

Author: Ellis

Task: Skill Extraction

Scripts:

- `PhaseTwo/Utility/llm_remote_skill_extractor.ipynb`
- `PhaseTwo/Utility/llm_local_skill_extractor.py`
- `PhaseTwo/Utility/skill_csv_generator.ipynb`

Data:

- Overwrote "skills" column in: `PhaseTwo/Data/new_england_indeed_jobs.csv`
- `PhaseTwo/Data/new_england_indeed_job_skills.csv`

Methodology:

- Used `PhaseTwo/Data/new_england_indeed_jobs.csv` for the "description" column to process and extract job "hard" skills.
- Researched sentence transformers and keyword extraction models (like [YAKE](#)), but these wouldn't take out *just* job skills.
- Was able to find various models on HuggingFace (HF), like a [BERT based model](#), however this would be a victim of the same issue and recognize non-job skills.
- I stumbled upon [SkillNER](#) which is a job specific skill named-entity-recognition (NER) tool. However, this hasn't been updated much in the past 4 years, and is bloated with dependencies. Most likely was "out of commission" with large language models (LLM).
- Eventually decided on using an LLM. I was hesitant at first because I wanted to find a more specialized model that was light weight.
- Started to use the HF inference API with a free account using the `gpt-oss-20b` model with the `PhaseTwo/Utility/llm_remote_skill_extractor.ipynb`
- This required frequent timeouts and writes to the CSV to keep progress. Essentially a very manual process, but I wanted a way to avoid needing the actual tensors on my local machine.
- After 5 hours and only ~450 rows were processed, I decided to move on. I am glad I still attempted it as I learned a lot in the process and refreshed my memory on a few topics.
- I ended up going with `LLaMA-3.1-8B-Instruct` on HF because I have access and experience working with the model from the Information Retrieval course. I tried running it on my 12GB 4070Ti GPU, but with 16GB in weights it was simply too slow going in between memory. I decided to run it on the school's Linux servers.
- The results were very good, just sometimes at the very end of a skill listing the model would add in extra comments. The instructor suggested re-prompting, but when looking at enough samples, I deduced it was not necessary.
- Instead, for each job and its respective skills entry, I made the text lowercase and cut off the string when a newline or period character occurred. This result then overwrote the already existing `PhaseTwo/Data/new_england_indeed_jobs.csv` file.
- Then, I generated a new dataframe which concatenated all of those skill entries and appended their frequency to it. I dropped skills that had only one entry, resulting in ~1700 unique skills. This was then output to a new file:
`PhaseTwo/Data/new_england_indeed_job_skills.csv`

