Empowering Job Seekers with Advanced NLP: A Revolutionary Approach to Career Navigation in the Modern Job Market

DATS 6202– Natural Language Processing

Professor: Amir Jafari

By: Amjad Altuwayjiri, Muhannad Alwhebie, Nammin Woo

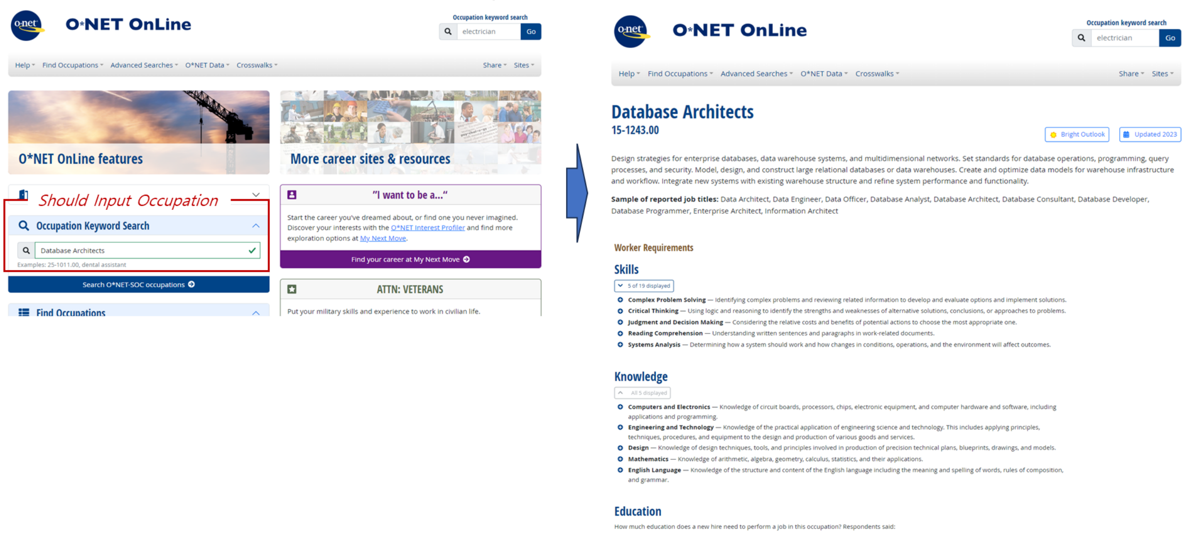
Table of Contents:

1. Introduction

This project addresses the gap in current job search tools, such as the ONET® Database shown in Fig.1, which primarily focuses on manual navigation and job title-based searches, often not adequately aligning with a job seeker's unique skills and career level. The advanced Natural Language Processing (NLP) program is designed to overcome these limitations by offering a more dynamic, user-centric approach. It uses modern NLP techniques to streamline job searching, helping individuals find positions that precisely match their skills and career aspirations. The project comprises two main components: a Text Classification Model for accurately predicting job levels in resumes, and a Semantic Similarity Search using Siamese-BERT Networks. By deploying these innovative tools, we aim to transform the job-hunting experience into one that is more efficient, targeted, and aligned with the individual career development goals of job seekers.

Fig.1

Existing Tool: O-NET Online



Database Architects

Examples: 25-1011.00, dental assistant

Start the career you've dreamed about, or find one you never imagined. Discover your interests with the O*NET Interest Profiler and find more exploration options at My Next Move.

Put your military skills and experience to work in civilian life.

Database Architects
15-1243.00

Design strategies for enterprise databases, data warehouse systems, and multidimensional networks. Set standards for database operations, programming, query processes, and security. Model, design, and construct large relational databases or data warehouses. Create and optimize data models for warehouse infrastructure and workflow. Integrate new systems with existing warehouse structure and refine system performance and functionality.

Sample of reported job titles: Data Architect, Data Engineer, Data Officer, Database Analyst, Database Architect, Database Consultant, Database Developer, Database Programmer, Enterprise Architect, Information Architect

Skills

- **Complex Problem Solving** — Identifying complex problems and reviewing related information to develop and evaluate options and implement solutions.
- **Critical Thinking** — Using logic and reasoning to identify the strengths and weaknesses of alternative solutions, conclusions, or approaches to problems.
- **Judgment and Decision Making** — Considering the relative costs and benefits of potential actions to choose the most appropriate one.
- **Reading Comprehension** — Understanding written sentences and paragraphs in work-related documents.
- **Systems Analysis** — Determining how a system should work and how changes in conditions, operations, and the environment will affect outcomes.

Knowledge

- **Computers and Electronics** — Knowledge of circuit boards, processors, chips, electronic equipment, and computer hardware and software, including applications and programming.
- **Engineering and Technology** — Knowledge of the practical application of engineering science and technology. This includes applying principles, techniques, procedures, and equipment to the design and production of various goods and services.
- **Design** — Knowledge of design techniques, tools, and principles involved in production of precision technical plans, blueprints, drawings, and models.
- **Mathematics** — Knowledge of arithmetic, algebra, geometry, calculus, statistics, and their applications.
- **English Language** — Knowledge of the structure and content of the English language including the meaning and spelling of words, rules of composition, and grammar.

Education

How much education does a new hire need to perform a job in this occupation? Respondents said:

figure 1:ONET database page where job seekers type the job title they are looking for and the database shows results of the skills, knowledge, and education needed for this occupation.

2. Methodology

2.1. Description of the Data Set

In this project, an employment the ONET® Database, a comprehensive and authoritative resource sponsored by the U.S. Department of Labor, Employment and Training Administration (USDOL/ETA), as the foundational data set. This database is remarkable for its extensive coverage of work and worker characteristics, emphasizing skill requirements for various occupations. It offers a rich data set with detailed information, including job responsibilities, required skills, and educational requirements, which is crucial for creating detailed job profiles and understanding the nuanced skill sets needed for different jobs. The ONET® Database's government sponsorship enhances its credibility and reliability, making it a trusted source in workforce development, HR, and education fields.

Fig 2.

Framework of ONET DB

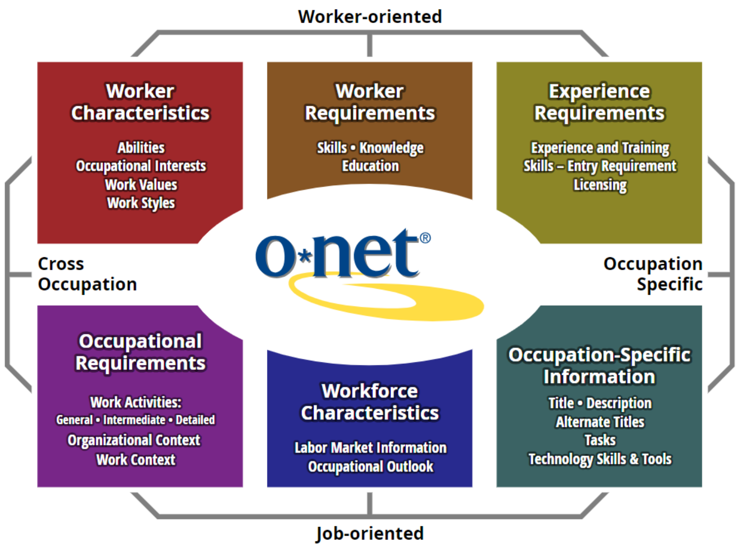


Fig 2. ONET Data involves various categories both worker ability and job description side. Specifically, 40 subcategories databases oriented from statistical survey or analyst research are defined, and they are fully connected by primary keys.

For this project, this database serves as the basis for deriving the job description corpus, providing a diverse and in-depth data pool. This is instrumental in training the Text Classification model as well as semantic similarity search using Siamese-BERT Networks, enabling us to build comprehensive profiles for various occupations and ensure the effective training of the models. Particularly, The project aimed to create a unique JOB Corpus by defining the following preprocessing steps and aggregating rules:

Step 1. Read bulk data (*.zip file) and unzip to unit files

- 40 individual text files contain either Base information like code-description mapping or specific Element Contents specified in each subcategory information

Step 2. Pick key categories and Generate Individual Corpus

- Choose Resume related 5 categories - Ability, Skill, Tech-Skill, Knowledge, Interest Category, and perform primary aggregation (Contents data + Base information)
- Create 'Job Corpus' of each Category

  * Each category has about 50 unique elements which describe specific contents in a category, and the corpus was constructed by extracting 5 highest demand elements in each category

Step 3. Create Aggregated Job Corpus data based on Occupation dataset

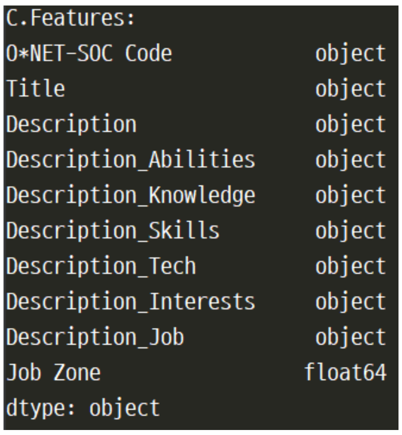
- Generate Full Corpus and Add Job Level label (classification model target)

  *Summarized job description + Combination of job corpus in each category

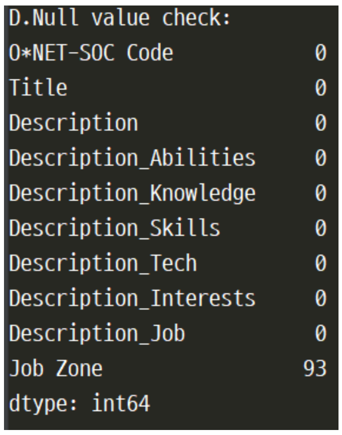
Below are a result of basic exploration of the data:

Fig 3.

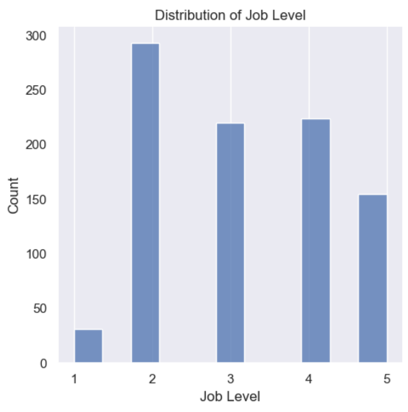
Job Corpus Data

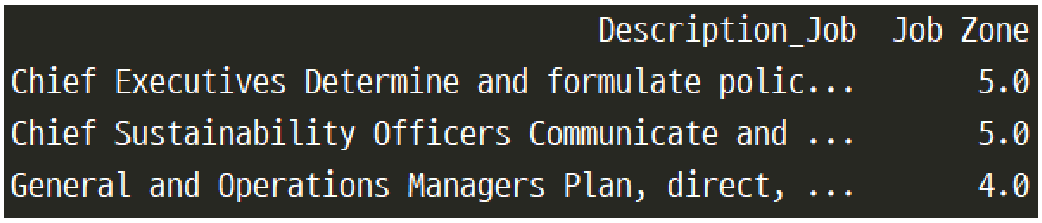


```
C.Features:
O*NET-SOC Code            object
Title                     object
Description               object
Description_Abilities     object
Description_Knowledge     object
Description_Skills        object
Description_Tech          object
Description_Interests     object
Description_Job           object
Job Zone                 float64
dtype: object
```



```
D.Null value check:
O*NET-SOC Code            0
Title                     0
Description               0
Description_Abilities     0
Description_Knowledge     0
Description_Skills        0
Description_Tech          0
Description_Interests     0
Description_Job           0
Job Zone                 93
dtype: int64
```





```
                             Description_Job  Job Zone
Chief Executives Determine and formulate polic...       5.0
Chief Sustainability Officers Communicate and ...       5.0
General and Operations Managers Plan, direct, ...       4.0
```

Fig 3. The Final ONET Job Corpus Data has 1,016 records with 10 features. Job levels are uniformly distributed except level 1.

2.2 Classification model

Implementation of a text classification model was conducted to ascertain the job level by analyzing the contextual information present in the Job Corpus data. The concept of job levels in this context refers to the categorization of various occupations based on comparable levels of experience, educational background, and requisite training as in Table 1. This approach enables a nuanced understanding of where a particular occupation fits within a broader professional hierarchy.

Table 1.

Definition of Job Level

Level	Expected Position	Description
1	Intern	Little or No Preparation Needed
2	Entry Level	Some Preparation Needed
3	Junior	Medium Preparation Needed
4	Mid Level	Considerable Preparation Needed
5	Senior	Extensive Preparation Needed

Table 1: description of job level categories

The modeling strategy included a spectrum of techniques, from the training of classical models from the ground up to the fine-tuning of pretrained transformers. Architectural modifications were also explored, specifically by augmenting the transformer model with additional layers at the head. Moreover, there was an experimentation with the recomposition of the job corpus, selectively concentrating on certain categories to sharpen the model's capacity for recognizing and emphasizing pertinent data.

Fig 4.

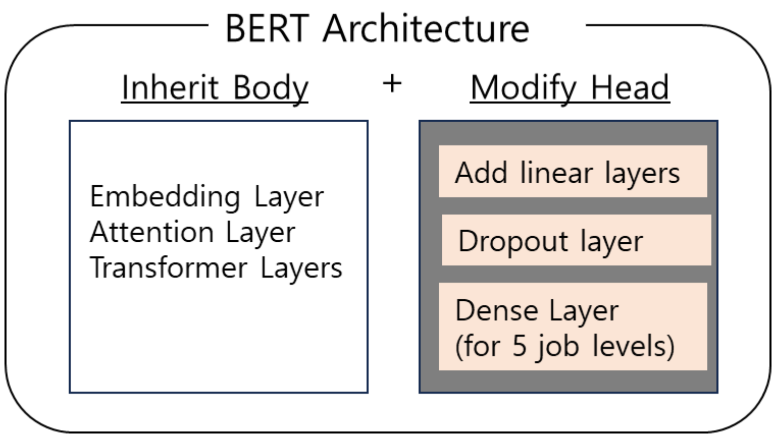


Fig 4. Modifying trials in BERT Architecture

2.2.1 BERT transformer with a Linear layer

Job level classification component was designed to categorize job roles into various levels. This task was accomplished using a classifier based on the BERT model, which was further enhanced with a linear layer. This BERT-based classifier effectively leveraged natural language processing (NLP) techniques to interpret and categorize the complexities of different job roles.

A vital part of this methodology involved the BERT tokenizer, which transformed textual job descriptions into a numerically tokenized format. Following this tokenization, the classifier, integrated with the pre-trained BERT model, uses a linear layer for the classification task. This layer performed a mathematical transformation of the input features from the BERT model, aligning them with the predefined job-level classes. The linear transformation was represented by the equation:

$$n = w \times p + b$$

Furthermore, the softmax function was applied to the output of the classifier's linear layer. This function mathematically transformed the output logits into a probability distribution over the job levels. The softmax equations for the output layer were given as

$$a = \text{softmax}(W \times a + b)$$

to ensure that the output probabilities summed up to one and were interpretable as class probabilities.

The training of this model focused on accurately classifying job descriptions by minimizing a loss function, typically Cross-Entropy Loss in multi-class classification tasks.

This loss function quantified the difference between the model's predicted probabilities and the actual class labels.

To fine-tune this BERT model with a linear layer, a hyperparameter tuning process using grid search was used. The parameters adjusted included learning rates, batch sizes, and the number of epochs. Specifically, the learning rates considered were (1e-5, 3e-5, 5e-5); batch sizes were set at (8, 16, 32); and the number of epochs options were (3, 4).

Upon completion of the grid search, the best-performing model configuration was identified as having a learning rate of (5e-05_), a batch size of (16), and a duration of (3) epochs. This configuration was selected for its superior performance to be used.

2.2.2 MLP

The Multilayer Perceptron (MLP) classifier in the project was optimized using Grid Search, a method chosen for its efficiency in finding the best combination of hyperparameters. Key hyperparameters tuned included the number of hidden layers, activation function, solver algorithm, L2 regularization strength, and initial learning rate. These parameters were varied within predefined ranges: hidden layers (50 or 100), activation function (tanh or relu), solver (sgd or adam), L2 regularization strength (0.0001 or 0.05), and learning rate (constant or adaptive).

A Grid Search was done, and the best model obtained from the search is activation: relu, alpha: (0.05), hidden layer sizes: (100,), learning rate: constant, solver: adam which was evaluated on the test set. Its performance was measured in terms of accuracy.

To complement the analysis, visual representations were also created. A learning curve graph displayed the relationship between training size and accuracy, and a confusion matrix was plotted to provide a visual insight into the model’s classification accuracy across different classes. These visuals offered a comprehensive view of the MLP model's performance and its effectiveness in the classification task.

2.2.3 Logistic Regression

In the project, the Logistic Regression model was employed as a comparative classifier to evaluate against the advanced machine learning models. This model's development began with feature extraction using TF-IDF and setting a maximum feature limit of 5000, to transform the job descriptions into a TF-IDF matrix, thus preparing the data for the Logistic Regression model. Mathematically, TF-IDF is computed as:

$$\mathrm{tf\ idf}(t,d,D) = \mathrm{tf}(t,d)\ .\mathrm{idf}(t,D)$$

It is calculated by multiplying two components: Term Frequency (TF), which counts the number of times a word appears in a document, and Inverse Document Frequency (IDF), which gauges how common or rare a word is across all documents.

Logistic Regression, a staple in classification tasks, was selected for its simplicity and effectiveness. The logistic regression model estimates probabilities using a logistic function, which is an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits. The logistic function is defined as:

$$f(x) = \frac{L}{1 + e^{-k(x-x_0)}}$$

Then it was optimized using Grid Search, a systematic approach to hyperparameter tuning. This process involved adjusting parameters which are the regularization strength values of (0.1, 1, 10) and the maximum number of iterations values of (100 and 200). The best model configuration was regularization strength: (10), maximum number of iterations: (100).

2.2.4 Naive Bayes

Further in terms of comparatives in this project, we leveraged the Naive Bayes model for job classification. Beginning with data preprocessing, we employed the TF-IDF (Term Frequency-Inverse Document Frequency) method to mathematically represent textual features. The Naive Bayes model, specifically the Multinomial Naive Bayes algorithm, was selected for its simplicity and effectiveness in handling text data. Through grid search, we fine-tuned the model's 'alpha' hyperparameter, governing probability smoothing. Subsequently, we evaluated the model's performance on a test set, utilizing the accuracy metric to quantify the ratio of correctly predicted job zones to the total cases examined. Furthermore, we constructed a learning curve to observe the mathematical evolution of accuracy with varying training data sizes, shedding light on the model's scalability. To visually assess its proficiency, we generated a confusion matrix, offering a clear mathematical breakdown of true positives, true negatives, false positives, and false negatives. This approach seamlessly integrated mathematical techniques and algorithms. The Naive Bayes classifier works based on the following formula;

$$P(c|d) = (P(c) * P(d|c)) / P(d)$$

This framework aimed to examine the model of job zone classification.

2.3 Semantic Similarity

To enrich inferencing hidden context in job seekers' resumes, this project leveraged searching semantic similarity between occupation and resume corpus as well as predicting the job level of an applicant. Specifically, methodology extracting embedding vectors in Siamese-BERT Networks was applied.

2.3.1 Siamese-BERT Networks

A key feature of the approach is the use of Siamese-BERT Networks (Sentence-BERT) to perform semantic similarity searches. This advanced model, an evolution of BERT, employs cross-encoder networks and is fine-tuned using Siamese and triplet network structures. It generates fixed-sized sentence embedding vectors for effective semantic analysis (Reimers et al., 2019).

2.3.2 Embedding Vectors

We produce embedding vectors for both job description corpus and resumes, allowing us to infer semantic similarities between the two. This approach is crucial in identifying occupations that match the context of a resume, thereby enabling more accurate and relevant job recommendations for job seekers.

Fig 5.

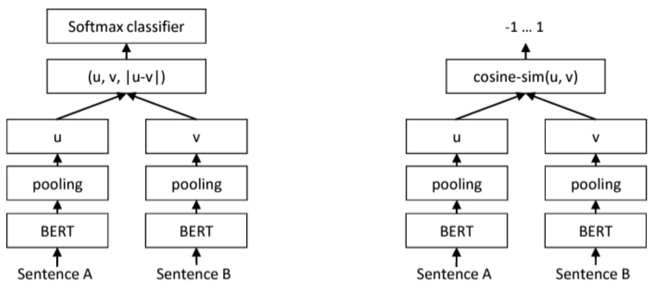


Figure 1: SBERT architecture with classification objective function, e.g., for fine-tuning on SNLI dataset. The two BERT networks have tied weights (siamese network structure).

Figure 2: SBERT architecture at inference, for example, to compute similarity scores. This architecture is also used with the regression objective function.

Fig 5. Structure of Sentence-BERT (Reimers et al., 2019)

3. Result

In this study, the performance of two neural network models and two machine learning algorithms: BERT with a Linear Layer, MLP, Logistic Regression, and Naive Bayes were assessed. Their effectiveness was evaluated in terms of accuracy with the results detailed in Table 2.

Table 2.

Evaluation Results

Algorithm	Accuracy
BERt and Linear layer	72%
MLP	69%
Logistic Regression	69%
Naive Base	66%

Table 2: Performance Metrics of the Neural Network Algorithms

As shown in Table 2, the BERT algorithm performed the best, achieving 72% accuracy. MLP and Logistic Regression performed close to BERT with 69% accuracy. While they are marginally lower by 3% in accuracy, they show drastically faster processing times and lower computational resource requirements, making them more practical for real-time applications. also performed well, with accuracy above 92% and SVM at 93%.

According to BERT confusion matrix Figure 6, most of the entries that are predicted correctly in class 1 that is ‘Entry Level’ however we have to note that this class has the highest number of entries across all levels. While class 2 that is ‘Junior‘ got the lowest class to be predicted correctly where 13 entries were predicted as 1 ‘Entry Level‘, 2 were predicted as 4 ‘Mid Level‘and 2 were predicted as 4 ‘Senior‘

Fig 6

BERT Confusion Matrix

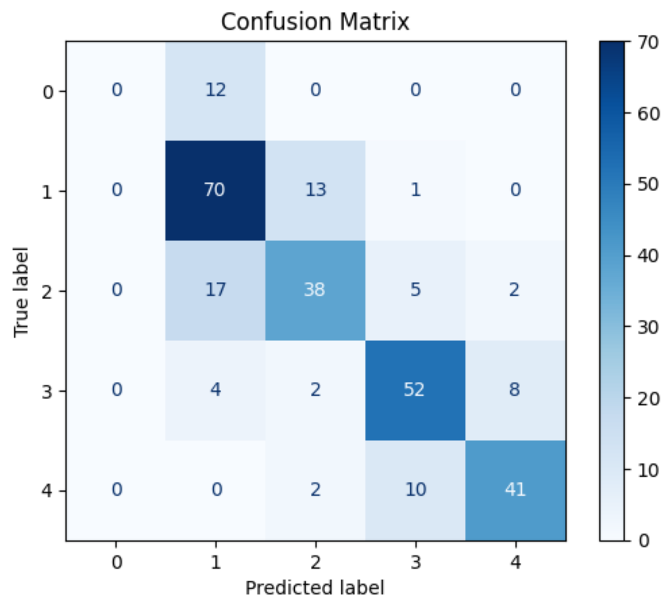


Fig 6: Confusion matrix of true and predicted labels of BERT model

The same as BERT, the MLP and Logistic Regression confusion matrix Figure 7 shows the same result but with more errors in predicting \`Entry Level\` and \`Juinior\` that it predicted 16 true \`Entry level\` as \`Juinior\` and 14 true \`Junior\` as \`Entry Level\` in the MLP model. For the Logistic Regression 19 true \`Entry level\` were predicted as \`Junior\` and 16 true \`Junior\` as \`Entry Level\`.

Fig 7.

MLP and Logistic Regression Confusion Matrix

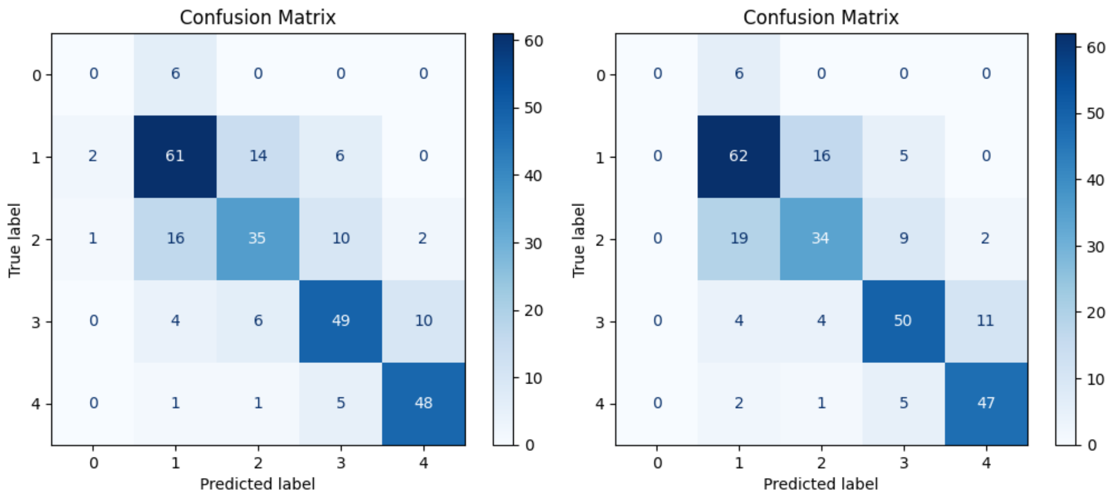


Fig 7: Confusion matrix of true and predicted labels of MLP model on the left and the Logistic Regression on the right

The learning curve for the MLP and Logistic Regression models in Figure 8, both show an improvement in training score as the number of training examples increases, indicating that the models are effectively learning from the data. However, the cross-validation score, although improving, remains consistently below the training score. This gap suggests that while the model is learning the training data well, it's not generalizing quite as effectively to new, unseen data. This implies that further model improvements might require techniques other than adding more training examples, such as feature engineering or model parameter tuning to reduce overfitting and improve their ability to generalize.

Fig 8.

MLP and Logistic Regression Learning Curve

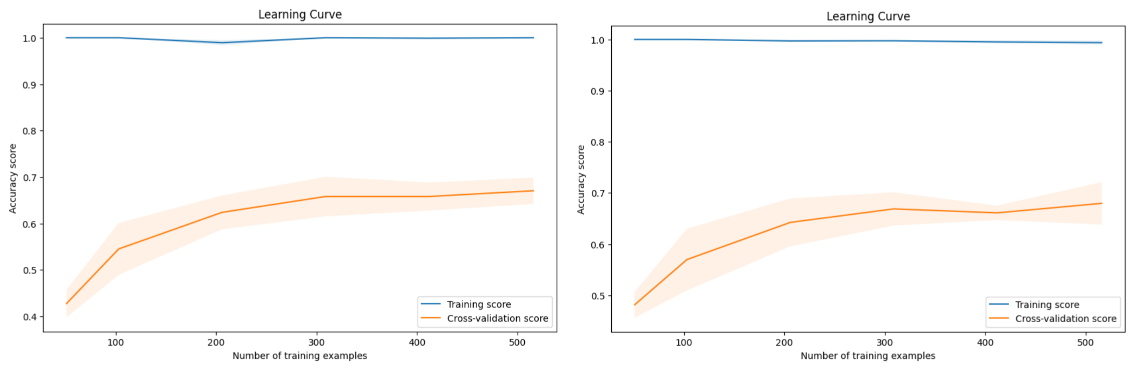


Fig 8: MLP training learning curve on the left and Logistic learning curve on the right

Fig 9.

Naive Bayes's Learning Curve

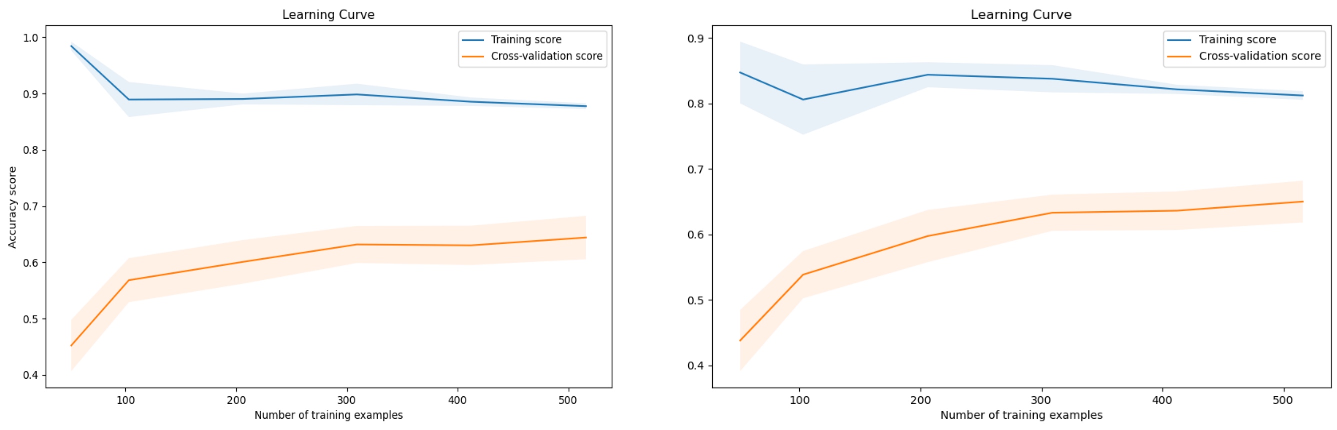


Fig 9: Contrasts learning curves: the left shows overfitting; the right reflects post-adjustment improvements

The learning curve in Figure 9 indicates that the initial implementation of the Naive Bayes model exhibited signs of overfitting, as evidenced by the substantial gap between the training and cross-validation accuracy scores. The model performed well on the training data but was unable to generalize effectively to the cross-validation set. In response to this, adjustments were made by reducing the complexity of the feature space through limiting max_features in the TfidfVectorizer to 3000 and employing a more granular grid search to optimize the alpha parameter. Additionally, the use of StratifiedKFold in the cross-validation process would have contributed to a more balanced representation of the dataset's inherent class distribution.

These modifications led to a discernible improvement in the model's generalization ability, reducing overfitting as the learning curve now presents a closer alignment between the training and cross-validation scores. As a result of these refinements, the accuracy of the model on the test set improved from 63% to 66%. This indicates not only a more reliable model but also one that is better at making predictions on unseen data. It is important to continue this iterative process of model evaluation and tuning to further enhance the model's performance, considering the specific nuances of the dataset in question.

4. Application

The Streamlit application serves as the interactive front-end of our system, designed to seamlessly integrate the sophisticated backend models with a user-friendly interface. This application is essential for processing resumes uploaded in PDF format as in Figure 9, where it extracts and structures text data for analysis, transforming unstructured resume content into a format amenable to computational processing.

Key to the application's functionality is the employment of a BERT-based classifier for predicting job levels. This classifier analyzes resumes against the backdrop of job descriptions, determining the most suitable job level for each user. The results are presented in a clear, interpretable format, aiding users in understanding their career prospects.

Additionally, the application offers job recommendations based on semantic similarity analysis. It compares the content of the user's resume with pre-encoded job descriptions, suggesting jobs that align with the user's skills and experience.

Fig 10.

Streamlit architecture

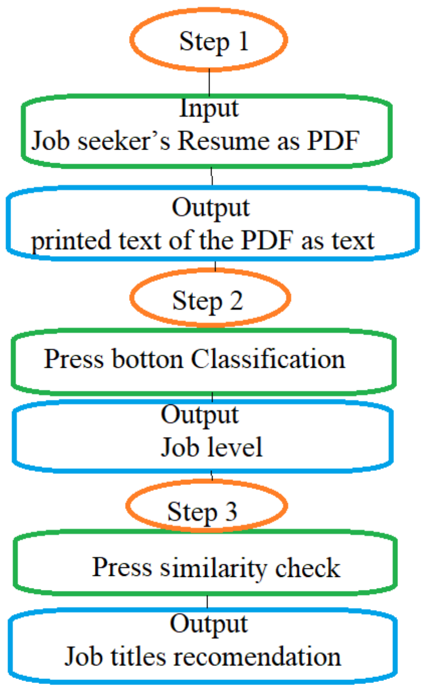


Fig 10. Final Streamlit looks like with three main functions: reading a file and tokanize, classifying job level, and suggesting job titles

This feature not only enhances job relevance but also personalizes the job-seeking experience. By helping them to gain the initiative to search specific information about the occupations that they would be interested in rather than randomly inputting keywords that would not fit them.

5. Conclusion

Natural language processing techniques such as a BERT model with a linear layer, MLP, Logistic Regression, and Naiev Base have been applied to create an advanced job classification system, achieving accuracies ranging from 64% to 72%. These models have been instrumental in accurately categorizing job levels and aligning job seekers with suitable career opportunities based on their resumes. The application of these methods promises to enhance the job search process significantly, offering a more personalized approach that aligns with the career goals and skill sets of individuals.

A limitation of this project is the reliance on a dataset that may not encompass the full diversity of job roles and descriptions found in the broader job market. The models' performance, could potentially be improved with a more extensive and varied dataset.

Reference

ChatGPT, personal communication, December 11, 2023

Cardiff NLP. (n.d.). Twitter RoBERTa Base Emotion Classification Model. Hugging Face. Retrieved from https://huggingface.co/cardiffnlp/twitter-roberta-base-emotion

Dixon, N., Goggins, M., Ho, E., Howison, M., Long, J., Northcott, E., Shen, K., & Yeats, C. (2023). Occupational models from 42 million unstructured job postings. Patterns, 4(7), Article 100757. https://doi.org/10.1016/j.patter.2023.100757

Hagan, M. T., Demuth, H. B., Beale, M. H., & Jesús Orlando De. (2016). *Neural network design*. s. n.

National Center for O*NET Development. *O*NET 28.0 Data Dictionary*. O*NET Resource Center. Retrieved December 11, 2023, from https://www.onetcenter.org/dictionary/28.0/text/

National Center for O*NET Development. *O*NET OnLine*. Retrieved December 11, 2023, from https://www.onetonline.org/

Reimers, N., & Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084.

Truong, M. (2023, January 11). Quick Semantic Search using Siamese-BERT Networks Medium.https://towardsdatascience.com/quick-semantic-search-using-siamese-bert-networks-1052e7b4df1