JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Resume Scorer-Resume Filtering using Machine Learning

B.Jagadeesh, A.D.N.Kishore, B.Vishnu Teja, M.Sardeesh

Under the guidance of Mrs.G.Sujatha (Associate Professor)

Department of Computer Science and Engineering

Vignan's Institute of Information Technology, Jawaharlal Nehru Technological University - Gurajada Vizianagaram (JNTU-GV), Vizianagaram.

Abstract

In this fast paced world where people rush their daily lives from their home to work place or try their best to get into their dream job. Thousands of people might apply for the same job role in a single day. It is not feasible for the recruiting team to go through all the applicant's resumes manually. Here, our resume scorer comes into action. Given a job description and the resume, the resume scorer uses machine learning algorithms to match the profile with job description and give it a score. This score helps filter out all the profiles which are not appropriate for the job and thus makes the overall process easier. In the perspective of an applicant, resume scorer helps him/her to check their resume score for the job they are applying to see where they stand, or even make any necessary changes to their resume.

Key Words: Extraction, word2vec, spaCy, scoring

1. Introduction

Resume Scorer automates the process of scoring the resumes based on the job description, can help Hr teams in many organizations while recruitment of candidates for several job positions. A central challenge is scoring the resumes based on the job description, which is to predict which candidate matches the job description.

We introduce an accurate method to predict suitable resumes based on the different skills like experience, education skills, technical skills etc., using technical methods that automates the process of manually checking thousands of resumes.

This is provided with technologies such as Machine Learning. Machine Learning algorithm is used for predicting scores i.e. to show the best resumes for specific job descriptions. In this system, we have used algorithms like Word2Vec, spacy library.

The outputs of the system will be scores of several resumes for a job description.

2. Literature Study

2.1 Numpy

The term **NumPy** refers to a software library in the Python programming language, which is commonly known as 'Numerical Python'. It is the core library for scientific computing, which contains a powerful n-dimensional array object, provides tools for integrating C, C++, etc. In addition to its general usefulness, NumPy is also beneficial for tasks related to linear algebra and generating random numbers, among other capabilities. We can simply use the statement “import numpy as np” to use numpy.

2.2 Pandas

The primary purpose of **Pandas** is to facilitate the manipulation, analysis, and cleaning of data. Python pandas are well suited for different kinds of data, such as:

- Tabular data with heterogeneously-typed columns
- Ordered and unordered time series data
- Arbitrary matrix data with row & column labels
- Unlabelled data
- Any other type of statistical or observational data sets

We can simply use the statement “import pandas as pd” to use pandas.

2.3 Matplotlib

Matplotlib is a powerful visualization library for creating 2D plots of arrays in Python. Developed by John Hunter in 2002, this multi-platform library is designed to integrate with the larger SciPy ecosystem and is built on NumPy arrays. One of the primary advantages of using visualization techniques is that it enables us to quickly comprehend large datasets by presenting them in a visually accessible format. Matplotlib provides a wide range of plotting options, including line, bar, scatter, and histogram plots.

2.4 Seaborn

Seaborn is a Python library for data visualization that is based on matplotlib. It offers a user-friendly interface for creating visually appealing and informative statistical graphics..

2.5 Scikit Learn

Scikit-learn is considered one of the most valuable machine learning libraries in Python. Built on NumPy, SciPy, and matplotlib, this library provides a comprehensive range of powerful tools for machine learning and statistical modeling, including regression, classification, clustering, and dimensionality reduction. It's worth noting that while scikit-learn is an excellent tool for building models, it's not intended for data manipulation, summarization, or reading. For those tasks, it's recommended to use more suitable libraries such as NumPy or Pandas.

2.6 Google Colaboratory

Colaboratory is a Jupyter notebook environment that is completely free and runs entirely in the cloud. You don't need to set up anything on your local machine, and you can write and execute code directly in your browser. In addition to writing and executing code, Colaboratory enables you to save and share your analyses, and provides access to powerful computing resources for free.

2.7 Anaconda

Anaconda is an open-source and free distribution of both the Python and R programming languages that is specifically designed for scientific computing. Its main goal is to make package management and deployment easier for users. Anaconda uses the conda package management system to manage different package versions, providing users with an efficient and streamlined way to manage their libraries and dependencies.

3. Exisiting Model

There are many existing systems which can give your resume a score using which you can analyse the faults and modify your resume for a better score. The recruitment process in today's world has witnessed a major change with the evolution of technologies like the Internet. The proposed solutions use various approaches with the aim of achieving automated screening of candidates. The work presented as EXPERT (2013) proposed the use of ontology mapping for screening candidates for the given job description. It included three phases of operation which were the creation of candidate ontology, construction of job criteria ontology document and hen finally mapping of both of these to evaluate which candidates are eligible for the job. In 2012, an automated job screening system was proposed . It uses Support Vector Regression to create a list of ranked candidates for the given job and discusses different machine learning algorithms. Another work presented (Weathington and Bechtel, 2012) that described how social media (e.g. LinkedIn, Facebook, etc.) information of the applicants can be used for recruitment decisions. In another approach, the work (Laumer, S. and Eckhardt, A., 2009) described a collaborative filtering based system to recommend applicants that best fit a job.

4. Proposed Work

Our work takes a different approach as it focuses mainly on the content of the resumes where we perform the extraction of skills and related parameters to match candidates with the job descriptions.

Upon providing multiple resumes and a job description, our model takes the key terms from the resumes and matches them with keywords from the job description and as a result gives us the scores of these resumes (highest to least).

5. Working

The dataset we used to train our model is from stackexchange dataset which we preprocessed by removing unncessary symbols and stop words. Followed by extraction,in which the dataset had some collection of words that are usually the heading in the resumes. For example 'education', 'academic', 'school', 'study', etc will mark the start of the education section.Iterated over all lines of all resumes, one by one.Next, categorized each line into one of the four sections. This is done by calculating its similarity to the existing words. If the similarity is higher than the threshold, we update the section and mark that point, on the other hand, if the similarity is below the threshold, we continue with the previous section.This enables us to separate the sections with good enough accuracy.Finally, wrote each section of a resume in a .csv file after removing the stop words and doing lemmatization.

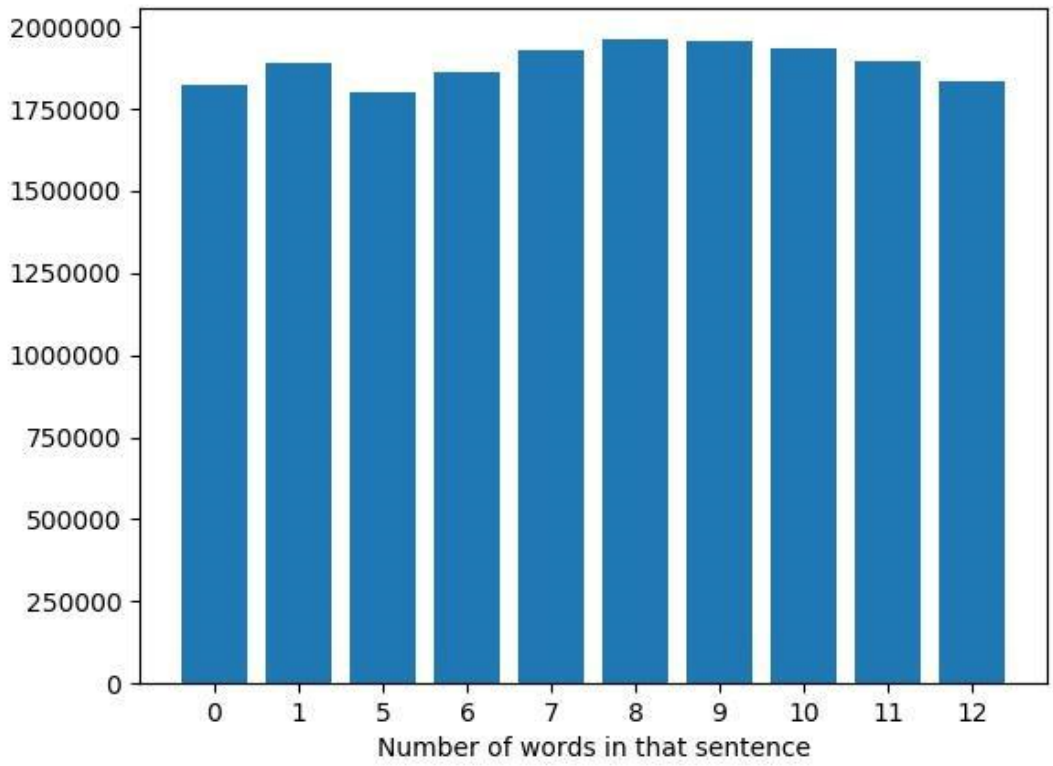

Figure: Number of words in that sentence vs number of sentences

A bar plot is a type of graph that shows the central tendency of a numeric variable using the height of each rectangle, and provides information about the uncertainty around that estimate using error bars. Bar plots typically include 0 on the quantitative axis, and are a good choice when 0 is a meaningful value for the variable being studied, and comparisons need to be made against it.

If a dataset has no meaningful value for zero, a point plot could be a better option since it can concentrate on the differences among one or more categorical variables without being constrained by the need to show zero on the quantitative axis. However, it's important to note that bar plots only show the mean (or other estimator) value, which may not be the most informative for certain datasets. In such cases, alternative approaches like box or violin plots may be more appropriate, as they provide a better representation of the distribution of values at each level of the categorical variable.

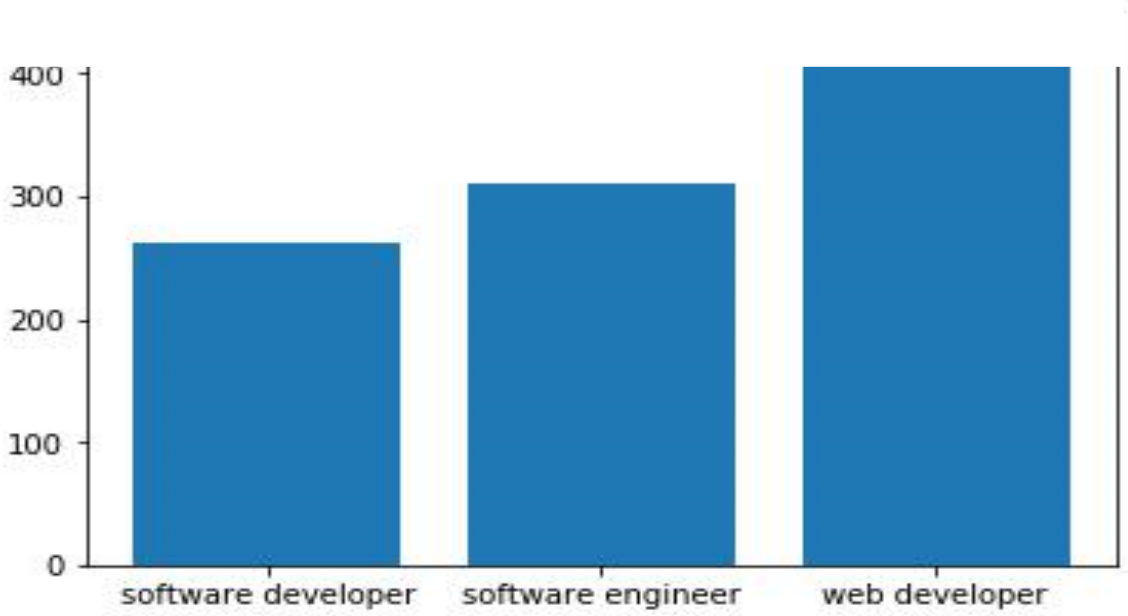

Figure: Job Role vs number of words in description

5.1. Vectorization

A group of objects known as vectors combine to form the geometric structure known as vector space. These can be multiplied ("scaled") by certain numbers, which in this context are referred to as scalars, then added together. It is an algebraic model for text information representation used in text mining, natural language processing, and information retrieval. Vectorization is the process of representing documents in a vector space model. It involves converting a piece of writing into a numerical vector. The fact that most machine learning models require numerical vector input rather than strings makes vectorization significant. The process of assigning a distinct integer to each word in a text is known as vectorization. Every word can fit into a different slot in an expansive array. The value at that index represents the word's frequency. Our array size is typically smaller than the corpus vocabulary. Therefore, a vectorization technique should be in place to take this into consideration.

5.1.1. word2vec

Word2vec is a common method for creating word embedding through a neural network with two layers. It takes a text corpus as input and generates a set of vectors as output. The word embedding produced by word2vec allows computers to process natural language, and mathematical operations can be performed on words to identify similarities. If the set of word vectors is trained effectively, similar words will be located near each other in the vector space. For example, words such as "women," "men," and "human" might be grouped in one area, while "yellow," "red," and "blue" might be clustered together in another.

5.2. spaCy

spaCy is a Python library that is designed to handle and comprehend large quantities of text. It is useful for developing systems that can extract information from text or comprehend natural language. Additionally, it can be used to prepare text for further processing. spaCy supports multiple languages and includes support for word vectors. However, it is known to consume a lot of memory resources.

5.3. TF-IDF

"Term Frequency - Inverse Document Frequency" is the abbreviation for this.When using text mining techniques, the TF-IDF weight is frequently utilised. For document search and information retrieval, TF-IDF was developed. This weight is a metric that quantifies the significance of a term in relation to a document in a corpus or collection. The number of documents that contain a term offsets the importance, which rises proportionately to its frequency in the document. Therefore, even though they may appear frequently, terms like this, and, what, whose, is, the, if, etc. rank low because they aren't really

important to that document. Equation (1) below demonstrates how to multiply two separate metrics to determine the TF-IDF value for a phrase in a document.

□□□□□□□□□□□□ □□□□□□□□□□ □□□□□□□□□□□

$$TF - IDF\ (t, d) = TF\ (t, d) * IDF\ (t, d) \quad (1)$$

5.3.1. Term Frequency: It counts the number of times a term appears in each piece of text in the corpus. You need to moderate or normalise this frequency because a word may appear more frequently in lengthy papers than shorter ones. By dividing the number of times a term appears in a document by the total number of terms in that document, a normalised term frequency is determined.Mathematically, we can write it as shown below in equation (2).

$$TF\ (t, d) = \frac{freq\ (t, d)}{\sum_i^n freq\ (t_i, d)} \quad (2)$$

□ □□ □□□□□□□□□□

Here, freq (t, d) is the count of the instances of the term t in document d,

TF (t, d) refers to the ratio of the frequency of term t in document d to the total number of distinct terms present in document d.

5.3.2. Inverse Document Frequency: It gauges a word's significance across the corpus of documents. In other words, this measure aids in determining how frequently or infrequently a word appears in the corpus. It scales up the uncommon terms while weighing down the more frequent terms. The IDF value for the rare terms is high, whereas it is near to zero for the terms that appear more frequently in the set of documents. It is calculated by dividing the total number of documents by the number of documents that contain a term and then

calculating the logarithm. Mathematically, we can write it as shown below in equation (3)

$$IDF(t) = log\left(\frac{N}{count(t)}\right) \quad (3)$$

Here, N represents the number of distinct documents in the corpus and count (t) refers to the number of documents in the corpus with the term ‘t’. The product of these two metrics results in a TF -IDF score of a word in a document. More relevance of a word in a document is reflected by its high TF-IDF score. In our system, we created a vector space model of the resumes and the job description paper. To do this, a dictionary of terms found in the documents is created, and then each term is changed into a dimension in the vector space. Using the CountVectorizer and TfidfTransformer Python libraries, we next generated the TF-

IDF matrix for the CVs and the job query. The similarity score between the resumes and the job description must be determined in the following stage.

5.4. Cosine similarity

A similarity measure is a metric that establishes the degree of similarity between two items. Regardless of the size of the two papers, cosine similarity is a method to determine how similar they are. When plotted on an N-dimensional space, where each dimension represents an aspect of the object, it displays the orientation of the documents. Being symmetrical means that the results of calculating the similarity of item X to item Y are the same as those of calculating the similarity of item Y to item X. Mathematically, we can represent it as shown below in equation (4).

$$cos(\theta) = \frac{\vec{a}.\vec{b}}{\left\|\vec{a}\right\|\left\|\vec{b}\right\|} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}} \quad (4)$$

Here, $\vec{a}.\vec{b} = \sum_{1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \ldots + a_n b_n$ is the dot product of the two vectors. With the help of this formula, we calculate the cosine similarity . The resume documents can then be ranked according to a given vector of search terms. Conclusions generated by cosine similarity are typically less dependable as it predominantly takes into account features related to the words in the text. Incorporating semantic information can enhance the efficacy of similarity metrics.

6. Result

A folder containing several candidates resumes of .pdf, .docx extensions taken as test data. These resumes of different candidates converted to text. PDFMiner is a specialized tool designed for extracting text information from PDF documents, which sets it apart from other PDF tools. It provides precise details on the text's location within a page, along with information about fonts and lines. In addition, it offers a PDF converter that can convert PDF files into other text formats like HTML. PDFMiner also features an adaptable PDF parser that can be utilized for purposes other than text analysis.

Step1:.For a given Job Description, removed all the stop words and do lemmatization, to get a selected few keywords.
Step2: For each keyword found, found 5 similar words and their corresponding similarity.
Step3 : Now, found **tf-idf** for each word, that was got in step 2.

Step4 : Got similarity values for all the resumes.
Step5 : The score of the CV is the sum of **tf-idf * similarity** for all words that were generated in step 2.
Step6 : Finally, sorting all the resumes based on their score and storing in output.txt as a result.

7. Conclusion

To conclude, our proposed model for filtering the resumes based on a given job description offers a simple and efficient solution for companies or organizations to make the work of shortlisting applicants for role specific jobs easier. All we have to do is give the job description and the set of resumes as input and the resume scorer does all the work for us and give the resumes in descending order of their score as output.

8. Future Work

The next phase of development for this system may include mining the candidates' social networking data (for instance, their Facebook, LinkedIn, and GitHub profiles) and utilising this social behaviour data in addition to the content of their resumes to produce even better suggestions. Another option is to employ a collaborative filtering-based method, which can match the current applicant with a position depending on how well other candidates who are similar to them are rated for it.

Acknowlegement

This research was supported by Vignan's Institute of Information Technology, Visakhapatnam.We would like to express our deep gratitude to Mrs. G. Sujatha, our guide, for her unwavering support and direction. We also appreciate the suggestions made by Dr. B. Dinesh Reddy, Mr. Ch. Sekhar, and other professors, who significantly enhanced the manuscripts. We also like to express our gratitude to all anonymous reviewers for their insightful comments.

References

[1] Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E.D., Gutierrez, J.B. and Kochut, K., 2017. A brief survey of text mining: Classification, clustering and extraction techniques. arXiv preprint arXiv:1707.02919.

[2] Huang, A., 2008, April. Similarity measures for text document clustering. In Proceedings of the sixth new zealand computer science research

student conference (NZCSRSC2008), Christchurch, New Zealand, 4, 9-56.

[3] Singh, A., Rose, C., Visweswariah, K., Chenthamarakshan, V. and Kambhatla, N., 2010, October. PROSPECT: a system for screening candidates for recruitment. In Proceedings of the 19th ACM international conference on Information and knowledge management, 659-668.

[4] Al-Otaibi, S.T., Ykhlef, M., 2012. A survey of job recommender systems. International Journal of Physical Sciences 7, 5127–5142.

[5] Breaugh, J.A., 2009. The use of biodata for employee selection: Past research and future directions. Human Resource Management Review 19, 219–231

[6] Mooney, R.J., Roy, L., 2000. Content-based book recommending using learning for text categorization, in: Proceedings of the fifth ACM conference on Digital libraries, ACM. pp. 195–204.

[7] spaCy Documentation**:** https://spacy.io/

[8] spaCy GitHub Issue Page: https://github.com/explosion/spaCy/issues

[9] Gensim Word2Vec Documentation: https://radimrehurek.com/gensim/models/word2vec.html

[10] Google Word2Vec: https://code.google.com/archive/p/word2vec