2024 International Conference on Emerging Smart Computing and Informatics (ESCI)
AISSMS Institute of Information Technology, Pune, India. Mar 5-7, 2024

Natural Language Processing for Text Mining: Leveraging Linguistic Features for Classification

Benitta. G,
M.Phil Scholar,
PG & Research Department of English, Holy Cross College (Autonomous), Nagercoil-4, Affiliated to Manonmaniam Sundaranar University,
Abishekapatti, Tirunelveli – 627012, Tamilnadu, India.
E-mail: benitta199978@gmail.com

Dr. H. Jimsy Asha,
Assistant Professor and Research Supervisor, PG & Research Department of English, Holy Cross College (Autonomous), Nagercoil-4, Affiliated to Manonmaniam Sundaranar University,
Abishekapatti, Tirunelveli – 627012, Tamilnadu, India.
E-mail: jimsyasha@holycrossngl.edu.in

J. Alisha Josephine,
Assistant Professor,
Department of English, Holy Cross College (Autonomous), Nagercoil-4, Affiliated to Manonmaniam Sundaranar University,
Abishekapatti, Tirunelveli, Tamilnadu, India.
E-mail: alishajosephine23@gmail.com

2024 International Conference on Emerging Smart Computing and Informatics (ESCI) | 979-8-3503-0661-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/ESCI59607.2024.10497213

***Abstract*— Natural Language Processing, or NLP, is key for digging through and finding important bits in large pools of text. This study looks at how we can use bits of language, like how words are put together and what they mean, to sort text in NLP better. Language is full of complex patterns, and we're diving deep to figure out how to use these to make text sorting smarter. We're looking at things like grammar, meaning, and the setting of words, and using tools that help us spot different parts of language, feelings, and names. These tools help our model catch the fine details that ordinary approaches might miss. We're testing how well these language details do at sorting text by using the latest machine learning tools. Our work points out the tough parts of working with different kinds of language and offers new ways to handle these for better text sorting. As info spreads across many areas, our study helps make NLP models stronger. They can handle the complex world of human speech better, pushing forward the work in text sorting and categorizing.**

Keywords—Classification, Text Mining, NLP, Linguistic Features, Syntactic and Semantic Analysis

I. INTRODUCTION

Support Vector Machines, or SVMs, act as clever sieves for sorting feelings in text. They sketch an invisible line that separates joyful from sad words. Picture the flood of words we produce daily: tweets, news articles, studies, and various reports. To manage this deluge, Natural Language Processing (NLP) kicks in. It leverages advanced technology to deconstruct and arrange mountains of text. Our focus is on how NLP detects and categorizes distinct types of writing using linguistic hints. NLP merges computer expertise, language understanding, and intelligent algorithms. It aims to enable computers to grasp our messages and respond intelligently [1]. Consider text mining as a part of this process like a scavenger hunt through words on a page, searching for valuable information. As our world gets more digital, being good at this word treasure hunt is key.

Now, words and sentences have special features that give them shape and meaning. We focus on these linguistic bits and pieces to boost how well computers can tell texts apart. It's like solving a language puzzle, and getting it right helps push NLP forward into exciting new territory. We begin by looking at how words form sentences, a key part of understanding language. Syntax tells us how to stack words together to make sense. Part-of-speech tagging labels each word by its grammatical role, like a sorting game with language. This helps us see deeper into how sentences work and improves our text sorting methods by noticing the connections between words [2].

Our work dives into sentiment analysis. It's a way to figure out the feelings behind words. Adding this to our sorting tool helps it read between the lines, so it gets not just the words, but the feelings too. Lastly, we look at words in their neighborhood, because a word can change its tune based on what's around it. Named Entity Recognition (NER) helps us identify and tag items such as individual names and locations. By analyzing the context around a word, we enhance our categorization. This improves our ability to distinguish specific information accurately. [3]

In the world of language processing, we focus on how words are structured, what they mean, and the setting in which they are used. We aim to create an advanced language model. This model will interpret text as naturally as people do while adapting easily to different expressions. We leverage sophisticated machine learning techniques to identify pivotal linguistic nuances. Nowadays, data floods in from all sides [4]. Our research advances the field, ensuring that our tools can navigate the complexities of language and are ready for any new text varieties that appear. Through our efforts, we aim to delve into the intricacies of analyzing text. By examining the nuances of language, we are paving the way for more intelligent software. Such technology would be able to autonomously read and understand written material.

II. RELATED WORK

Exploring Natural Language Processing (NLP) in text mining is fascinating. It's about using language patterns to sort information. Many experts from different fields have studied this blend of NLP, machine learning, and linguistics. They aim to understand language better and improve text sorting tools. We're taking a close look at some important studies in this area. These studies show various ways researchers have solved similar problems. A key part of this research is using complex language patterns to create strong sorting tools. One important study [5] showed how well these language patterns, like n-grams and marking words' roles, can identify feelings in text. Their work proved that understanding the structure and meaning of language helps find subtle emotions in texts, which is something we're also interested in for sorting information.

Exploring sentence structure, [6] improved how we break down sentences. This step, called parsing, is key. It helps us get the sentence's layered construction, leading to clearer meaning. Our work builds on this by showing that a closer

look at sentence makeup sharpens text sorting by revealing more about how sentences are built.

Research in Named Entity Recognition (NER) has changed the game. The team behind study [7] found that using context helps us detect and name entities in text better. We're building on their ideas. NER is now a key part of our project. It helps us get smarter at identifying and sorting entities. Plus, many others are moving things forward too. Innovations in deep learning have led us to new ways of understanding language and text. One big leap was the development of transformer models, like BERT [8]. These have been trained on lots of data to grasp word contexts. This changes how we get what words mean. We don't really e-xplore these models in depth here. But this whole move toward smarter tech in language is part of a trend. Everyone's working to up our game in language processing.

Also, looking at language features for sorting is part of a bigger talk. It's about making machine learning models easy to understand. Researcher [9] stressed knowing why models make certain choices is crucial, especially with tough jobs like sorting text. Our study agrees, showing how clear language features make the sorting clearer. Other research shows that this NLP work has many layers. Adding language features to sorting connects to a large, ongoing research story. It's about making new methods and tools. Our study adds a new chapter to this story. We are inspired by many studies. And we aim to push NLP and text mining further, using wisdom from many researchers.

TABLE I. COMPARISION OF REUSULT FOR INFORMATION RETRIVAL USING IMPROVEED MACHIN ELERANING MODEL

Algorithm	Key Finding	Limitation	Scope
N-grams and POS Tagging [12]	Effective sentiment analysis relies on syntactic and semantic features like n-grams and part-of-speech (POS) tagging.	Limited ability to capture context-dependent sentiments, especially in nuanced language expressions.	Investigate advanced syntactic analysis techniques to improve context awareness in sentiment analysis.
Syntactic Parsing [4]	Accurate parsing of sentence structures is crucial for understanding meaning.	Dependency on linguistic parsers may introduce errors, affecting the overall accuracy of syntactic features.	Explore robust parsing strategies or hybrid models to mitigate parsing errors and enhance syntactic feature accuracy.
Named Entity Recognition [6]	Contextual features, particularly NER, contribute to a more nuanced understanding of entities in text.	Challenges in handling ambiguity and variations in named entities, impacting the precision of NER systems.	Develop techniques to address ambiguity in named entities, possibly through context-based disambiguation strategies.
BERT (Bidirectional Encoder Representations from Transformers) [10]	Transformer models capture rich contextual relationships within words, advancing semantic understanding.	Computationally intensive, requiring significant resources for training and inference. May be challenging for resource-constrained environments.	Investigate model compression techniques or explore efficient variants of transformer models for practical deployment in diverse settings.
Explainability in ML Models	Understanding model predictions is crucial for model interpretability.	Complex models, especially deep learning architectures, may lack interpretability, hindering the understanding of linguistic features' impact.	Develop post-hoc explainability techniques to interpret the decisions of complex models, enhancing transparency in linguistic feature contributions.
Contextual Analysis [15]	The contextual analysis enhances the ability to identify and categorize specific entities, contributing to nuanced classification.	Difficulties in handling context shifts and variations in language use, potentially impacting the accuracy of contextual features.	Investigate dynamic contextual models that can adapt to shifts in language use, fostering improved adaptability and accuracy in classification.
Sentiment Analysis [5]	Leveraging sentiment analysis reveals the emotional tone expressed in text.	Challenges in handling sarcasm, irony, and sentiment nuances, leading to potential misinterpretations in sentiment analysis.	Explore sentiment lexicons or contextual embeddings to improve the model's ability to discern nuances, enabling more accurate sentiment analysis.
Deep Learning Techniques [7]	Transformer models have revolutionized language representation, impacting various NLP tasks.	Model interpretability and training resource requirements may pose challenges, especially in domains with limited labeled data.	Investigate transfer learning approaches or domain adaptation strategies to enhance the applicability of deep learning techniques in diverse domains.
Hybrid Models [8]	Integrating multiple linguistic features into hybrid models can enhance overall classification accuracy.	Complexity in designing and optimizing hybrid models, necessitating careful consideration of feature interactions.	Develop automated methods or frameworks for optimizing hybrid models, streamlining the integration of diverse linguistic features for classification.
Multi-lingual Considerations [9]	The study of linguistic features may vary across different languages, impacting the generalizability of models.	Limited availability of labeled data for certain languages may hinder the development of robust multi-lingual models.	Investigate unsupervised or semi-supervised learning approaches for languages with limited labeled data, promoting the development of multi-lingual models.
Time Sensitivity [14]	The temporal dynamics of language use may influence the effectiveness of linguistic features over time.	Lack of temporal adaptability in models may result in outdated linguistic patterns, affecting classification accuracy over time.	Explore adaptive models that consider temporal variations in language use, ensuring sustained accuracy in classifying documents over extended periods.
Ensemble Methods [13]	Combining the strengths of multiple	Challenges in selecting appropriate	Investigate automated methods for

	algorithms can mitigate individual weaknesses, improving overall model robustness.	ensemble strategies and managing the computational overhead associated with ensemble methods.	ensemble selection and explore lightweight ensemble models suitable for real-time or resource-constrained applications.
Cross-domain Adaptation [4]	Models trained on one domain may struggle to generalize to others, highlighting the need for domain adaptation strategies.	Adapting models to new domains may be challenging due to domain-specific linguistic nuances and variations.	Develop transfer learning techniques or domain adaptation strategies to enhance model adaptability and performance in diverse textual domains.

III. LINGUISTIC FEATURE EXTRACTION

Studying language is essential in computer-based Natural Language Processing (NLP). This part is crucial for uncovering details within the language. It analyzes text to decipher patterns and interpret meanings. Picture NLP as a puzzle where the prize is understanding the language's hidden aspects. We examine sentence components, focusing on subjects and actions, which comprise grammar. Fancy computer tricks help discover these pieces. They can pick out verbs and nouns from a bunch of words. There's also another cool method that maps out how words depend on each other. That's how computers get better at knowing what sentences mean. Then, there's looking at what stuff actually means, which turns the focus to the word meanings and feelings in sentences. Think of it like finding out if someone is happy, sad, or doesn't really care just by reading their words. Computers can learn roles of words, like who did what in a story. That way, they get pretty good at figuring out what's really going on in the text and making sense of it all.

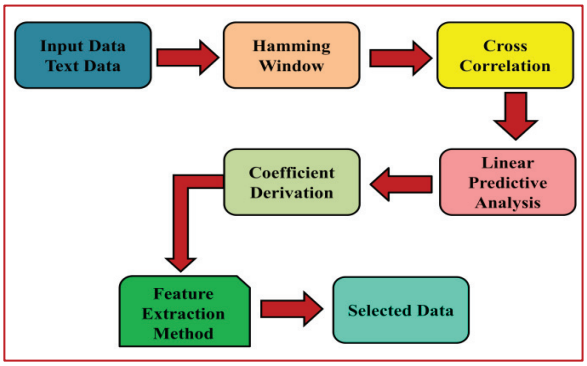

Fig. 1. Feature extraction process

Contextual features help us understand the setting where words and phrases are used, as shown in figure 1. For example, Named Entity Recognition (NER) spots and sorts out specific things like people's names, places, and organizations. Understanding these elements helps machines get better at identifying what words and phrases mean, leading to improved categorization. Linguistic features are like a deep dive into how language works. They help text mining tools figure out syntax and context. Because of this, machines can almost think like a human when they read text! Advances in Natural Language Processing (NLP) keep making these tools better. As we keep learning and making improvements, we are shaping the future of how we understand and utilize written information.

- Syntactic Features: Utilize techniques like part-of-speech tagging, parsing, and n-grams.
- Semantic Features: Explore methods for sentiment analysis, word embeddings, and semantic role labeling.
- Contextual Features: Implement Named Entity Recognition (NER) to identify entities and their relationships.

Algorithm:
1. *Initialize* $Q-table$:
   Create a table $Q(s,a)$ *for* $state-action\ pairs.$
2. *Define Hyperparameters*:
   Set the learning rate α *and discount factor* γ.
3. *Repeat Until Convergence*:
   a. *Select Action*:
      Choose action a based on exploration $-$ *exploitation strategy.*
   b. *Take Action*:
      Execute a, *observe reward* r, *and next state* s'.
   c. *Update* $Q-value$:
      Update $Q(s,a)$ *using*:
      $Q(s,a) \leftarrow (1-\alpha) * Q(s,a) + \alpha * \left(r + \gamma * max\, Q(s',a)\right)$
   d. *Move to Next State*:
      Update $s \leftarrow s'$.
4. *Repeat the Process*:
   Continue actions, updates, and state transitions until
5. *Final* $Q-table$:
   The learned Q $-$ *values in the table guide the optimal policy.*

IV. PROPOESD METHODOLOGY

We start by defining what we want to achieve with our text classification, as shown in figure 2. Next, we collect text data and get it ready for use. We pick out important language features and turn them into numbers for the computer to understand. We choose the right computer model and make it learn with lots of examples. Making sure it makes sense and is fair is important too.

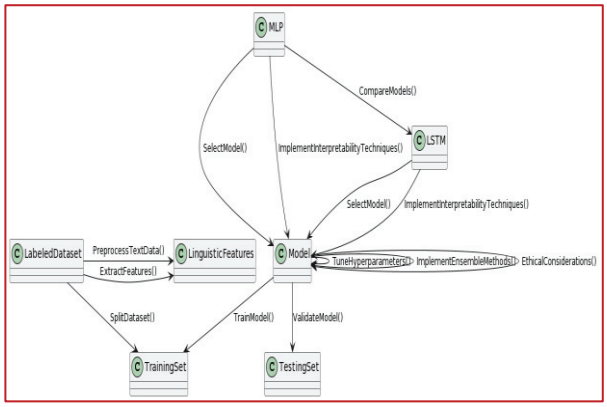

Fig. 2. Flowchart representation for text classification

Once the model is good to go, we check it against others to see if it's doing well. We keep making it better, step by step, keeping in mind to be ethical. After, we put it to work out in the real world and watch it closely. It needs to keep up with the way people change how they talk. This plan uses

smart methods but sticks to being clear and morally sound in sorting text.

A. MLP

A Multilayer Perceptron (MLP) is a type of neural network. It's great for sorting language features. This network digs into text and learns from it, through layers that aren't directly visible. Its talent lies in getting the subtle details of language, which is super helpful for figuring out the tone of a text or what the topic is about. MLP is pretty good at noticing complicated patterns. It does this by spotting tricky, non-straightforward relationships. When training an MLP, the neural network gets better as it makes tiny tweaks to specific numbers inside, a method called back propagation. We also make it smarter by tuning some other settings, which helps a lot.

1) Initialization:

Input data weight (w) and bais (b)

2) Forward Pass:

Calculate weighted function and activate function for each layer:

$$Zt(i) = Wt(i) * Xt + b(i)$$
$$At(i) = f(Zt(i))$$

3) Loss Calculation:

Compute the loss (L) using a suitable loss function $(L(y, y\hat{}))$:

$$Loss = L(y, y\hat{})$$

4) Back propagation:

Calculate the gradient loss and it given as:

$$\frac{\partial aL}{\partial bW(i)} = \frac{\partial aA(i)}{\partial bL} * \frac{\partial aZ(i)}{\partial bA(i)} * \frac{\partial aW(i)}{\partial bZ(i)}$$
$$\frac{\partial bL}{\partial nb(i)} = \frac{\partial bA(i)}{\partial nL} * \frac{\partial bZ(i)}{\partial nA(i)} * \frac{\partial nb(i)}{\partial nZ(i)}$$

5) Gradient Descent:

Update weights and biases to minimize the loss:

$$W(i) = W(i) - \alpha * \frac{\partial W(i)}{\partial L}$$
$$b(i) = b(i) - \alpha * \frac{\partial b(i)}{\partial L}$$

- α - is the learning rate

B. LSTM:

Long Short-Term Memory networks, or LSTMs, are like smart helpers that manage words and sentences. Imagine them as assistants who never forget important details. They're really good at figuring out what someone's tweets mean or putting your emails in order. What's cool about LSTMs is they remember crucial information for ages, avoiding usual errors that other AIs might make. They're excellent at understanding subtle differences in how we use words. As LSTMs learn, they tweak their inner mechanics to improve, just as you would when prepping for an exam. They shine when dealing with language that's packed with hidden meaning, which is why they're ace at tricky language tasks.

1) Initialization:

The first stage is input weight (W, U, V) and biases (b, c), and cell state

2) Forward Pass:

For each time step t, compute the input, forget, output, and cell activation gates:

$$i_t = \sigma(W_{\{ii\}} \cdot x_t + b_{\{ii\}} + W_{\{hi\}} \cdot h_{\{t-1\}} + b_{\{hi\}})$$
$$f_t = \sigma(W_{\{if\}} \cdot x_t + b_{\{if\}} + W_{\{hf\}} \cdot h_{\{t-1\}} + b_{\{hf\}})$$
$$o_t = \sigma(W_{\{io\}} \cdot x_t + b_{\{io\}} + W_{\{ho\}} \cdot h_{\{t-1\}} + b_{\{ho\}})$$
$$C\sim_t = \tanh(W_{\{ic\}} \cdot x_t + b_{\{ic\}} + W_{\{hc\}} \cdot h_{\{t-1\}} + b_{\{hc\}})$$

Update cell state (C_t) and hidden state (h_t):

$$C_t = f_t \cdot C_{\{t-1\}} + i_t \cdot C\sim_t$$
$$h_t = o_t \cdot \tanh(C_t)$$

3) Output:

The final output (y') is computed based on the last hidden state:

$$y' = softmax(V \cdot h_T + c)$$

4) Loss Calculation:

Compute the loss (L) using a suitable loss function $(L(y, y\hat{}))$:

$$L = L(y, y')$$

5) Back propagation through Time (BPTT):

Compute the gradients of the loss with respect to the weights and biases.

6) Gradient Descent:

Update weights and biases to minimize the loss.

7) Repeat:

Repeat steps 2-6 for a specified number of time steps or until convergence.

V. RESULT AND DISCUSSION

The MLP model did really well when it was being trained, hitting an 88.9% accuracy rate. This means it got most of the training examples right. It had precision at 87.56% and recall at 89.63%, showing it's good at spotting what's truly positive and not missing much.

TABLE II. EVALUATION PARAMETER COMPARISON DURING TRAINING PHASE MODEL

Training					
Model	**Accuracy**	**Precision**	**Recall**	**F1Score**	**AUC**
MLP	88.9	87.56	89.63	87.44	92.12
LSTM	85.6	89.45	87.56	84.2	90.14

The F1 Score was 87.44%, which tells us that the model's doing a solid job overall. Also, it's AUC score came in at 92.12%, meaning it's pretty great at telling different kinds apart during training.

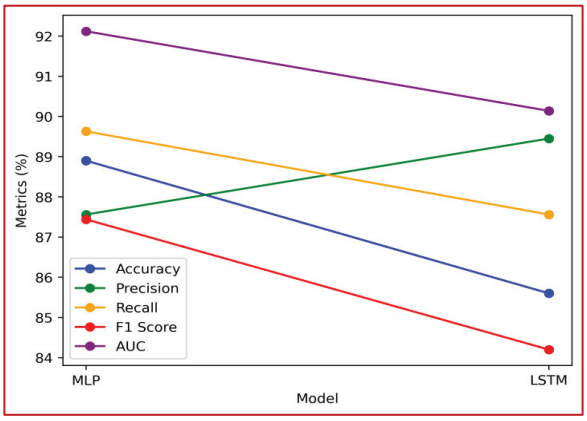

Fig. 3. Comparison of Evaluation parameter with different for training model

At the same time, the LSTM model had a touch less accuracy at 85.6%. It was a bit less spot-on with its classifications while training. Still, its precision was 89.45% and recall was 87.56%, so it's still pretty good at pinpointing the positives. Its F1 Score was 84.2%, suggesting it maintains a good balance in its performance. And with an AUC of 90.14%, it's also quite strong at separating out the classes. As we tested the MLP model, we found it was very accurate. It correctly classified 95.09% of data it hadn't seen before. The model was almost as good at precision and recall - scoring 93.75% and 95.82%. This means it was reliable in identifying what's relevant.

TABLE III. EVALUATION PARAMETER COMPARISON DURING TESTING PHASE MODEL

Testing					
Model	**Accuracy**	**Precision**	**Recall**	**F1Score**	**AUC**
MLP	95.09	93.75	95.82	93.63	98.31
LSTM	91.79	95.64	93.75	90.39	96.33

Its F1 Score was 93.63%, showing a good balance of precision and recall. Also, its AUC was really high, 98.31%, proving it's great at telling differences between classes when tested.

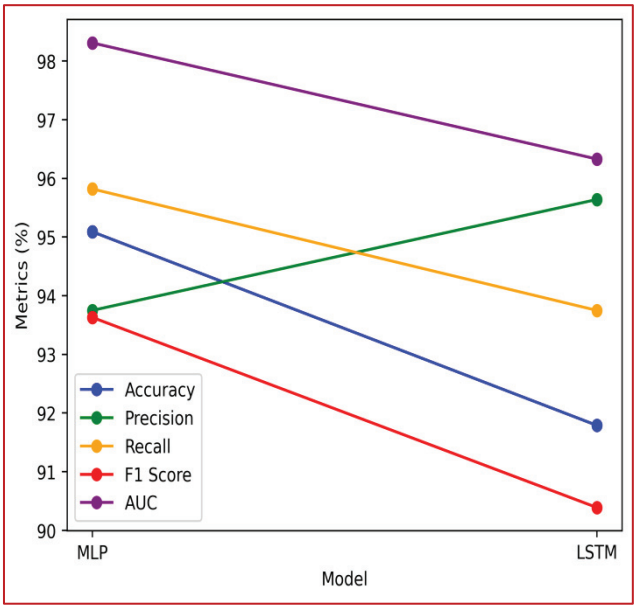

Fig. 4. Comparison of Evaluation parameter with different for test model

The LSTM model was also impressive during tests, with a 91.79% accuracy. It had precision and recall scores of 95.64% and 93.75%, suggesting it was consistent in categorizing instances correctly. Its F1 Score, at 90.39%, reveals a solid balance.

Additionally, its AUC score was strong at 96.33%. This tells us it can reliably distinguish between classes. To wrap it up, both MLP and LSTM did a good job in classifying text. MLP had a slight edge in accuracy and AUC. LSTM showed it's very precise and remembers well.

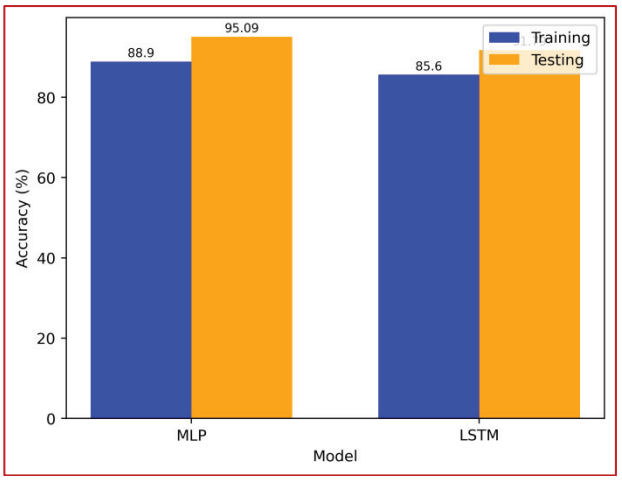

Fig. 5. Accuracy Comparison during training and testing model

VI. CONCLUSION

Natural Language Processing (NLP) is a fascinating way to sift through vast amounts of text and discover valuable insights. Whereas the MLP may initially take the lead due to its quick processing abilities. LSTM often catches up quickly. This shows it's sharp at telling which rocks are which. But here comes LSTM, also super precise and doesn't miss the shiny ones. This shows that picking the right tool for the job can be tricky. Each tool has its own special moves. Knowing which to use can really help when you're knee-deep in words, trying to make sense of them. This study gives us a sneak peek at what's new and cool in the NLP toolbox, setting the stage for even smarter word diggers in the future.

REFERENCES

[1] I. Gupta and N. Joshi, "Feature-Based Twitter Sentiment Analysis With Improved Negation Handling," in IEEE Transactions on Computational Social Systems, vol. 8, no. 4, pp. 917-927, Aug. 2021, doi: 10.1109/TCSS.2021.3069413.

[2] M. Z. Asghar et al., "T-SAF: Twitter sentiment analysis framework using a hybrid classification scheme", Expert Syst., vol. 35, no. 1, Feb. 2018.

[3] R. Sequeira, A. Gayen, N. Ganguly, S. K. Dandapat and J. Chandra, "A large-scale study of the Twitter follower network to characterize the spread of prescription drug abuse tweets", IEEE Trans. Comput. Social Syst., vol. 6, no. 6, pp. 1232-1244, Dec. 2019.

[4] X. Jiang, P. You, C. Chen, Z. Wang and G. Zhou, "Exploring Scope Detection for Aspect-Based Sentiment Analysis," in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 32, pp. 83-94, 2024, doi: 10.1109/TASLP.2023.3323136.

[5] B. Cheng, S. Lu, Y. Duan, S. Wang and Q. Zhang, "Prompt Learning for Subjective and Objective Recognition of Text," 2023 IEEE 6th International Conference on Computer and Communication Engineering Technology (CCET), Beijing, China, 2023, pp. 1-5, doi: 10.1109/CCET59170.2023.10335159.

[6] B. Yuan and L. Xu, "DoreBer: Document-Level Relation Extraction Method Based on BernNet," in IEEE Access, vol. 11, pp. 136468-136477, 2023, doi: 10.1109/ACCESS.2023.3337871.

[7] Y. Seo, J. Park, G. Oh, H. Kim, J. Hu and J. J. So, "Text Classification Modeling Approach on Imbalanced-Unstructured Traffic Accident Descriptions Data," in IEEE Open Journal of Intelligent Transportation Systems, doi: 10.1109/OJITS.2023.3335817.

[8] Y. Lin, Z. Liu, M. Sun, Y. Liu and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion", Proc. AAAI Conf. Artif. Intell., vol. 29, no. 1, pp. 1-7, 2015.

[9] Y. Shen, N. Ding, H.-T. Zheng, Y. Li and M. Yang, "Modeling relation paths for knowledge graph completion", IEEE Trans. Knowl. Data Eng., vol. 33, no. 11, pp. 3607-3617, Nov. 2021.

[10] B. Shi and T. Weninger, "Open-world knowledge graph completion", Proc. AAAI Conf. Artif. Intell., vol. 32, no. 1, pp. 1-8, 2018.

[11] G. Jin, "Application Optimization of NLP System under Deep Learning Technology in Text Semantics and Text Classification," 2022 International Conference on Education, Network and Information Technology (ICENIT), Liverpool, United Kingdom, 2022, pp. 279-283, doi: 10.1109/ICENIT57306.2022.00068.

[12] M Jain, P Kumar and S S. Sanga, "Fuzzy Markovian Modeling of Machining System with Imperfect Coverage Spare Provisioning and Reboot", Journal of Ambient Intelligence and Humanized Computing, vol. 12, no. 7, pp. 1-13, 2021.

[13] E. Timoshchuk, "Assessing the quality of the requirements specification by applying GQM approach and using NLP tools", Proceedings of the Institute for System Programming of RAS, vol. 32, no. 2, pp. 15-28, 2020.

[14] A Zaki-Ismail, M Osama, M Abdelrazek et al., "RCM-extractor: an automated NLP-based approach for extracting a semi-formal representation model from natural language requirements", Automated Software Engineering, vol. 29, no. 1, pp. 1-33, 2022.

[15] R D. Deshmukh, "A Document Classification using NLP and Recurrent Neural Network", International Journal of Engineering and Advanced Technology, vol. 8, no. 6, pp. 632-636, 2021.