TEAM NLP- FINAL REPORT RESUME EXTRACTION

Nilsu Bozan - bozannilsu@gmail.com - Binghamton University/Istanbul Technical University

Nishchay Vaid - Nishchay89@gmail.com - Rutgers University

Anish Mitra - anishmitra9666@gmail.com - Montana State University

Sukriti Macker - sm11017@nyu.edu - New York University

Specialization: NLP

Github Repo Link: https://github.com/nilsubozan/Resume-Extraction.git

Problem Description

Named entity recognition is a natural language processing system that involves extracting and identifying the important information from the text while discarding superfluous information. It is an information extraction subtype. It takes unstructured text and breaks it down into person names, organizations, locations, medical codes, time expressions, quantities, monetary values, percentages, etc.

Implementing named entity recognition algorithms, we hope to reduce these difficulties involved in applying for a job and thus streamline the process in the human resources industry. We have further context on how to improve the human resources application industry below.

Business Understanding

One of the industries where named entity recognition is most important is parsing resumes with very different formats and inputting the important information into the company database. This is something that sites with widespread usage such as workday do not do accurately and is very frustrating to both job seekers and human resources professionals and also the problem that we are trying to ameliorate for them.

Project Lifecycle

1. Introduction:

The project plan outlines the approach and timeline for the "Resume Extraction" project. The objective of this project is to automate the process of extracting relevant information from resumes using Named Entity Recognition (NER) techniques. Resumes often contain an abundance of information that needs to be manually processed by HR professionals, making the shortlisting task time-consuming and challenging. By leveraging NER models in Natural Language Processing (NLP), we aim to classify entities such as person names, college names, academic information, relevant experiences, skill sets, and more, thereby simplifying the resume screening process.

The project plan is divided into seven weeks, each focusing on specific tasks and deliverables. The plan begins with problem understanding, data collection, and data annotation. Then, it progresses to implementing NER models, training them on annotated data, and evaluating their performance. The subsequent weeks involve developing a Flask application for model deployment, testing the application for inference, and refining the model based on user feedback. Finally, the plan concludes with final reporting, including a project overview, methodology, results, and future recommendations.

This project plan provides a structured approach to effectively tackle the resume extraction problem, leveraging NER techniques to automate the shortlisting process and enhance HR efficiency.

2. Project Plan: Resume Extraction

- Duration: 7 weeks

Week 7: Problem Understanding and Data Collection

- Define the problem statement and objectives of the project.
- Conduct research on existing NER models and techniques for resume extraction and understand the given dataset
- Set up a version control system and create a project repository.

Deliverables: Problem statement, business understanding and objectives document.

Week 8: Data Preprocessing

- Define the entity categories to be extracted, such as person names, college names, academic information, experiences, skill sets, etc.
- Annotate the collected resumes with the corresponding entity labels.
- Ensure consistency and quality in the annotation process.

Deliverables: Data Preprocessing dataset of resumes.

Week 9: Named Entity Recognition (NER)

- Explore different NER techniques and models suitable for resume extraction.
- Preprocess the annotated dataset, including tokenization, normalization, and handling of special characters.
- Split the preprocessed dataset into training, validation, and testing sets.
- Implement and train an NER model using the training dataset, considering architectures like Bidirectional LSTM-CRF or Transformer-based models.
- Fine-tune the model's hyperparameters and optimize its performance.

Deliverables: Trained NER model, preprocessed dataset split into training, validation, and testing sets.

Week 10: Performance Evaluation & Reporting

- Evaluate the NER model's performance using the validation dataset, measuring metrics such as precision, recall, and F1-score.
- Analyze the model's strengths and weaknesses, and iteratively improve it by adjusting the architecture, hyperparameters, or training strategy.
- Generate a performance evaluation report, summarizing the results and providing insights into the model's performance and limitations.

Deliverables: Model evaluation results, Performance evaluation report.

Week 11: Model Deployment

- Develop a Flask web application that allows users to upload resumes for information extraction using the trained NER model.
- Implement the necessary backend functionality for processing resume files and applying the NER model for entity extraction.
- Set up a deployment environment (e.g., Heroku) and deploy the Flask application to a web server.

Deliverables: Flask application code, deployed web application.

Week 12: Model Inference and Refinement

- Test the deployed web application by uploading sample resumes and extracting entities using the NER model.
- Collect user feedback and incorporate improvements based on usability and performance.
- Perform any necessary refinements or enhancements to the NER model based on the insights gained from real-world usage.

Deliverables: Refined Flask application, Updated NER model (if applicable).

Week 13: Final Reporting and Submission

- Prepare the final project report, including an overview, methodology, results, and future recommendations.
- Create a presentation summarizing the project's objectives, approach, key findings, and potential applications.
- Practice and refine the presentation.
- Submit the final project report, along with the code and presentation.

Deliverables: Final project report, presentation slides.

Data Understanding

.json file provides an understanding of the individual’s professional background, skills, work experience and educational qualifications. It can be useful for assessing the candidate’s suitability for specific job roles, evaluating their expertise in relevant technologies and understanding their career progressions.

Type of Data

Unstructured data in .json format

Problems in the Data

The dataset exhibits certain challenges that need to be addressed for effective analysis and processing. Specifically, the number of columns and rows in each label is inconsistent, leading to a non-uniform structure. For instance, some entries in the skills dataframe contain only one skill, while others have up to five. This inconsistency results in numerous NA values, making it difficult to accurately determine the number of skills through resume parsing. Ideally, a consistent number of bullet points for each section would have facilitated a more straightforward extraction process.

It is important to note that there are no outliers or skewed data in this qualitative NLP data project. The primary objective of this project is to streamline the transfer of information from 200 raw resumes into a structured database. By leveraging NLP techniques, the project aims to identify and classify relevant entities such as person names, college names, academic information, relevant experiences, skill sets, and more.

To ensure the reliability and quality of the extracted information, steps will be taken to address the issues related to non-uniformity and NA values. This may involve applying data normalization techniques to establish a consistent structure across the dataset, filling in missing values using appropriate imputation methods, and implementing validation checks to validate the accuracy of the extracted entities.

By addressing these challenges, the project aims to streamline the process of resume analysis and enable HR professionals to efficiently shortlist promising candidates based on relevant qualifications and experiences.

Solving Problems in the Dataset

-Lemmatizing Words
-Removing Stopwords
-Removing/Replacing special characters
-Removing extra whitespaces
-Tokenizing
-TF-IFD
-Lower casing all characters
-Applying pos tagging
-Converting digits to words

Exploratory Data Analysis

After completing data cleaning part, we started our EDA with Part of Speech Analysis. We created a bar plot to visualize the distribution of Parts of Speech. Our results emphasized that:

- Nouns occur the most in Resume content.

We identified most common words used in Resumes. Our bar plot showed that:

- The term "experience" ranked as the fifth most commonly used word, highlighting the significant value placed on experience compared to education within the job market.
- Management experience is the fourth most sought after skill looked in the job being applied
- Microsoft and Oracle emerged as the most frequently cited companies on the applicants' resumes.

We computed and plotted the distribution of average word length, distribution of total word count, distribution of total characters count and we learned that:

- Bar plot shows that most of the resumes in our dataframe have avarage word length 7
- More than 70 rows(resumes) have avarage word lenght 7
- More than 120 resumes have total word count less than 500.
- There are less resumes that have more than 500 words
- Number of resumes which contains more than 1000 word is very few.
- Most of the resumes have total characters less than 5000.

After that, we found most commonly used stop words in Resume content and concluded that:

- Most common stopwords used in Resumes are ‘and’, ‘like’, ‘also’

We created a word cloud representation that shows the most frequent words in the text, with larger words indicating higher frequency.

- Word cloud emphasize that most common words in the resumes excluding stopwords are 'application', 'team', 'management', 'year', 'project', 'client', 'service'

We performed N-gram analysis which is used in natural language processing to extract contiguous sequences of n items (typically words) from a text. It helped us to identify patterns, relationships, and frequencies of word sequences, enabling insights into language usage and context. We computed unigrams, bigrams and trigrams.

- Unigram bar plot emphasized that the most commonly used unigrams are ‘:’, ‘two’, ‘and’,‘thousand’.
- Bigram bar plot emphasized that the most commonly used bigrams are ‘two thousand’, ‘thousand and’.
- Trigram bar plot emphasized that the most commonly used trigram is ‘two thousand and’.

We investigated the number of workers located in each state and performed visualization on the Indian map.

- Our results showed us that most of the workers are located in Karnataka.

Model Building

We decided to proceed with NER MODEL with python's SpaCy package to fetch resume details.

SpaCy's models are statistical and rely on patterns in training data to make predictions. These predictions are probabilistic and not absolute. The accuracy depends on the quality of training data.

Here's a breakdown of what the code does:

1-It defines the name of the model file as "ner_model" and sets the number of iterations for training as 20.

2-It reads a JSON file containing resumes data into a pandas DataFrame.

3-It defines a dictionary called entity_dict that maps custom tags for entities like name, college name, degree, etc.

4-The code includes a function called mergeIntervals that merges overlapping intervals.

5-The function get_entities extracts entity information (start index, end index, and label) from the 'annotation' column of the DataFrame using the entity_dict.

6-The extracted entities are merged using the mergeIntervals function, and the results are stored in a new column called 'entities' in the DataFrame.

7-The code then defines a function called trim_entity_spans that removes leading and trailing white spaces from entity spans.

8-The trim_entity_spans function is applied to the 'content' and 'entities' columns of the DataFrame.

9-It creates a blank spaCy NLP model.

10-The NER pipeline is added to the NLP model, and entity labels are added to the pipeline based on the entities in the data.

11-The code then trains the NER model using the provided training data (a), iterating for the specified number of iterations. The training data is shuffled at each iteration to avoid model bias.

12-After training, the model is saved to disk.

13-The saved model is loaded, and a specific resume is selected for testing.

14-The loaded model is used to process the selected resume, and the recognized entities are printed.