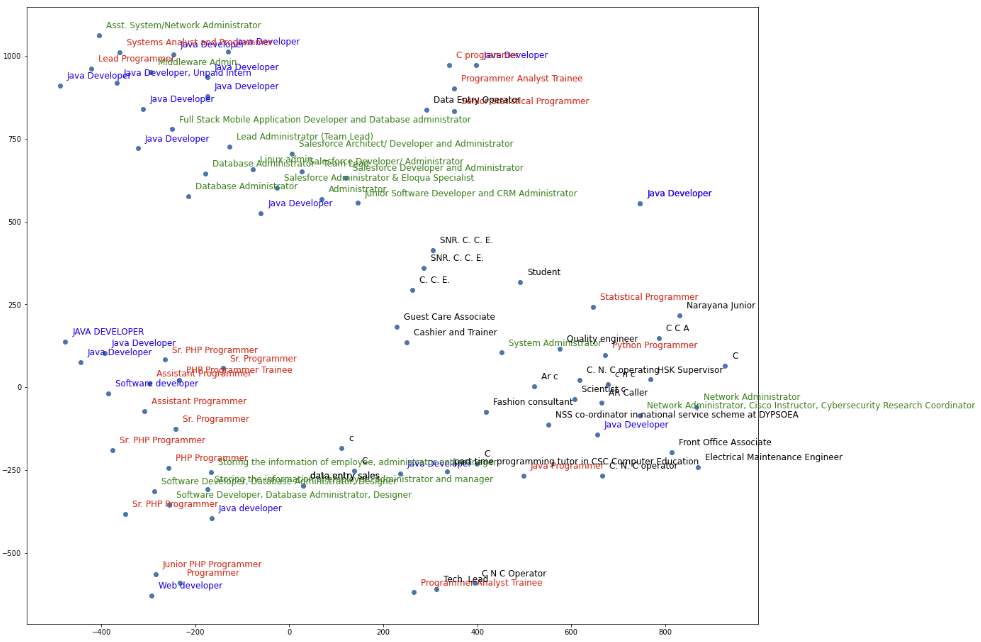


Figure 11: Resume Job Exp Embedded with LDA

* + - Job Description data points
      1. Here we can see that developers are quite similar to programmers, with IT admin in between and non-IT jobs clustered
      2. However, there are outliers too.
      3. This is consistent with our embedding with BERT.

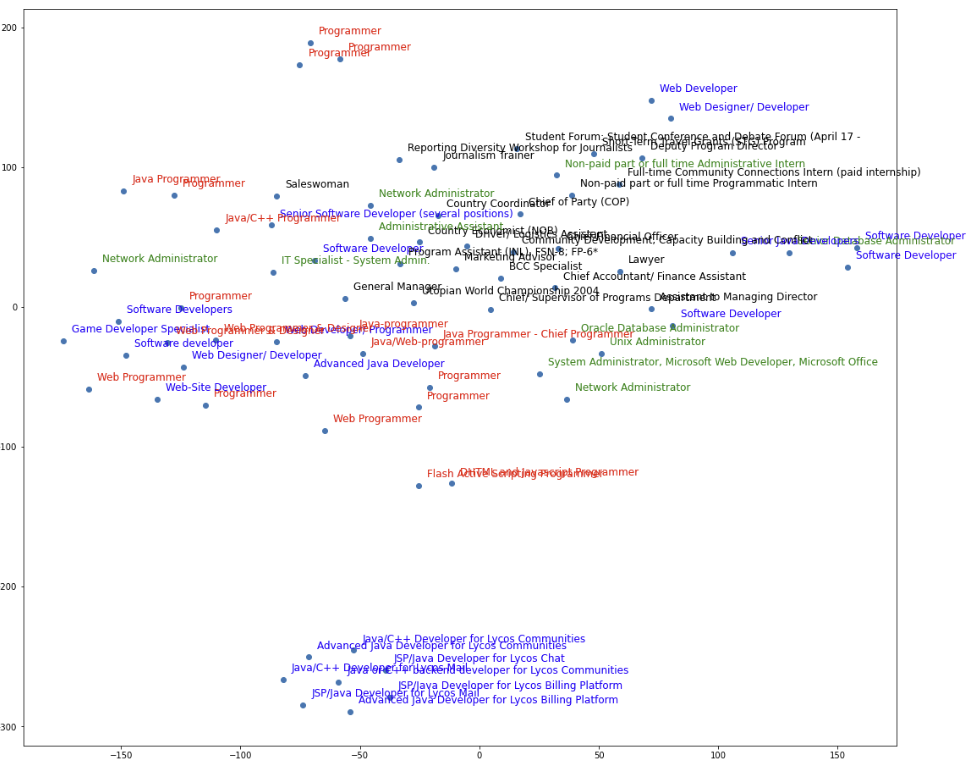


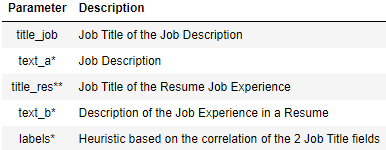
Figure 12: Job Descriptions Embedded with LDA

* 1. With BERT distributions generally more well-formed, it confirms our direction to use transformers like BERT, or RoBERTa for our sentence pair classification engine.

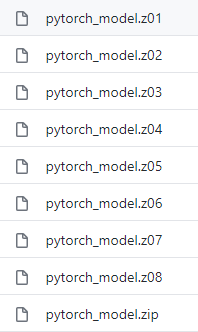
## 2. Create Labels from Job Description and Resume Datasets

As we do not have a labelled training set marking which resumes gain interviews for particular job descriptions, a heuristic is used to generate the label, which is the similarity of the job title as classified by a pretrained RoBERTa model.

* **Target:**
  1. The goal of this notebook is to create excels with the format of:



* Notes
  + \* fields will be fed into the final RoBERTa model for similarity training
  + \*\* This is not the content of a full resume, but the job description field of a particular job held by the resume's author currently or in the past - which is considered to have the highest chance to be worth of an interview for a job with similar title.
* **Technology used:**
  1. The model is trained by the GLUE task Semantic Textual Similarity Benchmark (STS-B), one of the benchmarks used to evaluate the original BERT model. While it is not specific to our task it performs reasonably well to label similarities between job titles.
  2. This model is of a class from the simpletransformers library, which is a wrapper of the huggingface transformers library.
     + simpletransformers reply on the latest huggingface transformers version (3.0.2) which is much newer than what is needed by spaCy in the previous notebook. A pip command is put here to upgrade that back
     + **Please note** that in the GitHub repository the file from which this model will be loaded is too big for GitHub and is cut to 8 pieces by zip, it will need to be recombined by zip to work.



* **Build the dataset with Job Titles pairs labeled with model:**
  + After loading the trained model, we load and process the 2 datasets again, then we put it into the trained model to get the predicted score, using the **Job Title** field as heuristic.
    - *Resume job titles:* In particular, as the Avani Resume dataset has fields like work\_experiences which are fields extracted from the original resumes and stored as dictionary, a procedure extracts the job title and job description from the dictionary and store in a seperate dataframe called wdescr\_df, this dataframe has over 35k records.
      * As the extracted individual work experiences has its own job title which may or may not align to the resume’s job title, the resultant job title in wdescr\_df will follow the content in the field, instead of avani\_resume.Resume\_title
      * As we mentioned, we target to train the model using IT related resumes – but we don’t have the flag like in the Armenian Job Descriptions to screen with, so here we manually add a flag for IT related titles using human knowledge.
      * Null fields or those with work experience < 20 tokens are removed to improve results
    - Then we join the job titles from both sides and calculate the predicted score. This joined set is huge with lots of low scores. To enable the training, we impose screening criteria:
      * 10% records with top scores are all selected
      * Then randomly select another 10% of records from the rest of the records – this will ensure range continuity which didn’t yield good results in previous trials.
      * The resultant table is **{Job Title from Jobs, Work Exp Title from Resumes, Labelled Score}**
* **Building the final dataset, with Job Description pairs labelled for training**
  + With those title pairs, we look up the original datasets to replace Titles with Job Descriptions
    - In other words, we will have a table of **{Job Desc from Jobs, Work Exp Desc from Resumes, Labelled Score}**
  + Before we proceed, there are lots of irrelevant info in the Job Descriptions, like application procedure, company descriptions that are not relevant to candidate selections. A procedure is run to screen only info in the sections of Job Description, Job Responsibilities and Qualifications
* **Split the dataset into training, validation and testing sets**
  + In our example, the trimmed down dataset has around 4k rows for training, and 2k rows for validation and testing. These are saved into excels.

## 3. Use Cosine Similarity to Estimate Interview Worthiness of Resumes to Jobs

We often wonder why we need to use the sentence pair classification model when we can embed sentences using the BERT models and find document similarity with cosine similarity. However, note that we are currently asking the question on whether a resume is suitable for interview for a particular job description. This may or may not mean the two passages are similar in context. Afterall, they are written for different objectives.

In our particular training set, the labels are similarities of job titles which may or may not correlate to how the job descriptions are different from each other. We have already seen from the data exploration that jobs and work experience descriptions are generally distinct in the vector space.

Due to the points above, simple job-resume cosine similarity makes a poor predictor on whether a resume is suitable for interview, and this notebook shows exactly that.

* Procedure
  1. **Embed Job Descriptions Pairs with RoBERTa model and compute cosine similarity**
     + As we are going to train a sentence classification model using RoBERTa on simpletransformers, to enable comparability, the STS-B RoBERTa pretrained model is again used to embed instead of spaCy. It also supports CUDA 10.1 that works with the GPU on Colab, Kaggle and my machine.
  2. **Compare with labelled score and calculate mean absolute error**
     + The mean absolute error is 2.12, meaning on average the scores have a 2.12 deviation out of a scale of 5. It is simply unacceptable.
     + In fact, in another trial not shown here, the final RoBERTa model we trained gives an error of 1.58, when cosine similarity is used to compare. While much better than the STS-B model, it is still not acceptable.

## 4. Similarity Classification Using Sentence Pair Comparison with RoBERTa

We have generated the labelled dataset from a previous notebook and are split into training set and the validation set and put into variables train\_df (~ 4k rows) and eval\_df (~2k rows) respectively.

simpletransformer is a wrapper to Huggingface's transformer library which helps to standardize many of the training tasks and has a nice integration with Wegihts and Bias (WandB) visualization.

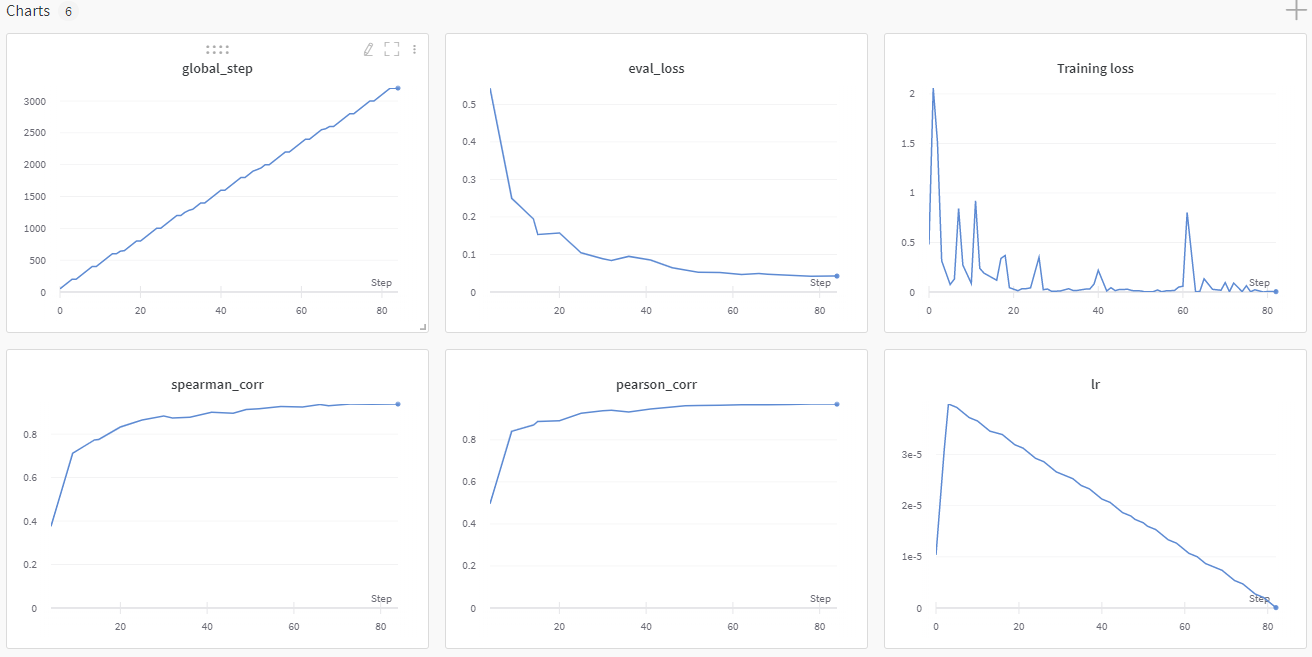
The task of finding the candidate from the resume is essentially the same as the GLUE task Semantic Textual Similarity Benchmark (STS-B). In here I am using the ClassificationModel with regression=True which is suitable for providing a single value, in this case the similarity of two sentences.

Instead of the original BERT for the pretrained model, I chose RoBERTa which uses the same engine but a different training method and more training time. It performs reasonably well even with a trimmed-down training/eval set to allow this runnable for the project submission.

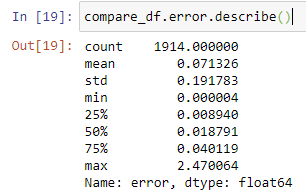
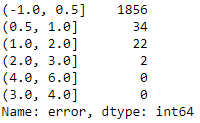
Comparatively, the original BERT base pretrained model (bert-base-cased) has a limited pretrained vocab and left lots of tokens marked as unknown [UNK]. Instead of using the BERT large I have picked the RoBERTa base (roberta-base) instead, which has a vocab of over 50k and covered most of the words used.

Here are the considerations when we select the parameters:

1. class ClassificationModel is selected with roberta-base as the pretrained model.
2. It will take in evaluation data to determine / save the best model under outputs/best\_model, happens every 200 steps.
3. The model is trained on GPU. If run on Kaggle GPUs, it can achieve train\_batch\_size of 16. My home PC can only use 6.
4. The max\_seq\_length determines the max length of the sentence pairs, based on previous statistics calculated 512 is enough.
5. The epoch is set to 5 which is often enough for larger datasets. It will also use the evaluation data performance to determine whether it can be stopped if there are no additional drops in the loss function. Given the small value of epochs this is also set small to avoid it from stopping prematurely.
6. The learning rate not shown here takes the default of 4e-5. However, the model will also automatically adjust lr by default.
7. WandB is configured to plot the progress of the training (shown below).

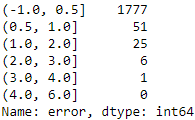
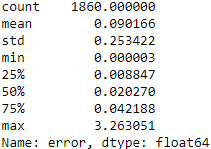


Training Results:

* The training completed all 5 epochs without hitting our intentionally low early stopping criteria in around 33 mins.
* The final loss on the eval set is 0.042 with both correlations higher than 94% which is quite good given the small dataset.
  + 'pearson\_corr': 0.9704873199597329
  + 'spearman\_corr': 0.9384291351353189
  + 'eval\_loss': 0.04246755579870296
* The eval loss is essentially the mean\_squared\_error, and we can also calculate the mean\_absolute error
  + 
  + In this case we can see that for all of the eval set on average we have an error of 0.071 on a 5-point scale, which is not bad.
* Among the 1914 records in the eval set, only 58 has error > 0.5
  + 

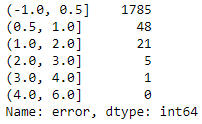
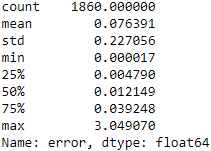
## 5. Model Trial on Testing Data

Here we run use the model to predict for the testing data and examine the performance. We also compare the result with what we ran on Kaggle with larger train\_batch\_size and sample size.

For the testing set with 1860 entries, the mean absolution error is 0.09 which is slightly higher than the eval set but still excellent on a 5-point scale. Only 83 (4.4%) records are classified with > 0.5 error

Below are the results of our better model:

For the testing set with 1860 entries, the mean absolution error is 0.076 which is slightly better than the current model. Only 75 (4.0%) records are classified with > 0.5 error.



# Project Conclusion

From our investigations, it seems that transformers models like RoBERTa would be able to learn the relationship between Job descriptions and Resumes work experiences descriptions.

While our labels are not real labels from the industry, the job titles correlations seem to be a good heuristic for job description correlations from which the model is able to learn.

On the other hand, we can see that simple similarities, even with BERT embeddings, perform far worse, which aligns to what we see in the t-SNE visualizations.

With better labelled data and larger training size and time we would be able to contribute something meaningful to the interview decisions.

# Deployment Strategy

This recommendation engine, after trained, can accept new resume-job pair for classification of worthiness of candidate for interviews. Initially, the training labels are relying on the same or similar job titles which are not really adapted to the particular company. However, as the HR process continues, each new job description will be paired with a list of resumes shortlisted scored according to their reaching the first interview, second interview, with the actual candidate hired to have the highest score (1). This way we can collect a set of good labels for the model to be trained. As the fine-tuning of the BERT pretrained model is relatively fast, it is recommended, once we have collected sufficient samples for training, that older resumes / job descriptions be phased out from the training on a, say, quarterly basis, so that we can divest from older representations, say those created greater than 5 years ago, of the job definitions and keep the latest interpretation of fitness of a candidate’s resume.

# References

Dhainje, A., 2019. *ResumeDataSet.* [Online]   
Available at: https://www.kaggle.com/dhainjeamita/resumedataset

Elmers, D., n.d. *The Job-Search Statistics All Job Seekers Should Know.* [Online]   
Available at: https://www.topresume.com/career-advice/7-top-job-search-statistics

Hab, M., 2017. *Online Job Postings - Dataset of 19,000 online job posts from 2004 to 2015.* [Online]   
Available at: https://www.kaggle.com/madhab/jobposts

Huggingface, n.d. *Transformers - State-of-the-art Natural Language Processing for PyTorch and TensorFlow 2.0.* [Online]   
Available at: https://github.com/huggingface/transformers

JobsPikr, 2019. *Australian job listings data from Gumtree - More than 8000 job listings extracted in 2018 from Gumtree.* [Online]   
Available at: https://www.kaggle.com/JobsPikrHQ/australian-job-listings-data-from-gumtree

Liu, Y. et al., 2019. *RoBERTa: A Robustly Optimized BERT Pretraining Approach.* [Online]   
Available at: https://arxiv.org/abs/1907.11692

Palan, M., 2019. *resumes.* [Online]   
Available at: https://www.kaggle.com/maitrip/resumes

PromptCloud, 2020. *Australian job listings data from Seek job board - 20k+ job listings extracted in 2018.* [Online]   
Available at: https://www.kaggle.com/PromptCloudHQ/australian-job-listings-data-from-seek-job-board

Rajapakse, T., 2020. *Simple Transformers.* [Online]   
Available at: https://github.com/ThilinaRajapakse/simpletransformers

Raman, V., 2019. *How I used NLP (Spacy) to screen Data Science Resume.* [Online]   
Available at: https://towardsdatascience.com/do-the-keywords-in-your-resume-aptly-represent-what-type-of-data-scientist-you-are-59134105ba0d

Siddhapura, A., 2020. *Resume Dataset - Annotated resume data to create text corpus for NER system.* [Online]   
Available at: https://www.kaggle.com/avanisiddhapura27/resume-dataset

Wu, O. T., 2019. *Resume Rater.* [Online]   
Available at: https://github.com/ongteckwu/Resume-Rater