International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(5s) 2026 pp. 376-389
https://doi.org/10.70917/ijcisim-2026-2717
Article

Design and Implementation of a Job Skill Entity Recognition (JSER) for Custom Entity Annotation using Trie Structure

Pareshkumar Prajapati[1], Dr. Sandeep Vasant[2], Dr. Jaideepsinh Raulji[3]

[1] M.K. Institute of Computer Studies, Bharuch; Research Scholar, Navrachana University, Vadodara, Gujarat, India
pareshkumar.prajapati@nuv.ac.in, ORCID: 0009-0004-3711-7402
[2] Department of Computer Science and Engineering, Navrachana University, Vadodara, Gujarat, India
sandeep.vasant@nuv.ac.in, ORCID: 0009-0008-1371-2898
[3] Department of Computer Science and Engineering, Navrachana University, Vadodara, Gujarat, India
Email: jaideep.raulji@nuv.ac.in, ORCID: 0000-0003-1787-8496

Abstract: Abstract—Named Entity Recognition in computer science is tricky because it uses terms and needs detailed annotations. Even though Named Entity Recognition has gotten better in uses it is not widely used in academic computer science. This paper presents JSER, a method that uses Natural Language Processing techniques to find and classify both known and unknown entities. Our approach uses Part-of-Speech tagging to pull out entities from unstructured text. We make annotations better and faster with techniques like tuning models adjusting thresholds and adding more data. JSER seems promising for tasks, like finding information analyzing text automatically and managing knowledge.

Keywords: JSER, Named Entity Recognition, NLP, Part-of-Speech tagging

1. Introduction

Domain-specific Named Entity Recognition in Computer Science presents challenges. This is because it requires annotation making it more complex than general-domain Named Entity Recognition. While Named Entity Recognition techniques have advanced applications tailored to the Computer Science domain remain underexplored. Though growing interest is expected.

This paper highlights progress in Computer Science- Named Entity Recognition by introducing the Job Skill Entity Recognition system. The Job Skill Entity Recognition system is designed to annotate both known entities, stored in a data repository and unknown entities not pre-existing in the database.

The proposed methodology uses Natural Language Processing techniques. These techniques include Part-of-Speech tagging. They help to identify and classify entities from unstructured text. This approach addresses the challenges of recognizing and annotating entities across text formats. It has applications in information retrieval automated text processing and knowledge management. Natural Language Processing enables computers to understand and process language. It plays a role in this system. A key Natural Language Processing application is Text Mining. Text Mining is aimed at discovering patterns and insights from textual data. Part-of-Speech tagging assigns grammatical roles to words. It is foundational for Named Entity Recognition. Part-of-Speech tagging facilitates the identification and categorization of entities. These entities include names organizations and locations. The annotation workflow follows a process:

Raw Text → Tokenization → Part-of-Speech Tagging → Named Entity Recognition → Final Annotated Text.

For example, in the sentence "Pandora is one of the Programming Language " Part-of-Speech tags would be:

Pandora/Proper Noun is/Verb, my/Possessive Pronoun, one/Cardinal Number, the/Determiner, best/Superlative Adjective, Programming/Noun, Language/Noun. This tagging aids in analyzing sentence structure. It supports Natural Language Processing tasks like Named Entity Recognition.

This study focuses on developing an entity recognition model. The model classifies skillsets from resumes. Using 150 resumes collected from an organization the model extracts job-related skills.

It combines database matching and context-based analysis. Known entities are identified using a Trie data structure. Unknown entities are inferred through context analysis Part-of-Speech tagging and proximity analysis. Key tools include NLTK for preprocessing tasks.

These tasks include tokenization and classification. WordNetLemmatizer is used for text normalization. PyPDF2 is used for extracting content from PDFs. Additionally, Flask is used to create a web interface for user interaction. Regular expressions enhance string manipulation and lexical analysis. They improve the extraction of both unknown entities. By categorizing and extracting skills the model enhances resume analysis and job skill identification. It offers a solution for entity recognition. This versatile approach is highly applicable, in talent assessment and recruitment. It leverages both predefined data stores and innovative methods for recognizing entities.

2. Related Work

The paper presents a job title classification system that uses computers to help with job searching. It utilizes two methods: Support Vector Machines (SVM) and k-Nearest Neighbors (k-NN) to classify jobs into categories. The system also uses -supervised methods to make it more scalable. Future improvements, such as using a job taxonomy could make the system even better. The results show that the system can improve job search accuracy.

* "Automated Custom Named Entity Recognition and Disambiguation" introduces FastEnt, a system that can identify entities like names and locations. FastEnt uses Word2Vec and FastText embeddings, which are types of computer algorithms and a special type of network. It achieves accuracy in identifying specific terms, such as drug names and can be used in specialized fields. The use of Named Entity Recognition (NER) methods, including rule-based, machine learning and hybrid approaches analyzes how well these methods work.

A Fuzzy Support Vector Machine (FSVM) is proposed to improve accuracy. The results offer comparisons and applications in information retrieval and bioinformatics.

* Salman Naseer et al. Review NER techniques categorizing them into rule-based learning-based and hybrid approaches.

They evaluate tools like SpaCy, StanfordNLP and ensorFlow. Find that SpaCy excels in efficiency.

Thomas Green et al. Develop a benchmark corpus for job description entity recognition. Their model achieves a score and addresses key challenges like defining "skills."

* Lev. Dan Roths paper explores key NER challenges. They integrate -local features and external resources achieving a high F1 score on a specific dataset.

Mihaela-Irina Enăchescu presents an automated system for screening IT candidates by extracting skills from resumes. Using Web technologies it converts data into a specific format for precise job matching.

* Finkel and Manning introduce a discriminative constituency parser for nested NER.

By structuring sentences as trees their approach boosts accuracy in biomedical domains. Pech et al. Introduce a semantic annotation approach using ontologies to enhance search in documents. By leveraging concept similarity it improves term disambiguation and linking to ontology entities.

* Mungi Naga Venkata Sai Raghavendra presents a machine learning-based resume screening system. The system uses NLP and classifiers like KNN, SVM and MLP. Proves that MLP is the most accurate.

"Streamlining Resume Annotation with Named Entity Recognition" introduces an automated NER approach for annotating resumes.

Using a -LSTM-CRF model it extracts entities with high accuracy from a large number of resumes. Recent developments in document processing and information retrieval demonstrate the growing impact of intelligence and NLP.

* Kurisappan and Pandiyan reviewed datasets shaping AI-driven reasoning. They note the challenges that large language models face in solving -step mathematical problems. The paper presents an implementation of Hearsts TextTiling algorithm for automated text segmentation. It achieves a precision and recall of 0.77 on New York Times articles.

* Tannous et al. Present the algorithm for segmenting unstructured documents. With a 96% F1 score and 2% segmentation error TSHD is adaptable for NLP applications.

This paper introduces "FastEnt," an approach for Custom Named Entity Recognition (NER). It automates dataset generation and annotation achieving state-of-the-art results on entities.

* This survey reviews methods for analyzing resumes. It explores keyword, grammar and statistical parsers to evaluate coverage, readability and comprehensibility. The paper presents a system for converting resumes into structured formats using NLP techniques. It extracts details like personal information, skills and qualifications.

* Dr. R. Vijaya Kumar Reddy et al. Present a method for converting data into structured formats. By applying MapReduce and HBase on HDFS they enhance data retrieval speed and accuracy.

Bhushan Kinge et al. Present a system for automating resume screening using NLP and machine learning algorithms. By leveraging NER and cosine similarity it ranks resumes based on job relevance.

* Shubham Bhor et al. Introduce an NLP-based resume parsing and ranking system. It extracts details like education, experience and skills.

Sharma and Sharma proposed semantic query expansion using BERT and CRF. The dual approach, including video resume parsing improves accuracy.

* This paper presents an NLP-based system that automates resume screening. Using NER and part-of-speech tagging it ensures extraction and streamlines recruitment. The paper presents a parser model using NLP and Big Data tools. It extracts details like academic background and personal information.

* Tiwari et al. Provided a domain- analysis of machine learning techniques. The model achieves a F1 score, for POS tagging and NER.

Stratos, Collins and Hsu present an approach using Anchor Hidden Markov Models. By applying -negative matrix factorization they achieve competitive results.

Table I: COMPARISON OF TECHNIQUES AND APPLICATIONS IN JOB TITLE CLASSIFICATION, NAMED ENTITY RECOGNITION, AND RESUME PROCESSING IN VARIOUS PAPERS

Title	Domain	Approach	Key Findings	Challenges
Automated Custom Named Entity Recognition and Disambiguation [11]	Named Entity Recognition (NER) for custom entities	Word embeddings (Word2Vec, FastText), CNN, and recurrent mechanisms	Achieved SOTA performance (85% accuracy on drug-related terms); dataset enrichment with 93% overlap between training and testing words.	Limited real-world validation; potential scalability issues with custom datasets.
Screening IT Candidates Using Semantic Web Technologies [16]	IT candidate screening	Ontology-based RDF representation, Apache Tika, GATE	Automated CV screening with semantic web technologies; improved matching of technical competencies to job requirements.	Limited scalability and domain applicability beyond IT.
Streamlining Resume Annotation with NER [20]	Resume annotation	Bi-LSTM with CRF model	High accuracy in automated annotation of 5,350 resumes; reduced manual effort in resume processing.	Lacks extensive real-world validation; needs testing across diverse industries.

Title	Domain	Approach	Key Findings	Challenges
Decoding Math: A Review of Datasets Shaping AI-driven Mathematical Reasoning [21]	Mathematical Reasoning, Education Technology, NLP	Survey of 20+ benchmark datasets (e.g., GSM8K, MATH), Review of DL architectures (Seq2Seq, Seq2Tree, Graph2Tree, GPT-4)	GPT-4 and LLMs outperform traditional DL models; Domain-specific datasets improve parsing & reasoning in math problems	LLMs struggle with theorem-based tasks; Limited dataset diversity; Interpretability of deep models remains a concern
Conversion of Unstructured Data to Structured Data with a Profile Handling Application [26]	Resume Management, NLP	NLP techniques in Python; Tools like Apache PDFBox for PDF-to-text conversion, NLTK for data extraction.	Efficient structuring of resumes with scalable solutions for various fields.	Focused on resume data only; potential challenges in extending methods to other domains.
Enhancing information retrieval through advanced query expansion techniques and semantic analysis [30]	Search Engines, Information Retrieval Systems Semantic Search	BERT + CRF Model, Named Entity Recognition, WordNet, Query Expansion (QE), Learning-to-Rank, AHP	Achieved 99.7% accuracy in classification; Semantic understanding significantly improves relevance of search results	Model complexity and high training time; Overfitting risk when expanding queries; Needs high-quality labeled data
A CV Parser Model using Entity Extraction Process and Big Data Tools [32]	CV Parsing, Big Data	NLP and entity extraction techniques with Big Data tools like Hadoop MapReduce.	Efficiently processes large volumes of unstructured resumes; integrates metadata for structured data.	Requires diverse environment testing for validation; challenges in scalability beyond CVs.
A comprehensive survey and review of machine learning techniques in document processing : Industry applications and future directions [33]	Human Resource Management, Resume Filtering, Job Matching	NER, Classification Algorithms (Random Forest, Naive Bayes, SVM), Resume Parsing Techniques, and Text Mining	Combination of ML models enhances accuracy in job classification and resume screening; Automated systems reduce manual HR workload	Accuracy can be affected by unstructured or inconsistent resume formats; Limited generalizability across industries
Unsupervised Part-Of-Speech Tagging with Anchor Hidden Markov Models [34]	Low-Resource NLP	Anchor HMMs using non-negative matrix factorization (NMF); Tested across languages with universal POS tagset.	Excels in low-resource settings; reduces annotation costs; adaptable to underrepresented languages.	Dependence on unique anchor words; challenges in some linguistic contexts.

2.1 Approach and Implementation

This section provides a detailed of the system architecture and implementation of a Job Skill Entity Recognition (JSER) system for resume analysis. Each sub-section explains a specific component of the system, its purpose, and the methodologies employed.

2.2 Data Design

The initial phase involves categorizing skills and competencies, such as "Programming Language" and "Database," into predefined groups based on job roles and industry standards. In the next phase, resume data in PDF format is processed using Python's PyPDF2 library to extract common words and phrases. These extracted elements are used to build a dataset of skills and entities, focusing on known entities available in a defined database as shown in Fig.1 and flowchart of JSER as shown in Fig. 2

To identify unknown entities, the system extends its capabilities beyond predefined database recognition to detect new or uncommon entities. This is achieved using techniques from compiler design, such as lexical analysis and syntactic processing, combined with Part-of-Speech (POS) tagging. Once a potential entity is identified, linguistic

analysis helps categorize it, even if it lacks a predefined category. This approach enables the system to recognize and tag new entities not present in its initial database, enhancing its adaptability and scope.

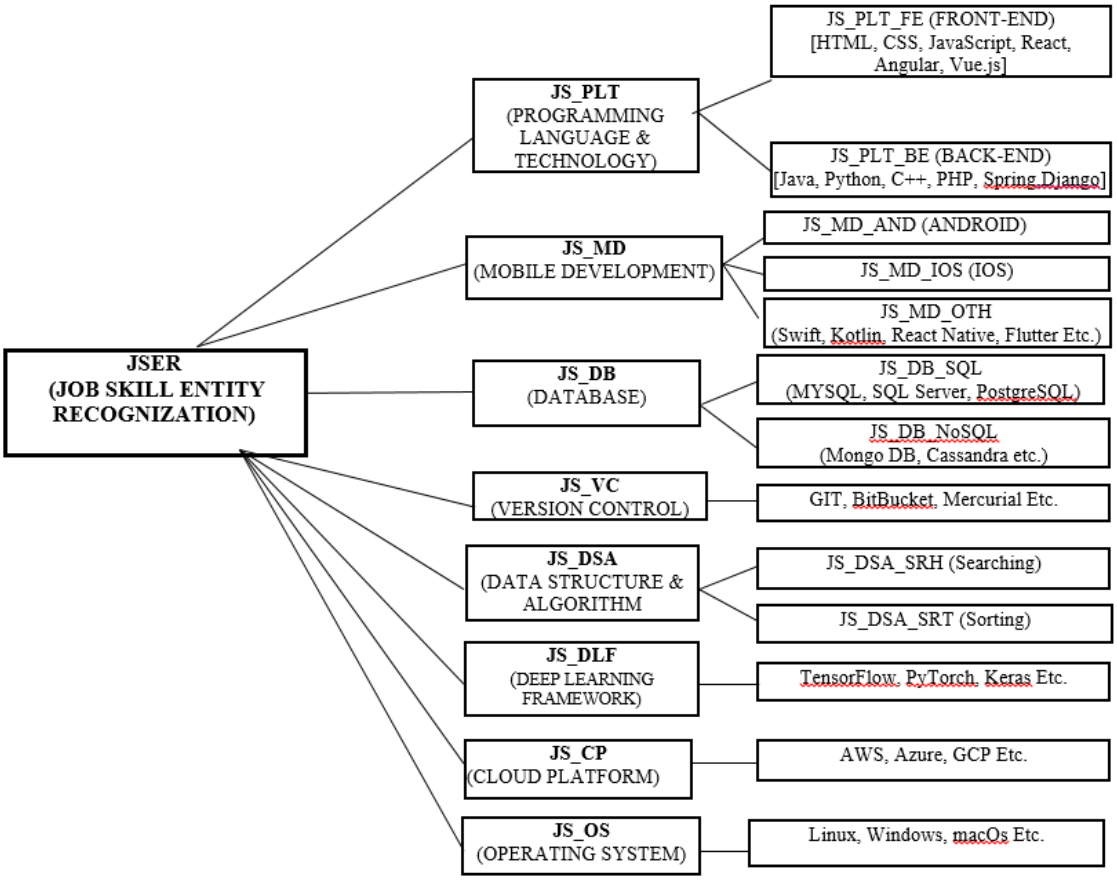

Figure 1: Nomenclature of tags

2.3 Framework

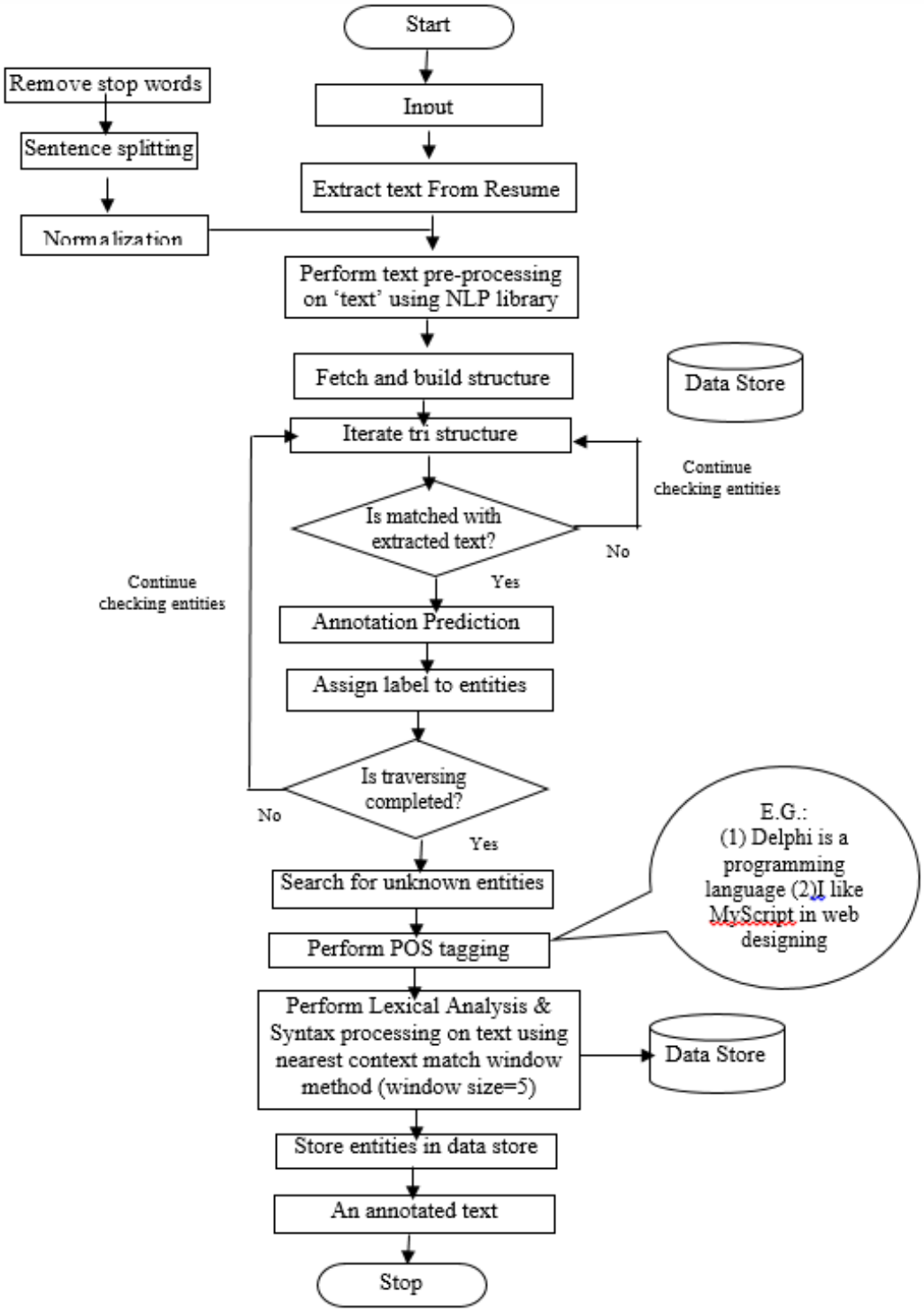

Figure 2: Flowchart JSER

2.4 The Lemmatizer

In Natural Language Processing a Lemmatizer is a tool that helps convert words into their form. This basic form is also called a lemma. The process starts with a word then it figures out what kind of word it is, like a noun or a verb. It uses a dictionary to find the form of the word. Finally it gives us the form of the word.

In the implemented methodology, the lemmatizer analyzed words from 150 resumes. They got rid of the variations of the words until they found the correct basic forms. Then they checked these forms against the JSER database to make sure they were right. There is a flowchart that shows how the tool works. It is in Fig.3. The basic forms of the words came from different dictionaries that were printed on paper. The Lemmatizer is really good, at finding the form of words, which is also called the lemma. The lemma is a part of Natural Language Processing.

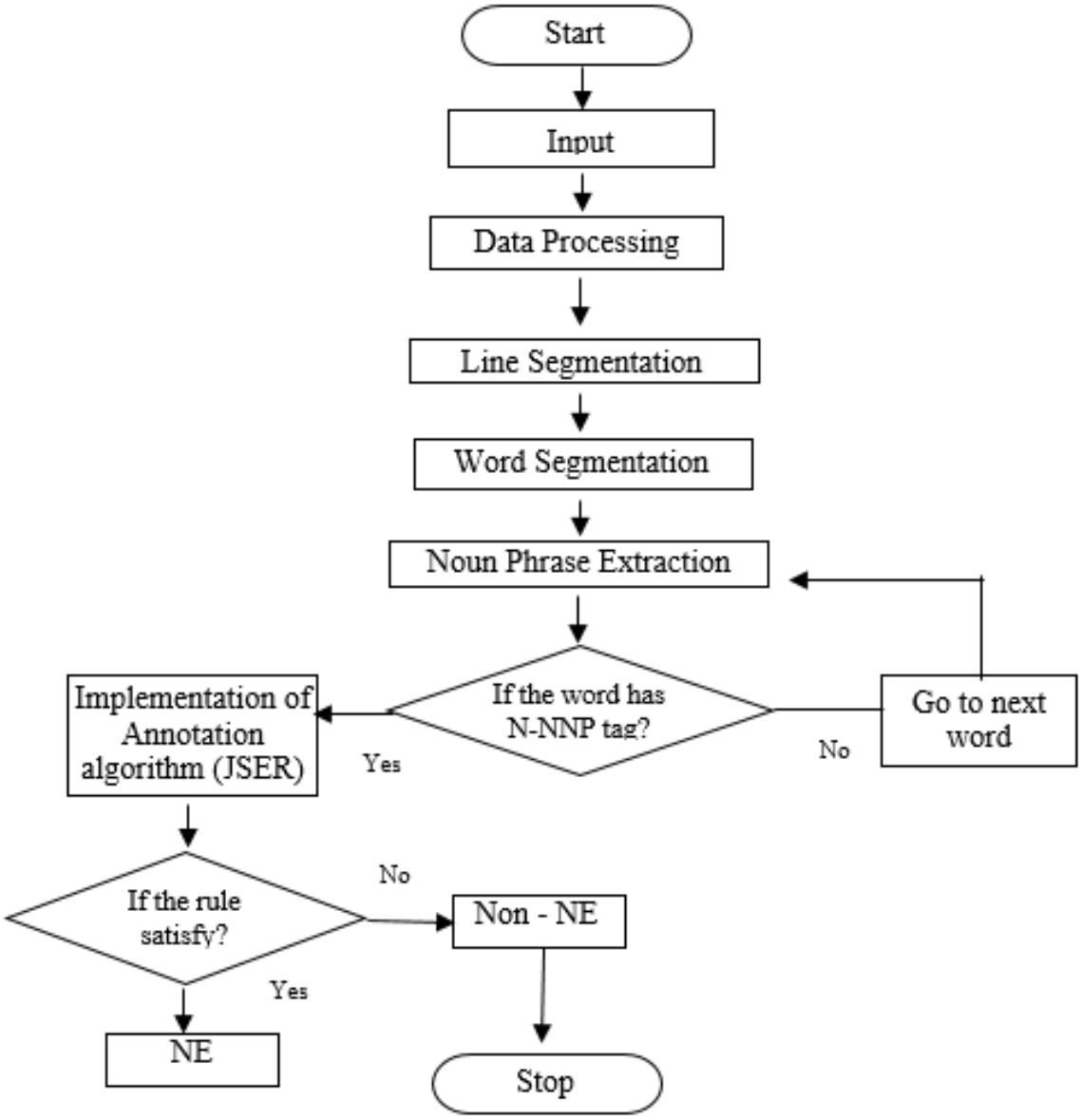

Figure 3: Flowchart Lemmatizer

2.5 Root Word Extractor (RWE)

A Root Word Extractor is a tool used in language processing to find and extract the part of a word. This part or root word, is what gives the word its basic meaning. It is often the word without any added prefixes, suffixes or endings.

The Root Word Extractor does a key things:

* It identifies the root of a word, which's the part that holds its core meaning.

* It simplifies forms of words to a basic form.

* It helps normalize words to their form, which makes text data easier to analyze.

There are two techniques used by Root Word Extractors: Stemming and Lemmatization.

Stemming is a technique where prefixes and suffixes are removed from words to get the root form. This is often done using a set of rules. For example the word "programming" becomes "program" and "nationalization" becomes "nation".

Lemmatization is a bit different. It looks at a words structure. Uses a dictionary to find its base form, which is called a lemma.

Unlike stemming lemmatization tries to produce a root word by considering the context and the type of word it is. For instance as shown in the proposed approach data, in table [4].

The Root Word Extractor uses these techniques to help with text analysis. It makes it easier to understand and work with amounts of text data.

2.6 Hidden Markov Model (HMM) for Language Processing

The Hidden Markov Model is an approach widely used for entity annotation. It is good at capturing dependencies in text.

Unlike methods like Trie or linear search HMM uses probabilistic transitions between states and observations to predict entities within unstructured text.

For example in annotating resumes HMM can accurately. Classify entities such as skills like "Python" and "Data Analysis" and job titles like "Software Engineer" by analyzing contextual patterns. The strength of HMM lies in modeling word sequences. This enables annotation even in noisy or ambiguous data.

The proposed system is a solution designed for analyzing resumes particularly in the computer science domain. The system processes the data. Converts it into JSON format. This facilitates data integration, analysis and automation thereby enhancing interoperability and utility. The stage utilized file manipulation and directory access.

Powerful libraries are used to handle tasks such as:

* Tokenization

* Part-of-speech tagging

* Stemming

* Lemmatization

It also handles tasks such as reducing words to their base forms. Stop-word filtering is used for data clarity. This allows text searching and matching. Processed entities are stored in a data store for analysis. Data can be represented in either a tri-structure or a linear way. A linear way of representing data is simple and straightforward.

However for systems with interdependent components a tri-structure is essential. The tri-structure effectively models interrelationships, dynamic changes and multiple processes simultaneously. This ensures decision-making and optimized data organization. Additionally the tri-structure enables annotation compared to the linear structure.

The system employs a trie structure to categorize entities stored in the database. Categories such as programming languages, databases and mobile development are split into subcategories for classification. For instance the programming category has subcategories like front-end tools and back-end tools.

This hierarchical structure facilitates entity comparison and identification. A web-based user interface enables users to upload resumes. These resumes are then processed to extract and highlight categorized skills. For example "Python" might be tagged under the "Programming" category. The system demonstrates accuracy in identifying predefined entities.

However recognizing entities that are not present in the database posed a significant challenge. For skill annotation tasks the Trie data structure offers an efficient approach than Linear Search especially when working with large datasets.

A Trie uses a tree- structure that allows for quick prefix matching. The time complexity of a Trie is O(L) where L is the length of the search string.

This is much faster than Linear Search, which has a time complexity of O(N×L) where N's the number of items in the dataset. The Hidden Markov Model and Trie data structure are used for entity annotation.

The Hidden Markov Model is good, at capturing dependencies in text.

The Trie data structure allows for prefix matching.

Here is an example of searching for skills that start with "Data". Let us consider the following dataset:

Dataset: ["Developer", "Designer", "Data Analyst", "Data Engineer", "Doctor", "Database Administrator", "Data Scientist"]

Trie: Efficiently retrieves all skills starting with "Data" by traversing the shared prefix, resulting in O(L) operations.

Matches: ["Data Analyst", "Data Engineer", "Data Scientist", "Database Administrator"]. Linear Search: Checks each item sequentially, requiring O(N×L) operations to find the same matches.

The system uses ideas from compiler design to identify and categorize things. This way it can handle both new things in a systematic way. Known things are treated like predefined keywords. Unknown things are understood based on the context in which they are used.

The process involves steps.

First the text is split into parts called tokens and labeled with grammatical labels like nouns or verbs. Then known things are matched to a dictionary. Labeled. For example "Database" is tagged as "DB". This process uses a method called Lexical Analysis and Syntax Processing on the text. It looks at the context to understand the meaning.

The system uses a window size of 5 which means it looks at 2 words before and 2 words after the target word to capture its context. If we increase the window size to 7 or 9 it affects the result and the accuracy of the model becomes lower. That is why we use a window size of 5. The process of identifying known things involves recognizing predefined things in the text using a dictionary and tagging them with labels. For example tagging "Database" as "DB". This is followed by Lexical Analysis and Syntax Processing using a sliding window approach. With a window size of 5 the context includes 2 words and 2 words after the target, which ensures that only the most relevant information is considered.

This focused context reduces noise. Makes the model more accurate. Increasing the window size to 7 or 9 introduces words, which can cause confusion and reduce the accuracy of the model. It also increases the complexity of the system and the likelihood of errors. To balance the need for context and accuracy we use a window size of 5 for the Data skills.

This way the system can efficiently retrieve all skills starting with "Data", like Data Analyst, Data Engineer, Data Scientist and Database Administrator. The comparison of different window size is shown in below table [2].

Table II: COMPARISON TABLE ILLUSTRATING THE IMPACT OF DIFFERENT WINDOW SIZES ON THE CONTEXT AND MODEL ACCURACY

Window Size	Context for "Programming Language"	Relevance	Noise	Accuracy Impact
5	" Pandora is one of the best Programming Language "	Focused and relevant. Captures nearest meaningful words (2 before and 2 after).	Minimal	High

Window Size	Context for “Programming Language"	Relevance	Noise	Accuracy Impact
6	" Pandora is my one of the best Programming Language "	Adds one extra word ("my") which may or may not add value.	Slight increase	Slight decline
7	" Pandora is also my one of the best Programming Language "	Includes less relevant word ("also"), which dilutes contextual relevance.	Moderate	Moderate decline
9	"Pandora is also my one of the best ever Programming Languag"	Adds significant irrelevant words ("ever") causing semantic ambiguity.	High	Significant decline

Context-Based Extraction: Unlisted entities are inferred based on their proximity and relationship to known entities (e.g., associating "Prish" with "Database").

Storing Results: All identified entities are stored in a SQLite database for future use and display known entities with color as shown in below Fig.4.

2.7 Result

This model extracts the texts, identifies the entities, calculate performance metrics and display the result. Among the 150 resumes, 5000 words are processed first by using trie structure and then by lemmatizer which examine 396 incorrect or missing entities and 4604 correct entities. As shown in table [3]. The evaluation of root word extractor counts are shown in table [4]

Detected Skills Summary

Document: 974-CV-Manish CV.pdf | Analysis Date: 2025-08-02

Detected Skills by Category

#	Skill Name	Nomenclature	Category	Subcategory	Count	Tag
1	React	JS_PLT	PROGRAMMING LANGUAGE & TECHNOLOGY	FRONT-END	6	React
2	Go	JS_PLT	PROGRAMMING LANGUAGE & TECHNOLOGY	GENERAL PURPOSE	2	Go
3	Rust	JS_PLT	PROGRAMMING LANGUAGE & TECHNOLOGY	LOW LEVEL PROGRAMMING	1	Rust
4	C	JS_PLT	PROGRAMMING LANGUAGE & TECHNOLOGY	LOW LEVEL PROGRAMMING	3	C
5	C++	JS_PLT	PROGRAMMING LANGUAGE & TECHNOLOGY	LOW LEVEL PROGRAMMING	1	C++
6	R	JS_PLT	PROGRAMMING LANGUAGE & TECHNOLOGY	GENERAL PURPOSE	443	R
7	MATLAB	JS_PLT	PROGRAMMING LANGUAGE & TECHNOLOGY	GENERAL PURPOSE	1	MAT...
8	sed	JS_PLT	PROGRAMMING LANGUAGE & TECHNOLOGY	SCRIPTING	8	sed
9	Ada	JS_PLT	PROGRAMMING LANGUAGE & TECHNOLOGY	LOW LEVEL PROGRAMMING	2	Ada
10	Spring	JS_WF	WEB FRAMEWORK	JAVA	1	Spring
11	Expo	JS_MD	MOBILE DEVELOPMENT	CROSS PLATFORM	1	Expo
12	IAM	JS_CP	CLOUD PLATFORM	GOOGLE CLOUD PLATFORM	1	IAM
13	Ant	JS_BT	BUILD TOOLS	JAVA	7	Ant
14	ERM	JS_MGT	MANAGEMENT & LEADERSHIP	RISK MANAGEMENT	5	ERM
15	Derivatives	JS_FIN	FINANCE & ACCOUNTING	INVESTMENT & TRADING	1	Deriv...
16	MRI	JS_MED	MEDICAL & HEALTHCARE	DIAGNOSTICS	7	MRI
17	LIS	JS_MED	MEDICAL & HEALTHCARE	MEDICAL TECHNOLOGY	1	LIS
18	SEM	JS_MKT	MARKETING & ADVERTISING	SEO & SEM	4	SEM
19	Velocity	JS_PM	PROJECT MANAGEMENT	AGILE METHODOLOGIES	1	Velocity
20	XP	JS_PM	PROJECT MANAGEMENT	AGILE METHODOLOGIES	1	XP

Figure 4: Known Entities result

TABLE III. LEMMATIZER EVALUATION RESULT

Total Words Processed by Lemmatizer	Words with Incorrect or Missing Lemmas	Words with Correct Lemmas/ Roots
5000	396	4604
Accuracy = 92.08 %		

(1)

TABLE IV. EVALUATING ROOT WORD EXTRACTOR

Total Word Count in Corpus	5000
No. of Sentences	1347
Correctly Tagged Words in Identified Sentences	4604
Incorrectly Tagged and Unidentified Words	396 (220 + 176)
Total Incorrectly tagged Words	220
Completely Unidentified Words (untagged words)	176

Below equation (1) is evaluated for check the accuracy of individual POS tagging and generate the result for each pro-nouns, verbs and nouns as shown in table [5]. POS category wise evaluation results.

$$\textbf{\textit{Accuracy of Individual POS tags}} =$$

$$\frac{\textbf{\textit{No.of correctly tagged entities}}}{\textbf{\textit{Total no.of entities in corpus}}} \times \mathbf{100} \qquad (1)$$

Where, entities are Pronouns, Verbs and Nouns.

Table V: POS CATEGORY WISE EVALUATION RESULTS

POS category	Status	No. of words	No. of words	Individual POS Category Accuracy %
Pronouns	Correctly tagged	657	674	97.48
Pronouns	Incorrectly tagged	17	674	97.48
Verbs	Correctly tagged	1442	1584	91.03
Verbs	Incorrectly tagged	101	1584	91.03
Nouns	Correctly tagged	2384	2607	91.45
Nouns	Incorrectly tagged	223	2607	91.45
Completely Unidentified Words (untagged words)	Completely Unidentified Words (untagged words)	176		
Total Words	Total Words	5000		

The accuracy of tagged entities data are shown in table [6]

Table VI: ACCURACY OF TAGGED WORDS

Total no. of Words processed	Total no.of tagged words	No. of words with invalid (220) + missed POS tag (176)	No. of words with valid POS tag
5000	4824	396	4604
Overall Accuracy tagged words = 92.08 %	Overall Accuracy tagged words = 92.08 %	Overall Accuracy tagged words = 92.08 %	Overall Accuracy tagged words = 92.08 %

3. Conclusion

The proposed Job Skill Entity Recognition (JSER) system demonstrates an efficient and innovative approach to domain-specific named entity recognition (NER) in the Computer Science domain. The system works by using

things like Part-of-Speech tagging and syntactic processing to find entities that are already known and stored in a database and also entities that are not known but can be figured out by looking at the context. It uses a kind of data structure called a Trie to categorize entities, which makes it faster and more scalable than traditional methods.

There are ways to make the system better in the future. For example using models like BERT or RoBERTa can help it understand the context better when finding entities. The system can also learn from feedback and get better over time with active learning. If the system can understand languages it will be useful to people all around the world. It can also look at how skills change over time. Testing it with datasets will help people see how well it works and want to use it.

The system can also work with organizations without sharing private information thanks to federated learning.

The system is very good at finding entities it got 92.08% of them right. It can handle datasets and is very precise. The system is also flexible so it can be used for things like looking at resumes finding information and managing knowledge.

JSER is a solution for these tasks. However it is still hard to find entities that're complex and not known and to be accurate when the text is not formatted in a standard way.

In the future the system can be improved by making it understand the meaning of things using advanced deep learning models and making it useful for other professional fields. With these challenges JSER is a big step forward, for automated entity recognition and will help make talent assessment and recruitment technologies better.

Reference

1. "Named Entity Recognition," Medium, Available: https://medium.com/hr-ai/named-entity-recognition
2. "Named Entity Recognition (NER)," Shaip, Available: https://www.shaip.com/solutions/named-entity-recognition-ner/.
3. "Natural Language Processing (NLP)," Cloudflare, Available: https://www.cloudflare.com/learning/ai/natural-language-processing-nlp.
4. "Natural Language processing," DeepLearning.AI, Available: https://deeplearning.ai/resources/natural-language-processing/.
5. "CompilerDesign,"Available: https://www.tutorialspoint.com/compiler_design/index.html.
6. "Flask Tutorial," Available: https://www.javatpoint.com/flask-tutorial.
7. "PdfReaderDocumentation,"Available: https://pdfreader.readthedocs.io
8. Part-of-Speech(POS)tagging: https://journalofbigdata.springeropen.com
9. "WordNet Lemmatizer," Natural Language Toolkit (NLTK), Available: https://www.nltk.org/api/nltk.stem.wordnet.html.
10. F. Javed, M. McNair, F. Jacob, and M. Zhao, "Towards a Job Title Classification System," arXiv, 2016. DOI: 10.48550/arXiv.1606.00917
11. L. Stepanyan, "Automated Custom Named Entity Recognition and Disambiguation," International Journal of Signal Processing, Mar. 2020. Available: http://iaras.org/iaras/journals/ijsp.
12. A. Mansouri, L. S. Affendey, and A. Mamat, "Named Entity Recognition Approaches," IJCSNS International Journal of Computer Science and Network Security, vol. 8, no. 2, Feb. 2008.
13. Salman Naseera, Muhammad Mudasar Ghafoorb, Sohaib bin Khalid Alvia, Anam Kiranc, Shafique Ur Rahmand, Ghulam .Murtazae, 'Named Entity Recognition (NER) in NLP Techniques, Tools Accuracy and Performance', publication
14. T. A. Green, D. Maynard, and C. Lin, "Development of a Benchmark Corpus to Support Entity Recognition in Job Descriptions," in Proc. 13th Conf. on Language Resources and Evaluation (LREC 2022), Marseille, France, Jun. 2022, pp. 1201–1208.
15. L. Ratinov and D. Roth, "Design Challenges and Misconceptions in Named Entity Recognition," in Proc. Thirteenth Conf. on Computational Natural Language Learning (CoNLL-2009), Association for Computational Linguistics, 2009.
16. M.-I. Enăchescu, "Screening the Candidates in IT Field Based on Semantic Web Technologies: Automatic Extraction of Technical Competencies from Unstructured Resumes," Informatica Economică, vol. 23, no. 4, 2019.
17. J. R. Finkel and C. D. Manning, "Nested Named Entity Recognition," in Proc. 2009 Conf. on Empirical Methods in Natural Language Processing, Singapore, 2009, pp. 141–150.
18. F. Pech, A. Martinez, H. Estrada, and Y. Hernandez, "Semantic Annotation of Unstructured Documents Using Concept Similarity," Hindawi Scientific Programming, vol. 2017, Art. ID 7831897, 2017. DOI: 10.1155/2017/7831897.
19. M. N. V. S. Raghavendra, "Resume Screening Using Machine Learning," Journal of Engineering Sciences, vol. 13, no. 9, 2022.
20. V. Gadesha, K. D. Joshi, and M. Bachika, "Streamlining Resume Annotation with Named Entity Recognition," in Proc. 2023 IEEE 20th India Council International Conference (INDICON), Dec. 2023.

21. M. Kurisappan and S. S. Pandiyan, “Decoding math: A review of datasets shaping AI-driven mathematical reasoning,” Journal of Interdisciplinary Mathematics, vol. 28, no. 2, pp. 607–625, 2025. DOI: 10.47974/JIM-2105.
22. L. Chiticariu, R. Krishnamurthy, Y. Li, F. Reiss, and S. Vaithyanathan, "Domain Adaptation of Rule-Based Annotators for Named-Entity Recognition Tasks," in Proc. 2010 Conf. on Empirical Methods in Natural Language Processing, MIT, USA, Oct. 2010, pp. 1002–1012.
23. B. Fitzgerald, "Implementation of an Automated Text Segmentation System Using Hearst’s TextTiling Algorithm," in Proc. Annual Meeting of the Association for Computational Linguistics. DOI: 10.3115/981732.981783.
24. M. E. Tannous, W. H. Ramadan, and M. A. Rajab, "TSHD: Topic Segmentation Based on Headings Detection (Case Study: Resumes)," Hindawi Advances in Human-Computer Interaction, vol. 2023. DOI: 10.1155/2023/6044007.
25. V. R. Kudatarkar, M. Ramannavar, and N. S. Sidnal, "Survey on Unstructured Text Analytics Approaches for Qualitative Evaluation of Resumes," Int. J. Emerg. Technol. Comput. Sci. Electron. (IJETCSE), vol. 14, no. 2, Apr. 2015.
26. N. Sivaramakrishnan, V. Vandana, M. Vishali, and Dharshana, "Conversion of Unstructured Data to Structured Data with a Profile Handling Application," Int. J. Mech. Eng. Technol. (IJMET), vol. 8, no. 8, pp. Aug. 2017. Available: http://iaeme.com/Home/issue/IJMET
27. R. V. Kumar Reddy, G. Venugopal, G. Rajanala, V. Sambasivarao, N. Harshavardhan, and T. Ajay, "Transforming Unstructured Data to Structured Data Using Map Reduce and HBase," Int. J. Emerg. Trends Eng. Res., vol. 8, no. 9, Sep. 2020. DOI: 10.30534/ijeter/2020/137892020
28. B. Kinge, S. Mandhare, P. Chavan, and S. M. Chaware, "Resume Screening Using Machine Learning and NLP: A Proposed System," Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol., DOI: 10.32628/CSEIT228240
29. S. Bhor, V. Gupta, V. Nair, H. Shinde, and M. S. Kulkarni, "Resume Parser Using Natural Language Processing Techniques," Int. J. Res. Eng. Sci. (IJRES), vol. 9, no. 6, pp. 01–06, 2021. Available: www.ijres.org
30. H. S. Sharma and A. Sharma, “Enhancing information retrieval through advanced query expansion techniques and semantic analysis,” Journal of Information & Optimization Sciences, vol. 46, no. 5, pp. 1587–1603, 2025. DOI: 10.47974/JIOS-1839.
31. N. Patil, S. Yadav, and V. Biradar, "Resume Parser and Analyzer Using NLP," Int. Res. J. Modernization Eng. Technol. Sci., vol. 5, no. 4, Apr. 2023. Available: www.irjmets.com
32. P. Das, M. Pandey, and S. S. Rautaray, "A CV Parser Model Using Entity Extraction Process and Big Data Tools," I.J. Inf. Technol. Comput. Sci., vol. 9, pp. 21–31, Sep. 2018. Available: http://www.mecs-press.org/. DOI: 10.5815/ijitcs.2018.09.03
33. M. Tiwari, P. Aital, and P. Joshi, “A comprehensive survey and review of machine learning techniques in document processing: Industry applications and future directions,” Journal of Information & Optimization Sciences, vol. 45, no. 4, pp. 1177–1188, 2024. DOI: 10.47974/JIOS-1701.
34. K. Stratos, M. Collins, and D. Hsu, "Unsupervised Part-Of-Speech Tagging with Anchor Hidden Markov Models," Trans. Assoc. Comput. Linguist. Dec. 2016. DOI: 10.1162/tacl_a_00096