# Automatic Candidate Recommendation and Statistical Analysis of Resumes

***Abstract*— Today's industry is clogged with thousands of unfit resumes and candidates. The human resources departments are the busiest when job postings are open to applicants. A solution to help bring in better candidates and clear the clutter of unfit resumes is very well needed. Also, students who are near graduation can benefit from knowing their chances of getting into certain companies and getting their desired position. The project discussed in this paper can help job seekers determine their eligibility for a certain position and receive feedback for improvement. We describe what we look for in a resume and how we utilize the word2vec and bag of words approaches for word vectorization. After resume preprocessing and vectorization we apply a supervised learning algorithm in order to classify a given resume against the dataset of available resumes to determine what tier of companies and jobs does the applicant fall in. In today's world, HR is considered a busy part of a company. Despite their diligence, HR spends huge amount of time reviewing resumes and filtering through candidates. This project, if fulfilled with its final goals, can simplify the process and divert companies' precious time for better talent acquisition and improvement.**

## I. INTRODUCTION

Our main motivation behind this project was our own experience. As university students, there will be a time in our life in which we will have to apply for industry jobs. The process can be overwhelming for some, especially for international students. The idea of using **A**utomatic **C**andidate **R**ecommendation and **S**tatistical **A**nalysis of **R**esumes, is to provide future employees the chance to view their chances and fit to positions they are applying to. The system will extract relevant information from resumes such as skills and experience of the applicant. Next, it will compare their resumes against a database of resumes and determine their class `Tier I` or `Tier II`. These tiers classify companies and jobs. `Tier I` being the top tech companies and related job positions and `Tier II` companies based in Houston, Texas and their related job positions.

We believe this tool, ACRSAR, can help many students getting ready to join the industry, receive a good insight on their resumes and skill sets. This project not only benefits the incoming workforce, but the human capital management sector. This way, they will avoid filtering through thousands of incompatible resumes and interviewing wrong candidates.

In this paper we will discuss related work in section 2 that have shown results in using tools such as word2vec and bag of words for text parsing. In section 3 we will discuss methods that we have used and algorithms that seemed reasonable for this project. Section 4 illustrates our experiments and shares the results gained from implementing the methods of section 3. Final section, 5, we share our opinions on possible research directions and ideas that can improve this project and deliver better quality in talent acquisition and resume classification.

## II. RELATED WORK

### A. Semantic Similarity Strategies for Job Title Classification

Yun Zhu et al. [1] have proposed the usage of word2vec (W2v) developed by Mikolov et al. that uses shallow neural networks to build vectors of words and phrases. These vectors produce good word representations that can be used to formulate relationships. For example, $V_{KING} - V_{MAN} = V_{QUEEN} - V_{WOMAN}$. They have developed a system called Carotene that helps in automatic job classification which has a taxonomy discovery component that leverages clustering techniques to discover job titles from data sets to create a custom job title taxonomy, and a two stage coarse and fine level classifier that uses an SVM-KNN cascade to classify input text to the most appropriate job titles in the custom taxonomy. Carotene can be used along side with ACRSAR to provide job title classification that can correctly identify what position/job is the candidate more suitable for according to their resume classification.

## III. METHODS

### A. General Outline

ACRSAR process is three-fold. First step is to convert resumes from PDF to TXT and then perform some natural language processing using a tool such as the `nltk` library. Using a `snowballstemmer` [8], it handles all strings of text with the `english` language parameter.

The second step is to take all documents and use most common words among all resumes for the Bag-of-Words approach. We create vectors by counting them for every resume. Then get the term frequency for vectors using `tfidf`. We classify with term frequency vectors that we created. Also, vector representation helps us to visualize the dataset. With the numerical data, we can apply a dimensionality reduction technique, PCA (Principal Component Analysis).

The third step would involve the usage of a supervised learning algorithm that we chose, logistic regression, to classify resumes accordingly to their quality and place them in tiers. For sake of simplicity we have created two tiers, I and II. Tier I represents top tech companies in the USA, such as Google, Facebook, IBM, etc. Tier II represents top companies in Houston, Texas area (domain knowledge).

The final step involved outputting the result for the user which indicated what Tier they have been placed in and what

are their chances of receiving an offer from those companies based on their classified resume skills and experience.

### B. Libraries Utilized

We mentioned the usage of BOW (bag-of-words) module to vector all common words across resumes. Another approach that was discussed here[1], was the usage of word2vec[2]. A semantically rich word vector representation. But these modules alone do not perform well as you will see in section 4, thus reinforcing them with `tfidf` technique gives the best output. We use logistic regression function as our machine learning algorithm. It uses the sigmoid function also known as the logistic function which returns the probability of being in a certain class. The predictions are drawn according to the assigned probabilities.

## IV. EXPERIMENTS

In this section we will illustrate our conducted experiments with the methods mentioned in the former section. We will begin with a disclaimer stating the experiments are biased due to lack of data (e.g. privacy concerns) thus resulting in high classification accuracy.

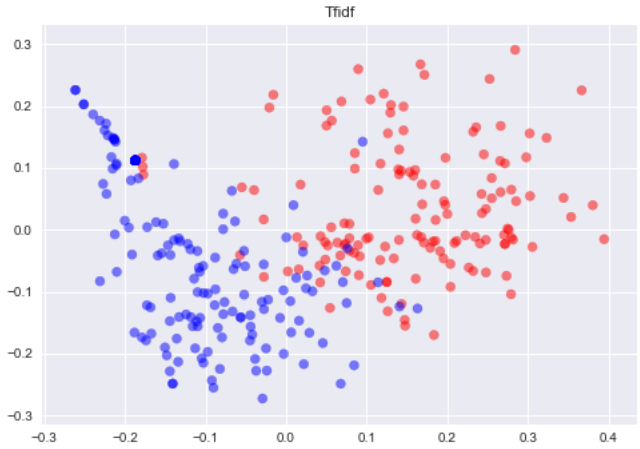


Fig. 1. Word term frequency derived from tfidf showing two classes of resumes, Tier I and Tier II.

In the preprocessing step we took our resume data and applied a tokenizer. Tokenization is a technique to separate words. We use tokenization instead of splitting every space and every punctuation, because tokenization is more complex and complements regular expressions. As an example to compare tokenization to simple splitting used in Python, the word 'C++' would be correctly parsed as 'C++' while the splitting would remove '++' and store only 'C'. After word tokenization, we used a stemming algorithm, `snowballstemmer`. Stemming is the process of reducing inflected (or sometimes derived) words to their word stem, base or root form. We also apply the 'english' parameter to specify certain grammar rules. And at last we remove stop words. Since stop words such as the, I, and etc... do not contain any meaningful information, we can get rid of them.

Next, we setup the experiment for supervised learning regression problem. We classify resumes based on the collected `tfidf` term frequency vectors. As seen in figure 1, the regression can be done using a linear separator such as logistic regression. The output that we expect is to be a confidence level of how good the resume is for a job position/company.

We used two methods as described in the section 3, word2vec and bag-of-words to vectorize our words. Figure 2 illustrates the accuracy of each method. They might seem equal, do not forget our data was biased and thus resulting in high accuracy. But, if we examine [1], We see their experiments have demonstrated that word2vec to be most efficient out of all methods available to count vectorized words.

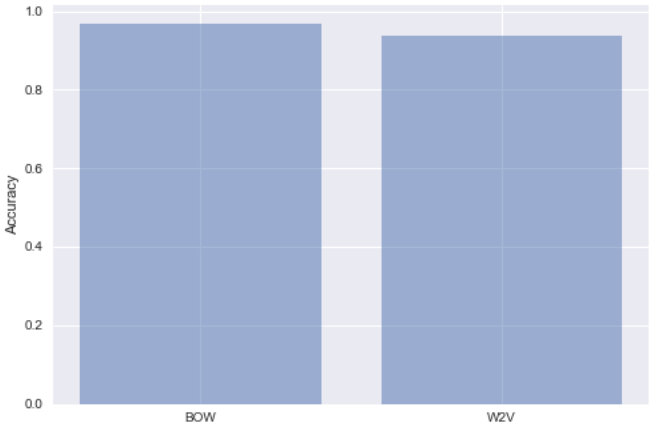


Fig. 2. Accuracy outputted from BOW and W2V methods.

The outcome is interpreted as what chance of getting the job or getting into a certain company does the applicant have. The output might be `Tier I, Google, Quora` due to applicants Python experience and previous work experience similar to those companies or their job posting requirements.

## V. CONCLUSION

Our described ideas and project in this paper are incomplete according to our final goal. Which is, creating a system that can correctly classify a given resume and assign a set of companies that person has the potential to work at, return a set of job titles most related to the applicant's skills and experience, and output a statistical analysis of the applicant's resume showing where they can improve in order to maximize their employment to relevant positions. Our experiment results are far from final and we are working on more detailed and complex experiments to update the results and demonstrate that such a system is in fact not only feasible but necessary for future human capital management.

## VI. RESEARCH DIRECTIONS & FUTURE WORK

As mentioned in the former section, our project is not complete and we are currently working on improving it. Important direction that needs to be researched is understanding what exactly to look for in a resume that every

candidate has in common. Unfortunately, due to no single resume writing style, this is a hard thing to predict. As part of our project upgrade we are looking forward to provide statistical analysis for the applicant on their resume. This incorporates determining what skills they need improving to do on, or how much needs to be improved.

There have been efforts made in creating such similar systems that we speak of but with obstacles such as user privacy, lack of data, and no single universal style of resumes, such projects might seem unfeasible but with enough dedication and effort we believe we are on the right track on providing the industry with a tool to alleviate HRs time wasted on resume filtering and futile candidate screening.

## ACKNOWLEDGMENT

We would like to thank Dr. Ozgur Ozturk and his coworkers. [1] for their motivation and assistance in providing information on what human capital management is looking for in candidates.

## REFERENCES

[1] Yun Zhu, Faizan Javed, Ozgur Ozturk, Semantic Similarity Strategies for Job Title Classification, arXiv preprint arXiv:1609.06268., 2016.

[2] Goldberg, Yoav and Levy, Omer, word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method. arXiv preprint arXiv:1402.3722, 2014.

[3] E. B. Albitar, S. and S. Fournier, Semantic enrichments in text supervised classification: application to medical domain, Florida Artificial Intelligence Research Society Conference, 2014.

[4] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient Estimation of Word Representations in Vector Space. arXiv preprint arXiv:1301.3781v3 2013

[5] T. Mikolov, I. Sutskever, K. Chen, G. Corrodo, J. Dean, Distributed Representations of Words and Phrases and their Compositionality, arXiv preprint arXiv:1310.4546v1 2013

[6] Edward Loper, Steven Bird, NLTK: The Natural Language Toolkit, arXiv preprint arXiv:cs/0205028v1 2002

[7] J.B. Lovins, Development of a Stemming Algorithm. Mechanical Translation and computation Linguistics. 11 (1) March 1968 pp 23-31.

[8] M.F. Porter, Snowball: A language for stemming algorithms, http://snowball.tartarus.org/texts/introduction.html, 2001