Week 8 Progress Report

Group Name: Natural Language Processing Engineer

Specialization: Natural Language Processing

Topic: Resume Extraction

Team Member Name: Ashraf Moumin

Email: ashrafmoumin1@gmail.com

Country: Turkey

University: Istanbul Technical University

Problem Description: Natural Language Processing is the application of computational techniques to the analysis and synthesis of natural language and speech. As such, this field is very important to a large number sectors, in particular those dealing with extensive bureaucratic processes. In particular, in this project, we want to work on automating resume information extraction. This problem is a text classification one in which the goal is to assign each resume to skillsets.

Data Understanding: The type of data needed is different than usual. The dataset we have is a .json file with a 'content' part consisting of text written by job applicants, labels about the text and some metadata. The actual information that is really needed is in the 'content' part of the resumes along with labels of skills of each applicant or other information to extract. As for the statistics of the dataset, the 'content' for each instance has on average around 3453 characters, which translates approximately to between 690 and 863 words (taking an average of 4 to 5 characters per word).

Data Issues: To load the data, I decided to load the .json file manually as a regular text file and to seperate each instance with the special characters used ("}]}]}" was at the end of every instance). This method worked well and there was only a minor issue with an instance potentially having wrong ending characters. The above-mentioned method was used because the file contained some issues (structural…) and the json library had issues reading it.

Model Choice and Modeling: I think embeddings are very important in these kind of text classification tasks. Now, I am thinking of a model taking the text content, using an embedding of the whole text, and leveraging a similarity algorithm with the embedding vectors of the skills we want to detect and then taking all skills that have a similarity score with the text higher than a threshold as being skills of the applicant. A variant of this procedure can also be a solution. I am looking at getting the embeddings of an open-source LLM such as BERT. I downloaded the weights of one such model in .h5 keras format, however extracting the weights from the file seem to require building and compiling the exact same model with keras before downloading the weights. I am still working on finding open-source embeddings and LLM models.