Resume Analysis
Data Mining Group 20 Project Report

Jayan Agarwal*
University of Colorado Boulder
Boulder, CO, USA
jayan.agarwal@colorado.edu

Tuan Nguyen*
University of Colorado Boulder
Boulder, CO, USA
tuan.nguyen@colorado.edu

Shivani Madan*
University of Colorado Boulder
Boulder, CO, USA
shivani.madan@colorado.edu

Abstract

Recruitment processes are essential for bridging the gap between job seekers and employers. However, the exponential growth in job applications and the increasing complexity of job roles have rendered traditional recruitment methods inefficient and time-consuming. With manual resume screening and job matching, recruiters often struggle to identify the most suitable candidates, leading to suboptimal hiring decisions and delays in the recruitment process. Additionally, job seekers face challenges in understanding market demands, identifying relevant skills, and tailoring their resumes effectively for targeted roles.

This project addresses these challenges by leveraging resume data to develop machine learning (ML) and deep learning (DL) models that assist job candidates in identifying roles aligned with their skills and aspirations. Drawing from personal experience, we recognize that researching the job market, identifying key skills, and crafting effective resumes is often a labor-intensive process. Moreover, the diversity of self-reported job titles, overlapping skill sets across industries, and unstructured resume formats pose significant challenges to automating these processes.

Our methodology combines classical ML models like Logistic Regression and Random Forest with state-of-the-art DL architectures such as Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs). These models classify and differentiate resumes based on key features such as skills, experience, and education. We implemented robust data preprocessing techniques to standardize resumes, extract skills using Named Entity Recognition (NER), and transform textual data into numerical formats for optimal model performance.

The broader vision of this project extends beyond classification. Future work aims to develop advanced recommendation systems that provide actionable feedback to candidates, enabling them to refine their resumes, bridge skill gaps, and align better with market demands. Furthermore, we propose the creation of a platform for employers to efficiently filter, rank, and classify resumes, streamlining recruitment while reducing biases and enhancing fairness. By combining data-driven insights with scalable AI technologies, this project addresses inefficiencies in traditional recruitment processes and offers innovative solutions for job seekers and recruiters alike. Ultimately, this work lays the groundwork for modernizing recruitment practices, ensuring better job matching, and enhancing the overall hiring experience.

Keywords

Resume Analysis, Machine Learning, Resume Classification, Named Entity Recognition, Natural Language Processing, Skill Extraction, Data Pre-processing, Feature Selection, TF-IDF, Logistic Regression, Random Forest, Support Vector Machine, Decision Trees, XGBoost, Convolutional Neural Networks, Long Short-Term Memory, Hyperparameter Tuning, Class Imbalance, Large Language Models, Job Description, Ranking System, Recommender Systems

1 Introduction

The central aim of this project is to bridge this gap by leveraging data-driven methodologies to create an automated and scalable recruitment system. Specifically, the project focuses on utilizing resume data to extract valuable insights, classify candidates into relevant job categories, and recommend roles tailored to individual profiles. By employing state-of-the-art machine learning and deep learning techniques, we aim to enhance the accuracy, efficiency, and fairness of resume classification and job matching systems. Such automation not only reduces the workload of human recruiters but also provides job seekers with data-driven insights and actionable feedback to improve their employability.

A unique aspect of this work lies in its dual perspective, addressing the needs of both job seekers and recruiters. For job seekers, the system offers personalized job recommendations and targeted suggestions for enhancing their resumes. For recruiters, it automates tedious processes like filtering and ranking resumes, allowing them to focus on strategic decision-making and candidate engagement. Additionally, this system addresses common challenges in resume analysis, such as inconsistent formatting, self-reported job titles, and unstructured text, by implementing robust preprocessing and feature engineering techniques.

This report is structured to provide a comprehensive overview of the project. It begins with an exploration of the dataset, highlighting its characteristics, challenges, and potential for analysis. Subsequent sections detail the methodologies employed, including data preprocessing, feature engineering, and the implementation of classical and deep learning models. A thorough evaluation of model performances, supported by comparative analyses and key metrics, offers insights into the effectiveness of various approaches. Finally, the report

*All authors contributed equally to this research.

concludes with a discussion of the project's implications, key findings, and directions for future work.
By addressing inefficiencies in traditional recruitment and leveraging the power of AI, this project seeks to modernize the hiring process. It aims to create a robust and scalable solution that empowers candidates, streamlines recruiter workflows, and fosters better alignment between skills and opportunities in an increasingly dynamic job market.

2 Related Work

Matching resumes to job descriptions has been a critical focus in recruitment, with advancements in machine learning (ML) and natural language processing (NLP) offering promising solutions. Studies have highlighted the effectiveness of NLP techniques in extracting relevant information and aligning candidates with suitable roles (Tiwari et al., 2024) [7]. Approaches integrating state-of-the-art models such as BERT and SpaCy have achieved high accuracy in parsing and classifying resumes, as well as providing job recommendations (Jaiswal, 2023) [5]. AI-driven systems have demonstrated the ability to streamline recruitment processes by automating skill analysis and improving job-candidate matching, especially in addressing labor market challenges (Gangoda, 2023) [4]. Challenges such as biases in job postings and resumes, as seen in phenomena like "Résumé-Driven Development," underline the importance of developing robust methodologies to ensure relevance and fairness in talent acquisition (Fritzsch et al., 2021) [3].
Several other works have explored the role of NLP in resume analysis and job matching. Bhor et al. (2021) [2] proposed a system to parse resumes using NLP, extracting structured information such as education and experience, and ranking candidates based on company preferences. Irfan et al. (2022) [1] introduced a Resume Classification System (RCS) that leverages NLP and ML techniques to classify resumes into job categories with high accuracy, demonstrating the effectiveness of TF-IDF and SVM classifiers for this task. Tran (2023) [8] highlighted the challenges faced by HR departments in managing large volumes of resumes and presented a solution to improve efficiency in data entry, screening, and matching processes.
Traditional recruitment tools, such as keyword-based search engines and rule-based filters, offer simplicity and ease of use but often fail to capture the nuances of unstructured text. For instance, a recruiter using a keyword-based tool may miss a candidate with relevant skills described in non-standard terms. In contrast, AI-based systems excel in understanding context and semantic relationships, enabling a deeper analysis of resumes and job descriptions. However, these systems require significant computational resources, robust datasets for training, and careful tuning to avoid biases. While traditional methods remain suitable for smaller-scale operations, the growing complexity of recruitment necessitates intelligent, scalable solutions, as demonstrated by the integration of NLP techniques.

These comparisons highlight the trade-offs between simplicity and sophistication in recruitment tools. While traditional methods may still be suitable for small-scale operations, the growing complexity of modern recruitment demands scalable and intelligent solutions, as demonstrated by the integration of NLP techniques.

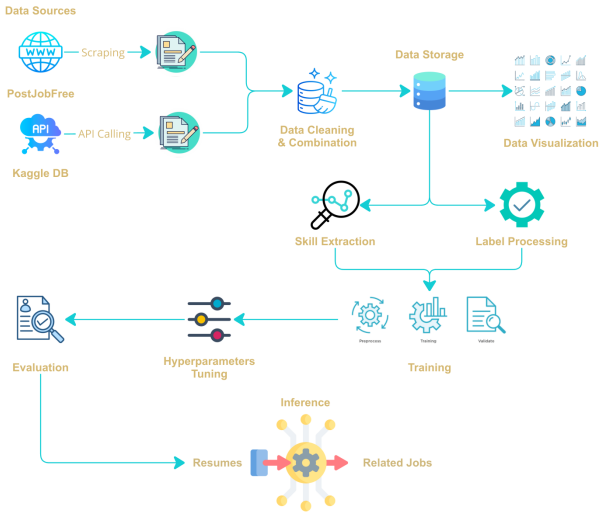
3 Methodology

3.1 Data Understanding and Preprocessing

One of the first steps in our methodology involved understanding and preparing the dataset. Initially, we planned to scrape resume data from Livecareer, and we had set up the required environment and scraping code. However, just as we were close to gathering a significant number of resumes, the website structure changed, rendering our code obsolete. Without an immediate workaround for the updated structure, we decided to pivot to data from PostJobFree, focusing on resumes related to the IT domain. This transition was smooth but resulted in several challenges, including handling duplicate resumes and ensuring data quality.Upon analyzing the collected data, we noticed a large number of duplicate entries and inconsistencies. Additionally, the dataset contained over 6,000 unique job titles due to the self-reported nature of resumes. A significant issue arose from ambiguous or overlapping job titles. For instance, the role of "Full Stack Developer" was represented in various forms such as "fullstack developer," "full-stack developer," "full stack dev," and other variations with different casing or special characters. To address this, we cleaned the data by:

- Removing special characters.
- Converting all text to lowercase.
- Merging similar job titles into standardized categories.

The revised dataset better suited our needs for the resume-job matching task, with titles consolidated into categories like Data Scientist, QA Engineer, and Cloud Engineer.



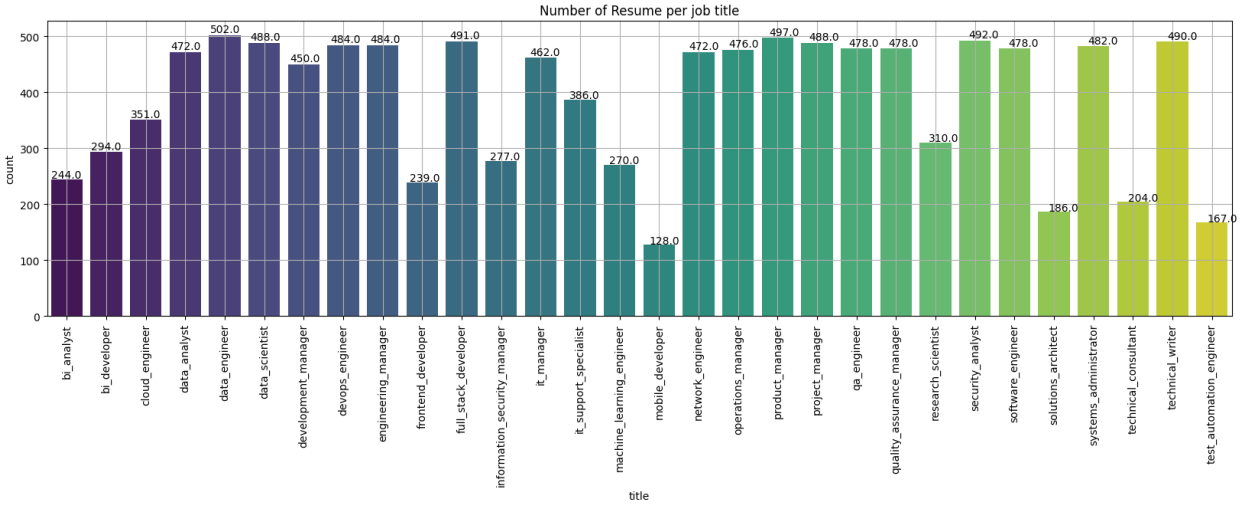


Figure 1: Number of resumes by job title.

Table 1: Comparison Between Traditional Tools and AI-Based Systems

Aspect	Traditional Tools	AI-Based Systems
Efficiency	Moderate; relies on manual filtering and keyword searches.	High; automates filtering and matching processes.
Accuracy	Limited; prone to missing nuanced or non-standard terms.	High; captures semantic context and nuanced text.
Scalability	Limited; struggles with large volumes of data.	High; designed for handling extensive datasets.
Fairness	Risk of implicit bias in rule-based filtering.	Risk of model bias but can be mitigated with proper training.
Implementation Cost	Low; easy to set up and use.	High; requires computational resources and expertise.
Personalization	Low; generic recommendations based on basic filters.	High; tailored recommendations based on deep insights.

Figure 2: Entire project process.

These preprocessing steps reduced the number of unique job titles significantly, yet 3,000 distinct titles remained. To further streamline, we grouped similar titles based on skills and responsibilities, ultimately reducing them to 300 unique titles. These titles were then categorized into six overarching "super titles" to minimize ambiguity and aid in effective modeling.

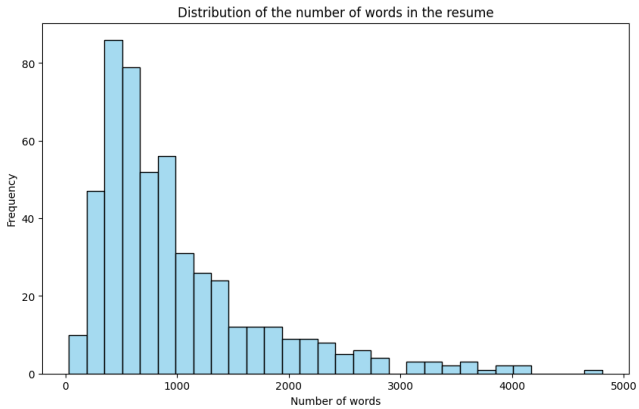


Figure 3: Histogram of resume length by words.

3.2 Feature Engineering and Transformation

Given the textual nature of resume data, skill extraction was critical for feature engineering. To achieve this, we utilized Named Entity Recognition (NER) techniques, as they are highly effective for identifying specific entities like skills, education, and experience within unstructured text. Tools like SpaCy and NLTK were employed to implement NER models, which provided structured insights into the resumes. This process helped us extract key features and skills, reducing the complexity of unstructured resume text while retaining essential information for downstream modeling tasks.

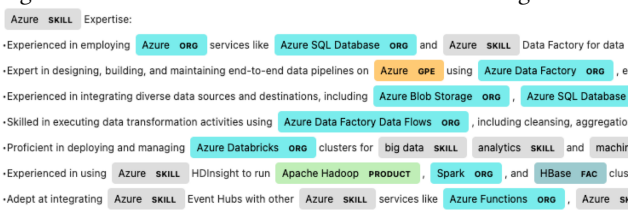


Figure 4: Using Named Entity Recognition to extract information from resume content.

Tokenization and vectorization techniques like word embeddings and TF-IDF were applied to convert text data into a format suitable for machine learning models. This transformation facilitated more effective analysis and model training.

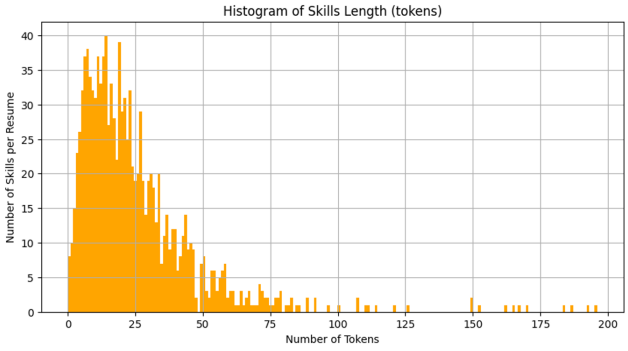


Figure 5: Histogram of skills per resume.

Data was further prepared through one-hot encoding, enhancing model compatibility and efficiency.

	3d	access	accounting	acquisition	active	actuarial	actuator	adaboost
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figure 6: One hot encoding result.

3.3 Modeling Approaches

We explored both classical machine learning (ML) models and deep learning (DL) models for resume classification. Based on our research, we observed that DL models often perform better on text-based data due to their ability to capture contextual relationships and patterns. However, this advantage comes with a trade-off in terms of computational requirements and training time. Classical ML models like Logistic Regression and XGBoost were computationally efficient, requiring less training time and fewer resources. These models delivered competitive results with accuracies of up to 82%. On the other hand, deep learning models such as Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks achieved slightly lower accuracies of around 80% but were more effective in capturing intricate textual patterns. To manage the high computational demands of DL models, we utilized CUDA for GPU acceleration. Despite this optimization, training deep learning models was significantly more time-intensive compared to classical ML models. Ultimately, while DL models offered robust performance, the slight edge in accuracy of classical models, along with their efficiency, made them favorable for this task.

3.4 Model Performance and Limitations

The performance of the models was evaluated using key metrics, including accuracy, precision, recall, and F1-score. Classical models such as Logistic Regression and XGBoost emerged as top performers, achieving an accuracy of 82%. These models were not only efficient but also consistent across various metrics, making them reliable choices for resume classification.

Deep learning models, including CNN and LSTM, demonstrated strong capabilities in handling complex text data and capturing contextual relationships. However, they achieved slightly lower accuracies of 80%. Despite this, their ability to process and learn from intricate text structures positioned them as promising options for future scalability.

One of the primary limitations of the models was overfitting observed in Decision Trees due to their sensitivity to noise and complex patterns in the data. Overfitting led to high training accuracy but poor generalization on unseen data. This was mitigated by employing techniques like pruning and setting depth constraints.

For XGBoost, hyperparameter tuning posed a challenge. While this model consistently delivered high accuracy, fine-tuning parameters like learning rate, max depth, and number of estimators was computationally intensive. This required significant time and resources to identify the optimal configuration.

4 Evaluation and Results

4.1 Evaluation Metrics

To assess the performance of the implemented models, a comprehensive set of metrics was employed, including accuracy, precision, recall, F1-score, Matthews Correlation Coefficient (MCC), and G-Mean.

- **Accuracy:** Measures the proportion of correct predictions (both true positives and true negatives) out of all predictions made by the model.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{Total Instances}}$$

- **Precision:** Evaluates the proportion of correctly predicted positive cases out of all cases predicted as positive. It measures the model's ability to avoid false positives.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- **Recall:** Measures the proportion of actual positive cases that are correctly identified by the model. It focuses on minimizing false negatives.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **F1 Score:** The harmonic mean of precision and recall, providing a balance between the two metrics.

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

- **G-Mean:** Geometric Mean measures the balance between correctly identifying positive cases and correctly rejecting negative cases.

$$\text{G-Mean} = \sqrt{\text{Sensitivity} \cdot \text{Specificity}}$$

- **Matthews Correlation Coefficient (MCC):** Measures the correlation between actual and predicted classifications, providing a balanced metric even for imbalanced datasets.

$$\text{MCC} = \frac{(\text{TP} \cdot \text{TN}) - (\text{FP} \cdot \text{FN})}{\sqrt{(\text{TP} + \text{FP}) \cdot (\text{TP} + \text{FN}) \cdot (\text{TN} + \text{FP}) \cdot (\text{TN} + \text{FN})}}$$

We used a variety of metrics to carefully evaluate how well each model performed. These metrics gave us a clear picture of the models' strengths and weaknesses. For example, we looked at accuracy, precision, and recall to see how often the models made correct predictions. The F1 score helped us understand the balance between precision and recall. We also used advanced metrics like G-Mean and MCC to check how well the models handled imbalanced data. By looking at all these factors, we got a complete view of each model's performance and could choose the best one.

4.2 Baseline Performance

Baseline models, such as Logistic Regression and K-Nearest Neighbors (KNNs), were implemented to establish reference points for performance comparison. Logistic Regression delivered strong initial results, achieving an accuracy exceeding 80%, which highlighted its capability to generalize well across the dataset. In contrast, KNNs faced challenges in handling data complexity and noise, leading to lower precision and recall scores. These baseline results underscored the need for more advanced models to address the nuances of the resume classification task effectively.

4.3 Hyperparameters Tuning

Hyperparameter tuning is the process of finding the best settings (hyperparameters) for a model to improve its performance. These settings, such as the learning rate or maximum depth, are predefined and cannot be learned directly from the data, but they have a big impact on how well the model works.

In our solution, we used RandomizedSearchCV and GridSearchCV methods from the sklearn library. Both methods allow us to test different combinations of hyperparameters for the model. Based on a specific performance metric (like accuracy or F1-score), it identifies the best combination of settings.

Once we find the best parameters, we use them to finalize the model, ensuring it achieves the highest possible performance. GridSearchCV and RandomizedSearchCV also include techniques to reduce overfitting, such as cross-validation. By setting the 'cv' parameter, the model is trained and validated on multiple random data splits (folds). This ensures the model learns from all the data while avoiding overfitting or focusing too much on any single subset.

Model	Baseline	Hyper-tuned
Logistic Regression	0.8	0.82
KNeighborsClassifier	0.74	0.78
DecisionTreeClassifier	0.68	0.72
RandomForestClassifier	0.78	0.78
SVC	0.8	0.8
GradientBoostingClassifier	0.79	0.8
AdaBoostClassifier	0.67	0.74
XGBClassifier	0.8	0.82
LSTM	0.76	0.76
CNN	0.8	0.8

Table 3: F1 Scores for Different Models Before and After Hyperparameter Tuning

The table shows the F1 scores of various models before and after hyperparameter tuning. Generally, most models showed slight improvements after tuning, with models like Logistic Regression and XGBClassifier experiencing notable gains, while others, such as SVC and CNN, did not show any change in performance.

4.4 Model Comparisons

Among the classical machine learning models, Logistic Regression and XGBoost emerged as the top performers, with accuracies of 82% and 80%, respectively. These models consistently provided balanced results across metrics such as precision, recall, F1-score, and MCC, demonstrating their effectiveness for this classification task.

Table 2: Model Performance Metrics

Model	Accuracy	Precision	Recall	F1-Score	G-Mean	MCC
Logistic Regression	0.82	0.82	0.82	0.82	0.89	0.75
KNeighborsClassifier	0.78	0.78	0.78	0.78	0.87	0.70
DecisionTreeClassifier	0.72	0.72	0.72	0.72	0.83	0.61
RandomForestClassifier	0.78	0.78	0.78	0.78	0.86	0.70
SVC	0.80	0.80	0.80	0.80	0.88	0.73
GradientBoostingClassifier	0.80	0.80	0.80	0.80	0.88	0.72
AdaBoostClassifier	0.74	0.74	0.74	0.74	0.84	0.64
XGBClassifier	0.82	0.82	0.82	0.82	0.89	0.76
LSTM	0.76	0.76	0.76	0.76	0.85	0.72
CNN	0.80	0.80	0.80	0.80	0.88	0.73

Gradient Boosting and Random Forest were competitive, achieving accuracies around 79%, though their performance in precision and recall lagged slightly behind the leading models. Simpler models, such as Decision Tree and AdaBoost, displayed weaker performance, with accuracies ranging from 73% to 74%, likely due to over-fitting and their sensitivity to noisy data.

Deep learning models, including Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs), exhibited strong capabilities in handling the complexities of the dataset. LSTM excelled at capturing sequential dependencies in textual data, while CNN effectively extracted meaningful features through its hierarchical structure. Although these models required more computational resources, their ability to process large-scale data effectively positioned them as promising candidates for future implementations.

4.5 Key Insights

Hyperparameter tuning proved pivotal in optimizing model performance. Techniques like RandomizedSearchCV and GridSearchCV enhanced the accuracy of Logistic Regression, Gradient Boosting, and XGBoost by fine-tuning parameters such as learning rate, maximum depth, and the number of estimators. Cross-validation ensured that the models generalized well across diverse subsets of the data, mitigating the risk of overfitting.

Logistic Regression and XGBoost stood out as the most reliable models, balancing strong performance metrics with computational efficiency. These models effectively addressed data inconsistencies and imbalances, outperforming simpler models such as Decision Tree and KNN. While deep learning models provided promising results, further exploration is required to fully leverage their potential for large-scale implementations.

Our project emphasized the critical importance of feature selection and data preprocessing in machine learning pipelines. The high imbalance in our dataset, coupled with the diverse and self-reported nature of job titles, highlighted the challenges of working with untrustworthy data sources. For instance, the dataset initially contained over 6,000 unique job titles due to inconsistencies in how candidates reported their roles. Through rigorous preprocessing—such as standardizing titles, removing noise, and consolidating overlapping roles—we reduced these to 300 titles, grouped under six overarching categories. This transformation was essential for improving model interpretability and performance.

A deeper analysis revealed several interesting patterns:

- **Impact of Feature Selection:** The lack of a sufficient number of diverse and meaningful features affected the performance of deep learning models. With our current feature set, classical models like Logistic Regression and XGBoost performed better. However, we observed that deep learning models might outperform if additional features, such as text embeddings or external knowledge graphs, were incorporated.
- **Effectiveness of Feature Engineering:** Proper standardization and grouping of job titles, combined with skill extraction using Named Entity Recognition (NER), significantly improved the quality of the dataset. This not only reduced noise and redundancy but also ensured that the models trained on cleaner, more meaningful data.
- **Effectiveness of Word Embeddings:** Word embeddings, such as TF-IDF and one-hot encoding, played a crucial role in capturing semantic relationships within the resume text. Models leveraging these representations showed higher recall and precision compared to those relying on raw text.
- **Computational Trade-Offs:** Classical models like Logistic Regression and Random Forest demonstrated shorter training times and lower computational costs. In contrast, deep learning models, such as CNNs and LSTMs, required significantly more resources but showed potential for scalability and handling complex textual patterns.

- **Performance on Overlapping Job Titles:** Overlapping roles, such as "Data Scientist" and "Data Engineer," posed challenges for the models. Despite extensive preprocessing, the classifiers occasionally miscategorized these roles due to subtle skill variations.
- **Class Imbalances:** Imbalanced datasets, with some job titles being overrepresented, influenced model performance. While metrics like precision and recall improved with weighted loss functions, achieving consistent accuracy across all job categories remains an area for further optimization.

The evaluation of deep learning models highlighted their potential for sequential and contextual text analysis. LSTM networks, for instance, excelled in processing long sequences of text, making them particularly effective for resumes with detailed descriptions. CNNs, on the other hand, showcased their strength in extracting hierarchical features, especially in identifying keywords and patterns. However, in our case, the limited number of features restricted their ability to outperform classical models. Expanding the dataset to include additional features, such as candidate achievements, project descriptions, or even sentiment analysis, could enable deep learning models to achieve better results.

These insights underline the importance of balancing computational efficiency, model complexity, and dataset quality to achieve optimal performance. The project's findings pave the way for developing more robust, scalable solutions in resume classification and job matching.

5 Application

Following the previous completion, which mostly focused on the resume classification task, we takes a step further to enhance its reliability and practical applicability by developing an end-to-end application. The application streamlines the process of handling new resumes, classifying them, and recommending job descriptions based on relevance.

For a quick and effective matching algorithm, we chose Term Frequency-Inverse Document Frequency (TF-IDF). In this approach, the extracted skills from the resume are used as input to compare against the skills required in the job description. Theoretically, TF-IDF (Term Frequency-Inverse Document Frequency) is a statistical method used to evaluate the importance of a word (or term) within a document relative to a collection of documents (corpus). For our purpose, it is applied to match resumes and job descriptions by analyzing the extracted skills from both sources.[6]

- **Term Frequency (TF):** Measures how often a term (skill) appears in a specific document.

$$TF(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$$

  Where $f_{t,d}$ is the frequency of term t in document d, and the denominator is the total number of terms in d.

- **Inverse Document Frequency (IDF):** Reduces the weight of common skills that appear in many documents across the corpus.

$$IDF(t) = \log\left(\frac{N}{1+n_t}\right)$$

  Where N is the total number of documents in the corpus, and n_t is the number of documents containing term t.
- **TF-IDF Score:** Combines TF and IDF to calculate the significance of a skill for matching purposes.

$$TF\text{-}IDF(t,d) = TF(t,d) \times IDF(t)$$

To match new resume content with job descriptions, we follow these steps:

(1) Processing New Resume Content: Clean the text and extract relevant skills.
(2) Resume Classification: Use trained models to identify the best-matching job titles based on the candidate's skills.
(3) Filtering Jobs by Titles: Narrow down job descriptions to those that match the predicted titles.
(4) Matching Using TF-IDF: Apply the TF-IDF algorithm to compare the skills from the resume with those required in the job descriptions and generate a ranking of the best matches.

The figure below shows an example of the TF-IDF output. The most relevant job is suggested to the candidate.

	id	title	score
346	vh7xdn_data-scientist-atmospheric-canada	Data Scientist	0.308377
297	vibzf9_data-scientist-oakville-on	Data Scientist	0.289100
317	vii2s4_data-scientist-the-san-francisco-ca	Data Scientist	0.285358
330	venwzf_data-scientist-rockville-md	Data Scientist	0.285284
262	vii0fs_sr-staff-data-scientist-myrtle-point-or	Data Scientist	0.280223
339	vhy7ln_software-engineer-and-data-lexington-ma	Data Scientist	0.277594
360	vhel1u_lead-data-scientist-toronto-on-canada	Data Scientist	0.269674
277	vh4mo5_senior-data-scientist-east-whiteland-pa	Data Scientist	0.269249
244	vf07hl_data-scientist-atlanta-ga-30324	Data Scientist	0.265500
209	vdaf40_data-scientist-los-angeles-ca	Data Scientist	0.263458

Figure 7: TF-IDF results for job suggestion based on resume skills.

6 Conclusion

This project successfully demonstrated the potential of automated systems in addressing the challenges associated with resume classification. By leveraging data-driven techniques, we implemented and evaluated a range of machine learning (ML) and deep learning (DL) models. Logistic Regression and XGBoost emerged as the most effective approaches, achieving accuracies exceeding 80%. These models balanced key performance metrics, including precision, recall, F1-score, and computational efficiency, proving to be reliable tools for handling real-world recruitment challenges. Additionally, deep learning models such as LSTM and CNN exhibited strong capabilities in capturing complex patterns and

sequential dependencies, highlighting their potential for future scalability and adaptability.

The system developed in this project serves as a comprehensive framework for streamlining recruitment processes. By automating resume classification, it significantly reduces recruiter workloads, enhances decision-making efficiency, and improves the relevance of job recommendations. For job seekers, the system provides actionable insights to refine their resumes, align their skills with market demands, and make informed career choices. This dual perspective ensures that both recruiters and candidates benefit from a more efficient and equitable recruitment process.

Furthermore, the project demonstrated the critical role of feature engineering and data preprocessing in determining model success. From cleaning and standardizing resume data to extracting meaningful features, these steps laid the groundwork for accurate and interpretable model predictions. The use of advanced techniques like Named Entity Recognition (NER) and word embeddings, combined with robust preprocessing, ensured that the system could effectively handle unstructured text data and derive insights from it. This highlights the importance of investing time and resources in data preparation when developing machine learning systems.

However, the project also underscored certain limitations. Deep learning models, while capable of capturing complex patterns, were constrained by the limited number of features in the dataset. Expanding the dataset to include additional dimensions, such as detailed project descriptions, quantified achievements, or sentiment analysis of cover letters, could unlock the full potential of these models. Additionally, addressing challenges like overlapping job titles and dataset imbalances remains an important area for further research and optimization.

In conclusion, this work not only addresses current challenges in the recruitment domain but also opens the door for future innovations. Enhancements such as real-time resume evaluation, the integration of advanced NLP models like BERT or GPT, and broader dataset inclusion from diverse industries can transform this system into a robust, scalable platform capable of adapting to the dynamic needs of modern recruitment. Furthermore, integrating explainable AI techniques can ensure transparency and trust in the system's recommendations, a crucial factor for adoption by both recruiters and job seekers.

The methodologies and findings established through this project lay a strong foundation for developing interactive, scalable recruitment solutions that empower candidates and streamline recruiter workflows. By bridging the gap between job seekers and employers, the system has the potential to democratize access to high-quality job recommendations and foster a more inclusive labor market.

7 Future Work

While this project successfully demonstrated the feasibility and effectiveness of automated resume classification, there are several areas for improvement and extension:

- Incorporating Advanced NLP Models: Future iterations can leverage transformer-based models, such as BERT or GPT, to capture contextual nuances in resumes and job descriptions. These state-of-the-art models have the potential to significantly enhance the accuracy, interpretability, and overall performance of the classification system.
- Improving Overall Accuracy and Predictions: Further parameter tuning is necessary to enhance prediction accuracy, particularly for differentiating similar job titles with overlapping skills. This requires training the model on additional features beyond skills to better segregate job titles.
- Expanding Dataset Diversity: The current dataset focuses on Information Technology and Computer Science domains. Expanding the dataset to include resumes and job descriptions from other fields would improve the system's generalizability and robustness, enabling it to cater to a wider range of industries.
- Real-Time Resume Evaluation System: Developing an interactive, real-time system for candidates to receive instant feedback on their resumes could significantly enhance user engagement. Such a system could offer actionable recommendations for skill improvement and tailored job matching.
- Prediction Based on Job Descriptions: Implementing a model trained on job descriptions would enable matching resumes to specific job requirements. This approach could also help identify what an ideal resume for a given job description should include, providing both candidates and recruiters with valuable insights.
- Improving Computational Efficiency: Deep learning models, while effective, are computationally intensive. Optimizing these models or exploring lightweight alternatives could make the system more accessible and scalable for broader applications.
- Incorporating Employer Features: Future enhancements could include features that allow employers to upload multiple resumes and generate rankings to better match job descriptions. This would provide value to recruiters and streamline their decision-making processes.

Most of these enhancements are interconnected, such as resume ranking and job description matching. Implementing these improvements would not only increase the system's accuracy and usability but also establish it as a comprehensive and scalable tool for addressing modern recruitment challenges.

References

[1] Irfan Ali, Nimra Mughal, Zahid Hussain Khand, Javed Ahmed, and Ghulam Mujtaba. 2022. *Mehran University Research Journal Of Engineering Technology*, 41, 1, 65–79. https://search.informit.org/doi/10.3316/informit.263278216314684.

[2] Shubham Bhor, Vivek Gupta, Vishak Nair, Harish Shinde, and Manasi S Kulkarni. 2021. Resume parser using natural language processing techniques. *Int. J. Res. Eng. Sci*, 9, 6.

[3] Jonas Fritzsch, Marvin Wyrich, Justus Bogner, and Stefan Wagner. 2021. Résumé-driven development: a definition and empirical characterization. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*, 19–28. DOI: 10.1109/ICSE-SEIS52602.2021.00011.

[4] Nikethani Gangoda, Kavindu Piumal Yasantha, Chamina Sewwandi, Navindu Induvara, Samantha Thelijjagoda, and Nishantha Giguruwa. 2024. Resume ranker: ai-based skill analysis and skill matching system. In *2024 Sixth International Conference on Intelligent Computing in Data Sciences (ICDS)*, 1–8. DOI: 10.1109/ICDS62089.2024.10756304.

[5] Gautam Jaiswal, Aryan Uttam, Devesh Dhar Dubey, and Pawan Kumar Mall. 2024. Resume analyser and job recommendationsystem based on nlp. In *2024 2nd International Conference on Disruptive Technologies (ICDT)*, 1584–1587. DOI: 10.1109/ICDT61202.2024.10489058.

[6] Claude Sammut and Geoffrey I. Webb, (Eds.) 2010. *Tf–idf. Encyclopedia of Machine Learning*. Springer US, Boston, MA, 986–987. ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_832.

[7] Aniket Tiwari, Sonali Nalamwar, and S.M.P.S. 2024. A review of resume analysis and job description matching using machine learning. *International Journal on Recent and Innovation Trends in Computing and Communication*, 12, 2, (May 2024), 247–250.

[8] Thanh Tung Tran, Truong Giang Nguyen, Thai Hoa Dang, and Yuta Yoshinaga. 2023. Improving human resources' efficiency with a generative ai-based resume analysis solution. In *Future Data and Security Engineering. Big Data, Security and Privacy, Smart City and Industry 4.0 Applications*. Tran Khanh Dang, Josef Küng, and Tai M. Chung, (Eds.) Springer Nature Singapore, Singapore, 352–365. ISBN: 978-981-99-8296-7.