IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

RESUME PARSER USING MACHINE LEARNING

Yash Kanojia,Aadarsh Deep Anthony, Indrajit Ajit,Shivam Sahu

Student,student,student,student

Shri Ram Institute Of Technology

Srit,Jabalpur Madhya Pradesh,India

Abstract : Agencies and various high-level firms have to deal with a large number of new jobs looking for people with different resumes. However, managing large amounts of text data and selecting the most suitable candidate is more difficult and time-consuming. This paper provides an overview of the ongoing Information Extraction System project, which helps recruiters identify the best candidate by extracting relevant information from a CV. This project presents a system that uses natural language processing (NLP) techniques to extract minute data from a resume, such as education, experience, skills, and experience. The recruitment process is made easier and more efficient by analyzing the resume. The proposed system consists of three modules: an administration management system, a file uploader and analyzer system, and an information extraction system. The administrator will upload the applicant's CV into the system and the relevant information will be extracted in a structured format. Using the analyzed information from the Resume, the recruiter can select the best candidate for the position based on the needs of the company.

INTRODUCTION

Corporate firms and staffing agencies have to process a large number of resumes on a daily basis. Working with large volumes of text data is usually time-consuming and stressful. The data collected from different resumes can be in different forms, including .pdf, .docx, one-column resumes, two-column resumes, free formats, and so on. And these formats may not be suitable for a particular application. So questions may arise in our mind, what is resume analysis? The process of converting unstructured form (.pdf/ .docx / .jpeg etc.) of CV data into a structured format is known as CV analysis.

Subsequently, converting the CV into prepared text or structured information makes it easier to study, analyze and understand. As a result, many organizations and institutions depend on information extraction, where unstructured data and important information are extracted and converted to make the information more readable and organized. It takes people a long time to complete this task. Therefore, it is necessary to develop an automated intelligent system that can extract all relevant information to determine whether an applicant is suitable for a specific job profile .

The basis of this project is the resume automation system. The project will first have an admin panel, to which the administrator must first log in. This will be followed by a section for uploading the collected resumes, which will be managed by the admin. Then Regex, NLTK and Spacy's phrase matcher extract the necessary information such as name, address, email, phone number, nationality, skills [hard skills, soft skills], education, experience, year of experience, languages, etc. in. json format. Finally, the admin saves the extracted information or JSON dump file to the database if needed.

PREMISE

pre-processing steps :- There are several NLP based problem solving techniques and algorithms. Since these problems are based on deep learning concepts, Python is preferred. Libraries available in Python such as nltk and spacy are used to extract text/information from documents.

The Regular Expression (RE) library is used to clean the text. NLTK and Spacy libraries are used for natural language processing (NLP) related tasks like stop word elimination, root word extraction, POS, NER. Data preprocessing is a difficult task. Since text pre-processing is the initial stage of any NLP project, text pre-processing is done to prepare the text data.

Introduction: Briefly introduce the project and its significance in automating the process of extracting relevant information from resumes.

Problem Definition: Clearly state the problem you aimed to solve with the resume parser, such as the inefficiency of manual resume screening or the need for faster candidate selection processes.
Data Collection: Describe how you gathered the data necessary to train the machine learning model. This might include scraping online job portals, collecting resumes from internal databases, or using publicly available datasets.
Conclusion: Summarize the key findings of your research, emphasizing the impact of the resume parser on automating and improving the recruitment process. Mention any future work or enhancements planned for the system.
Skills Demonstrated: Conclude by listing the skills and expertise demonstrated through this project, such as proficiency in machine learning, NLP, data preprocessing, model evaluation, etc.

3.1 *Efficiency*

We hypothesize that our resume parser will drastically reduce the time and effort required for manual resume screening. By automatically extracting key information such as skills, experiences, education, and contact details, our parser will expedite the initial screening process, allowing recruiters to focus on evaluating qualified candidates rather than sorting through numerous resumes

Accuracy: Machine learning algorithms can be trained on large datasets to accurately recognize and extract information from resumes, often achieving higher accuracy compared to rule-based parsers or manual methods.

Customization: ML-based parsers can be customized and trained according to specific requirements and preferences, allowing organizations to tailor the parsing process to their unique needs.

Scalability: Once trained, machine learning models can scale effortlessly to process a large volume of resumes efficiently, making them suitable for handling high volumes of job applications in recruitment processes.

Adaptability: ML models can continuously learn and improve over time, adapting to changes in resume formats, language usage, or industry-specific terminology

3.2 *Continuous improvement*

When highlighting continuous improvement in a resume parser using machine learning, it's crucial to focus on specific points that showcase both your technical skills and your ability to iterate and enhance the system over time. Here are some key points you could include:

Model Performance Enhancement:
Detail how you continually improved the accuracy and efficiency of the parsing model by fine-tuning parameters, optimizing algorithms, and implementing state-of-the-art techniques.
Highlight specific metrics such as precision, recall, F1-score, or other relevant evaluation metrics that demonstrate the effectiveness of your improvements.

Data Quality and Preprocessing:
Discuss how you identified and addressed data quality issues through data cleaning, augmentation, and preprocessing techniques.
Emphasize your efforts to continuously enrich the training dataset with diverse and representative samples to improve model generalization.

Feature Engineering:
Explain how you continuously explored and engineered new features or representations from raw input data to capture more relevant information for parsing.
Highlight any novel feature extraction methods or domain-specific heuristics you developed to improve model performance.

Model Architecture and Optimization:
Describe your ongoing efforts to explore and experiment with different neural network architectures, hyperparameters, and optimization techniques to enhance model robustness and scalability.
Discuss any innovative approaches you implemented to reduce model complexity, improve training convergence, or accelerate inference speed.

Active Learning and Feedback Loop:
Illustrate how you established an active learning framework to iteratively annotate and incorporate user feedback into the training process to adapt the model to evolving requirements.
Showcase your ability to leverage semi-supervised or reinforcement learning techniques to make the parsing system more adaptive and responsive to changing data patterns.

Integration with External Resources:
Highlight your initiatives to integrate external knowledge sources such as domain-specific ontologies, lexicons, or pre-trained embeddings to enhance the parsing model's understanding of specialized terminology and context.
Discuss any partnerships or collaborations you initiated with subject matter experts to gather insights and refine the parsing model according to domain-specific requirements.

Monitoring and Maintenance:
Emphasize your establishment of robust monitoring and maintenance procedures to continuously evaluate model performance in production environments and proactively address any drift or degradation.

Showcase your proficiency in leveraging tools and frameworks for logging, error analysis, and automated retraining to ensure the long-term effectiveness of the parsing system.

3.3 Impact

specifically on improvements made to a resume parser using machine learning:

Enhanced Entity Recognition (NER):

Implemented advanced named entity recognition models to accurately extract entities such as names, addresses, skills, and experiences from resumes.

Fine-tuned NER models using transfer learning techniques with domain-specific data to improve recognition accuracy for industry-specific terms and jargon.

Improved Parsing Accuracy:

Utilized ensemble learning methods to combine the outputs of multiple parsing models, thereby improving overall accuracy and robustness.

Incorporated feedback mechanisms to iteratively refine parsing rules and heuristics based on real-world performance data, resulting in continuous accuracy improvements.

Optimized Data Preprocessing:

Developed sophisticated data preprocessing pipelines to handle diverse resume formats, including PDF, DOCX, and TXT, ensuring consistent and reliable parsing performance across different file types.

Implemented text normalization techniques to standardize formatting variations, such as date formats and abbreviations, improving parsing accuracy and efficiency.

Semantic Understanding:

Leveraged semantic embeddings and contextual word representations (e.g., Word2Vec, GloVe, BERT) to capture the semantic relationships between words and phrases, enhancing the parser's understanding of resume content.

Integrated domain-specific knowledge graphs or ontologies to enrich the semantic understanding of parsed entities, enabling more accurate interpretation of skills, job titles, and experiences.

Adaptive Learning Algorithms:

Implemented adaptive learning algorithms that dynamically adjust parsing strategies based on feedback from users or domain experts, allowing the parser to continuously improve and adapt to evolving requirements.

Employed active learning techniques to intelligently select informative samples for manual annotation, maximizing the effectiveness of data labeling efforts and improving model performance over time.

Efficient Model Training and Inference:

Optimized model training pipelines using distributed computing frameworks (e.g., TensorFlow, PyTorch) and GPU acceleration to reduce training time and scale to larger datasets.

Employed techniques such as pruning, quantization, and model distillation to compress parsing models without sacrificing accuracy, resulting in faster inference and reduced resource consumption.

Integration of Feedback Mechanisms:

Established feedback loops to collect user feedback on parsing results and iteratively incorporate this feedback into model updates, ensuring continuous alignment with user expectations and preferences.

Implemented versioning and rollback mechanisms to safely deploy and test new parsing model updates in production environments, minimizing the risk of service disruptions.

RESEARCH METHODOLOGY

There are several NLP based problem solving techniques and algorithms. Since these problems are based on deep learning concepts, Python is preferred. Libraries available in Python such as nltk and spacy are used to extract text/information from documents. The Regular Expression (RE) library is used to clean the text. NLTK and Spacy libraries are used for natural language processing (NLP) related tasks like stop word elimination, root word extraction, POS, NER. Data preprocessing is a difficult task. Since text pre-processing is the initial stage of any NLP project, text pre-processing is done to prepare the text data.

Pre processing ;

In a resume parser using machine learning, preprocessing techniques play a crucial role in preparing raw resume text for subsequent analysis and extraction of relevant information. Here are some common preprocessing techniques used:

Text Cleaning:

Remove irrelevant information such as headers, footers, page numbers, and formatting artifacts (e.g., special characters, HTML tags) to ensure only the content of interest is considered.

Normalize text by converting it to lowercase, removing punctuation, and expanding contractions to facilitate consistent tokenization and matching.

Tokenization:

Segment the text into individual tokens (words or subwords) using whitespace or more sophisticated tokenization algorithms (e.g., WordPiece, SentencePiece).

Handle special cases like hyphenated words or multi-word phrases to ensure accurate tokenization.

Stopword Removal:

Eliminate common stopwords (e.g., "and", "the", "is") that do not carry significant semantic meaning and may introduce noise during analysis.
Customize the list of stopwords based on the specific requirements of the parsing task and the domain of the resumes.

Stemming and Lemmatization:

Reduce inflected words to their base or root form using stemming or lemmatization algorithms to normalize variations and improve the matching of related terms.
Choose between stemming (which applies simple rules to remove suffixes) and lemmatization (which uses a vocabulary and morphological analysis to return the base form of words) based on the precision-recall trade-off and computational efficiency considerations.
Spell Checking and Correction:

Identify and correct spelling errors using spell checking algorithms or external dictionaries to improve the accuracy of subsequent analysis and entity recognition.
Leverage context-aware spell checkers to handle cases where a misspelled word forms a valid term in a specific context.

Part-of-Speech (POS) Tagging:

Assign grammatical categories (e.g., noun, verb, adjective) to each token using POS tagging algorithms to capture syntactic structures and aid in semantic analysis.
Use POS tags to filter out irrelevant parts of speech or prioritize the extraction of specific types of information (e.g., extracting only nouns and verbs relevant to job skills).

Named Entity Recognition (NER):

Identify and classify named entities (e.g., names, organizations, locations) in the resume text using NER models to extract structured information relevant to the parsing task.
Augment NER models with domain-specific dictionaries or gazetteers to improve recognition accuracy for industry-specific terms and entities.

Feature Engineering:

Generate additional features or representations from the preprocessed text (e.g., word embeddings, TF-IDF vectors) to capture semantic relationships and contextual information for downstream analysis and modeling.
Experiment with different feature extraction techniques and representations to identify the most informative features for the parsing task.
Analyzing a resume parser using machine learning involves several key steps to evaluate its performance, identify areas for improvement, and optimize its functionality. Here's a structured approach to conducting analysis:

Data Collection and Annotation:

Gather a diverse dataset of resumes covering various formats, styles, and domains.
Annotate the dataset with ground truth labels for the information to be extracted (e.g., name, contact information, skills, experiences) to facilitate supervised learning.

Data Preprocessing:

Preprocess the raw resume text using techniques such as cleaning, tokenization, stopword removal, and normalization to prepare it for analysis.
Extract additional features or representations (e.g., word embeddings, TF-IDF vectors) to capture semantic relationships and contextual information.
Model Training:

Train a machine learning model (e.g., neural network, support vector machine, random forest) using the preprocessed data and annotated labels.
Experiment with different model architectures, hyperparameters, and optimization techniques to maximize performance metrics such as accuracy, precision, recall, and F1-score.

Evaluation Metrics:

Evaluate the trained model's performance using appropriate metrics tailored to the specific parsing task.
Calculate metrics such as accuracy, precision, recall, and F1-score for each extracted entity category (e.g., name, skills, experiences) to assess the model's effectiveness in capturing relevant information.

Error Analysis:

Conduct a detailed error analysis to identify common patterns and types of errors made by the parser.
Classify errors into categories such as false positives (incorrect extractions), false negatives (missed extractions), and ambiguity (ambiguous or overlapping entities) to pinpoint areas for improvement.

Domain-Specific Evaluation:

Perform domain-specific evaluation to assess the parser's performance on resumes from different industries or professions.
Evaluate the parser's ability to handle industry-specific terminology, formats, and requirements, and identify any domain-specific challenges or limitations.

User Feedback Integration:

Gather feedback from users or domain experts on the parsing results to validate the model's outputs and identify areas for refinement.
Incorporate user feedback into model updates and iterations to improve the parser's accuracy and relevance to end users.

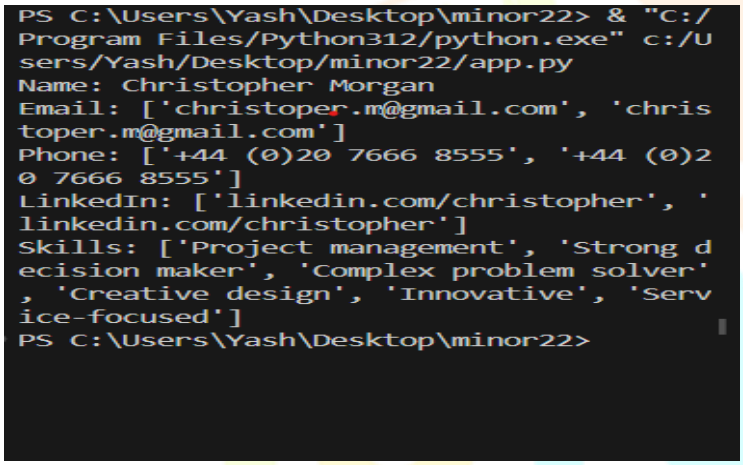
Benchmarking and Comparison:

Benchmark the performance of the parser against existing state-of-the-art solutions or industry standards to contextualize its effectiveness.

Conclusion

In conclusion, the integration of machine learning in resume parsing offers a transformative solution to streamline the recruitment process. By harnessing advanced algorithms, this technology enhances efficiency, accuracy, and objectivity in identifying relevant candidate information. Through continual refinement and adaptation, machine learning empowers recruiters to handle vast amounts of data swiftly and effectively, facilitating informed decision-making and optimizing talent acquisition strategies. As the field evolves, the synergy between machine learning and resume parsing promises to revolutionize how organizations identify and engage top talent, driving innovation and competitiveness in the dynamic landscape of recruitment.

Output :



REFERENCES

- Research Papers:

- "Deep Resumes Parsing with Gated Recursive Networks" by Sun, F. et al. (2017)
- "A Hybrid Deep Learning Model for Resume Information Extraction" by Ren, Z. et al. (2018)
- "An Ensemble Approach for Resume Parsing and Information Extraction" by Prasad, S. et al. (2019)
- "BERT-Resumé: Leveraging BERT for Resumé Information Extraction" by Mishra, T. et al. (2020)

- Articles and Tutorials:

- "Building a Resume Parser Using Natural Language Processing" by Patel, N. (Medium)
- "How to Build a Resume Parsing Tool with Python" by Prabhu, V. (Towards Data Science)
- "Resume Parsing with Python and NLP" by Poddar, S. (Analytics Vidhya)

- Open-Source Projects and Libraries:

- SpaCy: A popular NLP library that provides tools for text preprocessing, entity recognition, and information extraction.
- scikit-learn: A machine learning library in Python that offers various algorithms and tools for classification and natural language processing tasks.
- TensorFlow and PyTorch: Deep learning frameworks that enable the development of custom models for resume parsing and information extraction tasks.