International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Revue d'Intelligence Artificielle
Vol. 36, No. 4, August, 2022, pp. 575-580
Journal homepage: http://iieta.org/journals/ria

Deep Named Entity Recognition in Hindi Using Neural Networks

Rita Shelke[*], Sandeep Vanjale

Department of Computer Engineering, Bharati Vidyapeeth (Deemed to be University) College of Engineering, Pune 411030, India

Corresponding Author Email: ritashelke@gmail.com

https://doi.org/10.18280/ria.360409

Received: 8 July 2022
Accepted: 12 August 2022

Keywords:
NER, Hindi language, BiLSTM, ML

ABSTRACT

In Natural Language Processing, named entity recognition (NER) is a task of (NLP) that tries to automatically identify and annotate Named Entities in text, such as people, places, and organisations. We employ a deep learning-based architecture in this work to solve the problem of recognising named entities in a Hindi text phrase. In the literature, approaches based on bidirectional long short-term memory (BiLSTM) have been utilized for the NER task. We performed recursive BiLSTM in this study, which includes a de-noising autoencoder with conditioning logic. Experiments were undertaken to examine the behaviour of individual word embeddings and batch sizes, which is vital for training deep models. The findings of the suggested system architecture, as well as a comparison of performance characteristics with existing systems, are presented in this study.

1. INTRODUCTION

Natural Language Processing aims to recognise Named Entities (NEs) in a textual source and classify them into many conceptual categories (Name, Place, Party, and Designation) (NLP). This is acknowledged as Named Entity Recognition (NER). Information is abundant in this age of the Internet, yet there is very little work done on NLP and analytics in Indian languages. Information retrieval, etc are examples of applications and efficient search engines, NER is very important. Considering to the syntactic and semantic difficulties of Indian languages, as well as a shortage of processing tools, NER development for Indian languages continues a challenge. Since Hindi is India's official language, it was chosen for this research work to develop a NER tool using Machine Learning-based method to achieve greater performance than currently available. The current technique can be implemented to other Indo-Aryan languages such as Marathi, Bhojpuri, Rajasthani, Gujarati, Punjabi, Odia, and so on, as ideal NER tools are not available in these languages.

This system was established to solve some of the shortcomings of current systems. The following are some of the issues that were discovered:

- A lack of precision Tagger for parts of speech
- Although rule-based systems are more accurate, they are not configurable.
- The accuracy of NER systems varies depending on the text.
- Existing systems consider a limited tagset.
- In some circumstances, proper nouns are employed as common nouns.

These issues are addressed in the proposed system by a dynamically produced NLP algorithm that employs a neural network method. Cascaded BiLSTM has been trained in such a way that the concerns described above have been considerably resolved.

We noticed that Patil et al. [1] use a probabilistic Hidden Markov Model (HMM) trained on a manually tagged corpus to address the difficulty. The suggested system achieves when no pre-processing is employed, pre-processing is utilised [1]. The system described has great success at recognising people, places, numbers, and measurements, but not so well with other named items [2]. To increase efficiency, Patil et al. [1] use pre-processing techniques like as lemmatization, which can be complex.

Sinha [3] focuses on applying CRF to decipher Ambiguous Proper Names (APN) in Hindi. A common noun is a name that may also be used as an APN. Sinha [2] also cites the necessity for a relevant corpus to be derived in his study. After that, a relevant corpus was created, which was separated into three categories: names (NEP), dictionary terms that aren't, where the result is effectively OR-ed. Sharma and Goyal [4] looked at a total of 29 characteristics, including Context words, Word prefixes, Word suffixes, POS data, and Gazetteer Lists, which are lists of people's names, places' names, and organisations' names.

Chopra et al. [4] used the Hidden Markov Model (HMM) to perform NER in Hindi. HMM facilitates in the construction of language-independent NER systems, as they've stated. Scaling and evaluating these systems are also a breeze. Ekbal et al. [5] uses Bengali named entity recognition using classifier combination.

For the English language, Singh et al. [6] developed the algorithm they devised was put to the test on 500 sentences, with accuracy ranging from 60 to 70%. For Parts-of-Speech (POS) labelling, the system employs a Deep Neural Network. In Indian languages, POS is a crucial signal for distinguishing Named Entities. The custom rules additionally check for a missing first letter in the Entity in question.

Vora et al. [7] have also employed HMM for Gujarati language. The accuracy of their system is not mentioned in the study. However, named entity recognition in Gujarati is uncommon and understudied. Vora et al. used HMM for their NER in Gujarati since it is simple to implement in any language. Printed versions of Gujarati text are used as input. According to them, penmanship varies from person to person, hence their technology would not recognise handwritten documents.

Because the quality of the input data determines the quality of word embedding [8]. As previously stated, the fasttext technique for English language NER has shown results that are comparable to normal approaches for named entity recognition. However, it has been shown that in regional languages such as Hindi, due to the lack of a big corpus of data, tests are conducted using a conventional Deep learning algorithm and a traditional technique. In this paper, we examine the performance of the NER in relation to the BiLSTM neural network using a unique design [9].

BERT is a Google-developed embedding dimension with 768 dimensions that was trained on a sub-word level using the Wikipedia corpus in various languages. A bidirectional transformer was trained for language modelling as part of the model architecture. This unique strategy, known as Masked LM (MLM), improved the performance of their word embedding model for word representation as well as application in other NLP tasks [10].

Because the quality of the translation would decrease if they didn't, they took on the issue of dealing with named entities. Because a rule-based strategy proved insufficient, they used a hybrid technique, gathering 10,000 phrases from news websites as a corpus and identifying names using Stanford's NER tool. From a total of 9234 name entities, the system generated 9180 with an accuracy of 83.65% precision, 83.16 percent recall, and 83.40 percent F-Measure value [11].

In their article, they observed that Kannada is the only Indian language that lacks capitalization, larger gazetteers, standardisation, and spelling. They detected a lack of annotated data as well as a highly agglutinating and inflected language. They used Multinomial Nave Bayes classifiers to construct a Supervised Statistical Machine Learning system for Kannada Language. They must use 22 identified entities and a corpus of 95170-words. Their developed model for identifying named entities had 83 percent accuracy, 79 percent recall, and 81 percent F-Measure value [12].

They were able to accomplish success despite the limitations imposed by the nature of Arabic and the scarcity of linguistic resources. The relations don't have enough annotated instances to be used for learning approaches since the accessible corpora aren't annotated with name entities. They used Machine Learning and Genetic Algorithm concepts to increase the overall performance of the machine learning approach. A hybrid approach [13] achieves precision of 84.8 percent, recall of 67.6%, and F-Measure value of 75.22 percent.

By combining Rule Based with List Look, they were able to overcome the restrictions of knowledge corpus availability and resolve ambiguities in named entity identification in Hindi. For Location, Person, Organization, Date, Money, Direction, and Transportation, among other things, they discovered varied Precision, Recall, and F-Score values. The accuracy of their procedure is 95.77 percent [14].

They were able to overcome challenges such as confusing names, no capitalization, a lack of resources and tools, a lack of standards and spelling, the lack of labelled data, and the lack of a comprehensive Hindi gazetteer. To increase the performance of the Hindi language, they must use a variety of NER procedures as well as voting systems. Using the CRF approach, they got 71.43 percent Precision, 30.86 percent Recall, and 43.10 percent F-1 Measure for five testing files. MaxEnt received 76.92 percent accuracy, 19.8 percent recall, and 31.49 percent F-1 measure for five testing files. The rule-based work achieve 96.05 percent accuracy, 86.90 percent recall, and 91.25 percent F-1 measure for three testing files [15].

They ran into a variety of problems due to the intricate morphology of the Kashmiri language. The Kashmiri language, unlike English, lacks capitalization. They needed to perform a noun recognition test and used a dictionary gazetteer, lists, and morphological suffix mapping procedures to get satisfactory results. They were given the task of identifying nouns and received a score of 93.32 percent with 07.75 percent errors. In addition to using NE tags to resolve ambiguity, gazetteers lists and features are also utilised to do so [16].

They've run into difficulties with the Manipuri language, which is one of India's constitutional languages. No capitalization, redundant named items in the dictionary with additional specialised meanings, very inflectional language resulting in enormously intricate word forms, free ordering language impossible to compare to others, resource limited language The evaluation was done with the CRF approach, and the findings were 81.12% recall, 85.67% accuracy, and 83.33 percent Score value. Manipuri gazetteer listings are typically created using the NER [17].

They described named entity recognition for the Missing language. Assam has a Tibeto-Burman minority. It's a language with a limited number of resources. They used a 12 named entity tagset, as well as Roman Script, to extract features. The classification of recognised things necessitates the use of a Support Vector Machine. Because the language has little resources, authors must develop their own corpus. Out of 34000 data points, 16000 were assessed, yielding 90.58 percent recall, 85.14 percent accuracy, and 87.77 percent F-Score score [18].

No capitalization, a lack of resources, agglutinative nature feature, free-word order, complexity of spelling variants, borrow words, nested and compound named entities, and many more hurdles to identifying Urdu named entities are mentioned in their article [19]. In their research, they applied two machine learning algorithms to address common challenges in Punjabi named entity recognition: Hidden Markov Model and Entropy Markov Model. There is a lack of spelling uniformity. A substantial gazetteer is not accessible since Punjabi is not as commonly used on the internet as other languages. In Indian languages, there are several common words that are also used as Named Entities. There is no capitalization in this language, unlike English. There are few materials and tools available for the Punjabi language. The coaching corpus' Hidden Markov Model derived from various news items and Punjabi newspapers [20].

They observed that the 22 Indian languages' named entity recognition accuracy isn't comparable to that of foreign languages researched in depth in NER. There is a lot of material on Punjabi Language available, however it isn't organised in a way that is useful to locals. There are no web sources for various gazetteer listings throughout this language. They stated that a machine learning approach using the 12 Named Entity tagset is best suited for NER in Punjabi. They found that a context window with word sizes of 3, 5, and 7 yields the same F-Score. The Conditional Random Field technique [21] was used to conduct the experiment.

They claimed that, although being inflectional, free-order, and morphologically rich, Indian languages are resource-poor. It's hard to recognise Indian languages since they're so muddled. One of the biggest problems they ran into was that a NER system intended for one domain didn't work well in another. In Indian languages, a 90 percent F-Measure value was achieved using various NER techniques. During their examination, they noticed certain issues with the Urdu language. For their NER investigation, they required to use the Hidden Markov Model. They achieved 100% accuracy in seven lines of BBC Urdu news, as well as 100% performance in tourist corpuses. They observed that various NER tools are available, but that they are all language-specific. There is no such instrument that is shame language agnostic [22].

From the literature is it clear that there is a need to design a system in such a way that it can dynamically recognise the entities in the Hindi language. Also, there is a need to develop system in such a way that irrespective of the content of the sentence the type of entity should get detected. The proposed system is acting exactly in such a way that it fills this literature gap. The proposed approach has been tested on a variety of databases and has yielded positive results. The suggested system design is compared against a variety of algorithms, and the findings are presented in the results and discussion chapter. Ladkat et al. [23] propose systems time complexity. To achieve greater accuracy when utilising deep neural networks for classification, a mathematical model is crucial [24-26].

2. METHODOLOGY

Recognizes known items in text. The Normalized Named Entity Tag Annotation contains the normalised value of numerical entities that require normalisation, such as dates. As it is always better to feed numbers in the range of 0 to 1. After conversion of sentences or words into the vectors it may cross the range. So, to keep the values of vectors in the range of 0 to 1 vector are passed through the normalization process. The types of name entities along with their examples is tabulated in Table 1.

Table 1. Name entities along with their example

Entity	Example
Named Entity Person (NEP)	रीता, अजय, देवेंद्रसिंघ
Named Entity Designation (NED)	प्रधानमंत्री, राष्ट्रपती
Named Entity Currency (NEC)	रुपया, दीनार, डॉलर, यूरो
Named Entity Organization (NEO)	केंद्रसरकार
Named Entity Abbreviation (NEA)	सीटीओ
Named Entity Title Person (NETP)	महाशय, श्री, श्रीमती
Named Entity Location (NEL)	पुणे, महाराष्ट्र, भारत
Named Entity Time (NETI)	जनवरी, सुबह 9 बजे
Named Entity Number (NEN)	१९४७, १८५७
Named Entity Measure (NEM)	५ लीटर

The initial stage is to gather and categorise data on Hindi words, including their synonyms, antonyms, and meanings. Raw text is broken down into little chunks/words/sentences called tokens during pre-processing. These tokens are used to understand the context of the sentence. Without hampering meaning of the sentence, tokenization keeps the important words or phrases. Result of tokenization is grammatically wrong but it leads to remove the chances of the mis-classification at the time of classification. Also, it reduces the time complexity and space complexity. Due to tokenization, rather working on whole sentence, classifier can work on only important words or phrases in the sentence. We eliminated the stop words from the text since we didn't want them to eat up space in our database or take up precious processing time. Pre-processed text is used to extract important characteristics. On the basis of extracted information after pre-processing, a neural network (architecture as shown in Figure 1) is used to forecast related words, synonyms, and use of a particular term. The sensitivity, selectivity, specificity, and accuracy of the system are evaluated using the performance matrix which is well illustrated in third chapter.

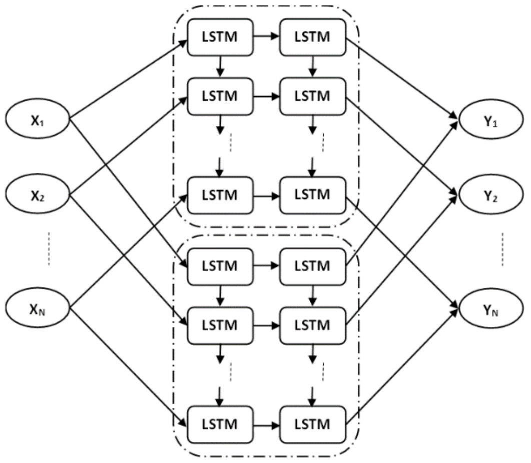

Figure 1. Proposed system architecture

Neural network is designed by using following steps:

(1) The input at time step t is $x_t \in L$. The d-dimensional feature vector is x_t.

(2) The output of the network at time interval t is $y_t \in L$.

(3) The hidden state at time t is stored in the $h_t \in L^m$ vector. The current context is another name for it. The number of hidden units is represented by the letter m. Initialize the h0 vector to zero. In the recurrent layer, $w_x \in L^m$ are weights linked with inputs.

(4) In the recurrent layer, $w_h \in L^{mxm}$ are weights linked with hidden units.

(5) Weights related with concealed to output units are $w_y \in L^m$.

(6) The bias linked with the recurrent layer is $b_h \in L^m$.

(7) The bias associated with the feedforward layer is $b_y \in L$.

At each time step, we can unfold the network for k time steps to get the output at time step k+1. In appearance, the unfolded network resembles a feedforward neural network. The unfolding of the network is carried out to get the maximum number of features from the extracted data. To apply the hysteresis to the network rectangle is chosen to get the result within limit. As illustrated by the rectangle, an operation is being carried out in the unfurled network. For instance, consider the activation function f:

$$\begin{aligned} h_{t+1} &= f(x_t, h_t, w_x, w_h, b_h) \\ &= f(w_x x_t + w_h h_t + b_h) \end{aligned} \tag{1}$$

The output y at time t is computed as:

$$yt = f(h_t, w_y) = f(w_y \cdot h_t + b_y) \tag{2}$$

where, · is the dot product.

As a consequence, after k time steps, an LSTM's feedforward pass computes the hidden unit and output values. The network's weights are distributed in a time-based manner. Each recurrent layer has two weight sets: one for the input and one for the hidden unit. In that it computes the final output for the kth time step, the final feedforward layer is similar to the ordinary layer of a classic feedforward network. To reduce the chances of the mis-classification, for each class new LSTM is working in parallel. By using this approach, in worst case scenario in result either we get entity classified or not classified. But that entity will not get mis-classified.

Eq. (3) is the loss function used for the classification of the name entity.

$$L(\Theta) = \frac{1}{n}\sum_{i=1}^{n} \| | F(Y_i; \Theta), X_i\|^2 \tag{3}$$

In the above formula, Y_i is the complete input data, X_i is the token of the input data, and n is the number of training samples.

3. RESULTS AND DISCUSSION

The results are calculated on the database which is having the sentences in Hindi language about weather enquiry. The length of the dataset is about 2600 sentences. The dataset is evaluated using the suggested system architecture as well as three current algorithms: spacy, coreNLP, and NLTK. The result of the spacy, coreNLP and NLTK is evaluated from the code given on their official websites.

By using confusion matrix performance parameters are calculated. Figure 2 shows the confusion matrix followed by the performance parameters equations.

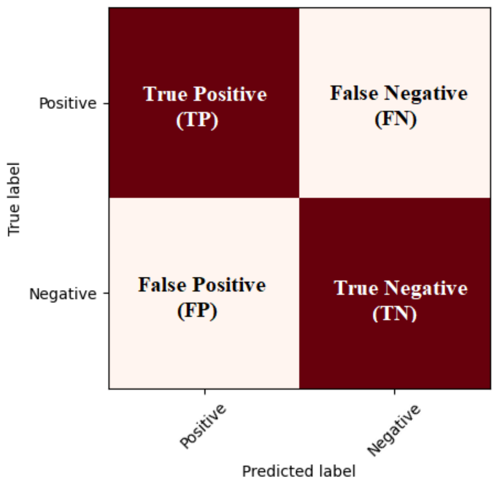

Figure 2. Confusion matrix

$$Accuracy = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{4}$$

$$Precision = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{5}$$

$$Recall = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{6}$$

$$F1\ score = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}} \tag{7}$$

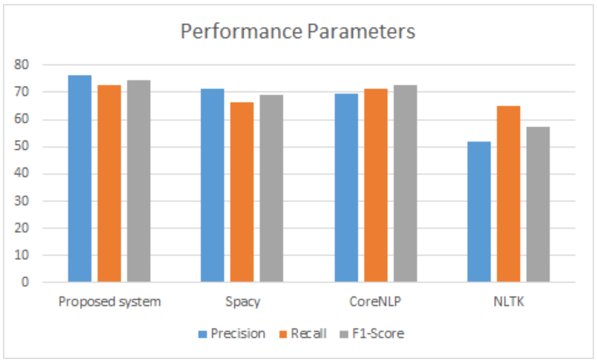

Figure 3. Performance parameters of the proposed system and existing systems are compared

The precision, recall and F1-score is calculated graphically illustrated in Figure 3. Performance parameters analysis illustrates that the proposed system is better in every aspect of the performance parameter.

Table 2. Precision, Recall and F1 score of the proposed system with existing systems taken on the same dataset

	Precision	Recall	F1-score
Proposed system	76.13	72.49	74.26
Spacy	71.36	66.18	69.21
CoreNLP	69.38	71.3	72.82
NLTK	51.94	65.12	57.15

Table 2 gives broad idea about how proposed system performs. The minimum precision value is 51.94% in case of NLTK whereas it is maximum in case of proposed system architecture. Similarly, for Recall and F1-score the proposed system is superior. Whereas Table 3 gives performance parameters of the classification for other neural networks.

Table 3. LSTM's performance parameters are compared to those of an existing classifier

	F1 score	Accuracy	Precision	Recall
Proposed LSTM	74.26%	98.35%	76.13%	72.49%
BiLSTM CRF	62.80%	95.70%	55.70%	72.00%
Residual BiLSTM	66.40%	96.60%	72.00%	61.70%
Residual GRU	69.00%	96.70%	77.50%	62.20%
Residual LSTM	66.40%	96.40%	74.30%	60.10%
Residual RNN	67.30%	96.60%	81.50%	57.30%

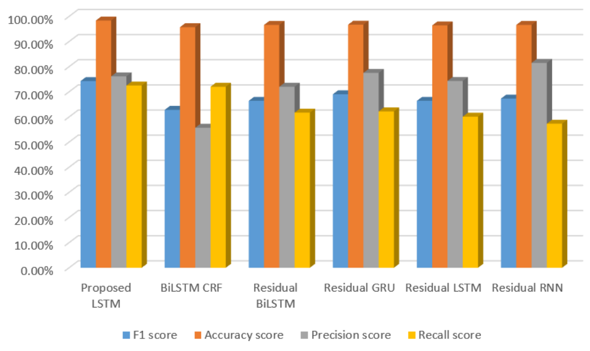

Figure 4. Classifier's performance characteristics are compared to those of existing classifiers

Figure 4 clearly shows that the proposed LSTM is far better than that of the existing classifiers. On each CPU, the time complexity is tested. Table 4 shows the average time it takes to receive a result on various hardware platforms.

Table 4. The time required to obtain the desired outcome utilising the suggested system design on various hardware platforms

Platform	Time required to get result (in seconds)
8GB RAM, Intel i3	0.204
8GB RAM, Intel i5	0.149
8GB RAM, Intel i7	0.143
Nvidia K80 GPU	0.0017

The time complexity is roughly equal while utilising a CPU, such as an i5, or an i7, however when the system is assessed on a GPU, the time required to achieve the results is substantially different.

4. CONCLUSIONS

The precision of the suggested system architecture is 76.13 percent, which is significantly higher than that of previous system architectures. Also, the value of recall and F1-score of proposed system architecture is 71.49 and 74.26 respectively. So, by comparing proposed system architecture with existing Spacy, CoreNLP and NLTK it is easy to conclude that proposed system architecture is reliable in all the sense. When the proposed classifier is compared with the existing classifiers as tabulated in 3.2 it clearly indicates that the proposed classifier is much superior than the existing classifiers. In addition, the suggested system is tested on a variety of hardware processors that produce results in a short amount of time.

REFERENCES

[1] Patil, N.V., Patil, A.S., Pawar, B.V. (2017). HMM based named entity recognition for inflectional language. In 2017 International Conference on Computer, Communications and Electronics (Comptelix), Jaipur, India, pp. 565-572. https://doi.org/10.1109/COMPTELIX.2017.8004034

[2] Chopra, D., Joshi, N., Mathur, I. (2016). Named entity recognition in Hindi using hidden Markov model. In 2016 Second International Conference on Computational Intelligence & Communication Technology (CICT), pp. 581-586. https://doi.org/10.1109/CICT.2016.121

[3] Sinha, R.M.K. (2011). Learning recognition of ambiguous proper names in Hindi. In 2011 10th International Conference on Machine Learning and Applications and Workshops, pp. 178-182. https://doi.org/10.1109/ICMLA.2011.87

[4] Sharma, R., Goyal, V. (2011). Name entity recognition systems for Hindi using CRF approach. In International Conference on Information Systems for Indian Languages, pp. 31-35. https://doi.org/10.1007/978-3-642-19403-0_5

[5] Ekbal, A., Bandyopadhyay, S. (2009). Bengali named entity recognition using classifier combination. In 2009 Seventh International Conference on Advances in Pattern Recognition, Kolkata, India, pp. 259-262. https://doi.org/10.1109/ICAPR.2009.86

[6] Singh, S.P., Kumar, A., Darbari, H. (2017). Deep neural based name entity recognizer and classifier for English language. In 2017 International Conference on Circuits, Controls, and Communications (CCUBE), Bangalore, India, pp. 242-246. https://doi.org/10.1109/CCUBE.2017.8394152

[7] Vora, K., Vasant, A., Adhvaryu, R. (2016). Named entity recognition and classification for Gujarati language. In 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Jaipur, India, pp. 2269-2272. https://doi.org/10.1109/ICACCI.2016.7732390

[8] Mikolov, T., Chen, K., Corrado, G., Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781. http://arxiv.org/abs/1301.3781

[9] Bojanowski, P., Grave, E., Joulin, A., Mikolov, T. (2017). Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5: 135-146. http://arxiv.org/abs/1607.04606

[10] Grave, E., Bojanowski, P., Gupta, P., Joulin, A., Mikolov, T. (2018). Learning word vectors for 157 languages. arXiv preprint arXiv:1802.06893. http://arxiv.org/abs/1802.06893

[11] Mathur, S., Saxena, V.P. (2014). Hybrid appraoch to English-Hindi name entity transliteration. In 2014 IEEE Students' Conference on Electrical, Electronics and Computer Science, Bhopal, India, pp. 1-5. https://doi.org/10.1109/SCEECS.2014.6804467

[12] Amarappa, S., Sathyanarayana, S.V. (2015). Kannada named entity recognition and classification (nerc) based on multinomial naïve bayes (mnb) classifier. arXiv preprint arXiv:1509.04385. https://arxiv.org/abs/1509.04385

[13] Boujelben, I., Jamoussi, S., Hamadou, A.B. (2014). A hybrid method for extracting relations between Arabic named entities. Journal of King Saud University-Computer and Information Sciences, 26(4): 425-440. https://doi.org/10.1016/j.jksuci.2014.06.004

[14] Kaur, Y., Kaur, E.R. (2015). Named Entity Recognition (NER) system for Hindi language using combination of rule based approach and list look up approach. Int. J. Sci. Res. Manag.(IJSRM), 3(3): 2300-2306.

[15] Srivastava, S., Sanglikar, M., Kothari, D.C. (2011). Named entity recognition system for Hindi language: a hybrid approach. International Journal of Computational Linguistics (IJCL), 2(1): 10-23.

[16] Malik, A.B., Bansal, K. (2015). Named entity recognition for kashmiri language using noun identification and NER identification algorithm. International Journal of Computer Sciences and Engineering, 3(9): 193-197.

[17] Nongmeikapam, K., Shangkhunem, T., Chanu, N.M., Singh, L.N., Salam, B., Bandyopadhyay, S. (2011). Crf based name entity recognition (ner) in manipuri: A highly agglutinative Indian language. In 2011 2nd National Conference on Emerging Trends and Applications in Computer Science, Shillong, India, pp. 1-6. https://doi.org/10.1109/NCETACS.2011.5751390

[18] Hussain, S., Kuli, J.J., Hazarika, G.C. (2016). The first step towards named entity recognition in missing language. In 2016 International Conference on Electrical,

Electronics, and Optimization Techniques (ICEEOT), Chennai, India, pp. 3013-3016. https://doi.org/10.1109/ICEEOT.2016.7755253

[19] Naz, S., Umar, A.I., Shirazi, S.H., Khan, S.A., Ahmed, I., Khan, A.A. (2014). Challenges of Urdu named entity recognition: A scarce resourced language. Research Journal of Applied Sciences, Engineering and Technology, 8(10): 1272-1278.

[20] Singh, J., Lehal, J.S. (2015). Named entity recognition for Punjabi language using Hmm and Memm. IRF International Conference, 8th March 2015, Pune, India.

[21] Kaur, A., Josan, G.S. (2015). Evaluation of named entity features for Punjabi language. Procedia Computer Science, 46: 159-166. https://doi.org/10.1016/j.procs.2015.02.007

[22] Malik, M.K. (2017). Urdu named entity recognition and classification system using artificial neural network. ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP), 17(1): 1-13. https://doi.org/10.1145/3129290

[23] Ladkat, A.S., Date, A.A., Inamdar, S.S. (2016). Development and comparison of serial and parallel image processing algorithms. In 2016 International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, pp. 1-4. https://doi.org/10.1109/INVENTIVE.2016.7824894

[24] Ladkat, A.S., Bangare, S.L., Jagota, V., Sanober, S., Beram, S.M., Rane, K., Singh, B.K. (2022). Deep Neural Network-based novel mathematical model for 3D brain tumor segmentation. Computational Intelligence and Neuroscience, 2022: 4271711. https://doi.org/10.1155/2022/4271711

[25] Shobana, M., Balasraswathi, V.R., Radhika, R., Oleiwi, A.K., Chaudhury, S., Ladkat, A.S., Naved, M., Rahmani, A.W. (2022). Classification and detection of mesothelioma cancer using feature selection-enabled machine learning technique. BioMed Research International, 2022: 9900668. https://doi.org/10.1155/2022/9900668

[26] Ladkat, A.S., Patankar, S.S., Kulkarni, J.V. (2016). Modified matched filter kernel for classification of hard exudate. International Conference on Inventive Computation Technologies (ICICT), pp. 1-6. https://doi.org/10.1109/INVENTIVE.2016.7830123