

**“Matching CV with job description”**

Aahil Ashiq Ali K214549

Khuzaima Ahsan K213328

Khubaib Lodhi K214596

BCS-6A

Department of Computer Science BS(CS)

FAST-NUCES Karachi

**Implementation As Per the Paper:**

Skill Finding Algorithm described in the "Skill Finder: Automated Job-Resume Matching System" paper is aimed at ranking student resumes according to their skills in relation to job requirements. The algorithm consists of two primary steps:

First Step:

Named Entity Recognition (NER) Training Model: This stage entails the utilization of a statistical classifier to identify skills needed from job descriptions and those possessed by students in their resumes. The algorithm scrutinizes each word in a sentence to ascertain if it constitutes a named entity, specifically a skill. To train this model, the training data is generated by downloading job excerpts and resumes, converting them into the OpenNLP name finder training format, and then training the model using the Apache OpenNLP framework.

Second Step:

Skill Storage: The identified skills extracted from job descriptions are stored in one database table (e.g., jobs\_skills), while the skills extracted from student resumes are stored in another table (e.g., resume\_skills).

Skill Matching: When a faculty user seeks applicants for a job, a Cartesian product and an equi join are performed on the job\_skills and resume\_skills tables to match students with the required skills for the job. The students are ranked based on the number of matching skills they possess.

**Methologoy Adopted In the Project:**

The algorithm begins by preprocessing both resumes and the job description (JD) text. Resumes are sourced from files within the 'cvs txt/' directory, where the text is converted to lowercase, special characters and stopwords are removed, and the content is tokenized for further analysis. Similarly, the JD undergoes a similar preprocessing routine to ensure uniformity in data handling.

The next crucial step involves entity extraction, facilitated by spaCy augmented with a custom EntityRuler. This ruler is loaded with patterns from a JSONL file specifically crafted for recognizing skills, whether soft or technical, from both resumes and the JD. These extracted skills are cataloged in dictionaries for easy access and subsequent processing.

Following entity extraction, the algorithm calculates TF-IDF values for both technical and soft skills found in the resumes. Initially, document frequency (DF) for each skill is determined, from which the inverse document frequency (IDF) is computed. Term frequency (TF) is then established for each skill in every resume, with TF-IDF weights derived by multiplying TF and IDF. This process is iterated for both technical and soft skills, yielding a comprehensive set of TF-IDF weights for each resume.

Similarly, TF-IDF weights are computed for the JD, utilizing the extracted skills and precomputed IDF values. Subsequently, cosine similarity metrics are computed for both technical and soft skills between the resumes and the JD. This measure quantifies the similarity between the skill sets of resumes and the JD.

Additionally, education levels are evaluated independently, with higher qualifications receiving elevated scores. These scores are integrated with the cosine similarity scores to generate a final ranking of resumes. Based on a predefined threshold (alpha) of 0.3, resumes are categorized as accepted or rejected, and the results are presented accordingly.

The results are displayed using streamlit library and job description file is inputed using tkinter. Alpha threshold being used is 0.3. All results greater than 0.3 will be accepted.

While OpenNLP offers rule-based named entity recognition capabilities, spaCy's EntityRuler serves a similar function in Python. However, spaCy distinguishes itself by providing an extensive range of efficient NLP tools and models within the Python ecosystem.

**Sample Results:**

* **Result for instructor**

****

* **Result for Lecturer**

****

* **PHP Developer**

**A screenshot of a computer

Description automatically generated**

* **Research Assistant**

****