Automating Resume Classification: Leveraging NLP and AI for Efficient Candidate Screening

Akshata Upadhye
Cincinnati, OH, USA

ABSTRACT
With the growth in the number of applications received for a job posting, it has become a time-consuming process for the recruiters and the HR teams to review every individual resume. On the other hand, technological advancements in the field of natural language processing and AI can help build efficient automated tools to process and classify the resumes into relevant categories to help speed up the initial review process. In this paper various text processing techniques that will help generate representations of resumes suitable to be used by various machine learning algorithms and various types of classifiers are explored. A Word2vec word embedding model is trained and used to generate resume level embeddings. Finally, the performance of various classifiers is evaluated, and the best performing classifier is used for classifying resumes. The effectiveness of the resume classifier developed using the proposed approach is demonstrated using several metrics such as precision, recall and F1 score.

Keywords
Candidate Screening, Resume Screening, Word Embeddings, Classifiers, Machine Learning, Natural Language Processing.

1. INTRODUCTION
Resume classification is important for an efficient hiring process in the current digital age. Companies that post a job opening receive a huge number of resumes. Reviewing these resumes manually is a very tedious and resource intensive process. Therefore, by automating the resume classification the initial screening process can be finished quickly and the recruiters can focus more on the candidates suitable for the position. Additionally, this can be very beneficial for the staffing and HR services industry since it helps speed up the hiring process. As the business grows a resume classification tool will be helpful to scale up the hiring process while maintaining the standardization in the resume screening process.

An automated resume classification tool that is designed to solely focus on skills and experience can also help reduce conscious or unconscious human bias and can help make the decisions objectively. Through automation of the resume screening process, the responses can be sent out to the candidates in a timely manner, thus enhancing the candidate's experience. Additionally classifying resumes into different categories can help the organizations to analyze trends within their talent database and make data-driven decisions that benefit the organization. An automated resume classification tool that is designed to solely focus on skills and experience can also help reduce conscious or unconscious human bias and can help make the decisions objectively. Through automation of the resume screening process, the responses can be sent out to the candidates in a timely manner, thus enhancing the candidate's experience. Additionally classifying resumes into different categories can help the organizations to analyze trends within their talent database and make data-driven decisions that benefit the organization.

Therefore, the goal of this research is to design an automated system to interpret, analyze and classify resumes into relevant categories based on the content in the resume using NLP techniques and various machine learning algorithms.

2. RELATED WORK
This section is used to describe the existing techniques used for text representation learning and the existing techniques used for text classification.

In recent years, due to the growth of the internet, the amount of unstructured text data has grown tremendously. Due to this the need to organize text information has grown rapidly which led to the development of machine learning techniques that could achieve great results in natural language processing [1]. The text classification has been researched in a variety of fields such as information retrieval, data mining and machine learning and they have a variety of applications such as document organization, sentiment analysis, etc.[2].

2.1 Text Representation
One of the preliminary steps in text classification is cleaning the text and then comes the most important stage which is generating text representation suitable to be used by classification algorithms. Many text representation techniques have been developed which can be used to represent text documents such as resumes in a meaningful representation suitable for machine learning.

2.1.1 Ontology based representation techniques
Ontology based text representations aim to capture the semantic relationships, concepts and properties and can be used in critical domain specific scenarios where the tolerance of error is low [3].

2.1.2 Bag-of-Words based representation techniques
Some of the effective techniques to represent a text as a fixed length vector are based on the Bag-of-Words (BoW) model. Examples of BoW based text representation models are One Hot Encoding, Term Frequency (TF), Term Frequency-Inverse Document Frequency (TFIDF), etc. These methods worked well to represent text but at the same time they suffered from the curse of dimensionality. Therefore, it became impractical to use them with larger datasets [4].

2.1.3 Topic model-based representation techniques
Topic modeling is a statistical technique used to find latent topics within text data. These hidden topics within a text document can be used as a low rank approximation method to represent a document-term matrix [5].

2.1.4 Embedding based representation techniques

The advancement in the field of deep learning has given rise to the research and development of deep neural networks for text data. These methods are helpful in generating low-dimensional representations of text while preserving the semantic relationship within the text data and are popular because they help overcome the curse of dimensionality [6].

2.2 Text Classification

Once the text is preprocessed and transformed into meaningful representations such as document-term matrix or the document-embedding matrix, these features can be used with various machine learning algorithms. The text features can be used by tree-based classifiers, rule-based classifiers, distance-based classifiers, SVM, regression-based classifiers, etc. to categorize the text data [7].

3. BACKGROUND

3.1 Text representation using Word2vec

Word2vec is a neural network-based algorithm used to learn distributed representation of words in the vector space [8]. There are two different frameworks proposed in the original paper:

3.1.1 CBOW model

The Continuous Bag-of-Words Model is used to predict the current word based on the words in its context from the past and the future.

3.1.2 Skip-gram model

The Skip-gram model is used to predict the words in the context given the current word.

The Word2vec model will help create low dimensional word embeddings which can be used to create embedding representation for the entire text which in this case is the resume.

3.2 Text classification

3.2.1 Support Vector Classifier

Support vector machine uses a set of points from the training data also known as the support vectors and they are used in regression, classification, and outlier detection [9]. They are effective in high dimensional spaces.

3.2.2 Decision Tree Classifier

Decision Tree is a supervised learning technique in which the classifier learns to classify the datapoints by learning a set of rules from the training dataset [10]. Decision trees are easy to interpret and visualize.

3.2.3 Random Forest Classifier

Random forest is an ensemble of Decision Trees, and it creates multiple decision trees fitted over different subsets of the training data and uses the average to increase the accuracy of prediction and to control the over fitting [11].

3.2.4 K Nearest Neighbor Classifier

K Nearest Neighbor classifier is a distance-based classifier that finds n nearest neighbors to a datapoint and then uses the vote from the nearest neighbors to classify the datapoint. The KNN classifier performs well when the decision boundaries are irregular.

3.2.5 Naive Bayes Classifier

Naive Bayes Classifier is a supervised learning technique based on Bayes Theorem. These classifiers assume conditional independence among the features given the value of the class and are hence known as Naive Bayes [12].

3.3 Evaluating the classifiers

For evaluating the performance of the classifiers, we will be using the following metrics:

3.3.1 Precision

Precision is computed by taking the ratio of number of true positive instances to the sum of number of true positive instances and number of false positive instances. It gives the percentage of instances correctly classified into that particular class.

3.3.2 Recall

Recall is computed by taking the ratio of number of true positive instances to the sum of number of true positive instances and number of false negative instances. It gives information about the percentage of relevant items retrieved.

3.3.3 F1 score

F1 score is the harmonic mean of precision and recall and in order for F1 score to be high both precision and recall should be high. Therefore, F1 score gives an overall idea about the classification.

Finally, after evaluating the performance of the classifier for every class, the weighted average of precision, recall and F1 score is taken into consideration since it will help in evaluating the results based on the number of instances in every class.

4. METHODOLOGY

This section describes the proposed methodology for building a resume classifier as shown in Fig 1.

4.1 Step 1: Preprocessing the resumes

In order to utilize the text data from the resumes *R* every resume *Ri* goes through several stages of preprocessing. The text in every resume *Ri* is tokenized, then the stop words are removed, and every token is lemmatized and the frequently occurring n-grams are added to the list of tokens.

4.2 Step 2: Generating resume embedding representations

In this stage a subset of preprocessed resumes is used to train a Word2vec model using the skip-gram architecture implementation from the gensim library. Then for every token in the resume *Ri* from the collection of resumes *R*, a 20-dimensional word embedding representation is generated using the Word2vec model. Then each of the words embedding representation *Wi* for the set of words in the resume *Ri* is added to get a sentence level 20-dimensional embedding representation of the entire resume *Ri*.

4.3 Step 3: Splitting the data

After generating the resume embedding matrix, the data is split into training data which is 80% of the dataset which is used for training and testing the performance of various classifiers with k-fold cross validation and for training the final classifier. The rest is reserved as testing data which is 20% of the dataset used for testing the final best classifier.

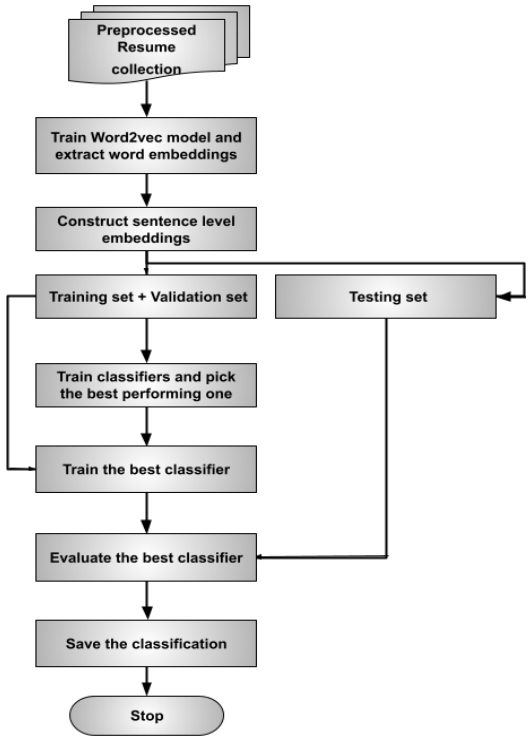

Fig 1: Data Preparation and workflow

4.4 Step 4: Training resume classifiers

In this phase the resume embedding matrix generated in the feature extraction in step 2, is used to train the following classifiers using 10-fold cross validation on the training data:

- Support Vector Classifier
- Decision Tree Classifier
- Random Forest Classifier
- K Nearest Neighbor Classifier
- Naive Bayes Classifier

4.5 Step 5: Evaluating classifiers

Each of the classifiers are evaluated for their performance from 10-fold cross validation on the training data using weighted average F1 score taken over all the 10 folds. The classifier with the best weighted average F1 score and the least standard deviation in the weighted average F1 score is selected as the best performing for resume classification.

4.6 Step 6: Training and testing the best performing classifier

After determining the best performing classifier, in this step the classifier is trained on the on 80% of the dataset reserved for training. The performance evaluation is conducted on the 20% data which was initially reserved for testing using precision, recall and F1-score.

5. DATASET

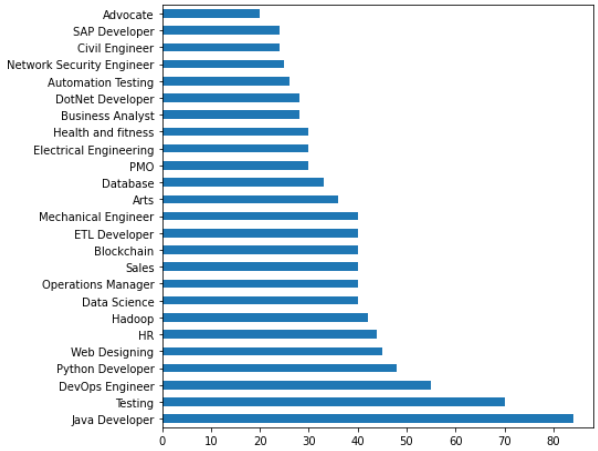

Fig 2: Class distribution in the dataset

To test the proposed methodology, a resume dataset from Kaggle containing 962 resumes from the following classes: Data Science, HR, Advocate, Arts, Web Designing, Mechanical Engineer, Sales, Health and fitness, Civil Engineer, Java Developer, Business Analyst, SAP Developer, Automation Testing, Electrical Engineering, Operations Manager, Python Developer, DevOps Engineer, Network Security Engineer, PMO, Database, Hadoop, ETL Developer, DotNet Developer, Blockchain, Testing was chosen. The class distribution within the dataset is shown in Fig 2.

6. RESULTS AND INTERPRETATION

This section is used to discuss the performance of each of the classifiers and how the best performing classifier is selected as the final model used as the resume classification tool. The results of the best performing classifier on the test data will also be discussed.

Table 1. F1 score computed over 10-fold cross validation for comparing the performance of classifiers

Classifier	Mean F1 score	Standard Deviation
Support Vector Classifier (SVC)	0.508538	0.052003
Decision Tree(DT)	0.989296	0.012411
Random Forest	0.985464	0.014489
K Nearest Neighbors	0.854867	0.057616
Naive Bayes	0.490398	0.063129

After extracting embedding representation for each resume in the dataset, the training split which is 80% of the dataset is used in the 10-fold cross-validation to train and evaluate the 5 different classifiers. For every classifier, in every fold the F1 score is computed and finally the mean F1 score over all folds and its standard deviation is used for comparison of the performance of classifiers. From the comparison of mean F1 score and the standard deviation shown in the Table 1 and Fig 3, we can observe that the Support Vector Classifier (SVC) and the Naive Bayes Classifier (NBC) have the least mean F1 score taken over the all the 10 folds of 0.50 and 0.49 respectively. They are followed by the K nearest neighbor classifier with a

mean F1 score of 0.85 which is still better compared to SVC and NBC. Finally, the Decision Tree classifier and Random Forest classifier have a mean F1 score of 0.989 and 0.985 respectively. Although both of these classifiers have similar performance, the Decision Tree classifier is picked as the best one because it performs slightly better and has smaller standard deviation when compared to Random Forest classifier which indicates that it is consistently performing well. Additionally, Decision Trees are also simple to interpret as compared to Random Forest which makes it easier to understand the classifier.

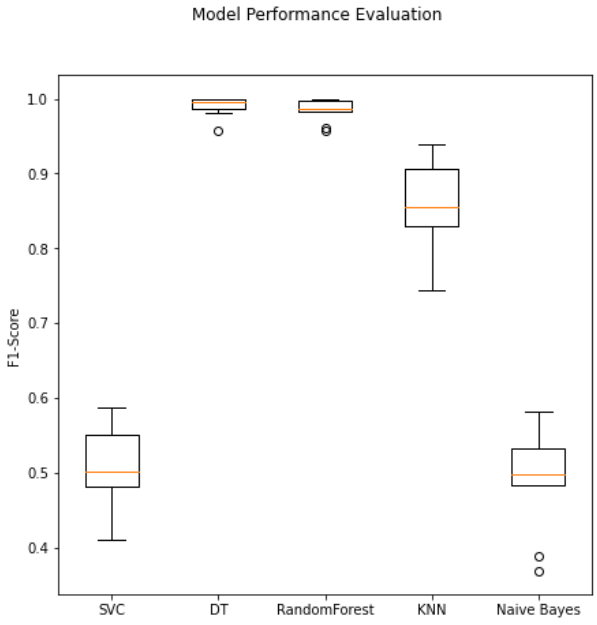

Fig 3: F1 score computed over 10-fold cross validation for comparing the performance of classifiers

Once the selection process is complete, the decision tree classifier is trained on 80% of the training data reserved for training purposes. Then the performance of the classifier is evaluated using the 20% data reserved for testing. The Decision Tree classifier is evaluated using metrics such as precision, recall and F1 score. As we can see in Fig 4, the Decision Tree based resume classifier has weighted average precision = 1.0, weighted average recall = 0.99 and weighted average F1 score = 1.0 taken over all the classes. Therefore, the Decision Tree based resume classifier can categorize the resumes into the relevant categories in most cases with good precision and recall.

7. CONCLUSION

In this paper the existing techniques for text representation and classification have been presented and its application in resume classification have been discussed. According to the proposed methodology after preprocessing the resume dataset, a word2vec model has been trained to extract low dimensional word embeddings which is then used to construct sentence level embeddings for every resume. Then a set of classifiers are tested for their performance on the training data using k-fold cross validation. Of all the classifiers the decision tree classifier performed the best with a weighted F1 score of 0.98. Then a final decision tree classifier was trained and tested on the test data and the final resume classifier performed well on the test data with a weighted average precision = 1and weighted average F1 score = 1 and a weighted average recall = 0.99. Therefore, a resume classification tool has been developed to help the recruiters and the HR teams to categorize the resumes from their resume database to make data-driven decisions.

	precision	recall	f1-score
Data Science	1.00	1.00	1.00
HR	1.00	1.00	1.00
Advocate	1.00	1.00	1.00
Arts	1.00	1.00	1.00
Web Designing	1.00	1.00	1.00
Mechanical Engineer	1.00	1.00	1.00
Sales	1.00	1.00	1.00
Health and fitness	1.00	1.00	1.00
Civil Engineer	1.00	0.93	0.96
Java Developer	1.00	1.00	1.00
Business Analyst	1.00	1.00	1.00
SAP Developer	1.00	1.00	1.00
Automation Testing	1.00	1.00	1.00
Electrical Engineering	1.00	1.00	1.00
Operations Manager	1.00	1.00	1.00
Python Developer	1.00	1.00	1.00
DevOps Engineer	1.00	1.00	1.00
Network Security Engineer	0.75	1.00	0.86
PMO	1.00	1.00	1.00
Database	1.00	1.00	1.00
Hadoop	1.00	1.00	1.00
ETL Developer	1.00	1.00	1.00
DotNet Developer	1.00	1.00	1.00
Blockchain	1.00	1.00	1.00
Testing	1.00	1.00	1.00
accuracy			0.99
macro avg	0.99	1.00	0.99
weighted avg	1.00	0.99	1.00

Fig 4: Evaluation of Decision tree classifier on test data

8. REFERENCES

[1] Kowsari, Kamran, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. "Text classification algorithms: A survey." Information 10, no. 4 (2019): 150.

[2] Aggarwal, Charu C., and ChengXiang Zhai. "A survey of text classification algorithms." Mining text data (2012): 163-222.

[3] Lai, Phung, NhatHai Phan, Han Hu, Anuja Badeti, David Newman, and Dejing Dou. "Ontology-based interpretable machine learning for textual data." In 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1-10. IEEE, 2020.

[4] Patil, Rajvardhan, Sorio Boit, Venkat Gudivada, and Jagadeesh Nandigam. "A Survey of Text Representation and Embedding Techniques in NLP." IEEE Access (2023).

[5] Sonbol, Riad, Ghaida Rebdawi, and Nada Ghneim. "The use of nlp-based text representation techniques to support requirement engineering tasks: A systematic mapping review." IEEE Access (2022).

[6] Upadhye, Akshata Rajendra. "Improving Document Clustering by Refining Overlapping Cluster Regions." Master's thesis, University of Cincinnati, 2022.

[7] Mirończuk, Marcin Michał, and Jarosław Protasiewicz. "A recent overview of the state-of-the-art elements of text classification." Expert Systems with Applications 106 (2018): 36-54.

[8] Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013).

[9] Chang, Chih-Chung, and Chih-Jen Lin. "LIBSVM: a library for support vector machines." ACM transactions on intelligent systems and technology (TIST) 2, no. 3 (2011): 1-27.

[10] Breiman, Leo. Classification and regression trees. Routledge, 2017.

[11] Breiman, Leo. "Random forests." Machine learning 45 (2001): 5-32.

[12] Zhang, H. "The Optimality of Naive Bayes Proc FLAIRS." (2004).