Final Project Report - The use of NLPK machine learning and AI for parsing resumes

12/12/2023
Michael Ginzburg
Benjamin Lindeen
ginzb010@umn.edu
lind1669@umn.edu

1 Abstract

This study is concerned with the fact that Companies and Hiring Managers sift through hundreds and sometimes even thousands of resumes every hiring cycle. With the abundance of job boards and the ease of applying for jobs, mixed with the amount of AI assistance tools that can significantly increase the number of potential applications who are applying for a few positions, it will be critical to have a system that can parse through and sort the candidate who are qualified for the job and those who aren't. This document is the final project using AI for CSCI 4511W to build a resume parser that can effectively go through a database of resumes and successfully interpret which people are the best for the job. The research paper consists of an Abstract, Introduction, Background or Related Work, Problem Statement, Methods, Experiment or Results, Conclusion, and References.

2 Introduction

Hiring Managers used to have to sift through hundreds, sometimes even thousands, of resumes in order to figure out who to hire. Now, however, a lot of this process has become automated through a technique called keyword matching. This website, created by PROJECTPRO, titled "20 Artificial Intelligence Project Ideas for Beginners [2023]", shows job searchers have found a way to game the system.[12] They do this by including as many keywords as possible in order to pass this automated system. The AI can detect too many redundant keywords and throw out the resumes from possible new recruits. This is a very interesting problem since it is very applicable to the real world and can be useful in day-to-day life if you are an HR manager or own a business. This makes the problem of building a successful resume parser very interesting since it can be used in the real world and has a lot of positive benefits to companies and individuals applying.

How resumes are looked over has changed a lot over the years. It used to be done by hand, but now computers do it, which makes the task faster. There are a lot of people who use phrase matching. This is a way for AI systems to find important words. This study, written by Aaron Matos, titled "Transforming Talent Acquisition Through the Power of Automation and Artificial Intelligence" explains that the use of AI will speed up the hiring process, reduce costs, reduce bias, and get better candidates for the job.[9]

However, people applying for these positions have found a way to get around automatic phrase matching. Some people are adding a lot of keywords and terms to their resumes in order to get past these automatic filters. If they do this, they can have a higher chance of getting picked, even if they're not qualified.[12] The fact that this could make these groups of people less useful to the company shows that the automatic screening method doesn't always work and needs some safety nets.

Luckily, AI can be further used to fix this issue by improving the screening process. This AI tool can find jobs that use too many redundant buzzwords and get rid of them. These words will help the AI determine how important they are to the resume and how well it flows. This method speeds up the hiring process by making sure that only the best applicants are looked at.

This is always the fairest and best way to pick candidates. Owners of businesses and HR workers will hire people who are actually good at their jobs more successfully than people trying to game the system. Companies using both resume and word matching and still getting rid of the words that aren't important or used more than once save time and money and earn more money by hiring people who are good at their jobs. Also, this way, people looking for work and having useful skills and experience are hired instead of people just trying to cheat the system. This makes the hiring process better for businesses and fairer for the employees attempting to get hired.

In addition to the main function of enhancing the accuracy and fairness of resume screening, this AI system offers several additional benefits. The main advantage is speed. Business owners can't spend their entire day reading hundreds of resumes. Companies that used to hire many HR people can now cut back on costs since eliminating some of the resumes early on through AI doesn't require as big of an HR workforce.[9]

One other improvement AI makes is being better than humans at hiring based on bias. This is very interesting and important since it can eliminate unfair hiring practices done by humans consciously or subconsciously. AI that only focuses on skills and not other characteristics can create a

fairer and merit-based work pace.[9] This could be an important and beneficial way to society by eliminating prejudice, racism, sexism, etc... from the hiring process.

The rapid evolution of AI in the field of resume parsing and talent acquisition marks a significant shift from traditional hiring practices to more efficient, data-driven approaches. The implementation of AI and machine learning technologies in the hiring process is not just a matter of convenience but a transformative step towards more objective and fair recruitment practices. This paradigm shift offers a multitude of benefits, extending well beyond the obvious time-saving aspects.

One more subtle yet impactful advantage of AI-driven resume parsing is its contribution to data analytics and strategic planning in human resources. Companies can leverage the data gathered through AI systems to identify trends in the job market, understand the evolving skills landscape, and adapt their recruitment strategies accordingly. By analyzing the frequency and relevance of certain skills and qualifications across a pool of applicants, businesses can better align their hiring practices with industry standards and future-proof their workforce.

Moreover, the AI-driven approach to resume screening can significantly enhance the candidate experience. In traditional methods, many qualified candidates might never get a chance to interview simply because their resumes did not include specific keywords or were not formatted in a particular way. AI systems, especially those equipped with advanced NLP capabilities [19], can interpret resumes more holistically, recognizing the value in diverse experiences and skills that might not match conventional keyword searches. This not only broadens the talent pool for employers but also ensures that candidates are evaluated more on their actual merits and potential rather than on their ability to optimize a resume for keyword scanning.

Furthermore, AI in resume parsing can be continuously refined and updated to respond to the dynamic nature of job requirements and skill sets. As job roles evolve with technological advancements, AI systems can be trained on the latest datasets, ensuring that they remain relevant and effective. This adaptability is crucial in today's fast-paced job market, where new skills and roles emerge rapidly.

Additionally, the integration of AI in resume screening can also play a pivotal role in upholding regulatory compliance and ethical standards in hiring. With the increasing focus on privacy and data protection, AI systems can be designed to anonymize and process candidate data in a way that complies with legal frameworks like GDPR [3]. This protects candidates' privacy and ensures that companies maintain ethical standards in their recruitment processes.

In conclusion, the application of AI in resume parsing and talent acquisition is not just a technical upgrade but a strategic revolution in the field of HR. It offers a more efficient, objective, and adaptive approach to hiring, aligning with the broader goals of business optimization, diversity and inclusion, candidate experience enhancement, and ethical compliance. As technology continues to evolve, it's likely that we will see even more innovative applications of AI in human resources, further streamlining the recruitment process and elevating the standards of workforce development.

3 Related Work

This literature review will summarize and synthesize the work that has gone into parsing resumes through machine learning and AI. These sources were all collected through academic journals and sources and will be instrumental in helping to create the final resume parser AI.

4 Related Work: Design and development of machine learning based resume ranking system

The study (Design and development of machine learning-based resume ranking system), written by Tejawini K, identifies problems with the present job market and application procedures.[18] HR faces difficulties in acquiring talent because of the scale of the Indian labor market and high attrition rates. Resume screening is time-consuming due to inefficiencies, such as the absence of a standard CV or resume format. The current systems for making recommendations can enhance the screening procedure. Finding qualified candidates in a crowded applicant pool is a challenge for recruiters. Recommendation systems are extensively employed in many different contexts. Various recommendation algorithms have been investigated in earlier studies. Finding qualified candidates in a crowded applicant pool is a challenge for recruiters. Recommendation systems are extensively employed in many different contexts. Various recommendation algorithms have been investigated in earlier studies. Finding qualified candidates in a crowded applicant pool is a challenge for recruiters. Recommendation systems are extensively employed in many different contexts. Various recommendation algorithms have been investigated in earlier studies. The paper provides an overview of the pertinent issues. Finding the best candidates quickly from large databases is the main challenge. The two main difficulties are interpreting resumes and verifying that applicants are capable of doing the work. An automation solution is suggested to deal with these issues. Finding the most eligible applicants for a position is the goal. A successful hiring strategy depends on effective resume screening. These remedies for resolving these problems are conjectured in the paper. Systems for recommendations are used in many fields, including human resources. Two categories of recommendation systems are content-based and collaborative filtering. In this paper, content-based filtering is applied. This paper approaches the problem in the following ways: Resumes and job descriptions are scaled using Gensim. Preprocessing text entails stemming, eliminating stop words,

and punctuation. Word relevance is assessed using the TF-IDF calculation. KNN techniques and cosine similarity are used to recommend resumes. Their ultimate outcomes were: Each candidate's resume is evaluated by the system, which then generates a relative rating and score. There is 85 percent accuracy in parsing and 92 percent accuracy in scoring.

5 Related Work: A Bibliometric Perspective on AI Research for Job-Résumé Matching

The study (A Bibliometric Perspective on AI Research for Job-Résumé Matching), written by S. Rojas-Galeano, J. Posada, and E. Ordoñez, shows the importance of hiring the best person for the job.[15] Usually, this is done by humans looking over resumes to see if the skills match open positions. Previous study into using algorithms to automate some parts of the applicant screening process was sparked by progress in artificial intelligence, especially in the areas of text analytics and natural language processing. The study looks at universities, writers, journals, trends, and research areas dealing with resume parsing focused on technology. The field is snowballing, and artificial intelligence technology for big language models and natural language processing is getting better very quickly. Researchers have been looking at how ideas about resume matching have changed over time to show how methods and technology have changed.

6 Related Work: Resume Analysis: A Comparison of Two Methods

The study "Resume Analysis: A Comparison of Two Methods", written by Celia Rose Lofink, looks at two different ways to look at resumes: using human readers and expert systems that AI runs.[8] The goal is to look at the possible pros and cons of using artificial intelligence (AI) in the first step of students' resume review. In the job market today, technology is being used more and more to judge prospects. Employers use algorithms based on artificial intelligence to quickly sort through a large group of applications. The study's goal is to find out if artificial intelligence expert systems can do a better job than human reviewers in college career preparation classes or even replace them. These are the main study questions: Research Question (Q1): Is there a clear difference between a human reader and an expert system in terms of bias, time, cost, or number of mistakes? There is no apparent difference between these aspects of the two methods (Absolute Hypothesis). Question 2 (RQ2): If expert system technology were used, would an expert's time be better spent helping students start their careers and reach their goals? Null Hypothesis: The amount of time needed for a professional reader and expert system technology to look over a resume is about the same. The study shows how important it is to understand how application tracking systems and AI work. If students do, they may have a better chance of getting through automatic screening processes and meeting with human reviewers. The results may also change how well human reviewers do in college courses that prepare students for jobs after graduation.

7 Related Work: Bias and Productivity in Humans and Algorithms: Theory and Evidence from Résumé Screening

The study (Bias and Productivity in Humans and Algorithms: Theory and Evidence from Resume Screening) by Bo Cowgill outlines a lengthy hiring process that consists of candidates for jobs visiting the business website.[4] Human screeners evaluate the applications. As part of a bilateral interview procedure, incumbent staff were interviewed both in person and over the phone. Most interviews are done in an unstructured fashion, with interviewers crafting their questions. The interviewers for the candidate made remarks and recommendations. To evaluate a candidate's ability, potential for leadership, and cultural fit, interviewers could be required to complete a survey. Employers make employment offers based on survey data and interviewer comments. Candidates might choose to haggle over the offer if they are receiving rival employment offers. The candidates have the option to accept or reject the offer, and those who accept begin working. The right to withdraw submissions is granted to applicants at any time. Notably, there is no set number of job positions; instead, the goal is to identify well-matched individuals rather than selecting the finest candidates for a limited number of jobs. The firm seldom rejects applications unless an applicant withdraws or is unable to pass the screening procedure. The technology employed in this process makes use of standard text-mining and machine-learning approaches, including resume parsing, content classification, and analysis of several types of variables, including education, job experience, self-reported talents, and resume keywords. Machine learning methods like support vector machines and LASSO is used to predict the results of prior job applications to identify eligible candidates for available openings.

8 Related Work: Resume Analysis Using Machine Learning and Natural Language Processing

The study (Resume Analysis Using Machine Learning and Natural Language Processing) written by Alkeshwar, Jivtode, Kisan Jadhav, and Dipali Kandhare says that it is hard to sort through resumes for professional jobs, especially in IT, where there are a lot of applications.[6] Because resume forms are so hard to understand, recruiters have a tough time finding good candidates. Natural language processing tools, such as NLTK, could be used to get helpful information from resumes. This would help fix issues more quickly and more effectively. The system sorts resumes into the correct groups by using machine learning models such as KNN, SVM, MLP, and LR. It's built to hold a lot of paper. The idea is to

improve the accuracy and speed of the hiring process so that screening is done automatically and no one has to do it by hand as much. This will also make better use of resources.

9 Related Work: Resume Screening Using Machine Learning

The study written by Mungi Naga Venkata Sai Raghavendra titled "Literature Search Introduction: Resume Screening Using Machine Learning" states that there are more specialty recruitment companies in India.[13] These firms use machine-learning models to speed up the hiring process. This paper talks about those businesses. Machine learning and natural language processing (NLP) tools will be used to get helpful information from resumes and give correct ratings based on business standards. This will speed up the hiring process and make it more fair. The study also offers ways to improve the system in the future, which would help both people looking for work and those hiring. For instance, it suggests giving comments and personalized tips on how to make resumes better. Right away, it is emphasized how hard it is to match job needs with candidates' skills and how resumes must now be electronic instead of paper. It shows how these problems can be fixed with innovative hiring systems that make better use of resources, match job candidates with open positions, and keep workers happy while the process is going on. One job's keywords are different from another job's keywords. But if you want to work in computer science, it could be Python, SQL, machine learning, artificial intelligence, or something else. These help the computer easily separate the resume into smaller, simpler chunks.

10 Related Work: NLP Based Resume Parser Analysis

The study(NLP Based Resume Parser Analysis), written by Mhaske Harshada and others, looks at the issues that HR teams face when they need to sort through a lot of resumes.[1] They say that Natural Language Processing (NLP) and phrase matching can be used to quickly and easily get valuable information from jobs. Managers and HR teams should find it easier to find the best people with this method. There are a lot of face-to-face meetings and background checks in the current hiring process, which takes a lot of time and work. The paper calls for a digital method instead. This method would make it easier to get data from unstructured resumes and put it in order. This would help sort and select resumes a great deal since people wouldn't have to study or work by hand as much.

11 Related Work: Semi-supervised deep learning-based named entity recognition model to parse education section of resumes

The study titled "Semi-supervised deep learning-based named entity recognition model to parse education section of resumes," by Bodhvi Gaur and others discusses how hard it is for computers to read resumes and find schools and degrees in the education section. Neural networks are often used for named entity recognition (NER), but they need a lot of labeled data. The study suggests An unsupervised method since there isn't a lot of this kind of data.[2] The first thing that is used to train a deep neural network model is a small set of resume schooling parts. It then makes guesses about what the things are that are in the parts that aren't labeled. A correction tool then makes those guesses more accurate. By adding the fixed things back to the training set, the model gets better at what it does. This process works for 92.06 percent of NER jobs, even though there isn't much data marked up. That is why the paper talks about how hard it is to read resumes with semi-structured text and why the AI model needs automated data labeling. A number of NER methods and models are also talked about. The main goal is to find a new way to correctly find educational skills on resumes. This is very important for recruiters who need to find people with specific academic backgrounds.

12 Related Work: AI based suitability measurement and prediction between job description and job seeker profiles

This study titled "AI based suitability measurement and prediction between job description and job seeker profiles" by Sridevi discusses an AI system that examines resumes to make an educated guess about how qualified job candidates are, which makes the hiring process more manageable.[17] There are four groups comprising primary skills, secondary skills, words, and adverbs. Each of these groups has a job description (JD) and a candidate resume (CR). The AI then looks at the Jaccard similarity between said groups to find a fit measure for each choice. The system uses three kinds of classifiers to try to guess which options will be best: linear regression, decision tree, Adaboost, or XGBoost. Features are made using the bag-of-words method. The model that did the best on average was XGBoost, which got a score of 95.14 percent. This method makes it much easier for businesses to quickly find people who best meet job posting instructions.

13 Related Work: Resume Parser using hybrid approach to enhance the efficiency of Automated Recruitment Processes

A new way to read resumes is shown in this study titled "Resume Parser Using Hybrid Approach to enhance the Efficiency of Automated Recruitment Processes" by Bhoir.[11] Spacy Transformer BERT and Spacy NLP are both used in it. The main goal is to make a resume parser that can quickly get helpful data from unorganized resumes. These resumes don't always follow a standard format, and the data isn't always shown in the same way. Video resumes are also looked at in the study as a new source of candidate information. A new way to read video resumes is suggested, one that combines methods for processing both speech and video. Spacy Transformer BERT is a deep learning model that has already been trained to understand the semantic meaning of text. Spacy NLP is used to pull out important information. This mixed method aims to get the most accurate and quick feedback from resumes about essential things. Tests with a set of resumes show that the system does a great job of getting accurate information about candidates, such as their names, contact information, qualifications, work experience, and other essential factors.

14 Related Work: Resume Ranking using NLP and Machine Learning

"Resume Ranking using NLP and Machine Learning" is the name of the study written by Zubeda and others.[7] It makes use of both Machine Learning (ML) and Natural Language Processing (NLP). The goal is the same as in the other summaries: to make the work of hiring easier to complete. The project's goal is to make it easy for people looking for work and companies to hire each other. An NLP method is used to pick out the most favorable resumes from a group of different types. They are ranked by how well they meet the job standards and what skills they have. A few crucial parts work together to make it whole. The client service and resume library can be used by people outside the company. It has a parser system, a list of possible skill sets, and a complicated algorithm for ranking resumes. This is the more complex system for ranking resumes. Some of the systems and tools that are used in this project are Python, NLTK, PostgreSQL, Elasticsearch, and various machine-learning techniques. In order to use this method, you need to quickly look through samples, pick out skills, and rate them accordingly. The method goes through many tests in the paper to ensure it works right. This AI can be adjusted and used for any business, including tech, healthcare, e-commerce...etc. This paper has a cutting-edge way to use AI to match job seekers with good chances, which improves the hiring process.

15 Related Work: Resume classification system using natural language processing and machine learning techniques

This paper, written by Irfan Ali, presents an automated Resume Classification System (RCS) that leverages machine learning (ML) and natural language processing (NLP) to choose qualified job candidates from a large resume pool.[10] Companies save time and effort by having the classification process automated by RCS. The system uses many natural language processing (NLP) techniques and machine learning algorithms. The machine learning classification models for testing include SVM variants, Naïve Bayes, KNN, and LR. The optimal feature representation was discovered to be Term-Frequency-Inverse-Document-Frequency (TF-IDF). As demonstrated by the RCS, SVM classifiers are pretty good at this multi-class classification task, which used tests including the Confusion Matrix, F-Score, Recall, Precision, and overall Accuracy. SVM classifiers achieved over 96 percent accuracy on a dataset of over 960 resumes. This demonstrates how effective and efficient RCS in accelerating the candidate screening process may be created using NLP and ML.

16 Related Work: Summary

In conclusion, these sources will be knowledge pillars when building the project. All of these sources combined state how this project is applied in real life as well as ways to implement it.

17 Problem Statement

People have found a way to cheat the system when applying for jobs using the knowledge that many companies use AI to screen resumes due to it being more time-efficient. Most AIs use a keyword-matching method that takes the skills that are posted on the job board and compares them with how many words match the resume. However, many people have a workaround for this by just throwing in as many keywords as possible on their resumes in order to game the system, even if they don't have any real substance or skills on their resumes. The goal will be to build a resume parser that still correctly identifies what resumes are good for specific positions while simultaneously not letting redundant or non-keywords result in better scores for the people trying to game the system.

18 Methods

Before making the Resume parser AI, a large set of resumes was collected, and a simple exploratory data analysis (EDA) was done using Python and the pandas library. This data set originated from Kaggle.[5] After the EDA was completed on the collected data, it became clear that it was mostly made up of two columns: "Category," which listed the types of jobs,

and "Cleaned Resume," which held the actual resumes. There are 962 rows in this collection, which means that each row has one unique resume.

A closer look showed that these applications were aimed at 25 jobs, including 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, and Testing listed in the "Category" column. Before any AI was involved, if an HR person were sent this large data set without knowing how to do EDA on it, the time it would've taken them to read through 962 resumes would be staggering. The Resume AI Parser is a solution.

After completing the EDA, the program began building the actual AI. National Language Toolkit (NLTK), a well-known Python library, is the project's main Natural Language Processing (NLP) API.[1]

In the early stages of making the Resume Parser AI, a lot of work went into cleaning up and standardizing text data.[6] At this point, the main goal was to make it possible to look at text data in more detail by cleaning and standardizing it. All the information was first changed into lowercase letters to ensure it would look the same in all resume formats and styles. After any unique characters were taken out of the document, the next step was tokenization, which breaks text down into individual words or "tokens" for analysis purposes.

Using the Natural Language Toolkit (NLTK) library to get rid of stopwords was a key turning point in the preprocessing phase, as described by WangHu in the literature review.[19] People who work with text often get rid of stopwords because they don't add much or any useful information to a summary. "The," "and," "it," and "in" are all examples of these so-called "stopwords." By removing these words, the focus could be put more on language elements that show off a person's skills and knowledge.

It was possible to successfully use the Natural Language Toolkit (NLTK) during this important pre-processing step.[19] This used lemmatization, a method that returns words to their original form so that they can be used consistently and in a wide range of situations. To use it, change verb forms like "developing," "develop," or "developer" to their most basic form. In this case, that would be "develop." The use of this feature in the study is noteworthy because it helps us combine different word formats into a single, standard template. So, figuring out the most important skills and experiences can go on without being slowed down by differences or inconsistencies in how people write their resumes. This will also be how the AI eventually will stop scoring certain resumes higher just because they used many similar keywords.

After the first part of pre-processing, the project turned its attention to feature extraction, an important part of machine learning processes. The study used a statistical tool called Term Frequency-Inverse Document Frequency (TF-IDF) [17] to find out how important words are in different documents, like resumes, compared to all texts together, called "corpus." The TF-IDF is the weight of the term that occurs in the resume document proportional to the term frequency of d. The equation is a count of t in d/ number of words in d.[18] When using the Term Frequency-Inverse Document Frequency with lemmatization, it normalizes the scores.

Now combining all of that, a score function was implemented, which is this(pseudo code).

def calculateScoreTfIdfNoDuplicates(tfidfRow, categorySkills): score = 0 skillsCounted = set()

for skill in categorySkills: if skill in tfidfRow.index and tfidfRow[skill] > 0 and skill not in skillsCounted: score += tfidfRow[skill] skillsCounted.add(skill) print(Found skill skill with score tfidfRow[skill]) elif skill not in tfidfRow.index or tfidfRow[skill] <= 0: print(Skill 'skill' not found)

This function takes in two arguments, the tfidfRow, which is a pandas series that represents the TF-IDF vector of a resume, and categorySkills which is a list of skills that are relevant for a specific job category. skillsCounted keeps track of what skills have already been counted in order to not count the same skill more than one time. The function then iterates over each skill in categorySkills, and for each skill, it checks if the skill is in the TF-IDF row and whether the TF-IDF score is greater than zero. This is to make sure that the skill needed for the job title is actually in the resume. If the skill gets found and is not already in the skillCounted set, the skill will be added to the skillCOunted set. This function makes sure that each skill contributes to the score only once, even if more than one of that keyword appears in that resume. If a word is similar to another word, like developer or develop, it turns to the root word as mentioned earlier, with lemmitization, so that will not be a problem for this function.

This method was able to get a better sense of how important different words were in different resumes based on how common they were in our dataset. The algorithm paid extra attention to words that only appeared in certain job applications and were used often between different resumes(not the same one). These words were specialized skills or relevant knowledge unique to those job categories, like, for example, python, AI, machine learning, etc... This basically means that it won't reward a person for having people skills in their resume if they are applying to a data science job as much if for example, they were applying to HR.

In conclusion, Creating the Resume Parser involved tokenization, Term Frequency-Inverse Document Frequency, lemmatization, and stopwords, all stemmed from the NLTK library. [19]However, the EDA and the scoring function that was done on the dataset before the parser was created using the pandas library.

19 Expirement and Results

The experimental steps that yielded results that the AI took were grouping methods to put similar resumes with each other. At this point, the AI's goal was to group the newly transformed TF-IDF resumes into different groups based on similar traits. The AI accomplished this by using the K-Means machine learning algorithm, a well-known grouping algorithm that is known for how quickly and easily it can divide data into k groups via a manually chosen k value. This was not in the literature review in order to test a new machine learning algorithm not mentioned before. The program used "the Elbow Method" to determine how many sets or clusters were beneficial for the AI to get the best scores. It's an efficient way to find the best value for k by trying different values until an "elbow" curve appears when plotting within-cluster sum squares (WCSS) against cluster numbers. After this process, the program was able to get very clear information about how many groups would work best for the dataset. According to the graph, the elbow ranges from 15 to 23. This is interesting since the assumption would be that the best number of clusters would be the number of categories of jobs, which is 25. However, this makes sense since many of these jobs overlap, such as Data Science and Business Analyst, Data Science and Python Developer, Python Developer and Java Developer, etc... This is not a problem with the k-means clustering or the AI since a lot of companies when they have shortages in Data Science for example sometimes takes resumes that are for a different job from their databases and sometimes asks the person if they would like to apply to Data Science. It is important for AI to give a chance, for example, to Python developers who have the necessary skills to be data scientists, even if they didn't apply for that position. Since the (WCSS) against cluster numbers, the graph displayed an elbow at k clusters' position from 13 to 23. A decision had to be made about which k value to choose. Since the job categories in the database were 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, and Testing, the decision was made to include all computer science related fields into the same cluster. This was what a cluster graph of k = 13 clusters looked like, with one cluster containing mostly Data Science, Java Developer, Python Developer, Business Analyst, Web Designing, Testing, DevOps Engineer, Database, SAP Developer, Automation Testing, and DotNet Developer. All the other clusters match up to the categorical job description quite well. There were outliers, like two resumes from PMO Developer got into the big Computer science cluster, and 7 resumes in total from Civil Engineer and Electrical Engineer got into the big computer science cluster. Network Security Engineer formed its own cluster with some overlap with the big computer science cluster. All the other clusters correlated with their categories(jobs).

After the program assigned each resume to a cluster based on how closely it matched the group's center, the final step was to judge each resume based on its relevance and compatibility with different job fields. The AI was able to reach its goal by giving each application a weighted number based on how often important skills were shown and how important those skills were for different job categories while simultaneously not adding to the score if duplicate keywords or very similar keywords were used.

According to the literature review, XGBBoost was another machine learning method that the AI used to sort the jobs into groups.[17] Adding XGBoost to the dataset analysis model made a big improvement to the process of sorting and ranking resumes. Using XGBoost, a machine learning algorithm known for being able to handle a lot of different datasets, resumes were put into different job groups based on the information on them. The point of the project was to find out how accurate XGBoost is at classifying documents based on a number of attributes taken from the TF-IDF vectorization process.

The dataset, which is made up of resumes that were each assigned to a related job category, went through a long preprocessing process. Still, Text normalization, lemmatization, and getting rid of stopwords remained important parts of this process. This made sure that the data that was fed into the model was clean and consistent. After that, the prepared text was turned into a TF-IDF matrix, which our algorithmic model used as input features. With the help of Scikit-learn's LabelEncoder tool, they were given numerical labels that fit into the database's job categories. This made them ready to be put into categories. In order to teach the model, the dataset had to be split into separate sections for training and testing, with 80 percent of the dataset being used for training and 20 percent set aside for later evaluation. This division was very helpful in figuring out how well the model could deal with data it had never seen before. This process is very common in data science, machine learning, and AI. After that, this train set was used to teach the XGBoost classifier how to understand how features are related to job categories using TF-IDF scores. The test set was used to check how well the model worked after its training phase. Accuracy, which is the percentage of correctly categorized CVs into their correct classes, was the most important factor used in this evaluation. To get a full picture of how well the model worked, other metrics like precision, recall, and F1-score were also checked.

According to the literature review, XGBBoost was another machine learning method that the AI used to sort the jobs into groups.[17] for reference Adding XGBoost to the dataset analysis model made a big improvement to the process of sorting and ranking resumes. Using XGBoost, a machine

learning algorithm known for being able to handle a lot of different datasets, resumes were put into different job groups based on the information on them. The point of the project was to find out how accurate XGBoost is at classifying documents based on a number of attributes taken from the TF-IDF vectorization process

The dataset, which is made up of resumes that were each assigned to a related job category, went through a long preprocessing process. Still Text normalization, lemmatization, and getting rid of stopwords remained important parts of this process. This made sure that the data that was fed into the model was clean and consistent. After that, the prepared text was turned into a TF-IDF matrix, which our algorithmic model used as input features. With the help of Scikit-learn's LabelEncoder tool, they were given numerical labels that fit into the database's job categories. This made them ready to be put into categories. In order to teach the model, the dataset had to be split into separate sections for training and testing, with 80 percent of the dataset being used for training and 20 percent set aside for later evaluation. This division was very helpful in figuring out how well the model could deal with data it had never seen before. This process is very common in data science, machine learning, and AI. After that, this train set was used to teach the XGBoost classifier how to understand how features are related to job categories using TF-IDF scores. The test set was used to check how well the model worked after its training phase. Accuracy, which is the percentage of correctly categorized CVs into their correct classes, was the most important factor used in this evaluation. To get a full picture of how well the model worked, other metrics like precision, recall, and F1-score were also checked.

As the last machine learning method, Support Vector Machines (SVM) are used by the AI resume creator. Ali's review of the research says that SVM does a good job of categorizing the resumes.The useful machine learning method called Support Vector Machines (SVM) works best in areas with a lot of dimensions. These work for problems like this one that involve sorting text into groups. The test was meant to see how well the SVM model could put resumes into predefined job groups using a big set of features from the TF-IDF vectorization of resume data. The dataset was carefully made before it was used, and it had a lot of resumes that were sorted into the right job groups. Getting rid of stopwords, lemmatizing, and standardizing the text were all parts of this process. These steps must be taken to make sure that the raw data is correct and uniform. An TF-IDF grid was made after the resumes were cleaned up. In this matrix, weights were used to show how important each term was in the dataset and to find specific resume content traits. The job groups were turned into numerical labels using LabelEncoder, a tool needed for machine learning models that need numerical input. This set up the job of sorting things into groups. After that, a part of the information was used to train the SVM model. Eighty percent of the information was used for training, and twenty percent was set aside for testing. To get a true picture of the model's ability to generalize, the data had to be split into tests and trains. The model was tested on the hold-out set after training, with accuracy being the main measure of success. At almost 99.48 percent, the model was very good at putting things into groups, as shown by the number of correct guesses compared to all predictions.

The resume parser AI performed on the obtained real-world dataset accomplished its purpose and got very good results. Each Resume in the dataset scored higher within its own category than when compared to another category. For example, the first 39 entries of the dataset were resumes that were applied for and had a category value of Data Science in their first column. This means the first 39 resumes were applying for data science positions. Keywords were gathered from a data science job posting from 2023 for Nation Grid Renewables.[14] Plugging these keywords into a Python dictionary titled jobCategorySkills, which maps the category of "Data Science" to the list of keywords and skills from the job posting by National Grid Renewables2023, provides a basis for the AI to grade the quality of resumes based on the number of keywords. A function titled calculateScoreTfIdfNoDuplicates then calculates the score of each row(resume) using the TF-IDF method, which doesn't consider duplicate words in the score or words that mean similar words. The resulting code then prints out scores for every single Resume for how good of a fit that resume would be for a data science position. The first 39 resumes had very high scores, with the best one being 67 out of 100, while the resumes within the category of HR had all scores of zero. However, Once the parser got to the developer resumes, especially the Python Developer resumes, it also got high scores. This makes sense since Data Scientists don't always hire strictly data science people, and most data scientist work is in Python. The Python developers' resumes got an average score of 43 with the averageScore function, which takes all the scores from one category (job), adds them, and then divides them by the number of rows of the same category. The average score obtained by the resumes in the data science category for a job listing for a data scientist was 54. The exact process was done with all the other categories(jobs) where a job posting was found, the keywords were taken from it, and the resume parser outputted a higher score for the resumes that were trying to apply for that job since they had more keywords.

The AI then underwent its final experiment. A row for each category of jobs was added to the original dataset that contained a copy of the first row in each category. Only this time, the copy had more keywords added to it that were very similar to the original. However, when this was done, the scores did not go up at all for any category, and when a lot of keywords were added, the score actually went down because TF-IDF makes it so if the amount of words proportional to the number of key terms is higher, then the score goes

down according to the equation t in d/ number of words in d. Combined with lemmatization and the score function, which only counts one word to the score, this makes sense. It has successfully parsed resumes that job searches require while simultaneously penalizing people who stuff their resumes with words. The scores for some resumes and there copies were. Data Science: Original 56.09, Copy 56.09, HR: Original 32.2 Copy 32.2 ... Python Developer Original 77.9 Copy 72.21.

20 Analysis of the results

The silhouette score measures how well clustering works in machine-learning applications. This score is usually used to evaluate clustering techniques such as K-Means clustering. The score is determined by evaluating the grouping of data points, which is done in several ways. The first of which is cohesion, which looks at how similar the data points are within a given cluster. The second of which is separation, which looks at how similar the data points are between all of the clusters. The scores for both of these can range from negative one to positive one. Making sense of the scores. A score of positive one means that the clusters are far apart from one another and that each one is clearly distinguished. A score of negative one means that the clusters are assigned in an incorrect way. While a score of zero would mean that the clusters are indifferent and the distance between them shows no significance.

The Calinski-Harabasz Index score is the last way to rate clustering algorithms like K-Means clustering, DBSCAN, or HDBSCAN that will be looked at and used in this work. You can use this way to see how well the data points are grouped together and spread out. Like the silhouette score, the data is looked at for harmony and separation. Cohesion is how the data points in a cluster connect to each other, and separation is how the clusters connect to each other based on how far they are from the world center. Some important changes exist between the silhouette score and the Calinski-Harabasz Index score. The main reason is that the Calinski-Harabasz Index doesn't care as much about how many groups are being looked at. This is helpful for algorithms or methods that consider many groups, like K-Means clustering. Second, the silhouette score looks at both cohesion and separation and gives a score that shows how well the clusters have been separated. The Calinski-Harabasz Index score, on the other hand, gives a ratio that compares the cohesion, or within-cluster variation, and the separation, or between-cluster variation.

The Davies-Bouldin index score is yet another metric for evaluating clustering algorithms such as K-Means clustering, DBSCAN, or HDBSCAN. The Davies-Bouldin index score works by measuring how different each cluster is from the other clusters and by how much. This metric is calculated by averaging the similarity of each cluster with the cluster that is most similar to it. The similarity of the cluster is calculated by taking the ratio between the inter-cluster and the intra-cluster distances. The Davies-Bouldin index will work well for working with small and large numbers of clusters. Makes no assumptions about the shape of the clusters, unlike silhouette scores. Evaluating the scores. When the index score is low, it means that this would indicate that the clustering is done in optimal configurations. This would predict that the clusters have strong connections. On the other hand, having a high index score would indicate that the clusters are not well separated. This would imply that there may be overlapping features. A high score is undesirable and shows that the model has room for improvement.

The evaluation for the Resume Parser AI model resulted in a Silhouette Score of 0.8340095367123582, a Calinski-Harabasz Index of 110.86635037517557, and a Davies-Bouldin Index score of 0.5740066848256452.

The Silhouette Score of 0.8340095367123582 is very high and close to the upper bound for a Silhouette score of 1. This would indicate the clustering is done well with a high degree of separation between the clusters.

Calinski-Harabasz Index score of 110.86635037517557 would suggest that the clusters are well separated and, simultaneously, very compact.

Finally, the Davies-Bouldin Index score of 0.5740066848256452 indicates that the clusters are more or less well separated and reasonably distinct from one another.

When we look at how well the XGBoost model can classify resumes, we can draw a number of important conclusions. The model's high accuracy of 98.45 shows how well it can sort applications into the right categories. This level of accuracy is really amazing when you think about how complicated and varied resume data is. This shows how strong and flexible the method is and how it can be used with different types of resumes.

The model's precision score of 0.86 is a good sign of how reliable it is. In everyday language, this means that there is an 86 chance that the program will correctly put a resume into the right job area. This is very important for a resume parser to make sure that there is less chance of misclassification and that recruiters get a pool of people who really fit the job description. The model is very complete, as shown by its 0.93 memory score. In this case, it shows how well the model can find a lot of resumes that fit each group. It is very important to do this so that qualified people don't get thrown out during the screening process. Memory and accuracy are both taken into account in the F1 score, which is 0.89. This score shows how fair the model's methodology is, which makes sure that the classification stays full and useful. A model must be accurate and complete for resume processing and classification to work. An F1 score close to 1 shows that the model is both. Good results were found: the model's accuracy rate was about 98.45, which means it was very good at identifying things. With a precision score of about 0.86, most of the resumes entered for a certain class were found

to meet the requirements of that class, showing that each category was very relevant. The method got an impressive recall score of 0.93, which means that most of the relevant resumes were found correctly for each group or part that was looked at in this case. The F1-score of about 0.89, which is just the average of precision and recall, shows that the model worked well and put resumes in the right group.

To sum up, the results of the XGBoost model look very good. They prove that the model can reliably and correctly sort resumes, making it a useful tool in the hiring process. Because the model did well in accuracy, precision, memory, and F1, it seems like it can make good decisions for a variety of job types. The SVM model works very well, as shown by its 99.48 success rate, even for difficult tasks like resume parsing. Based on the high accuracy rate, it looks like the computer does a great job of putting resumes in the right job categories. This level of accuracy points to a feature extraction method that is well-tuned and a model that is well-trained enough to be able to tell the difference between job types based on resume content. Even though getting very accurate results is generally what people want, they need to be careful to get them. Very high precision in machine learning can sometimes be a sign of overfitting, especially when the problem is a classification problem. If a model is too similar to the training set and picks up on details and noise that don't move well to new data, this is called "overfitting." Because of this, it is very important to test the model on a test set that is fairly representative and diverse. To get a better look at things, you should also think about using cross-validation method. It's also possible that accuracy alone might not give you the whole picture if the dataset isn't balanced (for example, if some job categories are clearly overrepresented compared to others). To fully understand how well the model works in these cases, we need to look at other factors like precision, recall, and F1-score. These measurements can tell you how well the model does at finding relevant resumes, remembering all relevant resumes, and balancing these traits (F1-score). To sum up, the SVM model's incredibly high level of accuracy in categorizing resumes is positive and shows a high level of effectiveness. That being said, more testing and checking is needed to make sure that the model is accurate and reliable, and that it works well across all of the job groups in the dataset. This all-encompassing plan makes sure that the model stays fair and can be used in other situations, in addition to being very accurate. These are two important qualities for any AI-based tool used in HRM and talent acquisition.

In conclusion, all of these metrics point to the fact that the Resume Parser AI model is K-means clustered effectively. This is of huge importance when determining the legitimacy of the Resume Parser AI model. Given this, the Davies-Doublin Index score was slightly lower than desired, which could serve as a point of improvement going forward. Furthermore, the way in which the resumes were scored was based on categorizing them and ranking them according to how relevant they were for each type of job. Higher scores were given to resumes that correlated to the skills and experience that were related to the job description in a particular category. On the other hand, the SVM had an accuracy score of almost one hundred percent, which doesn't seem very likely in the real world. This has led to the conclusion that the model is being overfitted and thus is not a good way of parsing resumes at least for categorizing this specific Resume Parser AI. The XGBBoost, as previously stated by Sridevi in the literature review, accomplishes categorization very well when it comes to large and very diverse datasets.[17] Knowing this, the Resume Parser AI model could accurately put resumes into groups and judge each one's fitness for different job positions while also getting rid of those who simply tried to game the system and jam as many keywords into the resume as possible.

However, XGBBoost and K-means clustering all depend on which categorization the employer prefers. There are a lot of HR managers and business owners who prefer to hire people with different backgrounds and experiences. However, there are also some very specific jobs, and HR/business owners want a very specific employee. As shown by the K means cluster method, many people applying to certain jobs overlapped clusters and merged into one cluster(computer science) that hires might want to use to hire people who have not just one skill but many in a broad field. However, it also makes sense to want to hire a person for a very specific role in which XGBBoost might be a better model to use.

21 Conclusion

In conclusion, the days in which human beings had to painstakingly read hundreds upon thousands of resumes are a relic of the past. The future will be dominated by AI. The modern-day large to medium business is already using the powers of artificial intelligence to sift through resumes. This has become the norm. In this research, it has been proven through the use of an artificial intelligence model using Natural Language Toolkit (NLTK) in Python, Pandas, Scikit-learn, K-means, and XGBBoost that it is not only capable but competent in clustering and evaluating the quality of candidates for different job positions based on parsing their resumes.

The strength of the artificial intelligence resume rating system comes from the speed and consistency and the Natural Language Processing (NLP) techniques such as tokenization, lemmatization, and Term Frequency-Inverse Document Frequency (TF-IDF) analysis. [1]This process seeks to find a harmonious balance between separating the qualified from the unqualified or the people trying to game the system. This is done by maintaining a balance between keyword significance and redundancy. Consequently, this system adeptly

identifies the most suitable candidates by focusing on genuine skills and experiences rather than being misled by an overabundance of keywords.

The three machine-learning techniques that the Resume Parser employed to categorize the different resumes were K-means clustering, XGBBoost, and SVM. K-means clustering resulted in a less rigid approach of hiring people who were applying in different fields but still had useful skills to offer the company for a different role. While, The XGBBoost had a very strict categorization which resulted in the AI only considering the person for the position they intended to apply to. Overall it really is up to the employer which method to use for their Resume Parser AI. There is validity in doing either one, depending on the circumstance.

Now moving on to more of the ethical and practical reasoning for this project. The system that was devised here is far more unbiased and impartial compared to a human screener, to mention nothing of the fact that the artificial intelligence will never get tired or misinterpret anything as long as it has enough computing power to function. The resume parser does not know nor care if the candidate went to their alma mater, nor if it is from the same hometown. It does not know race, gender, or nationality. The system is fair to everyone equally, which is something that study after study has proven false about human hiring systems.

Being ethical is nothing if the results are still flawed, but that is not the case in this situation. Given the high scores seen when evaluating the artificial intelligence resume rating systems - K-means clustering through the silhouette score, the Davies-Bouldin Index score, and the Calinski-Harabasz Index score - it can be said with certainty that the results that were given are accurate beyond a reasonable doubt and should serve well in matching the correct candidates to the correct jobs, in addition to that the XGB Boost machine learning algorithm also performed even better than the K-means technique of data clustering.

Given this pedigree, it is safe to say that there is currently a paradigm shift happening in the way in which companies recruit new talent. The efficiency and equitability stand unparalleled compared to artificial intelligence human counterparts. This development marks a notable progression from traditional methods, aligning with the evolving dynamics of the job market and the growing reliance on artificial intelligence tools. The future implications of this research are manifold, suggesting potential enhancements and adaptations to refine the parser's capabilities further.

This study proves that, at least for this application, Humans need not apply. The downsides are far outweighed by the upsides. This, however, is not a condemnation or obituary for hiring managers and human relations teams everywhere but simply a tool that can be used to automate the most tedious and boring process away from them, leaving only the uniquely human and fulfilling roles to be done. However, it would be remiss not to mention the growing threat of artificial interference replacing at least faster-than-can-be-recreated jobs that have been dependent on human intervention for the entirety of human history. Oddly enough, this artificial intelligence resume rating parser may just be put out of business, but the very advancements it relied on exist in the first place. From an ethical perspective, it is important to balance the need for human work and livelihood with the automation and easier life that AI can and will create. In conclusion, The AI parser was created, accomplished its goal of parsing resumes successfully, and made sure that people trying to game the system didn't get around keyword matching by stuffing resumes with keywords.

22 Attribution

This paper has two authors, Benjamin Lindeen and Michael Ginzburg. The two of them will be accredited for the work done in this paper as such. Tejaswini2021 [18] was gathered, read, and analyzed by Benjamin Lindeen. RojasGaleano2022 [15] was gathered, read, and analyzed by Benjamin Lindeen. Lofink2021 [8] was gathered, read, and analyzed by Benjamin Lindeen. Cowgill2020 [4] was gathered, read, and analyzed by Benjamin Lindeen. Jivtode2023 [6] was gathered, read, and analyzed by Benjamin Lindeen. Raghavendra2022 [13] was gathered, read, and analyzed by Benjamin Lindeen.

Harshada2023 [1] was gathered, read, and analyzed by Michael Ginzburg. Gaur2020 [2] was gathered, read, and analyzed by Michael Ginzburg. Sridevi and Suganthi 2022 [17] were gathered, read, and analyzed by Michael Ginzburg. Bhoir2023 [11] was gathered, read, and analyzed by Michael Ginzburg. Zubeda2016 [7] was gathered, read, and analyzed by Michael Ginzburg. Ali2022 [10] was gathered, read, and analyzed by Michael Ginzburg. Chandra2018 [3] was gathered, read, and analyzed by Michael Ginzburg. Michael Ginzburg gathered, read, and analyzed PROJECTPRO2023 [12]. Sarker2021 [16] was gathered, read, and analyzed by Michael Ginzburg. Wang and Hu 2021 [19] were gathered, read, and analyzed by Michael Ginzburg. Matos2023 [9] was gathered, read, and analyzed by Michael Ginzburg. Dutta2020 [5] was gathered, read, and analyzed by Michael Ginzburg. National Grid Renewables 2023 [14] was gathered, read, and analyzed by Michael Ginzburg.

Benjamin Lindeen and Michael Ginzburg devised and worked on the idea and preparation for the project. The libraries for Python and the dataset were found by Michael Ginzburg. Michael Ginzburg wrote the abstract. The introduction was written by Benjamin Lindeen. The summary of related work was written by Michael Ginzburg. The problem statement was written by Benjamin Lindeen. The methods were devised, planned, coded, and written about by both Michael Ginzburg and Benjamin Lindeen. The experiments and results were conceptualized, planned, and documented by both Benjamin Lindeen and Michael Ginzburg.

The text processing was handled and coded by Benjamin Lindeen. The vectorization was handled and coded by Michael Ginzburg. The Scikit-Learn and K-Means clustering were done by Michael Ginzburg. The Scikit-Learn metrics were performed, coded, and analyzed by Benjamin Lindeen. The creation and implementation of the XGBBoost was handled by Michael Ginzburg. The conclusion was reached by Benjamin Lindeen that there are positive to both algorthims K-means clustering and XGBBoost. Michael Ginzburg handled the creation and implementation of SVM. Benjamin Lineen reached the conclusion that SVM was not viable for our specific code and dataset, most likely due to overfitting. handled the analysis and evaluation metrics of the model. The performance score calculation was coded by Benjamin Lindeen. The analysis of the results was researched, tested, and written about by both Michael Ginzburg and Benjamin Lindeen. Benjamin Lindeen wrote the conclusion. Benjamin Lindeen wrote the attribution. Michael Ginzburg then reviewed the final Research Paper and write-up before submission.

23 Authors and Affiliations

Author: Michael Ginzburg
Email: ginzb010@umn.edu
Author: Benjamin Lindeen
Email: lind1669@umn.edu

24 Citations and Bibliographies (References)

References

[1] Mhaske Harshada A. N. Kshirsagar Nevase Sonali Pimparkar Ankita. 2023. NLP Based Resume Parser Analysis. *Journal of Emerging Technologies and Innovative Research* 10 (2023). Issue 5. https://www.jetir.org/view?paper=JETIRFX06037

[2] Hamsa Bharathi Sivakumar Sanjay Singh Bodhvi Gaur, Gurpreet Singh Saluja. 2020. Semi-supervised deep learning based named entity recognition model to parse education section of resumes. *Springer Link* (2020). https://link.springer.com/article/10.1007/s00521-020-05351-2

[3] Ronil V. Chandra, Amir H. Razavi, Shiwei Huang, Mark Brooks, Michael J. Lee, Hamed Asadi, Guy S. Handelman, and Hong Kuan Kok. 2018. Peering Into the Black Box of Artificial Intelligence: Evaluation Metrics of Machine Learning Methods. (2018). https://ajronline.org/doi/pdf/10.2214/AJR.18.20224

[4] Bo Cowgill. 2020. Bias and Productivity in Humans and Algorithms: Theory and Evidence from Résumé Screening. (2020). https://conference.iza.org/conference_files/MacroEcon_2017/cowgill_b8981.pdf

[5] Gaurav Dutta. 2020. UpdatedResumeDataSet.csv. (2020). https://www.kaggle.com/datasets/gauravduttakiit/resume-dataset

[6] Alkeshwar Jivtode, Kisan Jadhav, and Dipali Kandhare. 2023. Resume Analysis Using Machine Learning and Natural Language Processing. (2023). https://www.irjmets.com/uploadedfiles/paper//issue_5_may_2023/39611/final/fin_irjmets1684836041.pdf

[7] Gunduka Rakesh Narsayya Godavari Sayed ZainulAbideen Mohd Sadiq Naseem Juneja Afzal Ayub Zubeda, Momin Adnan Ayyas Shaheen. 2016. Resume Ranking using NLP and Machine Learning. https://core.ac.uk/reader/55305289

[8] Celia Rose Lofink. 2021. Resume Analysis: A Comparison of Two Methods. *Journal of Business, Economics and Technology—Spring 2021* (2021). https://www.nabet.us/j_archives/JBET_2021.pdf#page=81

[9] Aaron Matos. 2023. Transforming Talent Acquisition Through the Power of Automation and Artificial Intelligence. (2023). https://www.paradox.ai/report/harvard-business-review-how-ta-is-transforming-with-ai-and-automation#report

[10] Irfan Ali; Nimra Mughal; Zahid Hussain Khand; Javed Ahmed; Ghulam Mujtaba. 2022. Resume classification system using natural language processing and machine learning techniques. (2022). https://search.informit.org/doi/abs/10.3316/informit.263278216314684

[11] Snehal Lavangare Aarushi Das Sujata Kolhe Nirmiti Bhoir, Mrunmayee Jakate. 2023. Resume Parser using hybrid approach to enhance the efficiency of Automated Recruitment Processes. (2023). https://www.authorea.com/doi/full/10.22541/au.168170278.82268853

[12] PROJECTPRO. 2023. 20 Artificial Intelligence Project Ideas for Beginners [2023]. (2023). https://www.projectpro.io/article/artificial-intelligenceproject-ideas/461

[13] Mungi Naga Venkata Sai Raghavendra. 2022. Resume Screening Using Machine Learning. *Journal of Engineering Sciences* 13 (2022). Issue 09. https://jespublication.com/upload/2022-V13I9053.pdf

[14] National Grid Renewables. 2023. Machine Learning Engineering Intern. (2023). https://www.indeed.com/cmp/National-Grid-Renewables/jobs?jk=551acd5a4f2fb433&start=0&clearPrefilter=1

[15] S. Rojas-Galeano, J. Posada, and E. Ordoñez. 2022. A Bibliometric Perspective on AI Research for Job-Résumé Matching. *The Scientific World Journal* (2022). https://www.hindawi.com/journals/tswj/2022/8002363/

[16] Iqbal H. Sarker. 2021. Machine Learning: Algorithms, Real-World Applications and Research Directions. (2021). https://link.springer.com/article/10.1007/s42979-021-00592-x

[17] S. Kamala Suganthi Sridevi GM. 2022. AI based suitability measurement and prediction between job description and job seeker profiles. (2022). https://www.sciencedirect.com/science/article/pii/S2667096822000520

[18] Shashank M Kadiwal a Tejaswini K, Umadevi V b and Sanjay Revanna a. 2021. Design and Development of Machine Learning-Based Resume Ranking System. *Global Transitions Proceedings* 3, 2 (2021). https://www.sciencedirect.com/science/article/pii/S2666285X21001011

[19] Meng Wang and Fanghui Hu. 2021. The Application of NLTK Library for Python Natural Language Processing in Corpus Research. (2021). https://tpls.academypublication.com/index.php/tpls/article/view/1400/1129