Towards Entity Recognition and Relation Extraction from Job Description Text

Shraddha Mukesh Makwana
University of Alberta
CCID:smakwana
smakwana@ualberta.ca

Pranjal Dilip Naringrekar
University of Alberta
CCID:naringre
naringre@ualberta.ca

Abstract

One of the most important aspects of the modern recruitment market is online job boards. With millions of job seekers perusing job posts on a daily basis, the need for accurate, effective, meaningful, and transparent employment suggestions is greater than ever more than ever before. Finding the perfect match of job to candidate's profile without trivial recommendation is difficult and not a straightforward task. How to successfully fill in the job vacancy heavily depends upon the skill-set mentioned in the job description and resume of the candidate. Traditional job search systems perform simple data-mining based on keyword similarity, and do not take into account the interlinks between entities. This paper focuses on extracting the entities and relations from the job descriptions, which are basic building blocks for constructing the job search knowledge graph. Here, we built the relation extraction model using BERT which will be a classifier that predicts a relation 'r' for a given pair of entities e1, e2. We evaluate our work using the F1 score measure.

1 Introduction

In this task, we performed joint Named Entity Recognition and Relation Extraction between the entities to build a better match between job and candidate's profile skills. A survey says that 42% of Job Applicants don't meet skill requirement and hence we aim to construct a KG that will help match right candidates with right job. For an instance, a candidate might have extensive skills in "Java" programming, but the job description of interest requires knowledge in "J2EE" framework, which is essentially based on "Java"; hence relation extraction will make sure it understands that "Java" is entangled with "J2EE" and perform better resume shortlisting.

Figure 1, depicts the basic architecture of the task, wherein the job description dataset will be passed to the relation extraction classifier and the output of the same will be relation triples from the input text material. The entire task is performed in three stages called Named Entity Recognition, Relation Extraction between the entities to build a better match between job and candidate's profile skills and Knowledge Graph (KG) construction. The job description is annotated using UBIAI (an open-source tool) for entities and relations. Spacy's internal rel-component is used as our base project. We trained the BERT model on this input data to predict relations on test data. We then constructed the KG using Neo4j which can be queried. An example, if the job description contains entities like "Bachelor" and "Computer Science" and their relationship is "degree_in", in that case the output relation triples will be ("Bachelor", "degree_in", "Computer Science").

Therefore, solving this task of relation extraction from job description will help to address upon the foundation for a tool that finds field-relevant skills listed on the resume and to uncover which job skill it is more pertinent to. It's application is majorly focused on transferring the job searching task from keyword searching to candidate model matching (Guo et al., 2016).

2 Related Work

The task of Entity Recognition and Relation Extraction have been previously solved using various approaches. The architecture proposed by de Groot et al. (2021) studies link prediction methods for quantifying the relatedness between skills and occupations using Node2Vec. The performance of their model exceeded two different link prediction methods, which were based on preferential attachment (PA). Here, they took into account the aspect of similarity matching. If few words were used in the same context it was able to identify those

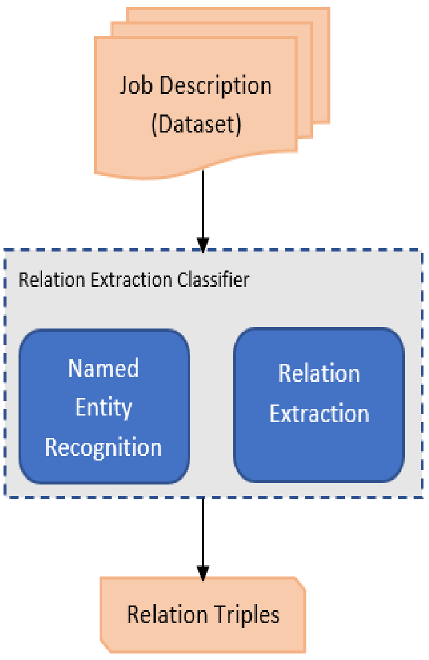


Figure 1: System Architecture

entities appropriately. They made use of Node2Vec model whereas we used the transformer based and non-transformer based (Tok2Vec) model.

Biancofiore et al. (2021) proposed GUapp which is a tool for searching and recommending job openings in the Italian public sector. The platform provides recommendation services with the goal of matching user skills and requests with job openings in a specific time frame. They extracted stronger relations by merging some sub-graphs from state-of-the-art solutions like Dbpedia and new triples scraped from external sources. Their approach combined the existing KG to enhance their system whereas our approach followed a more simple-from-base KG construction.

A method for rating applicants based on keywords related to competence was proposed by Wang et al. (2021) which constructed two KG unlike our approach that constructed a single KG to incorporate all possibilities. To begin with, they determined the score of matching between a competency keyword and a corpus of CVs using the TF-IDF Vectorizer. Second, based on two types of competency keywords they used the Weighted Average Method to create a global CV score. Third, they constructed a Knowledge Graph (KG) from a structured Competence Map that can classify bidirectional association. Finally, they proposed using BERT's Named-Entity Recognition to better identify tokens in the client's input questions and tested it on CVs from the Human Resource Department.

3 Methods

We aim to solve the problem by using a three stage approach: Named Entity Recognizer (NER), Relationship Extraction using transformer based model (BERT), and storing the constructed KG in Neo4j. We aim to chose this method as it divides a complex task into three simple stages. It uses a pre-annotated text that can be easily generated from UBIAI to depict the entities and relations that can be given to the model and also be used to construct the KG.

3.1 Data

For our implementation, we used the dataset of job descriptions as shown in Table 1, which is publicly available at Kaggle. The dataset is divided as 15% dev data, 70% as train data and 15% as test data. We utilise the UBIAI text annotation tool to collect training data because of its flexible interface, that allowed us to quickly switch between entity and relation annotation. UBIAI is an Oregon based startup that provides cloud-based solutions and services in the field of Natural Language Processing (NLP) to help users extract actionable insights from unstructured documents. We converted the annotated data to a binary spacy file before we can train the model and separated the UBIAI annotations into train/dev/test and saved them separately.

Field	**Value**
Id	12612628
Title	Systems Analyst
JobDescription	Our clients are looking for...
LocationRaw	Dorking, Surrey
LocationNormalized	Dorking
ContractTime	permanent
Company	Martin International
SalaryRaw	20000 - 30000/annum
SalaryNormalized	25000
SourceName	cv-library.co.uk

Table 1: Sample Data Record

3.2 Phase 1: Entity Recognition

In first stage, we extracted entities using the pre-trained fine-tuned NER model to find skills, diploma, diploma major and years of experience. We provided the entities from our golden corpus and used them to train the classifier. Spacy's internal rel_component was used for this purpose. SpaCy 3 uses a config file config.cfg that contains all the model training components to train the model that selects the language of the model, NER and hardware (GPU) to use and download the config file template. Later, auto-filled the config file with the rest of the parameters that the BERT model requires and trained the model and saved it under the folder called 'model-best'.

3.3 Phase 2: Relationship Extraction

The second stage focused on implementing the HD-SKG solution done in the paper by Zhao et al. (2017), which uses BERT model to extract the relationships. In this we made changes to the model file that helped to decrease the max_length in configs/rel_trf.cfg from the default 100 token to 20 to increase the efficiency of our model. The max_length corresponds to the maximum distance between two entities above which they will not be considered for relation classification. As a result, two entities from the same document will be classified, as long as they are within a maximum distance (in number of tokens) of each other. We then trained the transformer and non-transformer model (tok2vec) to predict these relations.

3.4 Phase 3: KG Construction

In third stage, we constructed the KG using Neo4j. While the Python module networkX allowed us to create graphics of our nodes and relationships, the actual graph was saved in Python memory rather than a database. When trying to build scalable systems that must hold an ever-growing knowledge graph, it proved to be an issue. This is why Neo4j is used as it allowed us to save the graph in a fully working database that can handle massive amounts of data and can also be queried. The output can be seen in Figure 2.

Lastly, we will evaluated our method on 15% test data. The F1-score (0.28) for relation extraction by Angeli et al. (2015) who proposed the openIE tool will remain our baseline and our aim would be to maximize this F1-score value.

4 Results

For assessing the performance of our work, we will be using the same evaluation metrics used by Zhao et al. (2017), which is F1 score. It's calculation is the harmonic mean of Precision and Recall, as follows:

$$F1 = 2 \times (\frac{(precision \times recall)}{(precision + recall)})$$

The evaluation measure is calculated based on the overlap between the predicted and actual extracted relationships. Our evaluation function will give credit to partial matches between gold and predicted relationships. The partial credit is proportional to the intersection of the two relationships, and it is normalized by the length of the two entities. The gold label standards for this task was handcrafted by us.

Table 3 highlights the result for entity recognition and relation extraction model. We received an F1 score of 0.8 for entity recognition and F1 score of 0.5 for relation extraction.

Along with this, we also ensured the correctness of the constructed KG using various queries.'Finding the most in demand skills', 'Skills that Require Highest Years of Experience' and 'Pair of Skills that Co-occur the Most' are some of the examples of the executed query.

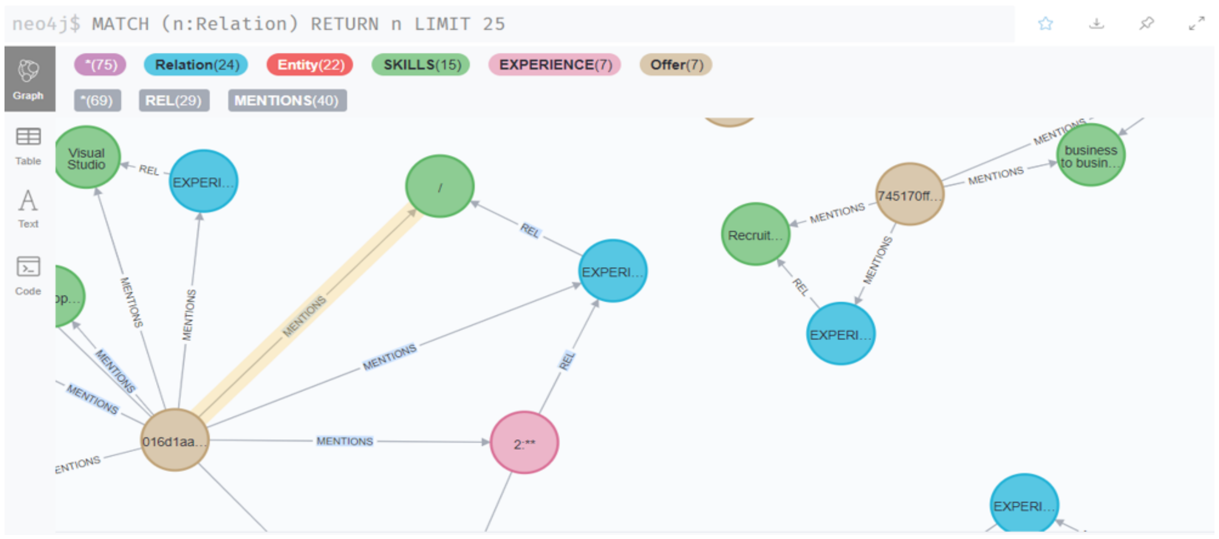
Model	**Precision**	**Recall**	**F1**
Entity Recognition	0.6	0.4	0.8
Relation Extraction	0.6	0.2	0.5

Table 2: Result of Model

Also, for better performance we tried implementing the Dictionary-based named entity recognition and Bi-LSTM-CRF model to extract entities. Dictionary-based named entity recognition categorized datasets based on collected dictionaries or user-defined dictionaries (Neelakantan and Collins, 2014). However, there is a disadvantage of having to manually organize dictionaries, and because it is necessary to deal with constantly changing and emerging new words over time we tried the Bi-LSTM-CRF model which showed meaningful performance in time series data, using supervised learning-based word embedding and non-supervised learning-based word embedding from a large corpus (Huang et al., 2015).

5 Discussion

The results are build on the existing evidences reported by Angeli et al. (2015). Even though our model was able to reach an F1 score of 0.8 and



```
neo4j$ MATCH (n:Relation) RETURN n LIMIT 25
```

Figure 2: Neo4j Graph

0.5 for entity recognition and relationship extraction respectively, it still depicted various faults. We were able to identify errors with respect to three scenarios in the constructed KG.

In first scenario, a skill called 'C#' was considered as 'C' and '#' separately causing an undefined entity to be constructed. In the second scenario, the number of experience was taken to be as 35 instead of 3-5 years. In the third scenario, a wrong relationship was predicted amongst these wrong recognized entities. In order to avoid these errors, categorize all types of entities which are not classified, along with the relationships. Include relation extraction to be trained on more precise entities to avoid interpreting wrong relations.

Apart from this, NEO4J helped in fast searching in the KG. This solution can be implemented for any other independent domain as well. With only a hundred of annotated documents, we were able to train a relation classifier with good performance. However, both tasks are pipelined currently and can't be used separately to train both models simultaneously

6 Conclusion

We were able to solve the problem of matching job description with perfect candidates by using Named Entity Recognizer (NER) and transformer based model (BERT) to extract entities and relationships, respectively and store the constructed KG in Neo4j that can be queried. We achieved this by building a KG linking jobs and skills together. Our results clearly depict that our entity recognition model performed better than the relationship extraction model. As a part of our error analysis we were able to identify few errors in the constructed KG and also proposed methods to resolve them.

In future, we will also try to achieve high performance for this task by exploring multiple state-of-the-art Natural Language processing based techniques such as Markov Logic Networks and experimenting with other different types of feature extraction to understand and evaluate Relation Extraction Triples. This will help in constructing a more precise KG. Combining NLP with Neo4j's graph DB can help us accelerate information discovery in many domains, with more notable applications in healthcare and biomedical.

7 Repository URL

The url for our project repository can be found at:

https://github.com/shr1911/cmput656-job-knowledge-graph

References

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354.

Giovanni Maria Biancofiore, Tommaso Di Noia, Eugenio Di Sciascio, Fedelucio Narducci, and Paolo Pastore. 2021. Guapp: Enhancing job recommendations with knowledge graphs. *11th Italian Information Retrieval Workshop*.

Maurits de Groot, Jelle Schutte, and David Graus. 2021. Job posting-enriched knowledge graph for skills-based matching. *arXiv preprint arXiv:2109.02554*.

Shiqiang Guo, Folami Alamudun, and Tracy Hammond. 2016. Résumatcher: A personalized résumé-job matching system. *Expert Systems with Applications*, 60:169–182.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging.

Arvind Neelakantan and Michael Collins. 2014. Learning dictionaries for named entity recognition using minimal supervision. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 452–461, Gothenburg, Sweden. Association for Computational Linguistics.

Yan Wang, Yacine Allouache, and Christian Joubert. 2021. Analysing cv corpus for finding suitable candidates using knowledge graph and bert. In *DBKDA 2021, The Thirteenth International Conference on Advances in Databases, Knowledge, and Data Applications*.

Xuejiao Zhao, Zhenchang Xing, Muhammad Ashad Kabir, Naoya Sawada, Jing Li, and Shang-Wei Lin. 2017. Hdskg: Harvesting domain specific knowledge graph from content of webpages. In *2017 ieee 24th international conference on software analysis, evolution and reengineering (saner)*, pages 56–67. IEEE.