Proceedings of the International Conference on Sustainable Computing and Smart Systems (ICSCSS 2023)
IEEE Xplore Part Number: CFP23DJ3-ART; ISBN: 979-8-3503-3360-2

2023 International Conference on Sustainable Computing and Smart Systems (ICSCSS) | 979-8-3503-3360-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICSCSS57650.2023.10169716

Resume Screening and Ranking using Convolutional Neural Network

Prof. Sonali Mhatre
Information Technology
Bharati Vidyapeeth College of Engineering
Kharghar, India
sonalinmhatre@gmail.com

Prof. Bhawana Dakhare
Information Technology
Bharati Vidyapeeth College of Engineering
Kharghar, India
bhawanadakhare@gmail.com

Vaibhav Ankolekar
Information Technology
Bharati Vidyapeeth College of Engineering
Kharghar, India
vaibhavank18@gmail.com

Neha Chogale
Information Technology
Bharati Vidyapeeth College of Engineering
Kharghar, India
nehachogale795@gmail.com

Rutuja Navghane
Information Technology
Bharati Vidyapeeth College of Engineering
Kharghar, India
rutujanavghane2501@gmail.com

Pooja Gotarne
Information Technology
Bharati Vidyapeeth College of Engineering
Kharghar, India
poojapgotarne@gmail.com

***Abstract*—Manual filtering becomes a tedious task for the recruiter as there are thousands of candidates applying for a single job posting. In this paper, a Long Short-term Memory (LSTM)-based and Convolutional Neural Network (CNN)-based approach are introduced for categorizing resumes submitted by candidates. Along with proposed CNN and LSTM model, also implemented other baseline models such as Random Forest, K-nearest neighbor, and Support Vector Machine. The resumes of the candidates and the job description are compared using Cosine Similarity. The list of job applicants is ranked based on the percentage obtained by Cosine Similarity.**

***Keywords*—Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Term Frequency - Inverse Document Frequency(TF-IDF), Cosine Similarity, Text Vectorization**

I. Introduction

In today's world, thousands and thousands of candidates apply for a single job. The recruiter cannot interview all the candidates applying for the job. It becomes tedious work for the recruiter to shortlist candidates for interview. While applying for the job, the candidates upload their resume to the job portal. In the traditional method, the recruiter has to go through all the resumes and look for the skills the job requires. After manually screening all the resumes, the recruiter has to shortlist only a few candidates who have the proper skills and are qualified for the job.

Candidates mention their skills in the resume and also mention some information about themselves that is not relevant for the job for which candidates are applying. As the recruiter has to shortlist the candidates having suitable skills for the job, recruiter have to read all the resumes thoroughly to determine how valuable each candidate is. The recruiter's time is wasted reading all the irrelevant information the candidates mentioned in the resume. Here aim is to create a system that automates the shortlisting of candidates from the thousands and thousands of job applicants. This reduced the work of the recruiter and saved the time of shortlisting candidates rather than using the traditional method.

II. Related Work

The work presented in [1] has proposed the ontology of a ranking feature for selecting algorithm for step counters which is based on machine learning. The features were extracted from the accelerometer and the Bhattacharyya coefficient was used to decide to which class the snippets belongs to. The resumes and job description uploads are not in the structured format so for deriving the keyword, information will be extracted using natural language processing [2]. Another work interpreted that described the effective and efficient ranking of resumes where the author used the ontology of vector space model which made the shortlisting of candidates easier for the recruitment process [3].

For making the shortlisting process quicker in an efficient way while assuring the selection of the most appropriate candidate for the given job role the classification of a job description and resumes have been performed in which the keywords from both documents will be extracted and matched with each other [4]. The authors of [5] collected the dataset from Kaggle and performed the preprocessing on it further for comparing the similarity, classification using various models was done and the score of accuracy was recorded for the resume recommendation system. Documents containing a larger amount of text can be a problem for natural language processing. So, for this purpose, the author used automatic text summarization to extract only the necessary parts of the document [6].

III. Methodology

A. Dataset Description

The resume dataset used for training the model contains three columns ID, Resume and Category. ID - the identification number of the resume, Resume – the text extracted from the resume PDF and Category – states the resume belongs to which domain. The dataset contains 962 resumes which belongs to 25 different category of industry sectors. Fig. 1 shows the distribution of resume in different domains.

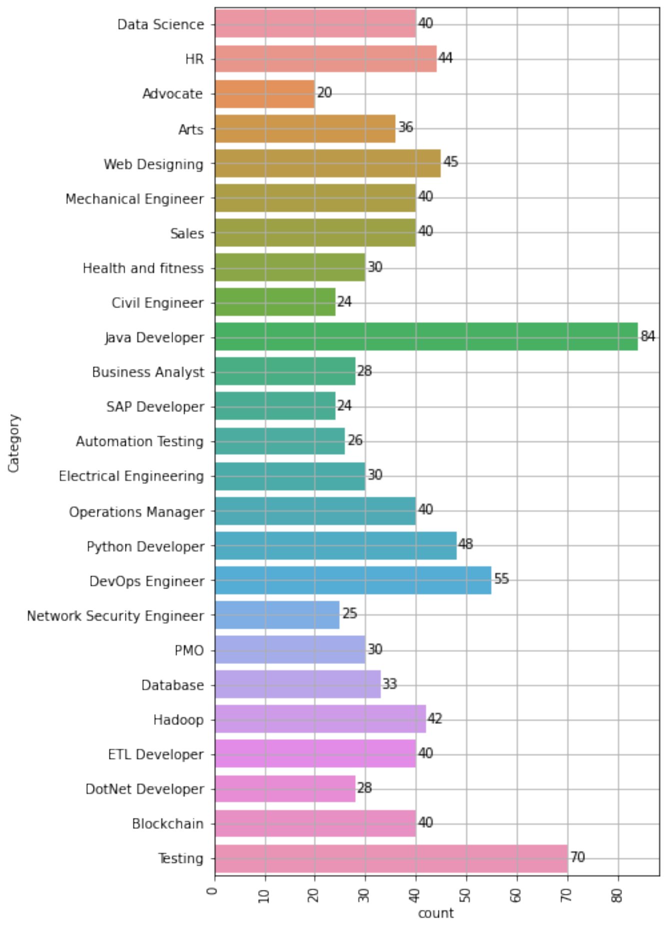


Fig. 1. Resume Dataset Distribution in 25 categories

B. Cleaning

Most of the candidates upload the resumes in PDF or Docx format. When converting from pdf to text, some unwanted characters are also extracted along with the text. These characters are used for styling and formatting the structure of PDF files. But they are not useful for feature extraction and comparison, as they do not provide any contextual meaning. Cleaning these characters is an important step for computer models. They are masked with empty characters and removed. The cleaning step includes:

- Converting text to lowercase
- Removing URLs
- Removing punctuation
- Removing non-ANCI characters
- Removing extra white-space by masking empty string fragments for escape sequences like \a, \b, \t, \n

C. Removing stop-words

Stop-words are words that regularly appear in text but add nothing to the sentences in which they appear. For the computer models, these words are meaningless. Prior to analysis, these terms should be removed from the text. By importing the pre-programmed English stop-word library with NLTK, these can be eliminated. Steps to remove stop-words:

1) Convert the input text into individual tokens using word_tokenize
```
text_tokens = word_tokenize(text)
```
2) Download the stopwords library using nltk
```
nltk.download('stopwords')
```
3) Import the stopwords library using nltk corpus
```
from nltk.corpus import stopwords
```
4) Get the set of English stop words
```
stopwords_set = set(
    stopwords.words('english')
    +['``', "''"])
```
5) Remove terms from the input text that are included in the list of stop words.
```
filtered_text = [word for word in
           text_tokens if not word in
           stopwords_set]
```

D. Stemming and Lemmatization

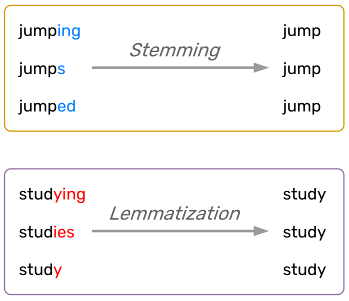
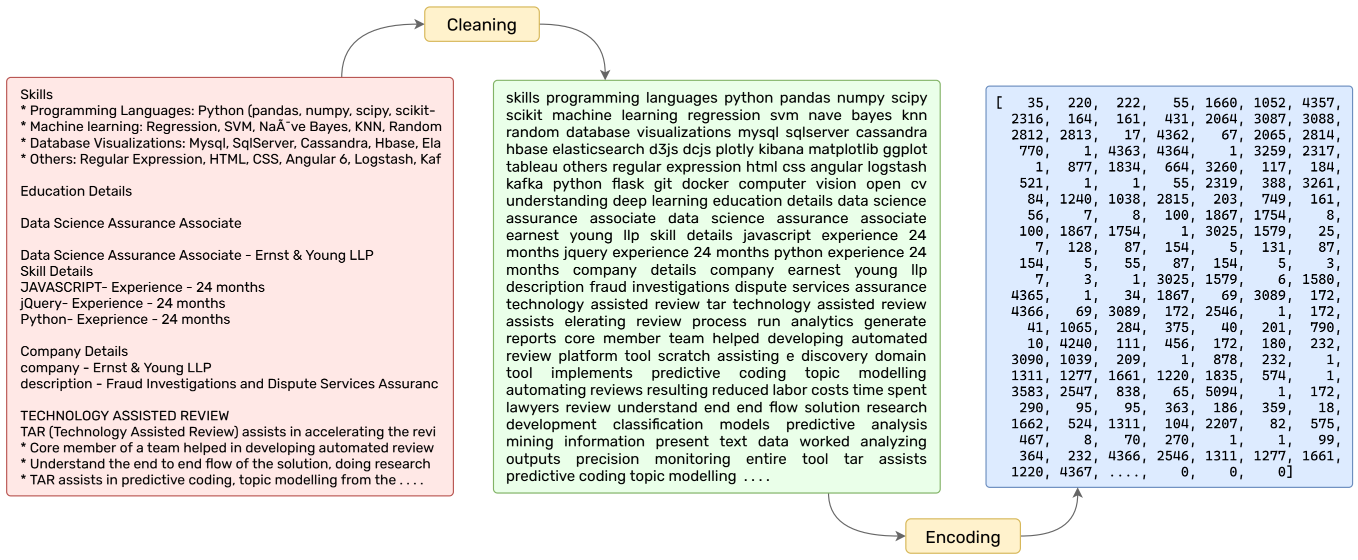


Fig. 2. Stemming and Lemmatization

The process of ”stemming” involves returning derivative or inflected words to its root or stem [8]. Words are grouped according to their root stem. This makes it possible for us to understand that the words ”jumped,” ”jumps,” and ”jumping” are all derived from the same root/stem verb, ”jump,” and as a result, refer to the same scenario. This method has two main challenges:

1) Over-stemming: In this, the term’s meaning is lost in this situation, and the resulting stem word is meaningless. As a result, words with the same stem but various meanings are produced. For search phrases like University, Universal, and Universe, the terms are shortened to ”univers,” for instance.



```
Skills
* Programming Languages: Python (pandas, numpy, scipy, scikit-
* Machine learning: Regression, SVM, NaÃ¯ve Bayes, KNN, Random
* Database Visualizations: Mysql, SqlServer, Cassandra, Hbase, Ela
* Others: Regular Expression, HTML, CSS, Angular 6, Logstash, Kaf

Education Details

Data Science Assurance Associate

Data Science Assurance Associate - Ernst & Young LLP
Skill Details
JAVASCRIPT- Experience - 24 months
jQuery- Experience - 24 months
Python- Exeprience - 24 months

Company Details
company - Ernst & Young LLP
description - Fraud Investigations and Dispute Services Assuranc

TECHNOLOGY ASSISTED REVIEW
TAR (Technology Assisted Review) assists in accelerating the revi
* Core member of a team helped in developing automated review
* Understand the end to end flow of the solution, doing research
* TAR assists in predictive coding, topic modelling from the . . . .
```

skills programming languages python pandas numpy scipy scikit machine learning regression svm nave bayes knn random database visualizations mysql sqlserver cassandra hbase elasticsearch d3js dcjs plotly kibana matplotlib ggplot tableau others regular expression html css angular logstash kafka python flask git docker computer vision open cv understanding deep learning education details data science assurance associate data science assurance associate earnest young llp skill details javascript experience 24 months jquery experience 24 months python experience 24 months company details company earnest young llp description fraud investigations dispute services assurance technology assisted review tar technology assisted review assists elerating review process run analytics generate reports core member team helped developing automated review platform tool scratch assisting e discovery domain tool implements predictive coding topic modelling automating reviews resulting reduced labor costs time spent lawyers review understand end end flow solution research development classification models predictive analysis mining information present text data worked analyzing outputs precision monitoring entire tool tar assists predictive coding topic modelling . . . .

```
[   35,  220,  222,   55, 1660, 1052, 4357,
  2316,  164,  161,  431, 2064, 3087, 3088,
  2812, 2813,   17, 4362,   67, 2065, 2814,
   770,    1, 4363, 4364,    1, 3259, 2317,
     1,  877, 1834,  664, 3260,  117,  184,
   521,    1,    1,   55, 2319,  388, 3261,
    84, 1240, 1038, 2815,  203,  749,  161,
    56,    7,    8,  100, 1867, 1754,    8,
   100, 1867, 1754,    1, 3025, 1579,   25,
     7,  128,   87,  154,    5,  131,   87,
   154,    5,   55,   87,  154,    5,    3,
     7,    3,    1, 3025, 1579,    6, 1580,
  4365,    1,   34, 1867,   69, 3089,  172,
  4366,   69, 3089,  172, 2546,    1,  172,
    41, 1065,  284,  375,   40,  201,  790,
    10, 4240,  111,  456,  172,  180,  232,
  3090, 1039,  209,    1,  878,  232,    1,
  1311, 1277, 1661, 1220, 1835,  574,    1,
  3583, 2547,  838,   65, 5094,    1,  172,
   290,   95,   95,  363,  186,  359,   18,
  1662,  524, 1311,  104, 2207,   82,  575,
   467,    8,   70,  270,    1,    1,   99,
   364,  232, 4366, 2546, 1311, 1277, 1661,
  1220, 4367, ....,    0,    0,    0]
```

Fig. 3. Cleaning and Word Encoding

Although if these three words have a similar etymology in this situation, their meanings are quite distinct.

2) *Under-stemming:* Even though the expressed words have distinct meanings, they share the same stem in this instance. When there are words that are distinct forms of one another, under-stemming is a problem. For the word ”alumnus,” for instance, there are three distinct forms: alumni, alumna, and alumnae. It should be observed that these synonyms are not combined and that this English word has a Latin morphology.

The process of reducing a word or search phrase to its basic lexical form is known as lemmatization. This method treats a word's component as a singular element by taking it into account in this manner [10]. As a result, ”studying,” ”studies,” and ”study” (which have various word-ends) can be identified as forms of the word ”study,” which serves as their lemma. Lemmatization is applied to the resume data to help computer models comprehend it better and perform better. Fig.2 shows how stemming and lemmatization works. Steps to perform lemmatization on the resume text:

1) Import the WorkNetLemmatizer library
   `from nltk.stem import WorkNetLemmatizer`
2) Initialize a WordNetLemmatizer class
   `lemmatizer = WordNetLemmatizer()`
3) Determine the POS tag for the word
   `POS = get_wordnet_pos(word)`
4) Convert the word to its lemmatized form
   `new_word = lemmatizer.lemmatize(word, POS)`

E. Named-Entity Recognition (NER)

Using named-entity recognition, the candidate's name is taken from the resume content [9]. An NLP library called SpaCy includes techniques for named-entity recognition within its framework. The name of the candidate can be extracted from the resume using matcher ['POS': 'PROPN']. Steps for extracting name from resume text:

1) Import SpaCy and load the 'en_core_web_sm' language set pack
   `nlp = spacy.load('en_core_web_lg')`
2) Parse the resume text into the NLP engine
   `nlp_text = nlp(text)`
3) Declare a pattern to match the pronoun from the part of speech
   `pattern = [{'POS': 'PROPN'}]`
4) Match the pattern to the NLP parsed resume text
   `matches = matcher(nlp_text)`
5) This will return the name of the candidate from the resume text

F. Word Encoding

The resume text is first pre-processed, which includes cleaning, eliminating stop words, and lemmatization before going through the encoding procedure. Text serves as the CNN model's input data. The CNN model cannot be trained directly from the text. The resume text is divided into individual words, known as ”tokens,” in this step. A dictionary is created, where each unique word present in the dataset of resume text is given a unique index. The mapping of each token to a distinct index value enables the representation of the resume text as an array of indices. The CNN model is then fed this array of indexes. Both the resume content and the resume categories are independently encoded.

G. Text Vectorization

Text is transformed into a numerical vector representation known as vector space by a process called vectorization. Text vectorization or word embedding is the process of converting

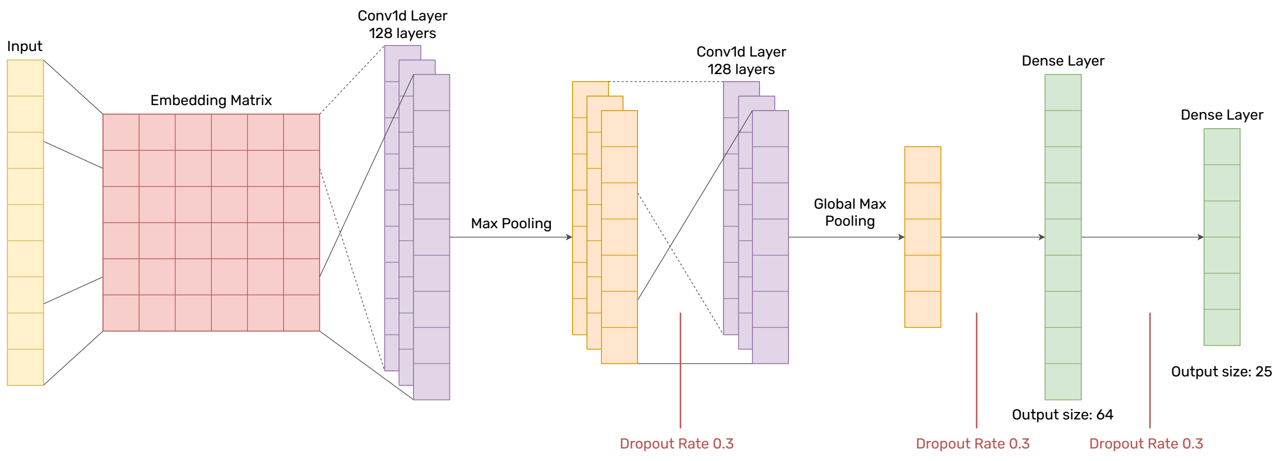


Fig. 4. CNN Model

words or texts contained in a corpus into numerical vectors of integers or real numbers [13]. Because the majority of machine learning algorithms operate on numerical input, this step is crucial in tasks dependent on sentiment analysis using machine learning. Other models require vectorized data as input, however the CNN model contains an embedding layer for this purpose. The resume text is vectorized using TF-IDF vectorization.

H. TF-IDF

The abbreviation TF-IDF stands for Term Frequency and Inverse Document Frequency. This method is commonly used to translate document into meaningful numerical representations which can be applied to fit machine learning prediction algorithms [11], [12].

The frequency with which a word (w) occurs in a document(d) is measured using the term frequency (TF) metric (d). The ratio of all the words in the text is used to describe the frequency (TF) of a word in a text. Each corpus document's length varies, so the denominator term in the computation normalises the result.

$$\text{TF(w,d)} = \frac{\text{frequency of word (w) in document (d)}}{\text{total number of words in document (d)}} \quad (1)$$

The inverse document frequency (IDF) is a metric used to assess a word's significance. Term frequency does not take into consideration the significance of words (TF). Certain words, like ”of,” ”and,” etc., can be used repeatedly but have little meaning. The IDF gives each word a weight based on how frequently it appears in the corpus D. A word's IDF is defined as

$$\text{IDF(w,D)} = \ln\left(\frac{\text{freq of docs (N) in corpus (D)}}{\text{freq of docs having word (w)}}\right) \quad (2)$$

Term Frequency - Inverse Document Frequency (TF-IDF): It is the result of product between TF and IDF which are calculated as mentioned above. TF-IDF gives terms that are uncommon in the corpus (all documents) more weight. TF-IDF gives more weight to the word that appears more frequently in the document.

$$\text{TF-IDF(w,d,D)} = \text{IDF(w,D)} \times \text{TF(w,d)} \quad (3)$$

I. Cosine Similarity

No matter how big the documents are, the cosine similarity statistic is utilised to assess how similar they are. [14]. By vectorizing two documents, item S and item T, and then using cosine similarity, the two documents can be compared based on the frequency of keywords in each of them. It figures out the cosine of the angle that two vectors created when they were projected onto a multidimensional space.

$$cos(\theta) = \frac{\vec{s}.\vec{t}}{|\vec{s}|.|\vec{t}|} = \frac{\sum_{i=1}^{n} s_i t_i}{\sqrt{\sum_{i=1}^{n}(s_i)^2}\sqrt{\sum_{i=1}^{n}(t_i)^2}} \quad (4)$$

The job description and resume text are pre-processed before being vectorized using TF-IDF. The cosine similarity, which calculates how similar a resume and job description are, is used to compare them. Each Resume can be compared to the job description using cosine similarity, and a list of percentages can be produced. The candidates' ranking can be determined by sorting this in descending order.

J. CNN Model

Convolutional Neural Networks typically employ convolution and pooling, two techniques that can be thought of as feature extractors. The system have a 1-dimensional array that represents the text in NLP tasks rather than images. Here, the ConvNets' design is modified to use 1D convolutional-and-pooling procedures. [7]

The first layer of the CNN model, called an embedding layer, looks for embeddings of the top 6,000 words in a 64-dimensional embedding. The input for this layer is specified as the resume's maximum length, i.e., 800. The model consists of

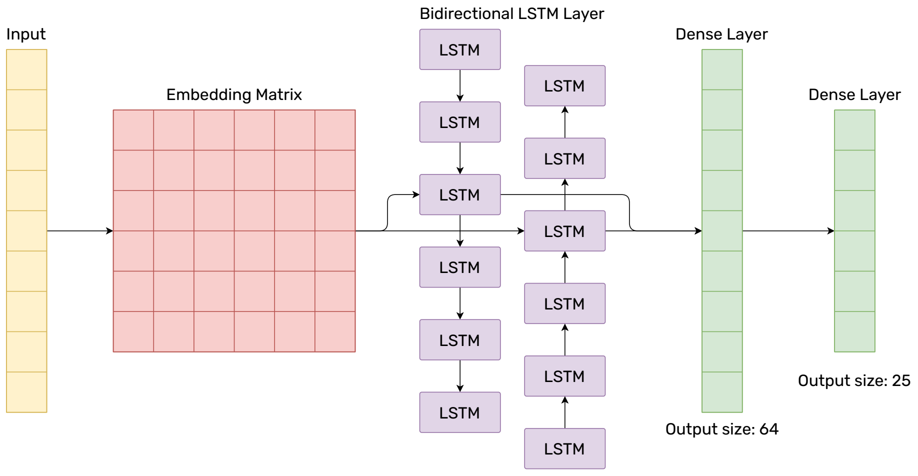


Fig. 5. LSTM Neural Network Model

two sets of convolutional layers, followed by a pooling layer and a dropout layer. The output filter for both convolution layers is 128. The second convolution layer is followed by a global max-pooling layer, while the first convolution layer is followed by a max-pooling layer with pooling window 3. A dropout layer with a dropout rate of 0.3 is introduced after each pooling layer to avoid overfitting. After global maximum pooling, a fully connected layer of size 64 with ReLU activation function is added, followed by a third dropout layer. The result of the SoftMax function is used to calculate the size of the dense layer, which represents the number of categorization classes. Fig. 4 shows the structure of the CNN model.

K. *LSTM Model*

Although LSTM is a type of recurrent neural network, it performs better in terms of memory than conventional recurrent neural networks. The performance of LSTMs is improved by having a strong grasp of memorising specific patterns [15]. Like any other NN, LSTM is capable of having several hidden layers. When it moves through each layer, the relevant data is retained while the irrelevant data is eliminated in each and every cell.

The embedding layer, the top layer of the model, represents words using dense vector representation [16]. When a word is used, the words around it define its position in the vector space [3]. The vocabulary size for the embedding layer is set to 6000, with an embedding dimension of 64. The maximum length of the resume, i.e., 800, is defined as the input for this layer. The following layer is a bidirectional LSTM layer with a 128-output filter. Dropout and recurrent dropout rates of 0.2 are used in the bidirectional LSTM layer. A 64-output sized fully connected layer with ReLU activation is added after Bi-LSTM layer. The SoftMax function determines the output with a dense layer of size 25 to represent the number of categorization classes. Fig. 5 shows the structure of the LSTM model.

L. *Baseline Models*

Other models are also trained for comparison purposes. These include K-Nearest Neighbors, Random Forest, and SVC. These models are ineffective for multi-class classification. As a result, One-vs-Rest is used. One classifier is fitted per class using One-vs-Rest. Each classifier is then combined to classify the data for multiple classes. Before training the model, the resume text data is pre-processed and converted to vector space using text vectorization. The vector-space data is then fed into the model for training.

IV. System Architecture

A. *Resume Screening Module*

The system accepts a resume from the job seeker and extracts information from it in this module. The file is initially converted to text because actions cannot be performed on files in the pdf or docx formats. After extracting text from a resume, it is pre-processed for further analysis. In the pre-processing stage, the resume text undergoes cleaning, stop-word removal, and lemmatization. The cleaning step first converts all the words to lowercase and then removes URLs, punctuation, non-ANCI characters, and white-space. Then the stop-words are removed and lemmatization is applied to provide meaning and contextual information. From this stage, a cleaned resume text is produced. This is stored in the database as a candidate resume for further operations.

Then information about the candidate is extracted from the cleaned resume text. Using Named-Entity Recognition, name of the job applicant is retrieved. The phone number and email address are extracted by performing regular expression matching. For education and skills mentioned in resumes, the keywords are matched with a list of education and skills from various domains and industrial sectors that the employees mention in their resumes. The name, email, phone number, education, and skill details of the job applicant are stored in the database for access when an employer requests it. The Figure 6 shows the flow of the resume screening module.

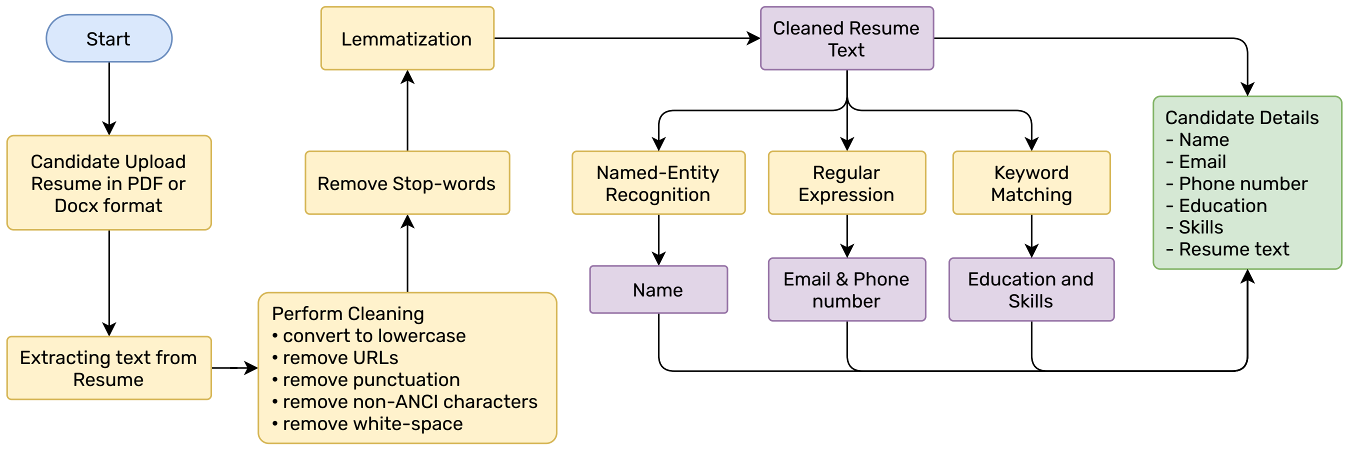


Fig. 6. Resume Screening Module

B. Resume Categorizing Module

In this module, the LSTM module is used for categorizing the resume and providing the top 5 categories in which the job applicant is most suitable for working. The cleaned resume text of the job applicant is retrieved from the database. As the model does not work directly on words, the resume text undergoes word encoding.

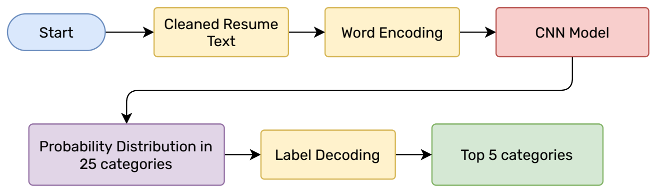


Fig. 7. Resume Categorizing Module

In this step, each word of the resume text is tokenized and replaced with the dictionary index with which the model was trained. This outputs an array of indices that represent the resume text. The array of indices is fed to the LSTM model. This model produces a probability distribution of the resume in 25 different domains/categories. Depending on this, the top 5 categories are selected, and the label is decoded to the original words. These top 5 category names, along with their respective probabilities, are then stored in the database. When an employer requests the list of job applicants based on the domain name, this information about the candidate stored in the database can be retrieved and displayed to them along with candidate details extracted from the resume. The Figure 7 shows the flow of the resume categorizing module.

C. Resume Ranking Module

In this module, employers first created a job post. In which, the job role, job salary, and other information are provided. Along with this, a job description in PDF or Docx format is also provided. This job description, when uploaded by the employer, goes through the same pre-processing step as the resume. The system produces a cleaned job description text and stores it in the database. When a candidate applies for the job, the candidate's cleaned resume text is retrieved from the database along with the cleaned job description text. They are then compared using Cosine Similarity. The resume text and job description text are first converted to vector space using TF-IDF. Then these two vectors are provided as an input to the cosine similarity function.

The cosine similarity algorithm evaluates the two documents and generates a percentage that indicates how well the applicant's resume matches the employer's stated job description. This percentage is stored in the database along with references to the job ID and candidate ID. The list of job applicants and their cosine similarity % for that job posting are fetched from the database when the employer wants to see the candidates who have applied for the position. In the following step, this list is sorted in descending order, and the employer is shown the ranked list of job candidates together with candidate information that was taken directly from the resume. The Figure 8 shows the flow of the resume ranking module.

V. Evaluation

The baseline models, the CNN model, and the LSTM model, were trained on the pre-processed and word-encoded resume dataset. To train the CNN and LSTM model, the Keras and Tensorflow libraries were used. The baseline models include N-Nearest Neighbors, Random Forest, and SVM.

Sr. No.	Model	Accuracy
1	K-Nearest Neighbors	96.68 %
2	Random Forest	98.34 %
3	SVM	99.17 %
4	CNN	99.48 %
5	LSTM	98.68 %

TABLE I
Accuracy of Models

These baseline models were trained on the resume dataset after applying text vectorization. As the labels of the dataset

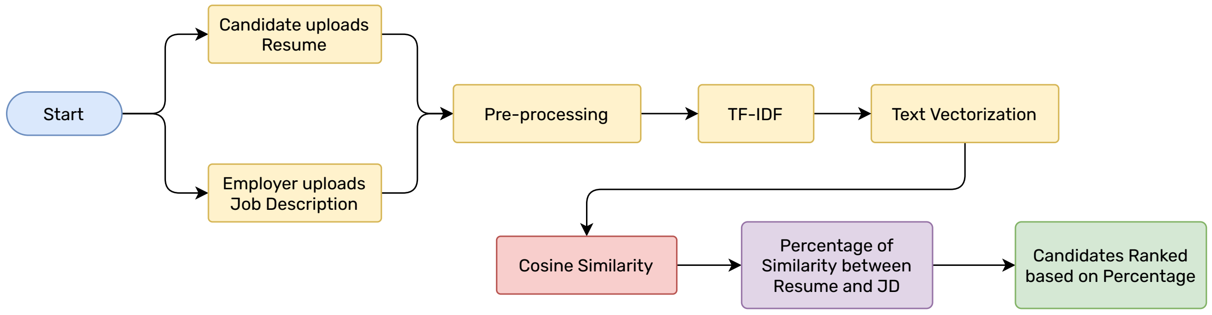

Start
Candidate uploads Resume
Employer uploads Job Description
Pre-processing
TF-IDF
Text Vectorization
Cosine Similarity
Percentage of Similarity between Resume and JD
Candidates Ranked based on Percentage

Fig. 8. Resume Ranking Module

are multi-class, the One-vs-Rest algorithm is used. This algorithm allows binary classification models to be trained on a multi-class dataset. The CNN and LSTM models are the neural network models trained on the data. The word encoded resume dataset is fed to the NN models, which provide the probability distribution among the 25 categories of domain. The index of the probabilities is then decoded to the original labels. The accuracy of each model trained on the resume dataset is shown in Table I. After testing the models, the KNN model yeilded an accuracy of 96.68%. The Random Forest and SVM classifier predicted the categories of resume with acuuracy of 98.34% and 99.17% respectively. The CNN model gave the highest accuracy among other models i.e. 99.48%. The LSTM model yeilded an accuracy of 98.68%. So by using the CNN Model the performance was improved.

VI. Conclusion

In this research work, the system is built using a CNN and LSTM model along with some baseline models like N-Nearest Neighbors, Random Forest, and SVM. Among these models, the CNN model gave the highest accuracy of 99.48% for the resume dataset. The CNN models provide results in the form of a probability distribution. From these, the domain category having the top five probabilities can be considered the domain in which the resume is most suitable. The cosine similarity algorithm is useful for comparing the candidate's resume and job description. Using the CNN model and Cosine Similarity, the system is useful for HR and employers to find the right candidates for the job role. The work of the employer is reduced as the system automatically compares the resumes of job applicants and ranks them.

References

[1] S. Vandermeeren, S. Van de Velde, H. Bruneel and H. Steendam, "A Feature Ranking and Selection Algorithm for Machine Learning-Based Step Counters," in IEEE Sensors Journal, vol. 18, no. 8, pp. 3255-3265, 15 April15, 2018, doi: 10.1109/JSEN.2018.2807246.

[2] Daryani, Chirag & Chhabra, Gurneet & Patel, Harsh & Chhabra, Indrajeet & Patel, Ruchi. (2020). AN AUTOMATED RESUME SCREENING SYSTEM USING NATURAL LANGUAGE PROCESSING AND SIMILARITY. 99-103. 10.26480/etit.02.2020.99.103.

[3] D. L. Lee, Huei Chuang and K. Seamons, "Document ranking and the vector-space model," in IEEE Software, vol. 14, no. 2, pp. 67-75, Mar/Apr 1997, doi: 10.1109/52.582976.

[4] A. Zaroor, M. Maree and M. Sabha, "JRC: A Job Post and Resume Classification System for Online Recruitment," 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI), Boston, MA, USA, 2017, pp. 780-787, doi: 10.1109/ICTAI.2017.00123.

[5] Pradeep Kumar Roy, Sarabjeet Singh Chowdhary, Rocky Bhatia, A Machine Learning approach for automation of Resume Recommendation system, Procedia Computer Science, Volume 167, 2020, Pages 2318-2327, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2020.03.284.

[6] I. Awasthi, K. Gupta, P. S. Bhogal, S. S. Anand and P. K. Soni, "Natural Language Processing (NLP) based Text Summarization - A Survey," 2021 6th International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 2021, pp. 1310-1317, doi: 10.1109/ICICT50816.2021.9358703.

[7] Hughes M, Li I, Kotoulas S, Suzumura T. Medical Text Classification Using Convolutional Neural Networks. Stud Health Technol Inform. 2017;235:246-250. PMID: 28423791.

[8] Rio Pramana; Debora; Jonathan Jansen Subroto; Alexander Agung Santoso Gunawan; Anderies, "Systematic Literature Review of Stemming and Lemmatization Performance for Sentence Similarity"2022 7th International Conference on Information Technology and Digital Applications (ICITDA), Yogyakarta, Indonesia, doi: 10.1109/ICITDA55840.2022.9971451.

[9] B. VeeraSekharReddy, K. S. Rao and N. Koppula, "Named Entity Recognition using CRF with Active Learning Algorithm in English Texts," 2022 6th International Conference on Electronics, Communication and Aerospace Technology, Coimbatore, India, 2022, pp. 1041-1044, doi: 10.1109/ICECA55336.2022.10009592.

[10] R.J. Prathibha; M.C. Padma, "Design of rule based lemmatizer for Kannada inflectional words" 2015 International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT), Mandya, India, doi: 10.1109/ERECT.2015.7499024.

[11] Gleen A. Dalaorao; Ariel M. Sison; Ruji P. Medina,"Integrating Collocation as TF-IDF Enhancement to Improve Classification Accuracy",2019 IEEE 13th International Conference on Telecommunication Systems, Services, and Applications (TSSA), Bali, Indonesia, doi:10.1109/TSSA48701.2019.8985458.

[12] Bakhyt Bakiyev,"Method for Determining the Similarity of Text Documents for the Kazakh language, Taking Into Account Synonyms:Extension to TF-IDF",2022 International Conference on Smart Information Systems and Technologies(SIST),Nur-Sultan,Kazakhstan,doi:10.1109/SIST54437.2022.9945747.

[13] Ali Mansour; Juman Mohammad; Yury Kravchenko,"Text Vectorization Method Based on Concept Mining Using Clustering Techniques", 2022 VI International Conference on Information Technologies in Engineering Education (Inforino), Moscow, Russian Federation, doi: 10.1109/Inforino53888.2022.9782908.

[14] Reza Fauzan; Muhammad Ikhwanul Atha Labib; Joshua Oktavianus Tarung Johannis; Herlinawati; Syamsudin Noor; Saifulah," Semantic similarity of Indonesian sentences using natural language processing and cosine similarity",2022 4th International Conference on Cybernetics and Intelligent System (ICORIS), Prapat, Indonesia, doi:10.1109/ICORIS56080.2022.10031439.

[15] Bhaskar Marapelli; Anil Carie; Sardar M. N. Islam, "RNN-CNN MODEL:A Bi-directional Long Short-Term Memory Deep Learning Network For Story Point Estimation", 2020 5th Interna-

tional Conference on Innovative Technologies in Intelligent Systems and Industrial Applications (CITISIA), Sydney, Australia, doi: 10.1109/CITISIA50690.2020.9371770.

[16] MingXia Gao; JiaYi Li,"Chinese short text classification method based on word embedding and Long Short-Term Memory Neural Network",2021 International Conference on Artificial Intelligence, Big Data and Algorithms (CAIBDA), Xi'an, China, doi: 10.1109/CAIBDA53561.2021.00027.