Resume Ranking System Using Word Embeddings And Euclidean Distance

Abstract

The hiring process involves screening numerous resumes to find the best candidates, which can be time-consuming and inefficient. This project automates resume ranking using **word embeddings** and **Euclidean distance** to compare resumes against a job description.

The system extracts text from resumes and the job description, cleans and tokenizes the text, and trains a **Skip-Gram model** to generate word embeddings. Each resume is converted into a numerical vector (document embedding), and the similarity between the job description and each resume is calculated using **Euclidean distance**. Resumes with a lower distance are ranked higher, while those with a higher distance are considered less relevant.

This automated approach improves efficiency, reduces bias, and speeds up the recruitment process. It can be further enhanced by integrating **pretrained models like SBERT or Word2Vec** for better accuracy.

System Modules

1. **Resume Extraction** – Reads resumes in **PDF format** and extracts text.
2. **Preprocessing** – Cleans text by removing special characters and stopwords.
3. **Word Embedding Generation** – Trains a **Skip-Gram model** to create word vectors.
4. **Document Embedding Computation** – Converts resumes and job descriptions into numerical vectors.
5. **Resume Ranking** – Measures similarity using **Euclidean distance** and ranks resumes.
6. **Result Categorization** – Classifies resumes as **highly relevant** or **less relevant** based on a threshold.

Workflow

1. **Input:** Upload a **job description** and a folder of **resumes** (PDFs).
2. **Text Processing:** Extract and clean text, tokenize words.
3. **Train Word Embeddings:** Use **Skip-Gram** to generate word vectors.
4. **Compute Similarity:** Convert resumes and job description into vectors, compare using **Euclidean distance**.
5. **Ranking & Output:** Rank resumes based on similarity; display **highly relevant** and **less relevant** resumes.