JOURNAL OF
EMERGING INVESTIGATORS

Article

A natural language processing approach to skill identification in the job market

Sai Harshith Suram[1], Sindhu Ghanta[2]
[1] MOT Charter High School, Middletown, Delaware
[2] AIClub, Mountain View, California

SUMMARY

The dynamic nature of the job market, with its constant growth and evolving demands, makes the assessment of skills needed for a particular job increasingly difficult, time-consuming, and tedious. This necessitates an automated approach to identify the required skills for job seekers and recruiters. This study aimed to utilize machine learning (ML) and deep learning algorithms to streamline skill identification from technology-related job descriptions. We tested the viability of the Bidirectional Encoder Representation of Transformers (BERT) language model to complete this multi-class, multi-label classification task. We hypothesized that BERT, utilizing an NLP transformer, would predict job-required skills more accurately than traditional statistical ML models. To test this, we utilized a dataset comprised of job descriptions with varying skills. We experimented with two preprocessing approaches, K-Means Clustering and Latent Dirichlet Allocation, which are both statistical ML models meant to cluster the job descriptions. We then compared the performance of the BERT tokenizer to a combination of text-to-number tokenization techniques (Term Frequency - Inverse Document Frequency and Word2Vec) and supervised classifiers (K-Nearest Neighbors, Random Forest, and Multi-Layer Perceptron). The NLP transformer-based BERT model achieved the highest accuracy. We conclude that while traditional ML models provide some insights, advanced models like BERT show potential for more accurate skill prediction since the NLP transformer model we tested outperformed the ML models. However, the complexity of real-world data necessitates further refinement and development of these technologies. Our study shows that advanced deep learning-based models have the potential to enhance the job market's efficiency by automating skill identification.

INTRODUCTION

The job market is an ever-evolving landscape, with new industries and technologies emerging all the time. Based on data from 2022, the job market is thriving, with an average of 11,012,000 job openings in the US alone (1). This growth is projected to continue in the coming years (2). As a result, it is important for both recruiters and job seekers to have access to information about the types of skills that are needed to succeed in a particular field.

Manually examining the language used in online job postings to identify the skills and qualifications that employers are looking for in candidates can be time-consuming and labor-intensive. This is especially true due to the dynamic nature of the jobs posted daily (3). To address this challenge, a data-driven approach that automates the identification of required skills is paramount. Such automation can facilitate job market analysis by people wanting to understand the market as a whole and analyze trends. Machine learning (ML) algorithms provide an opportunity to automate the process of identifying required skills within each job description, streamlining the recruitment process for both job seekers and recruiters.

ML has emerged as a powerful tool for automating tasks in various domains like image classification, deep learning, and reinforcement learning (4). Specifically, the literature on natural language processing (NLP) has seen a tremendous increase in the performance of algorithms due to the use of large language models (5). These large language models are trained on vast amounts of text data, enabling them to learn the semantics of the language. Such models can be leveraged to perform a range of NLP tasks such as text classification, question answering, and summarization (6). These advances have opened up new possibilities for automating tasks such as prediction of skill set from a job description.

To go further into depth with ML approaches to solving this problem, one study proposed a novel Multi-Graph Neural Network based Skill Prediction (MGNSP) model to learn the mapping between job position descriptions and their required skills (7). This approach represents a significant step toward the goal of developing ML algorithms that can accurately predict the skills needed for various job positions. However, there is still significant room for improvement with this approach since there are larger, more capable models that use NLP to understand the semantic and linguistic elements of the job description to provide a more comprehensive and adaptable view of skills needed for certain jobs.

Tokenization converts text into numerical data for NLP models, enabling them to process and learn from text inputs. It is essential for shallow machine learning models, which cannot interpret raw text. However, in some deep learning models, like transformers, tokenization is handled internally. There are many text processing methods to prepare data to feed into the NLP algorithms. Term-Frequency Inverse-Document Frequency (TF-IDF) is a tokenization method that assigns numbers to words based on the frequency at which each word appears across multiple documents (8). TF-IDF proves to be better than other tokenization methods such as Bag of Words (BOW) at encoding context (9). Word2Vec is another tokenization method that accounts for the

https://doi.org/10.59720/24-068

context of the word when assigning values. This contextual understanding occurs due to the words appearing in similar contexts, or surrounded by similar words, in the same document (10). Due to the ability of the Word2Vec algorithm to make mathematical relationships between words close in proximity, it could produce different results compared to other methods such as TF-IDF (11). TF-IDF and Word2Vec are traditional feature engineering techniques that are primarily used in shallow ML models.

In contrast to traditional ML models, a transformer-based encoder is used to help large-language NLP models understand the context of words. Transformers are simply ML models that account for the context of words using self-attention to focus on a singular word/phrase/token at a given time. This is done by first encoding the values into vectors of numbers and then using a decoder to understand the sequence of values and generate an output in text. Transformers are trained through transfer learning. Transfer learning utilizes a previous model trained for a specific task and applies it to a new task (12). The Bidirectional Encoder Representation of Transformers (BERT) encoder from transfer learning tokenizes the word by accounting for position, context, and attention using the BERT model (13). Specifically, in the context of a job posting, the BERT tokenizer encodes each word along with its position and surrounding context into vectors. The attention mechanism then assigns importance values to each word, determining which words in the job description are most relevant to the overall meaning and key qualifications. The BERT tokenizer that encodes words, position, and context in vectors is by far the most complex of all the tokenizers used in our experiments. When trying to interpret context, the algorithm can encode the position of a certain word in the job description from the dataset and uses an attention mechanism to assign importance values to the words (13).

In this work, we propose a multi-class, multi-label classification approach to identify skills across different job descriptions. Unlike conventional classification paradigms, which associate a single label with every instance, our multi-label paradigm allows for the assignment of multiple labels, in this case skills, to a single job description. This ensures a comprehensive assessment of the multiple skills that many modern job roles demand. We hypothesized that state-of-the-art NLP transformers, known for their advanced capabilities in understanding and processing complex language patterns, would exhibit superior accuracy in predicting the skills required for various job roles compared to conventional statistical ML models.

The primary difference between the BERT tokenizer used in this experiment and the other tokenizers used in the experiments is the attention mechanism used to assign importance values to words and understand the positional context. The BERT tokenizer paired with the BERT NLP language model was the primary NLP tested. We compared the results from the BERT NLP to other statistical ML algorithms. The BERT tokenizer cannot be used with the statistical ML models because BERT produces high-dimensional token embeddings; incorporating them into traditional statistical ML models can lead to the "curse of dimensionality" (15). This is because these traditional models struggle with high-dimensional input features, which can cause performance degradation and increase computation time (16).

We hypothesized that the transformer framework discussed above, utilizing the latest advances in NLP, especially BERT, would yield better predictive performance on skill classifications for job description data when compared to conventional statistical ML models. The sophisticated transformer-based model paired with the understanding of positional context paired with the attention mechanism will likely produce more accurate results than tokenization methods such as Word2Vec and TF-IDF (11, 14). Our results show that transformer-based models such as BERT are more capable of extracting information from the text than traditional statistical ML models. This would allow job candidates to stop worrying about deciphering the complexities of job descriptions and focus on building their profiles.

RESULTS

To build a ML model that can predict required skills based on a job description, we utilized a dataset with a job description and skills column, where the number of skills as well as how they were described varied per job description. The data was taken from job listings online and put into a spreadsheet for analysis. Our dataset was split into training (60%), testing (20%), and validation sets (20%). The complexity of this dataset is in line with the complexity expected of real-world datasets. However, to predict the relevant skills for a given job description, it was necessary to preprocess the skills column in the dataset and transform it into a categorical outcome that indicates the types of skills needed (**Figure 1**).

The words in the job description were converted into numbers in order to conduct complex computations on the vectors of numbers. The job descriptions in the dataset were converted into a vector of numbers using TF-IDF, Word2Vec, or BERT tokenizers. We conducted experiments using all three tokenization techniques so that the performance of the models in each case could be compared. Following this pre-processing step combined with K-Means or LDA, the model can predict the presence of each skill cluster or topic for a given job description. However, this task is more complex

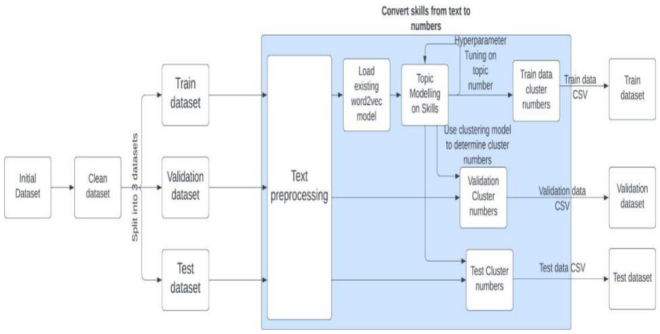

Figure 1: Dataset pre-processing with LDA. Flowchart of the dataset pre-processing pipeline using Latent Dirichlet Allocation (LDA) for topic modeling of skills. The initial dataset is cleaned and split into training, validation, and test subsets. The text preprocessing steps within the blue box include loading a pre-trained Word2Vec model, applying LDA for topic extraction, and tuning the number of topics. A clustering model then assigns cluster numbers to each dataset, which are saved into CSV files and reintegrated into the respective datasets. This process converts skills from text to numerical clusters, enabling further analysis and model training.

https://doi.org/10.59720/24-068

than simple classification, as each job description may require a varying number of skills, rather than just one. The sheer number of different skills involved in this task prevents it from being cast as a mere multi-label problem. The algorithm's goal is to predict the set of skills, rather than a single skill.
We found that there were multiple clusters of skills that tended to co-occur. Given that ML classification algorithms need distinct categories, we found it necessary to map the skills into distinct clusters or categories. We used two different algorithms to accomplish this: K-Means Clustering, and Latent Dirichlet Allocation (LDA) (17, 18).

For the traditional ML models in our study, we combined one of three supervised classifiers (K-Nearest Neighbors (KNN), Random Forest, or Multi-Layer Perceptron (MLP)) with one of two tokenizers (TF-IDF or Word2Vec). These traditional models were compared to the BERT NLP model. We aimed to cover a broad spectrum of ML approaches based on simple rules like nearest neighbors, ensemble decision trees, and neural networks. KNN is straightforward and effective, and it is especially useful for its ability to handle multiclass problems and provide insights into the dataset's structure through its instance-based learning (19). Random Forest was chosen for its robustness and performance with high-dimensional data, as it reduces overfitting risks and enhances the model's generalizability through ensemble learning (20). Lastly, MLP allowed us to explore deep learning capabilities and is particularly useful for capturing nonlinear relationships in data through its layered architecture (21). These models collectively cover a range of different types of model architectures. The use of these statistical ML models is to compare performance to a much more sophisticated BERT NLP model.

K-Means Results

The first set of experiments utilized K-means clustering to group the job skills. We first converted the skills into numerical representations using Word2vec or TF-IDF and then used K-means clustering to group them. Since the optimal value of K is unknown, we tested a range of K values to evaluate the distortion, or the average distance between the centers of different clusters as calculated by the algorithm (**Figure 2a**). We used the elbow method to determine the optimal K value, which balances between a small number of clusters and a low distortion score. We found the optimal K value to be 15. We saw that there was an uneven distribution in the number of skills contained in the clusters (**Figure 2b**). This could cause some differences in accuracy between the models using K-Means and LDA methods.

Next, we used KNN, Random Forest, and MLP to create the ML model. For the KNN algorithm, we tuned the hyper-parameter K between 1 and 10. The model performed the best when K = 1 regardless of the preprocessing method used. The KNN algorithm using TF-IDF tokenization had a testing accuracy of 24.96% compared to a testing accuracy value of 23.84% when using Word2Vec tokenization (**Figure 2c-d**). For the random forest algorithm, the number-of-trees and depth were tuned from 5–100 and 5–50, respectively. The best model, achieving a validation accuracy of 30.83%, used TF-IDF tokenization with 100 trees and a depth of 45 (**Figure 3a**). A similar validation accuracy of 28.80% was achieved using the same random forest algorithm and hyper-parameters with Word2Vec tokenization (**Figure 3c**). For the MLP algorithm, we used a hidden layer size of 100 and tuned the learning rate and epochs from 20–100 and 0.00001–0.01. The best result of 21.66% validation accuracy was achieved with a learning rate of 0.001 run over 100 epochs with TF-IDF tokenization (**Figure 3b**). Using a Word2Vec tokenization for the same algorithm resulted in a maximum validation accuracy of 19.11% (**Figure 3d**).

We then compared our three traditional models to the BERT NLP, which uses its own tokenizer to convert text into numbers after K-means clustering to group job skills. We used the DistilBERT tokenizer followed by the 'TFDistilBertModel' pre-trained model with custom layers added to it. The hyper-

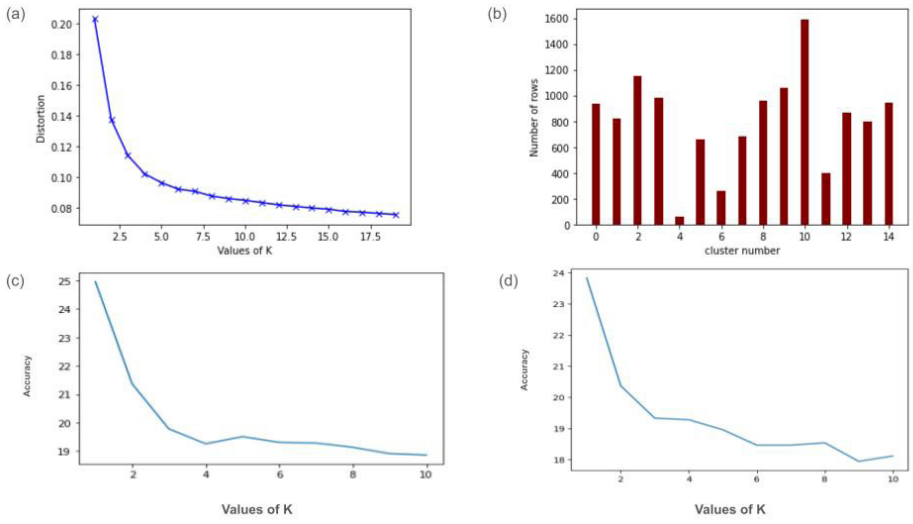

Figure 2: K-means performance metrics for model selection and accuracy of TF-IDF and Word2Vec tokenization methods using KNN. (a) The K-means elbow plot shows the relationship between the number of clusters and the distortion scores where the ideal score is around 0.08 to 0.09. **(b)** The number of skills in each cluster/group created by the K-means clustering algorithm when the number of clusters was 15. **(c)** Accuracy of the KNN algorithm for varying values of K after TF-IDF tokenization. **(d)** Accuracy of the KNN algorithm for varying values of K after Word2Vec tokenization.

https://doi.org/10.59720/24-068

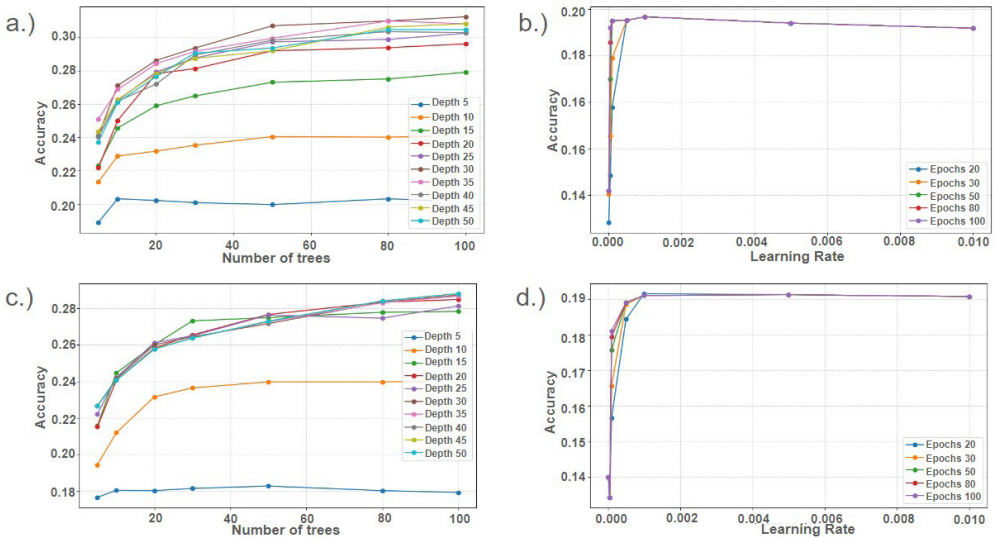

Figure 3: Accuracy of the K-Means algorithms using statistical ML models (RF, MLP). (a) Accuracy of the random-forest algorithm after TF-IDF tokenization for different numbers of trees and depths. **(b)** Accuracy values of the MLP algorithm for varying learning rates and number of epochs after TF-IDF tokenization. **(c)** Accuracy of the random-forest algorithm for different numbers of trees and depths after Word2Vec tokenization. **(d)** Accuracy values of the MLP algorithm for varying learning rates and number of epochs after Word2Vec tokenization.

parameters 'base_learning_rate' and 'epochs' were tuned from 0.00001–0.01 and 20–100, respectively. We were able to achieve a maximum testing accuracy value of 21.42% when the 'base_learning_rate' value was 0.005 and the epochs value was 100.

In this section of experiments, we found that BERT slightly underperformed the best statistical ML model using Random Forest, which performed at 31.76% testing accuracy while BERT performed at 21.42% testing accuracy. However, there was not much of a difference in accuracies between the Random Forest algorithm's performance when using either Word2Vec or TF-IDF tokenization. By comparing the two testing accuracies, we can conclude that BERT with K-Means was not the best-performing model. However, it appears that the tokenization technique plays a more pivotal role in the MLP algorithm as the accuracies were slightly more varied between TF-IDF and Word2Vec tokenization.

LDA Results

In the second set of experiments, we used LDA to group the skills, where the skills were represented by a distribution of topics inferred by the LDA model. Similar to K-means, LDA also requires the user to specify the number of topics. To determine the ideal number of topics, we calculated the coherence value of different topic numbers. The primary purpose of the coherence value is to see how similar words are within a given context. The score measures the strength of the semantic relationships between words. Using these coherence values, we found that the ideal number of topics for LDA topic modeling was 25 (**Figure 4a**). With the implementation of 25 clusters in LDA, there was still some variation in the number of rows in the job description dataset corresponding with each skill cluster (**Figure 4b**). This is expected since topic models do not aim to create topics with a perfectly even distribution but rather to make relevant topics to suit the dataset.

After LDA was used to convert the skills in the job descriptions into a fixed number of topics, we used TF-IDF and Word2Vec techniques to convert the text in the job description into numerical values. We then used KNN, Random Forest, and MLP to create the ML models.

For the KNN algorithm, hyper-parameter K was tuned between 1 and 10. Using TF-IDF, the model performed the best at K = 1 with a validation accuracy of 10.06% (**Figure 4c**). However, using Word2Vec resulted in a lower maximum validation accuracy of 5.06% at K = 7 (**Figure 4d**). For the Random Forest algorithm, hyper-parameters number-of-trees and depth were tuned from 5–100 and 5–50, respectively. Using TF-IDF tokenization, 50 trees and a depth of 35 resulted in the best model performance with a validation accuracy of 13.19% (**Figure 5a**). However, the Word2Vec tokenization with the same Random Forest model achieved a higher validation accuracy value of 28.80% (**Figure 5c**). For the MLP algorithm, hyper-parameters learning rate and epochs were tuned from 20–100 and 0.00001–0.01, respectively. Using Word2Vec tokenization, a maximum validation accuracy value of 12.10% was achieved with a learning rate of 0.0001 run between 50 to 100 epochs (**Figure 5b**). However, using a TF-IDF tokenization method for the same MLP model resulted in a validation accuracy of 15.10% (**Figure 5d**).

We then compared these traditional models to BERT. We used the DistilBERT tokenizer followed by the 'TFDistilBertModel' pre-trained model with custom layers added to it. The hyper-parameters 'base_learning_rate' and 'epochs' were tuned from 0.00001–0.01 and 20–100, respectively. This resulted in the highest validation accuracy value of 31.97% when the 'base_learning_rate' value was 0.00001 and the epochs value was 100. The validation accuracy tended to increase as the learning rate initially increased from 0.00001 to 0.0005, reaching a peak at 0.0005 for earlier epochs from 20 to 30. However, as the learning rate further increased to 0.01, the validation accuracy started to plateau and even slightly decline. This was particularly evident in longer training durations (80 and 100 epochs). Increasing the number of epochs generally leads to an improvement in model accuracy across all learning rates (22). However, when

https://doi.org/10.59720/24-068

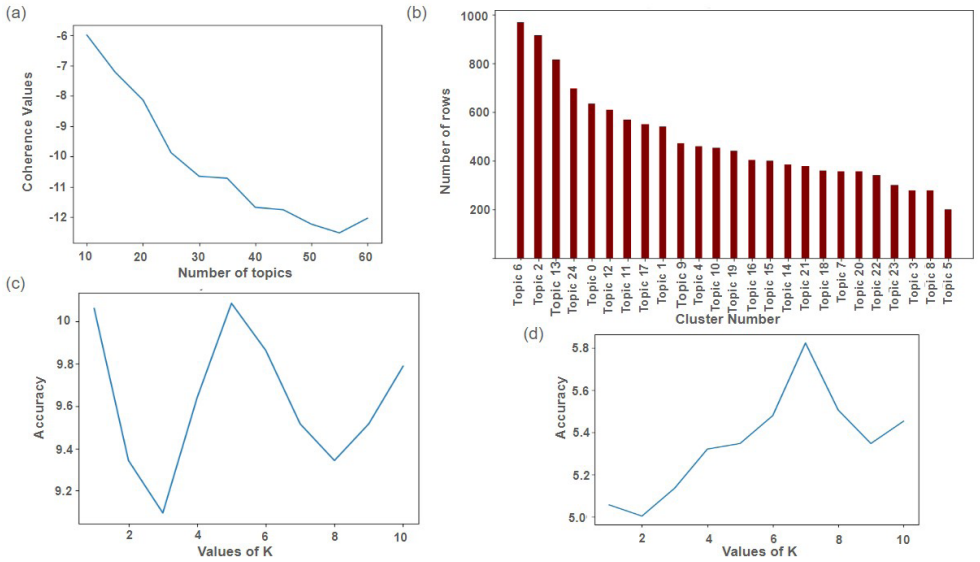

Figure 4: LDA performance metrics aiding in model selection. LDA was used to convert the skills columns into categories. (a) The LDA coherence value plot shows the coherence scores of using a varying number of topics. An increase in the number of topics showed diminishing returns. **(b) D**istribution of 25 topics in the dataset. **(c)** Accuracy values of the KNN algorithm for different values of K after TF-IDF pre-processing and clustering values using LDA. **(d)** Accuracy values of the KNN algorithm for different values of K after Word2Vec pre-processing and clustering with LDA.

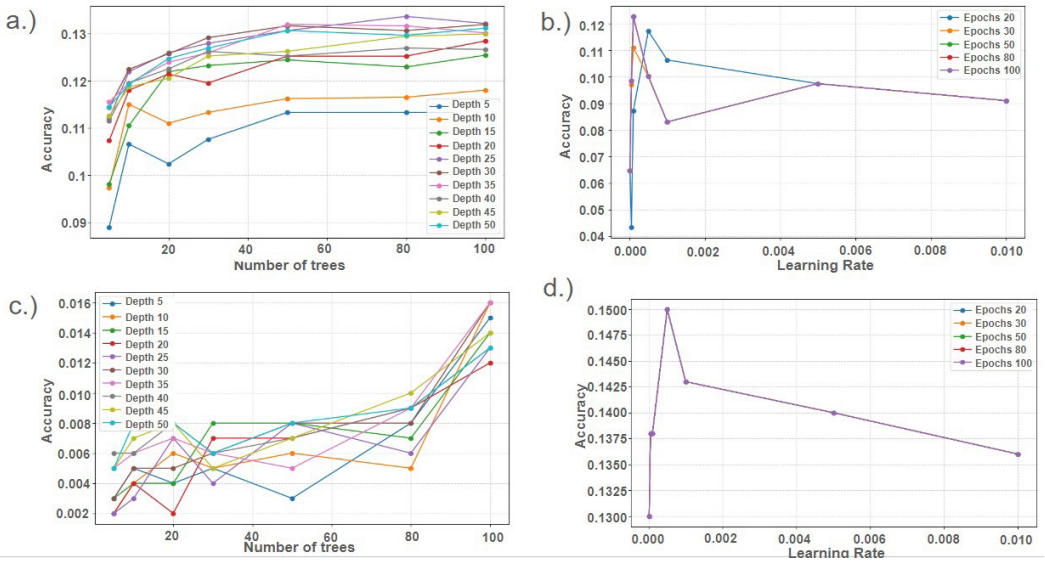

Figure 5: Accuracy of the LDA algorithms using statistical ML models (RF, MLP). (a) Application of the random-forest algorithm with TF-IDF tokenization with LDA. **(b)** Accuracy values of the MLP algorithm with TF-IDF tokenization in LDA. **(c)** Use of the random-forest algorithm with Word2Vec tokenization in LDA. **(d)** Accuracy values of the MLP algorithm with Word2Vec tokenization in LDA.

we increased the number of epochs to 150, we did not see an improvement in accuracy and hence stopped the experiments at 100 epochs. Even though the validation accuracy value was low, an inspection of the confusion matrix showed that some topics had very low performance.

In summary, the transformer-based BERT model was best able to predict required skills based on job descriptions, particularly when skills were predicted as one of 25 categories. The testing accuracy value was 47.19% (**Figure 6**). In addition to the accuracy, all categories' average precision and recall were 43.48% and 43.73%, respectively (**Figure 7**). This validated our hypothesis that transformer-based deep learning algorithms such as BERT perform better than traditional statistical algorithms in predicting the skills required for a given job description. The primary difference between the BERT results from K-Means and LDA is that LDA topic modeling created 25 unique and useful topic classifications whereas K-Means clustering created 15 topics. There was also a significant difference in validation accuracies between the K-Means and LDA models, with the K-Means approach having a much lower validation accuracy than the LDA approach.

https://doi.org/10.59720/24-068

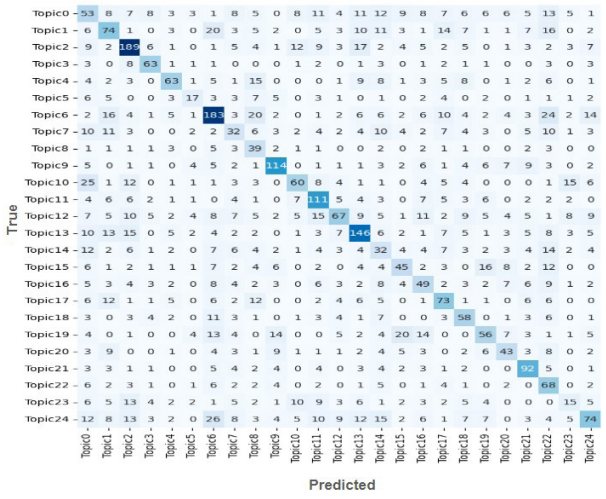

Figure 6: Test confusion matrix using BERT with LDA classified and pre-processed data. Test confusion matrix obtained using the BERT algorithm following clustering using LDA and BERT tokenization to predict the cluster numbers. The algorithm achieved a 47.19% accuracy with 25 topics.

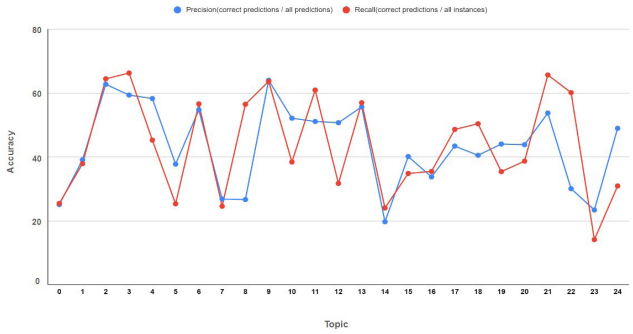

Figure 7: Precision and Recall values for BERT by class. Precision (blue, correct predictions to total predictions) and recall (red, correct predictions to total instances) metrics for each topic. The model's performance is evaluated after applying LDA for topic modeling with the BERT classifier and the BERT tokenizer, demonstrating variations in prediction accuracy across different clusters.

DISCUSSION

Our study integrates and expands upon previous research demonstrating the efficacy of advanced ML models, particularly BERT, in interpreting complex text data within job descriptions. When K-Means was used to cluster job descriptions' skills into groups, the KNN algorithm performed slightly better using TF-IDF with a validation accuracy of 24.96% compared to Word2Vec with a validation accuracy of 23.84%. Similarly, using LDA, TF-IDF featurization resulted in an optimal validation accuracy of 10.06% versus 5.06% for Word2Vec for the KNN algorithm. Overall, we saw that the KNN algorithm performed better when the job description was featurized using TF-IDF compared to Word2Vec in both K-Means and LDA experiments. Analysis of the random forest algorithm in K-Means experiments showed that TF-IDF featurization yielded a validation accuracy of 30.83% while the same random forest algorithm run on Word2Vec had a validation accuracy of only 28.80%. LDA also showed similar results with TF-IDF having better accuracy values with the Random Forest algorithm than Word2Vec. Lastly, the MLP algorithm in K-Means had a validation accuracy of 21.66% when used with the TF-IDF tokenizer. When used with Word2Vec, the validation accuracy dropped to 19.11%. A similar pattern was seen in LDA as well, where the TF-IDF tokenizer yielded higher accuracy than the Word2Vec tokenization. This is an interesting result because Word2Vec is considered to be a more sophisticated technique to convert text to numbers since Word2Vec has the ability to encode context more clearly (10). Utilizing a pre-built Word2Vec model trained on millions of documents like Wikipedia enables the model to have a huge vocabulary with meaningful associations between words. However, in the case of this dataset, the words that appear in job descriptions might be unique enough that they are not a part of the Word2Vec vocabulary. This may be why TF-IDF, which learns the vocabulary based on training data, performed better.

One of the more surprising results from our study was the relatively low performance improvement when utilizing advanced NLP techniques, such as BERT, compared to more traditional methods like TF-IDF and Word2Vec in certain phases of our experiments. Although BERT paired with the LDA topic modeling generally outperformed other models, the margin of improvement was smaller than anticipated. This was particularly true when using K-Means, where BERT achieved only slightly better accuracy than the random forest algorithm run on TF-IDF.

However, overall, the statistical ML models produced suboptimal results compared to the testing accuracy of the best-performing BERT model that used LDA; therefore, this reinforces our decision to use BERT as our primary NLP algorithm as opposed to statistical ML models. In the K-Means experiments, BERT achieved a testing accuracy of 21.42%, which did not seem to severely outperform the statistical ML algorithms just reviewed. However, the performance of BERT improved when using LDA, with the highest testing accuracy value at 47.19%, which was better than the other ML models. This relatively low accuracy value can partly be attributed to the complex and noisy nature of the job descriptions in the dataset. Job descriptions don't typically have uniform structure and contain a wide variety of language, making it difficult for even the most sophisticated models to consistently interpret them correctly. The inherent variability in how different roles and skills are described across industries could lead to challenges in model training, where the model struggles to generalize from the training data to unseen job descriptions. Additionally, the use of pre-trained models like BERT and the other statistical ML models, which are not specifically tuned for the domain of job descriptions, may not capture all the nuances needed to accurately predict skills from these texts. This suggests a need for more domain-specific pre-training or fine-tuning on job-related texts.

Another unexpected pattern we noticed was that BERT performed much better with LDA clustering than K-Means clustering. However, the performance improvement on BERT from using LDA was not reflected in the other ML algorithms (KNN, random forest, MLP). In some cases, LDA clustering made the accuracy values for the ML algorithms go down significantly. This might be because LDA has a much more

https://doi.org/10.59720/24-068

advanced clustering mechanism that was not detected by the standard ML models but was more easily recognized by BERT. The advanced capabilities of BERT likely made it easier for the model to recognize the more sophisticated clustering patterns of LDA.

The primary limitation of this study is the reliance on a single source of job descriptions that were all related to technology. This could have introduced a bias based on the style of writing specific to that source and may not generalize well across different platforms or job markets. Moreover, the conversion of skills into numerical clusters or topics might oversimplify the rich, nuanced information skills descriptions can provide, potentially leading to a loss of valuable insights about the interrelationships and subtleties among different skills.

The findings of Lê Văn Duyệt, et al. on Skill2vec highlighted the potential of neural network architectures derived from Word2vec for extracting skill-related information from job descriptions (24). Both our study and the Skill2Vec study underscore the importance of contextual understanding in text analytics within the recruitment domain, albeit using different technological approaches. Our use of BERT and transformer-based models drew a direct parallel to Skill2vec's reliance on neural networks inspired by Word2Vec. However, where Skill2vec focuses on generating a new vector space to explore skill relationships, our approach leveraged the pre-trained capabilities of BERT to enhance classification accuracy directly from textual descriptions, thereby reducing the need for extensive pre-processing typically associated with traditional models like KNN or Random Forest. Our study has made clear improvements compared to this approach since we are using a pre-trained model that has been trained on various types of data across the internet making it more capable of finding useful patterns for our classification task. While the Skill2Vec model claimed that 22% of the skills extracted were irrelevant, our model extracts only relevant skills and assigns probability values to these skills thereby ranking them by importance. After probability values have been assigned, the most relevant skills are assigned to the job description. Therefore, our model offers a unique approach to skill extraction.

In the future, we hope to develop more sophisticated methods for preprocessing and feature extraction that can better capture the complexities and variations in job descriptions and enhance model performance. Incorporating more data from other web sources could result in a more representative dataset that would help capture the variations and nuances in job postings across different websites. Our work shows how a transformer-based model paired with semantic analysis is superior compared to statistical ML models for multi-class, multi-label problems for analyzing convoluted textual data. This observation holds true specifically when using BERT to conduct multi-class, multi-label semantic analysis with the utilization of topic modeling algorithms such as LDA.

MATERIALS AND METHODS

Dataset

The dataset used for this study consisted of 22,000 job listings. These job listings were extracted from Dice.com, a prominent US-based technology job board, using Prompt Clouds's in-house web crawling service to store the data in Kaggle. The dataset we obtained from Kaggle contained 12 columns, including 'advertiseurl', 'company', 'employmenttype_jobstatus', 'jobdescription', 'jobid', 'joblocation_address', 'jobtitle', 'postdate', 'shift', 'site_name', 'uniq_id', and 'skills'. There were 1331 unique skills across all columns. The purpose of our ML and NLP models was to validate the selection of these skills done by the web crawler and train a more generalized model that can do the same tasks for any provided job description.

Dataset preprocessing

To ensure accurate model evaluation, we split the dataset into three distinct parts: training (60% of the dataset), testing (20% of the dataset), and validation (20% of the dataset). Text preprocessing techniques were fit on the training set and applied to two other parts of the dataset to ensure uniformity and consistency. The validation set was utilized to assess the model's accuracy during hyper-parameter tuning on the training dataset, and the testing set was used to evaluate the final model performance.

There are several methods of text preprocessing that were used to prepare the dataset for learning the relationships between job descriptions and skills. All punctuation, unnecessary whitespace, and null values were removed from the dataset using a Python script. Then, we removed the columns that provided no information regarding the job that would help with skills prediction. These columns included 'advertiserurl', 'site_name', 'jobid', 'uniq_id', and 'shift'. In the third step, we utilized another Python script to extract the individual skills using comma separation. Skills exceeding 15 characters were filtered out to eliminate any extraneous and excessively long skills. The length of this list of unique skills was 1,331.

In the first method, we converted each skill into a vector of numbers using the Word2Vec algorithm by Gensim or the TF-IDF algorithm by Sci-Kit Learn performed by creating Python scripts with the open-source packages specified above to perform the text tokenization on the data. Then, the algorithm splits everything into various clusters/topics by applying the K-means clustering algorithm using Sci-Kit Learn to assign a cluster of skills to each job description instead of individual skills. To determine the ideal number of clusters, we employed the elbow method while experimenting with various cluster numbers since increasing the number of clusters from 10 to 15 reduced the distortion value by less than 0.02 points (23).

In the second phase of the experiment, we replaced the K-Means clustering algorithm with LDA. Unlike K-Means, which clusters based on numerical values, LDA was chosen for its more advanced ability to create labeled topics. When using LDA, we utilized Sci-Kit Learn's LDA package to extract topics and their distribution across each set of skills in each column. We applied LDA to the skill sets in each column after tokenizing with Word2Vec/TF-IDF, and the resulting topic distribution was assigned to each job description. We selected the number of topics based on a graph of coherence scores. A balance was required between achieving a low coherence score and maintaining a manageable number of topics. While a low coherence score often results from having many topics, too many topics can reduce the usefulness of classifications by creating overly similar categories. It was important to carefully judge this balance to ensure the topics were both distinct and meaningful. Since many statistical ML

https://doi.org/10.59720/24-068

algorithms are incapable of handling multi-label classification, we made sure LDA only predicted a single class for each job description making the task easier to handle while retaining accuracy. We ensured this single class prediction by taking the probabilities of all classes and only assigning the highest probability class to the given job description.

The next step was to convert the job description column into numbers since ML algorithms only understand numerical data. Three different methods were employed in this step: TF-IDF, Word2Vec, and BERT tokenization, the only tokenization tool provided by the HuggingFace package.

As a result, many combinations of the dataset were generated with different clustering and tokenization techniques. These were used in training ML models. The TF-IDF and Word2Vec tokenization techniques were used in conjunction with the statistical ML models (KNN, Random Forest, and MLP) provided by Sci-Kit Learn, while BERT tokenization was utilized exclusively with the BERT algorithms.

ML Algorithms

For the traditional ML models, tokenization techniques such as TF-IDF and Word2Vec were utilized, resulting in six distinct models. Hyper-parameter tuning was performed on each model to identify the model with the highest accuracy value using a validation dataset, followed by testing the best model in each of the six cases with a separate test dataset.
In the case of KNN, the hyper-parameter K was varied between 1 and 10. In the case of random forest, the hyper-parameter number of trees and depth were varied from 5–100 and 5– 50, respectively. In the case of MLP, the two parameters tuned were learning rate and epochs. The learning rate was tuned between 0.00001 and 0.01 while the epochs were tuned from 20 to 100.

In the case of deep learning models, BERT was used for tokenization as well as training the ML algorithm. The transformer-based model was constructed on top of a pre-trained transformer taken from the HuggingFace open-source platform (25). Nine trainable layers were added to the initial pre-trained model to tailor it for a multi-class classification task. The pre-trained model's weights were frozen to avoid tampering with the imported model. The custom layer begins with two distinct input layers, one each for input IDs and input attention. The transformer's output is then processed to extract the hidden state of the classification (CLS) token, which is passed through a series of dense layers, each followed by a dropout layer for regularization. These layers consisted of two dense layers with 256 and 64 neurons, respectively, each using a LeakyReLU activation function and GlorotNormal weight initialization. The final output layer was a dense layer with 15 neurons (corresponding to 15 classes) and used a softmax activation function for multi-class classification. The model, compiled with an Adaptive moment estimation (Adam) optimizer and mean squared error loss function, aims to optimize classification accuracy. There were three additional dense layers added to the BERT model, one of which was an output layer containing the same number of neurons as the number of predictable classes. Hyper-parameters batch size and learning rate were tuned from 16–128 and 0.0001–0.01, respectively, and evaluated using the validation dataset to determine the model with the highest performance.

Received: January 23, 2024
Accepted: May 6, 2024
Published: October 29, 2024

REFERENCES

1. "Table 1. Job Openings Levels and Rates by Industry and Region, Seasonally Adjusted - 2023 M12 Results." *U.S. Bureau of Labor Statistics*, 30 January 2024, www.bls.gov/news.release/jolts.t01.htm.
2. "Overview of Projections to 2024: Monthly Labor Review." *U.S. Bureau of Labor Statistics*, Dec. 2015, www.bls.gov/opub/mlr/2015/article/overview-of-projections-to-2024.htm#:~:text=Total%20employment%20is%20expected%20to,160.3%20million%20jobs%20by%202024.
3. Deming, David J., and Kadeem Noray. "Earnings dynamics, changing job skills, and STEM careers." *The Quarterly Journal of Economics,* vol. 135, no. 4, 16 June 2020, pp. 1965-2005. https://doi.org/10.1093/qje/qjaa021.
4. Condie, Tyson, et al. "Machine learning for big data." *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, 22 June 2013, pp. 939-942, https://doi.org/10.1145/2463676.2465338.
5. Hoffmann, Jordan, et al. "Training Compute-Optimal Large Language Models." *arXiv*, 29 March 2022, https://doi.org/10.48550/arXiv.2203.15556.
6. Su, Dan, et al. "CAiRE-COVID: A question answering and query-focused multi-document summarization system for COVID-19 scholarly information management." *arXiv,* 8 December 2020, https://doi.org/10.48550/arXiv.2005.03975.
7. Liu, Liting, et al. "Learning multi-graph neural network for data-driven job skill prediction." *2021 International Joint Conference on Neural Networks*, 18 July 2021, https://doi.org/10.1109/IJCNN52387.2021.9533402.
8. Qaiser, Shahzad and Ali, Ramsha. "Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents." *International Journal of Computer Applications*. vol. 181, no. 1, 1 July 2018, https://doi.org/10.5120/ijca2018917395.
9. Akuma, Stephen, et al. "Comparing Bag of Words and TF-IDF with different models for hate speech detection from live tweets." *International Journal of Information Technology,* vol. 14, no. 7, 21 September 2022, https://doi.org/10.1007/s41870-022-01096-4.
10. Herremans, Dorien and Ching Chuan. "Modeling Musical Context with Word2vec." *arXiv*, 29 June 2017, https://doi.org/10.48550/arXiv.1706.09088.
11. Cahyani, Denis E., and Irene Patasik. "Performance comparison of tf-idf and word2vec models for emotion text classification." *Bulletin of Electrical Engineering and Informatics,* vol. 10, no. 5, 1 October 2021, https://doi.org/10.11591/eei.v10i5.3157.
12. Torrey, Lisa, and Jude Shavlik. "Transfer learning." *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, edited by E. Sora, et al., IGI Global, 2009, pp. 242-264.
13. Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems,* vol. 30, 2 August 2023, https://doi.org/10.48550/arXiv.1706.03762.
14. Jamshidian, Mina. "Evaluation of Text Transformers

https://doi.org/10.59720/24-068

for Classifying Sentiment of Reviews by Using TF-IDF, BERT (word embedding), SBERT (sentence embedding) with Support Vector Machine Evaluation." *Technological University Dublin*, 2022.
15. Köppen, Mario. "The curse of dimensionality." *5th Online World Conference on Soft Computing in Industrial Applications*. vol. 1, 17 April 2009, https://doi.org/10.1007/978-0-387-39940-9_133.
16. Berisha, Visar, et al. "Digital medicine and the curse of dimensionality." *NPJ Digital Medicine,* vol. 4, no. 1, 28 October 2021, p. 153, https://doi.org/10.1038/s41746-021-00521-5.
17. Blei, David M., et al. "Latent dirichlet allocation." *Journal of Machine Learning Research,* vol. 3, 1 March 2003, pp. 993-1022. https://doi.org/10.5555/JMLR.944919.944937.
18. Ahmed, Mohiuddin, et al. "The K-Means Algorithm: A Comprehensive Survey and Performance Evaluation." *Electronics*, vol. 9, no. 8, 12 August 2020, p. 1295, https://doi.org/10.3390/electronics9081295.
19. Cover, Thomas M., and Peter E. Hart. "A Study on the Nearest Neighbor Rule." *IEEE Transactions on Information Theory*, vol. 13, no. 1, 1967, pp. 21-27, https://doi.org/10.1109/TIT.1967.1053964.
20. Breiman, Leo "Random Forests." *Machine Learning*, vol. 45, no. 1, 2001, pp. 5-32, https://doi.org/10.1023/A:1010933404324.
21. LeCun, Yann, et al. "Backpropagation Applied to Handwritten Zip Code Recognition." *Neural Computation*, vol. 1, no. 4, 1989, pp. 541-551, https://doi.org/10.1162/neco.1989.1.4.541.
22. Xi, Haibin, and Wenjing Wang. "Deep Learning Based Uterine Fibroid Detection in Ultrasound Images." *BMC Medical Imaging*, 19 Aug. 2024, https://doi.org/10.1186/s12880-024-01389-z.
23. Lê, Văn Duyệt, et al. "Skill2vec: Machine Learning Approaches for Determining the Relevant Skill from Job Description" *arXiv,* 31 July 2017, https://doi.org/10.48550/arXiv.1707.09751.
24. Syakur, Muhammad. A., et al. "Integration k-means clustering method and elbow method for identification of the best customer profile cluster." *IOP Conference Series: Materials Science and Engineering*. vol. 336, 1 April 2018, https://doi.org/10.1088/1757-899X/336/1/012017.
25. "Distilbert/Distilbert-Base-Uncased · Hugging Face." *Distilbert/Distilbert-Base-Uncased · Hugging Face*, huggingface.co/distilbert/distilbert-base-uncased. Accessed 20 Aug. 2024.