Est. 1841 | YORK ST JOHN UNIVERSITY

Siddalingappa, Rashmi and Sekar, K. (2022) Bi-directional long short term memory using recurrent neural network for biological entity recognition. IAES International Journal of Artificial Intelligence, 11 (1). pp. 89-101.

Downloaded from: https://ray.yorksj.ac.uk/id/eprint/12879/

The version presented here may differ from the published version or version of record. If you intend to cite from the work you are advised to consult the publisher's version: http://doi.org/10.11591/ijai.v11.i1.pp89-101

Research at York St John (RaY) is an institutional repository. It supports the principles of open access by making the research outputs of the University available in digital form. Copyright of the items stored in RaY reside with the authors and/or other copyright owners. Users may access full text items free of charge, and may download a copy for private study or non-commercial research. For further reuse terms, see licence terms governing individual outputs. Institutional Repositories Policy Statement

RaY

Research at the University of York St John

For more information please contact RaY at ray@yorksj.ac.uk

IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 11, No. 1, March 2022, pp. 89~101
ISSN: 2252-8938, DOI: 10.11591/ijai.v11.i1.pp89-101

Bi-directional long short term memory using recurrent neural network for biological entity recognition

Rashmi Siddalingappa, Kanagaraj Sekar
Department of Computational and Data Sciences, Indian Institute of Science, Bangalore, India

Article Info

Article history:

Received Jun 20, 2021
Revised Dec 15, 2021
Accepted Dec 27, 2021

Keywords:

1-hot vector representation
Bi-directional recurrent neural network
Electronic medical records
GloVe
Long-short-term-memory
Named-entity recognition
Skip-gram model

ABSTRACT

Biomedical named entity recognition (NER) aims at identifying medical entities from unstructured data. A quintessential task in the supervision of biological databases is handling biomedical terms such as cancer type, DeoxyriboNucleic and RiboNucleic Acid, gene and protein name, and others. However, due to the massive size of online medical repositories, data processing becomes a challenge for a gazetteer without proper annotation. The traditional NER systems depend on feature engineering that is tedious and time-consuming. The research study presents a new model for Bio-NER using recurrent neural network. Unlike existing approaches, the proposed method uses bidirectional traversing with GloVe vector modelling performed at character and word levels. Bio-NER is performed in three stages; firstly, the relevant medical entities in electronic medical records from PubMed were extracted using the skip-gram model. Secondly, a vector representation for each word is created through the 1-hot method. Thirdly, the weights of the recurrent neural network (RNN) layers are adjusted using backward propagation. Finally, the long-short-term memory cells store the previously encountered medical entity to tackle context-dependency. The accuracy and F-score are calculated for each medical entity type. The MacroR, MacroP, and MacroF are equal to 0.86, 0.88, and 0.87. The overall accuracy achieved was 94%.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Rashmi Siddalingappa
Department of Computational and Data Sciences, Indian Institute of Science
Bangalore-560012, Karnataka, India
Email: drrashmis64@gmail.com

1. INTRODUCTION

The electronic medical records (EMRs) are a vast repository of data maintained by healthcare professionals over a long period containing vital information about a patient in structured and unstructured formats. They include patients' medical conditions, treatments, progress notes, physical condition, prognostic and diagnostic procedures, past medications, immunization, lab reports, discharge indications, and persistence of any other medical problems [1]. Artificial intelligence (AI), particularly natural language processing (NLP), helps discover various associations among these parameters, providing clinicians with opportunities to improve treatment outcomes and administer systematic medical delivery [2]. Bio-named entity recognition (NER) is a process of identifying, classifying, and tagging medical entities for a given context [3]. The main challenges of Bio-NER are: i) tagging entities for the same class (protein/ deoxyribonucleic acid (DNA) or cell type/cell line) does not confer to a standard naming convention; ii) there are a plethora of abbreviations used for different compounds for both long and short words; iii) use of special symbols such as hyphen, colon, and greek letters could create different annotations leading to defiance for a NER task; and iv) the technical terms in online repositories are increasing, the tenacity of disambiguating

medical terms becomes cumbersome for a gazetteer. The authors in [4] describe different methods to build an efficient Bio-NER system. The work concentrated on marking tags for bio-entities (DNA and protein terms) using conditional random fields (CRFs) and recurrent neural network (RNN). However, the baseline accuracy obtained on BioNLP 2004 corpus was 70.09% since the RNN does not retain the previous information. The traditional approaches to handle Bio-NER are: i) dictionary/corpus-based: here, a corpus contains entities and their associated tags. The target word is searched, and the corresponding label is retrieved from the corpus. Though this approach is simple, the efficiency is dependent on the type and size of the corpus. If a target word is not present, the system will not tag the entity. For instance, if the dictionary contains a protein name as "NF-Kaapa B" and the input context has "NF KappaB," then the system will not recognize this variation. Further, the approach also suffers from false recognition and low recall values, especially for small words and spelling variations [5]; ii) rule-based: hand-generated rules are created by a gazetteer. Though the accuracy is increased, manually creating rules is cumbersome and depends on expert knowledge and domain [6]; and iii) machine learning-based: the statistical models in machine learning (ML) recognize entities through the feature representation. The primary steps are training and testing; training, where annotations are marked based on an annotated document and later store the model; next, the annotations for the raw document are scored based on the model saved. This approach is the best compared to other methods. Here, the model can recognize new tags even when they are not present in the corpus [7]. Recent deep learning methods with traditional ML models are amalgamated to achieve good results [8]. Deep learning mimics human brain functions to help process data by creating functional patterns to support decision-making. More interestingly, deep learning techniques outperform other ML-based approaches such as support vector machines (SVM) [9], hidden markov model (HMM) [10], maximum entropy likelihood model conditional random fields [11].

Deep learning has recently led its way in Bio-NER, too, and researchers have achieved excellent results [12], [13]. Bio-NER to identify genes and proteins using RNN is proposed by Li *et al.* [14]. The information from the last node is considered to make new entity predictions. An extra class-based input layer was created using a brown clustering algorithm. The features are extracted using word-embedding, and the work is demonstrated on BioCreative II genetic mutation (GM) and obtained an accuracy of 81.06% on f-score. Though the extended-RNN framework attained a better accuracy, perhaps an additional feature layer could incur extra processing time. Ali *et al.* [15] have used sequence labelling and memory component of RNN to mark Arabic text labels. A pre-trained embedding using the LSTM network is used for training the corpus. The model is evaluated on Arabic-NER Corp achieving an f-score of 88.01%. Cho and Lee [16] focused on marking bio-medical entities using bi-direction long short term memory (BiLSTM). The critical associations between the adjacent labels were drawn using condition random fields (CRFs). The computational time of the contextual LSTM (CLSTM) network was computed, and it was compared with other models (bidirectional encoder representations from transformers (BERT) and bidirectional recurrent neural network (BiRNN)). It was found that CLSTM had a faster training time as compared to different character-level embeddings. However, it took 20% longer training time than the BiRNN models. The results are evaluated on three corpora; the National Center for BioTechnology, Gene Mutation, and Chemical Disease-Related Corpus, and the accuracy of 85% was recorded. Lyu *et al.* [17] proposed bioNER using CRFs on word-character representation. The CRF layer encodes the context information of a given sentence in two directions, forward and backward. The model was tested against two corpora, bio creative II GM and joint workshop on NLP (2004), and achieved an accuracy of 86.55% and 73.79%, respectively. The approach performed better than other models; however, contextual and external knowledge information was not considered. Bio-NER for Chinese texts was designed by Li *et al.* [18]. The orthographic and lexicon-semantic features were derived from the given context using the word (W) and character (C) embedding. The part-of-speech (POS) tags (P) of previous information are used to improve the overall performance and obtained an accuracy of 90%. However, the domain-specific features related to diagnosis or medications were not included in this WCP-RNN based research study. Chowdhury *et al.* have proposed a multitask Bi-RNN model for Chinese EMRs [19]. The work was divided into the shared and task-specific layers. Each word was represented using word and character embedding. The context information was extracted using Bi-RNN. In the next layer, POS tagging was marked to separate the POS tags from the given context, and in the next step, NER was performed for identifying the entities. This approach requires high training time as it contains two extra task-specific layers for POS tagging. NER implementation for national center for biotechnology information (NCBI)-disease and JNLPBA corpus (Joint Workshop on Natural Language Processing in Biomedicine and its Applications) have been discussed by authors in [20] using LSTM and convolutional neural network (CNN) models. The proposed model lacked the knowledge transferring approach. The accuracy was equal to 74.4% and 86.0%, respectively, for these binary datasets. Further, the authors in [21] have explained different efficient methods ruling in the industry for Bio-NER and relation detection (RD) to learn the interaction between protein, drug, diseases, and genes. In the present research

paper, the authors have tried to overcome the shortcomings of the existing literature in the following ways: i) by far, the global vector representation (GloVe) representation model is not used to address BioNER. The features are extracted at both character and word levels using GloVe embeddings. Thus, the accuracy was increased, unlike other state-of-art systems; ii) no additional layer was required for training features in the hidden layer; iii) the POS labels for all the entities are not marked, except for bio-medical phrases; thereby, the execution time gradually decreased; and iv) the LSTM layers treat each entity class as a different model; thus, the disease name is not confused with the drug or gene name. Consequently, each layer becomes a master in the training time and a candidate during testing time.

The objectives of the present work are: i) recognize the medical entities and the association between other vital terms like genes/proteins causing cancer or mutations indicating the up or down-regulation in cancer; ii) disease identification; iii) adopting deep learning strategies for perceiving the relevant features to understand the critical role played by cancer entities; iv) to automatically extract meaningful bio-entities found in unstructured data with biological relevance; v) to learn and annotate various biological terms from the biomedical repositories such as PubMed; and vi) study the efficiency of the model through evaluation metrics. The paper is structured as indicated here: section 2 talks about research methods such as different stages of the Bi-RNN and LSTM frameworks. Then, the experimental and implementation results are described in section 3. Finally, the study concludes with section 4.

2. RESEARCH METHODOLOGY

The present "problem-solution" research paper explains the problems faced by the existing bio-ner systems and finds suitable solutions using the advanced deep neural network model. In the pursuit of this, the following details are discussed in this section: i) methods adopted: RNN for learning features, GloVe embeddings at character and word level to extract essential features, and LSTM to retain the critical features and forget those features that are no more essential for learning tasks; ii) the bi-directional RNN and LSTM algorithms and pseudo codes; and iii) evaluation metrics: F-score components

2.1. Recurrent neural network and components

The RNN model uses sequential information to process text. In a traditional neural network, an input at each layer is self-reliant and does not depend on other layers' inputs. However, this is unlikely in a large number of prediction tasks. For instance, knowledge about the previous word is essential in predicting the next word of the sequence. Thus, in RNNs, the previous output is treated as inputs for the present state [22]. Besides, RNN also has a memory element to capture the computations performed during the pre-trained embedding task. A typical RNN model is shown in Figure 1. The layers of an RNN model is dependent on the number of input words in a given sentence 'S' such that each word is represented by 'W'The terms used are described below.

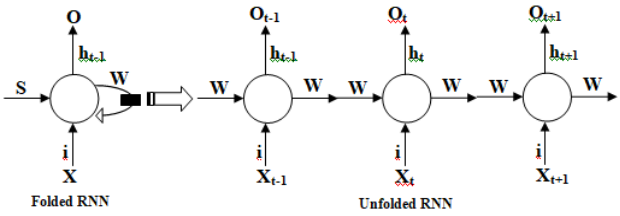

Figure 1. Basic Structure of an RNN model with two variations; folded and unfolded model

- x<t>: the 1-hot vector representation of a given word at time 't'
- h<t>: the hidden state at 't' that contains the output of the previous state at time 't-1' and the input of the current state at the time 't'. It is usually calculated as h<t>=tanh(cx<t>+ ph<t-1>), where 'c' indicates the current input at x<t> time and 'p' indicates the previous output at hidden state h<t-1>. Tanh is a non-linear function for activating input and hidden states.
- o<t>: output obtained at each layer in the time step <t>. If the bio-entity tagging is expected, then output at <t> step would be series of vector probabilities in the chosen corpus 'C' at the training stage. Softmax, expressed as o<t>=softmax[Ch<t>], is the activation function used for normalizing output value. Consider the following example for bio-ner task: "chronic myeloid leukemia (CML) is characterized by the presence of a breakpoint cluster-abelson kinase (BCR-ABL) fusion gene. Being an inhibitor of BCR-ABL, Imatinib rapidly and dramatically modified CML treatment. Nilotinib and

Dasatinib also have superior efficacy to Imatinib to diagnose CML. Gastrointestinal stromal tumors (GISTs) are defined by C-KIT expression (CD117) in tumor cells. Imatinib was found effective in patients carrying KIT mutations. Thus, the tyrosine kinase (TK) inhibitor, Imatinib has revolutionized the therapy of malignancies that are addicted to one of its target kinases, C-ABL, C-KIT and platelet-derived growth factor receptor (PDGFR)" [23]. * *ABL: Aberlson murine leukemia, C-KIT: cluster of differentiation 117- tyrosine-protein kinase*

In the above example, the dependency between each term is evident. Initially, CML is tagged as a cancer type. The same word CML appears after a long gap, enforcing a long-term dependency. RNN model operates smoothly only for trigram context dependency. RNN loops the input repeatedly, modifying the input weights/gradients. These gradients refer to the values used to update the network weights at each input layer. A substantial change in these input values leads to error gradients, which results in a poor network. An unstable network constitutes two main challenges; firstly, the gradient values can grow exponentially, at times greater than 1, because they multiply at each layer with the hidden values, leading to overflow or Nan values. Secondly, if the values are smaller than 1, they may quickly vanish due to recurrently operating on a smaller value. A low gradient value does not help much in the learning process. Thus, the LSTM technique is used to handle this problem. LSTM retains only the relevant information and forgets irrelevant data. For instance, LSTM considers words such as "being, was, an, have, are, dramatically, rapidly, and so on" as irrelevant. In addition to this, the forward pass of a conventional RNN model considers the data processing in the single, sequential direction and fails to look ahead for the context-dependency. On the other hand, Bi-RNN "looks ahead" for any dependencies between the key terms. For example, i) Teddy Douglas was the president during the drought and ii) she was excited to know that Teddy bear dolls were on sale at the sixth main avenue. In these examples, the word 'Teddy' indicates the person's name and a toy. In a unidirectional approach, the system would tag 'Teddy' as a toy. However, in a BiRNN system, the model looks forward and learns the context information and accordingly marks the tag for 'Teddy' as the person's name in example 1 and a toy in example 2. A loss function is used in the backward pass to quantify the wrongness of the model. A loss is measured as a square difference between the answer obtained and the expected correct output. Suppose a BiRNN model calculated output value as 0.6, and the result expected is 1; the loss is calculated as $[0.6\text{-}1]^2$=0.16. This value suggests that the weights have to be tweaked to minimize the value of the loss function. Thus, a backward pass is also referred to as gradient descent with the objective to descent (minimize) the gradients (weights) [24]. These steps are performed for an ideal number of times called the epoch, from which the network gets better every time.

2.2. Extracting features at word and character level using GloVe model

Tokenization is performed in step-1. Each token then serves as input for word embedding, Figure 2. The figure indicates the following terms: $<F_i>$: forward-pass features, $<B_i>$: backward-pass features and $<W_i>$: word-entity concatenation, d X T: input gradient at time-step. Here, a word is represented in a vector form, i.e., a unique number, based on the corpus. For instance, the word imatinib appears in 5536^{th} place in the corpus. Therefore, the 1-hot representation is Imatinib=$O_{5536.}$ Word-vectors cluster similar words and dissimilar words are repelled. Consider the subset of the example; imatinib treats CML. Nilotinib also treats leukemia. Oxaliplatin is used against Colorectal Cancer. The word and the dimensional vector are indicated in Table 1. By looking at the high numbers in the Table 1, we can say that 'treat' is closely associated with Imatinib, nilotinib, and leukemia, indicating that these terms are closely used for treating some 'X' (X is unknown, it could be a disease). Values of CML, cancer, colorectal, and leukemia are higher in the second row, extrapolating the disease names. The dimensional vector represents how closely the terms (row x column) are associated.

The GloVe model is used as a word-embedding task [25]. Unlike the traditional word2vec method, the GloVe model looks for global co-occurrence between words and later mark vectors. The GloVe model forecasts the input texts to their corresponding vector values. Here, certain dimensionality constraints are defined randomly. The dimensional vectors are; 'treat,' 'cancer,' drug names, and so on. These keywords are placed in a columnar fashion. For each dimensional vector, the value of a keyword is compared using a random function. For instance, Imatinib is used to treat cancer. Therefore, the number 1 indicates a valid substitution. However, CML is a cancer and is not a 'treat' entity. Therefore, a negative value is assigned. The 2d space representation of this analogy is indicated in Figure 3. Thick lines indicate a solid word-pair association, and dotted lines indicate a weak connection. With this, we can conclude that, for CML, imatinib is used, and similarly, for colorectal cancer, oxaliplatinis used.

Internally, $E_{Imatinib}$-E_{CML}~$E_{Oxaliplatin}$–$E_{?}$. Therefore, we find the word (w) so that this approximation holds good. We have to find a word to maximize the similarity of 'w', ii) $E_{Imatinib}$-E_{CML}=1-(-1)*=2, $E_{Oxaliplatin}$-$E_{colorectal}$=0.99–(-0.9)*=1.89~2. *Values taken from Table 1. Therefore, a GloVe draws a probability (P) and ratio (R), as shown in Table 2.

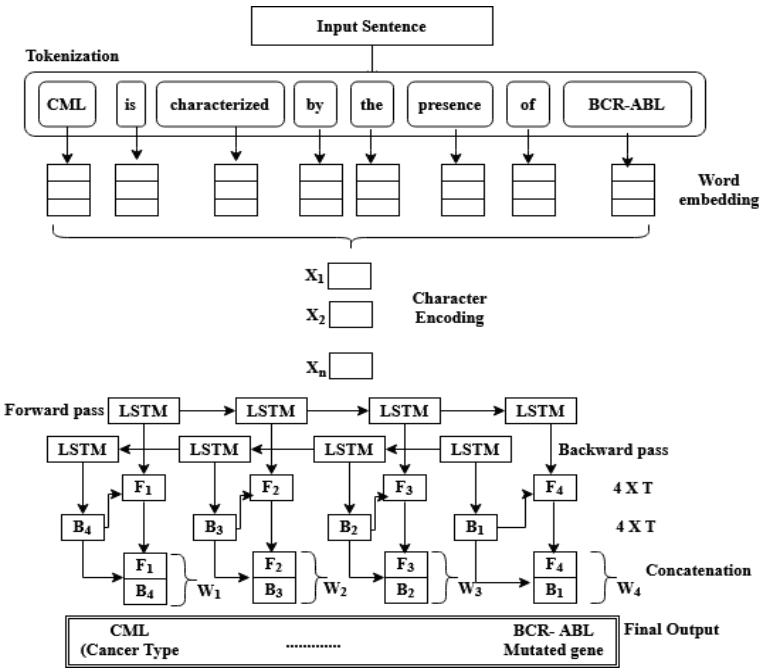

Figure 2. The architectural model for the proposed BiRNN network with LSTM cells for the NER task

Table 1. Dimensional vector representation for a given sample text

	Imatinib	CML	Nilotinib	Leukemia	Cancer	Oxaliplatin	Colorectal
Treat	1	-1	0.97	-0.95	-0.98	0.99	-0.9
Disease	0.09	0.93	0.01	0.99	0.98	0.02	0.97

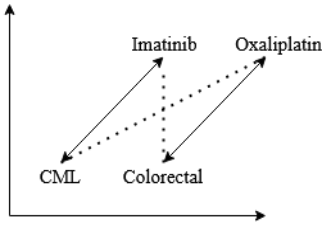

Figure 3. The 2D space representation of word vectors for GloVe word embedding

Here Table 2, P_iz/P_jz, indicating → p_iz=X_iz/X_i. P_iz: the probability of witnessing the words 'i' and 'z' jointly, calculated by dividing the total times 'i' and 'z' are seen together (X_iz) with the actual time 'i' is seen in the corpus (X_i). This allows drawing semantic similarities between the word-pairs in terms of probability and ratio of words, co-occurrences have higher values in the second row, extrapolating the disease names Let us consider two words, imatiniband oxaliplatin, and the word 'z' is: i) associated to Imatinib and not associated to oxaliplatin if P_iz/P_jz is greater than or equal to 1 and ii) associated to oxaliplatin and not associated to Imatinib if P_iz/P_ij is lesser than 1.

The GloVe approach will not recognize the target word if a word is not found in the dictionary. Further, for an out-of-vocabulary (OOV) word, the GloVe model simply assigns a random vector value. The character level embedding represents each word with a vector of numbers at the character level [26]. From a bird's eye view, a sliding window moves the window character by character, making the next character prediction. As the window slides every time, the embedding concentrates on the sequence of multiple characters, capturing the information indicated by these characters. Lastly, all character-level embeddings are joined to form a word. The main drawback of language modelling is that character and word-level embedding assigns the same vector number for every appearance of the word/character in the context. This becomes complicated for heteronyms like 'bank,' and 'orange'. So assigning the same vector will make the model inconsistent and create chaos in the network.

Table 2. Probability ratio between the target and the keyword

P & R	z=CML	z=colorectal
P(z|imitanib)	3.4	0.23
P(z|oxaliplatin)	0.66	1.83

2.3. Long-short-term-memory (LSTM) networks

LSTM, a variation of RNN model, learns a pattern even when the pattern is scattered across the full sentence [27]. The LSTM comprises four stages: i) eliminate stage, ii) input stage, iii) concatenation stage, and iv) output stage. Here, the vector representation of current and previous inputs is passed through activation functions tanh and sigmoid. The input layer performs a series of operations, making the values explode exponentially. With tanh activation, these values are squished between -1 to +1 [28]. The sigmoid activation function converts the values in the range 0 to 1 [29]. Each stage of LSTM is explained below in detail, Figure 4:

- Elimination stage: this takes the input from 'k' and 'O_{t-1}'. For instance, the previous hidden state is 'CML' with the tag 'cancer,' and the present input is BCR-ABL with no label. The vector for BCR-ABL and the associated weight of previously seen 'CML' is passed into a sigmoid function. The output is then multiplied by the previous output O_{t-1}. If the value is closer to zero, then the state information is forgotten, and if it falls closer to 1, the state information is retained. Therefore, the tag 'Cancer' is ignored.
- Input stage: it takes the hidden state information with the weight' h_{t-1}'andthe vector representation of the present state 'x_t'. In our example, the hidden weight of CML and vector representation of BCR-ABL is considered, indeed, multiplied by the tanh activation values of the previous hidden and present input state.

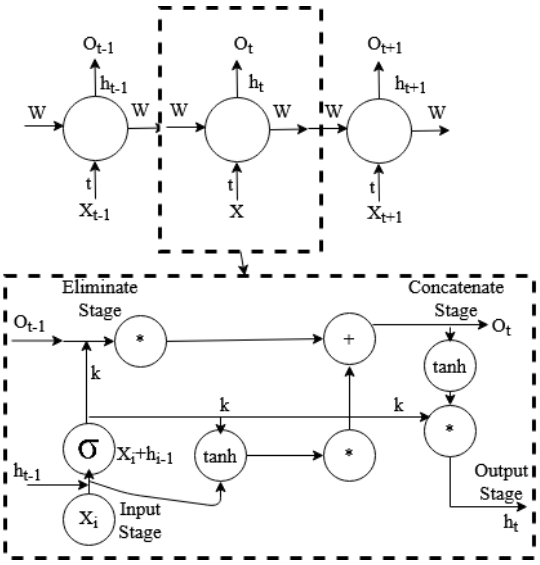

Figure. 4. The workflow of an LSTM network

The three stages of an RNN model are shown in the upper part. The lower part indicates the dissection of a single-stage, unfolding the four evaluation stages such as: i) input, ii) eliminate, iii) concatenation, and iv) output stage. Here, O_{t-1}: previous output, O_t: present output, σ: sigmoid activation function, tanh: tanh activation function, h_{t-1}: Previous hidden state, h_t: Present hidden state, X_t: Present input, k: Sigmoid value of X_t+h_{t-1}, *: point wise multiplication, +: point wise addition

- Concatenation stage: all the values are combined and form new information in an RNN memory unit. To do so, it concatenates the values of elimination and input stages and obtains a new value called O_t
- Output stage: the final stage reveals the tag to be marked for the current state information stored in the memory unit (concatenation stage). The O_t value is passed through the tanh activation function and multiplied with the sigmoid function of the present and previous hidden states. Therefore, BCR-ABL is

marked with the 'mutation' tag. In the backward pass, as the word 'gene' precedes the BCR-ABL, the labels are altered and finally fit the tag 'mutated gene' for BCR-ABL.

2.4. The flowchart and Pseudocode for Bi-directional RNN and LSTM based Bio-NER system

Figure 5 shows the flowchart of the entire process in a typical ML perspective. At first, the input data is fetched from the dataset, and then a series of ML actions take place to tag the associated tags for the features properly. The algorithmic procedure is indicated below in Algorithm 1. The algorithm illustrates how to extract the feature and tag appropriate labels.

Algorithm 1. Algorithm for feature extraction and bio-entity tagging using Bi-RNN-LSTM model

```
Input: 'in_sen' ➔ an input sentence for entity tagging
Output: 'out_sen ➔ the output with the bio-entities tagged (of the seven entity types)
Procedure:
 Remove stopwords and tokenize ['in_sen']
 For each word Wi in S, where, S={W1, W2, … Wn}
 For each character Ci in W, where, W={C1, C2, … Cn}
                  Draw probability ratio P and R (described in Table 2. under section: 2.2)
                  extract important features
                  Train features for LSTM model
 End For
            i) Calculate [h<t>], [x<t>] and [h<t-1>]
```

The input dataset is divided into training and test set. The essential features are extracted from the training data through GloVe embeddings at both character and word-level ('i' indicates the embedding model, as described in section 2.2). Further, the extracted features are passed as input for the Bi-RNN LSTM model ('ii' indicates algorithm 1). The model is trained, and features are learned. The test data is exposed for the learned model, and the prediction results are estimated using evaluation metrics, as shown in section 3.6.

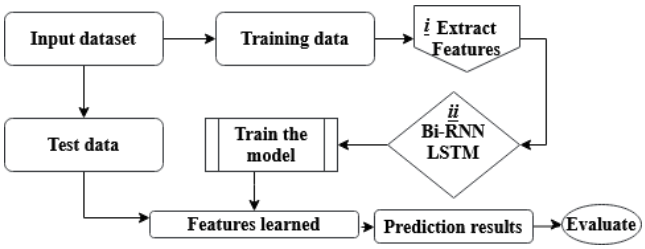

Figure 5. Flowchart of the neural network perspective for Bio-NER

3. RESULTS AND DISCUSSIONS

3.1. Datasets

Generally, a dataset is crucial in an implementation task containing 'enough' data for the training process and arrive at empirical hypotheses. Thus, in a data-driven learning mode, the accuracy is dependent on the type of corpus used. Initially, GENIA3.0 (source: www.nactem.ac.uk/genia/genia-corpus) was chosen for the training purpose; however, the tagging was not accurate as this corpus is suited for generic entities marked with the terms related to human blood lines and transcription factors. Therefore, 10,000 abstracts were selected randomly from MEDLINE using a PubMed search engine to have more specific medical terms. The abstracts were drawn with selective terms such as Cancer type, for instance, CML. There were a total of 1,550,970 words counted using a small python script. The sentences were then run through a python model called ScispaCy v0.2.4 (source: allenai.github.io/scispacy/) using a python code. The resulting tagged entities were tested against BioCreAtIvE II (source: https://biocreative.bioinformatics.udel.edu/tasks/biocreative-ii/) and GeneTag [30]. The entities were then compared, and wherever required, slight modifications were made manually to ensure the tagging is correctly completed.

3.2. Tools used for implementation

The proposed model is executed on the Spyder, v5.1.1 (https://www.spyder-ide.org/). Keras 2.4.0 is used for the neural network package (https://keras.io/), with TensorFlow v2.6.0(https://www.tensorflow.org/) running on top of the Keras application [31]. Python 3.9.0 (https://www.python.org/) is used for the coding. Keras is a neural network module that helps to design, fit, and assess a proposed model. An interesting fact about Keras is that it is more user-friendly as compared to Tensorflow. Keras library only provides low-level

APIs; however, it runs on Tensorflow (supports high-level APIs also); therefore, Keras supports both low and high-level APIs.

3.3. Data splitting method

The k-fold cross-validation (CV) approach is used to evaluate the performance of the model. First, the value of k is set to ten [32]. Next, the entire dataset is split into ten portions (for k=10). One part out of ten is reserved for the validation set. The remaining nine portions (k-1) are held for training, and the predictive performance is recorded at each evaluation stage. The process is repeated, changing the validation and test portions, and finally, the average of these measures is considered. The words are categorized into six entity classes: cancer type, genes, mutations, protein type, DNA structure name, and drugs and performed k-fold CV. The results are illustrated in Figure 6. Each parameter has ten recall, precision, and f-score values. The average of ten folds is calculated and presented in the Table 3.

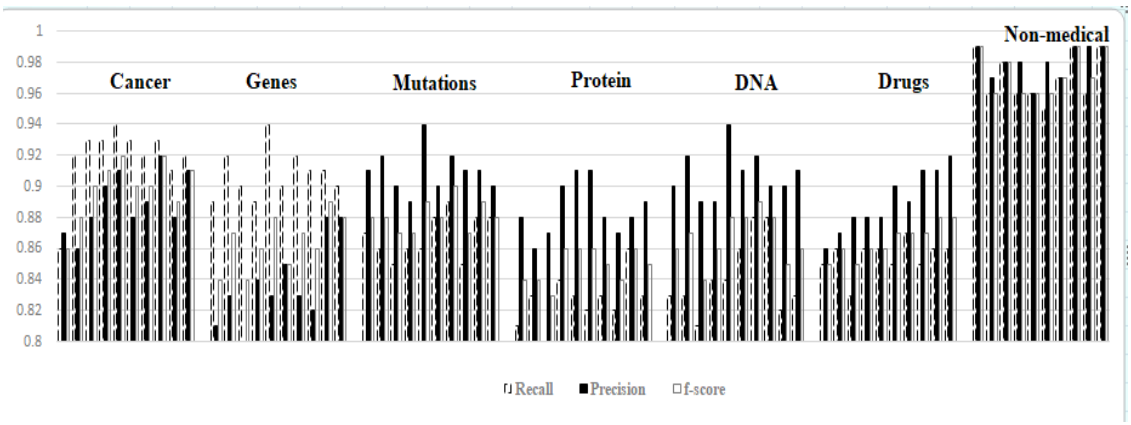

Figure 6. A graphical representation of 10-cross k-fold validation for seven parameters.

3.4. Baseline features

Keras is a popular deep learning framework that works efficiently with python code. The fascinating feature of Keras is that it consists of TensorFlow, CNTK, Theano, MXNet, and PlainML as the backend engines to support the deployment of deep learning mechanisms. The typical deep learning stages are defined in four layers in Keras; i) Dense layer: point-wise multiplication and the bias function is executed, ii) Activation layer: a transfer function to initiate the learning task, iii) Dropout layer: the layer is used to avoid the 'overfit' of the network through regularization techniques. During the training process, some percentage of layers are "dropped out" / left out. Thus, the model treats a layer differently based on the features of the previous layer [33], iv) Lambda layer: covers the inconsistent aspect of the learning and reduces the random fluctuations in the model.

3.5. Parameters setting

The total words in the input are the number of hidden layers in the network. The dropout value was initialized to 0.6 at the input layer and 0.5 at LSTM and softmax layers to address the overfitting problem of the network. Having fewer neurons often results in underfitting because the network will have insufficient data for training [34]. Therefore, 120 hidden neurons are used at each hidden layer. An optimizer must adjust the weights to avoid a high loss function value. Here, the AdaGrad optimizer learns the frequently occurring features, and the learning parameters are updated at every epoch. The learning rate and epsilon are set to 0.01 and 1e-08. The batch size=40, epoch=1000 with loss/error over training set indicated at each epoch execution.

3.6. Evaluation metrics

Figure 7 shows the final output obtained from the proposed model. It shows: i) the epochs and loss. In the beginning, the loss was equal to 67.3 and later reduced to 10.85 at the end of 1000 epochs; ii) input provided by the user; iii) entity-label tagged by the system. The LSTM model has removed all irrelevant words, and the tagging is performed only on keywords; iv) The cancer name and drug list. The performance of the output is evaluated through the following metrics; micro and macro F-measure alongside recall and

precision scores [35]. This is because macro scores show how the system performs across all the data provided in the testing phase and micro score indicates the performance for every input. The other terminologies used are: i) true positive: the training and the test data are matched. For example, CML is a cancer type. It is identified as the Cancer type in the testing phase; ii) false positive: the entity identified in the testing phase is incorrect and does not match the entity type mentioned in the training phase. For example, GIST (tumor or Cancer type) is identified as mutation; iii) false negative: the entity type though present in the training data, is mentioned as not present. For example; SMARCA4 is not tagged as mutation though it is present in the training data; iv) precision indicates the correctly classified entities across all the entity types, for example, correctly classified number of Cancer type, genes, mutations, drug name and so on, over the actual number of individual entities in the training set; v) recall measures, for a particular entity type how many were correctly classified, for example: out of all cancer type instances, how many were correctly identified as a cancer type; vi) f-score calculates the average of both recall and precision; and vii) accuracy is the fraction of correctly classified entities over the total number of entities.

$$P = \frac{TP}{TP + FP} \tag{1}$$

$$R = \frac{TP}{TP + FN} \tag{2}$$

$$F = \frac{2\ X\ R\ X\ P}{R + P} \tag{3}$$

$$MacroF = \frac{1}{N}\sum_{i=1}^{N} F_i \tag{4}$$

$$MacroP = \frac{1}{N}\sum_{i=1}^{N} P_i \tag{5}$$

$$MacroR = \frac{1}{N}\sum_{i=1}^{N} R_i \tag{6}$$

$$MicroF = \frac{2\ X\ MicroP\ X\ MicroR}{MicroP + MicroR} \tag{7}$$

$$MicroP = \frac{\sum_{i=1}^{N} TP_i}{\sum_{i=1}^{N} TP_i + \sum_{i=1}^{N} FP_i} \tag{8}$$

$$MicroR = \frac{\sum_{i=1}^{N} TP_i}{\sum_{i=1}^{N} TP_i + \sum_{i=1}^{N} FN_i} \tag{9}$$

where N indicates the total entities types and P, R, F indicates precision, recall, and F-measure, and lastly, P_i, R_i, F_i indicates precision, recall, and F-score, for instance 'i'. The micro-scores for each of the entity types are presented in Table 3. The average is indicated in the table. The dataset was evaluated using a 10-fold CV as described in Figure 6. To calculate the macro scores, each entity-type score for a particular R, P, and F-score is enumerated by the total number of entity types (N=6). Therefore,

$$macroR = \frac{0.92 + 0.90 + 0.86 + 0.82 + 0.84 + 0.85}{6} = 0.86$$

$$macroP = \frac{0.89 + 0.83 + 0.91 + 0.88 + 0.90 + 0.89}{6} = 0.88$$

$$macroF = \frac{0.96 + 0.86 + 0.88 + 0.84 + 0.86 + 0.86}{6} = 0.87$$

3.7. Complexity

Each sentence (S_i) (considering the longest sentence) is compared against the training corpus of length (t_l) for an entity match. Thus, the local alignment complexity achieved is O ($S_i t_l$). Further, each sentence has multiple entity types, which are checked across all the test sentences of length (t_m). Therefore, for each entity pattern (e_p), the complexity will be O (e_p t_m t_l).

3.8. Comparison of existing research and current study

Table 4 demonstrates a comparison study between previous research works and the present study. Values in bold represent the overall best results. The highest score achieved in each row is indicated in bold. The last row values are the accuracy of the proposed model. Although the work [36] is better than the proposed model, only one entity phenotype for human-was considered. However, in the present research study, six different entities were considered for the tagging process. R: Recall; P: Precision; F: F-Score From a thorough insight of the authors', no research work adopted forward and backward pass along with LSTM variation. Due to this inclusion, this study witnessed elevated scores in the recall, precision, and f-score for the six entity types. Further, the character-based RNN is essential to identify the grammatically correct sequences for any morphological representation. With word-based RNN, the model displays high accuracy at a minimal computational time. Owing to these factors, the proposed model can mark complex and lengthy words such as epoxygenase and antitrypsin-glutathione. Besides, incorrectly marked entity tags would be corrected in the backward pass. There were no gazetteers involved for manually tagging, removing a tiresome workload of marking entity tags. These strategies helped achieve a remarkable improvement in the performance metrics.

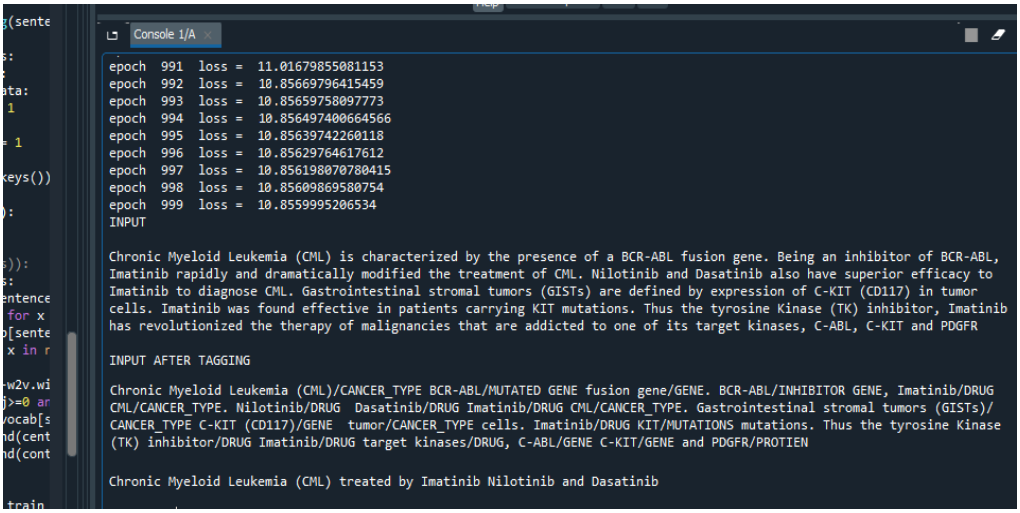

Figure. 7. The final output of the proposed system

Table 3. Micro-scores evaluated using [7]-[9] for each entity type

Entity Type–Entity Instances	Recall	Precision	F-Measure
Cancer Type-287926	0.92	0.89	0.90
Genes–156718	0.90	0.83	0.86
Mutations–379871	0.86	0.91	0.88
Protein Type-190171	0.82	0.88	0.84
DNA Structure Name–18027[illegible]	0.84	0.90	0.86
Drugs 245689	0.85	0.89	0.86
Non-medical Words-110020	0.97	0.98	0.97

Table 4. A comparison study with the existing systems

State-of-art systems	R	P	F
Bio-NER using Deep Neural Networks [36]	76%	66%	71%
NE using Statistical NER [37]	79%	68%	73%
Bio-NER using RNN and LSTM [20]	73%	74%	86%
Bio-NER using rich features of CRFs for human phenotype terms [38]	87%	85%	86%
NER on Arabic text (only on the regular text) [39]	-	-	90.6%
Proposed model BiRNN using both forward and backward pass	86%	89%	88%

3.9. Discussion: challenges and future directions

DL-based Bio-NER significantly overcomes the disadvantages of the traditional NER models, such as feature engineering and pre-trained embeddings. However, like any other knowledge representation system, the RNN-based approach also suffers from challenges. The proposed model performs different pre-processing tasks such as tokenization, word and character embeddings. Due to this, the time complexity depends on these tasks plus an RNN training phase. To avoid this, the loss function could be presented after removing the stop words from the input, such as 'a', 'an', and so on. The observations could be ranked based on the frequency of words apart from stop words. One more possibility of reducing the time is considering the gradients of only the last iteration epoch rather than the entire sequence. The output at each step could be merged without worrying about the previous input sequence at different time steps. Further, the quality of annotated corpus plays a major role in performance. For example, BRCA2 is marked as a gene in GENETAG and mutation in BioCreAtiVE II corpus, creating confusion in entity labeling, and 17% of entities in GENIA are marked with duplicate entity tags. The same was observed even with the IL-2 gene. Out of 72 instances, 16 times, it was annotated as DNA and 56 times as protein. Some of the tags in GENETAG are not marked appropriately to their base category. For instance, T-cell and b-cells are not marked as cell-type, and 10 times t-cell occurred in the MEDLINE abstract and were marked as gene since the "T-cells and gene regulation" term appeared 48 times in the corpus. Thus, in the backward propagation, the t-cell was tagged as gene type and not as a cell type. T-cell receptor (TCR) gene was correctly marked as gene type; however, with such influence, even b-cell are marked as gene type since t-cell and b-cell appeared together in the corpus 108 times. These kinds of incorrect tagging increase the false-negative ratio. Protein molecules names (e.g., Dystrophin) were marked as DNA type. However, this error was reduced for some instances when the context information was increased (e.g., the length of input was increased by 1-fold). Further, names such as 'antigenic' were incorrectly identified as entity thereby increasing false positives. Although the pre-trained embeddings enabled the model to learn the context information precisely, there were 14089 errors (0.9%), which contributed to the overall misclassification.

Future enhancements: there are still many unanswered questions related to Bio-NER. For instance, it is still a question dealing with overlapping entity classes, such as cell lines vs. cell types and protein molecules vs. DNA molecules. Thus, in the future direction of the present study, the authors wish to perform boundary identification to handle overlapping entities and implement semantic parsing rules to identify the impact and relationship between different vital terms present in the input corpus. The human annotation could be reduced by applying ML techniques. Further, a catalog could be created to understand the samples of healthy vs. diseased individuals. Additionally, pipeline architecture could be designed to recognize entities and connect a feedback network to interact with different entity types, resolving the error propagation in further stages.

4. CONCLUSION

The bi-directional RNN LSTM technique is applied for biomedical and embedding at the word and character level in the present study. A total of 1,550,970 words from MEDLINE are used as a dataset. The GloVe model is put forward on these words for word-vector representation at both character and word levels. The LSTM component is created in the next step to interpret the tagging process with both forward and backward pass with the BiRNN model. Thus, with the LSTM, lengthy sequential data can be handled efficiently. From the experiments carried out, it is evident that the RNN model is better than the other traditional ML techniques such as SVM and HMM. The performance evaluation of the proposed system is judged through recall, precision, f-measure, and accuracy scores. The proposed approach outperforms the existing methodologies. Thus, the results of this study make the clinicians' task easy by providing an accurate and robust biomedical annotated corpus.

ACKNOWLEDGMENTS

Author Rashmi S expresses sincere gratitude to Science and Engineering Research Board (DST-SERB), New Delhi, India, for providing a research grant (NPDF, sanction order no PDF/2019/000254). In addition, the authors would like to thank the Department of Computational and Data Sciences, Indian Institute of Science, Bangalore, India, for extending the essential support required to carry out the study.

AVAILABILITY OF DATA AND MATERIALS

The necessary code and datasets are stored in the GitHub repository (https://github.com/RashmiSKarthik/Bio-NER). The researchers may replicate any methodology used in the

present study with due citations. However, kindly write a mail to drrashmis64@gmail.com to obtain access to this repository.

REFERENCES

[1] T. Piliouras *et al.*, "Electronic health record systems: a current and future-oriented view," in *9th Annual Conference on Long Island Systems, Applications and Technology, LISAT 2013*, May 2013, pp. 1–6, doi: 10.1109/LISAT.2013.6578225.

[2] O. Fennelly *et al.*, "Successfully implementing a national electronic health record: a rapid umbrella review," *International Journal of Medical Informatics*, vol. 144, Dec. 2020, doi: 10.1016/j.ijmedinf.2020.104281.

[3] P. Sun, X. Yang, X. Zhao, and Z. Wang, "An overview of named entity recognition," in *Proceedings of the 2018 International Conference on Asian Language Processing, IALP 2018*, Nov. 2019, pp. 273–278, doi: 10.1109/IALP.2018.8629225.

[4] H. J. Song, B. C. Jo, C. Y. Park, J. D. Kim, and Y. S. Kim, "Comparison of named entity recognition methodologies in biomedical documents," *BioMedical Engineering Online*, vol. 17, Nov. 2018, doi: 10.1186/s12938-018-0573-6.

[5] J. Li, A. Sun, J. Han, and C. Li, "A survey on deep learning for named entity recognition," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2020, doi: 10.1109/tkde.2020.2981314.

[6] R. Phan, T. M. Luu, R. Davey, and G. Chetty, "Deep learning based biomedical NER framework," in *Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, SSCI 2018*, Nov. 2019, pp. 33–40, doi: 10.1109/SSCI.2018.8628740.

[7] M. Dias, J. Boné, J. C. Ferreira, R. Ribeiro, and R. Maia, "Named entity recognition for sensitive data discovery in Portuguese," *Applied Sciences*, vol. 10, no. 7, Mar. 2020, doi: 10.3390/app10072303.

[8] D. Campos, S. Matos, and J. Luis, "Biomedical named entity recognition: a survey of machine-learning tools," in *Theory and Applications for Advanced Text Mining*, InTech, 2012.

[9] S. Rashmi, M. Hanumanthappa, and N. M. Jyothi, "Text-to-speech translation using support vector machine, an approach to find a potential path for human-computer speech synthesizer," in *Proceedings of the 2016 IEEE International Conference on Wireless Communications, Signal Processing and Networking, WiSPNET 2016*, Mar. 2016, pp. 1311–1315, doi: 10.1109/WiSPNET.2016.7566349.

[10] S. Rashmi, M. Hanumanthappa, and M. V. Reddy, "Hidden markov model for speech recognition system—a pilot study and a Naive approach for speech-to-text model," in *Advances in Intelligent Systems and Computing*, vol. 664, Springer Singapore, 2018, pp. 77–90.

[11] I. Fernandes, H. L. Cardoso, and E. Oliveira, "Applying deep neural networks to named entity recognition in Portuguese texts," in *2018 5th International Conference on Social Networks Analysis, Management and Security, SNAMS 2018*, Oct. 2018, pp. 284–289, doi: 10.1109/SNAMS.2018.8554782.

[12] W. Saad, W. A. Shalaby, M. Shokair, F. A. El-Samie, M. Dessouky, and E. Abdellatef, "COVID-19 classification using deep feature concatenation technique," *Journal of Ambient Intelligence and Humanized Computing*, Mar. 2021, doi: 10.1007/s12652-021-02967-7.

[13] L. Yuwen, S. Chen, and X. Yuan, "G2Basy: a framework to improve the RNN language model and ease overfitting problem," *PLoS ONE*, vol. 16, Apr. 2021, doi: 10.1371/journal.pone.0249820.

[14] L. Li, L. Jin, Z. Jiang, D. Song, and D. Huang, "Biomedical named entity recognition based on extended recurrent neural networks," in *2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, Nov. 2015, pp. 649–652, doi: 10.1109/BIBM.2015.7359761.

[15] M. N. A. Ali and G. Tan, "Bidirectional encoder–decoder model for arabic named entity recognition," *Arabian Journal for Science and Engineering*, vol. 44, no. 11, pp. 9693–9701, Nov. 2019, doi: 10.1007/s13369-019-04068-2.

[16] H. Cho and H. Lee, "Biomedical named entity recognition using deep neural networks with contextual information," *BMC Bioinformatics*, vol. 20, no. 1, Dec. 2019, doi: 10.1186/s12859-019-3321-4.

[17] C. Lyu, B. Chen, Y. Ren, and D. Ji, "Long short-term memory RNN for biomedical named entity recognition," *BMC Bioinformatics*, vol. 18, no. 1, Dec. 2017, doi: 10.1186/s12859-017-1868-5.

[18] J. Li *et al.*, "WCP-RNN: a novel RNN-based approach for Bio-NER in Chinese EMRs," *The Journal of Supercomputing*, vol. 76, no. 3, pp. 1450–1467, Mar. 2020, doi: 10.1007/s11227-017-2229-x.

[19] S. Chowdhury *et al.*, "A multitask bi-directional RNN model for named entity recognition on Chinese electronic medical records," *BMC Bioinformatics*, vol. 19, Dec. 2018, doi: 10.1186/s12859-018-2467-9.

[20] Tian, Y., Shen, W., Song, Y., Xia, F., He, M., & Li, K. Improving biomedical named entity recognition with syntactic information. *BMC Bioinformatics*, 21(1), 2020, doi: 10.1186/s12859-020-03834-6.

[21] N. Perera, M. Dehmer, and F. Emmert-Streib, "Named entity recognition and relation detection for biomedical information extraction," *Frontiers in Cell and Developmental Biology*, vol. 8, Aug. 2020, doi: 10.3389/fcell.2020.00673.

[22] T. H. Yang, T. H. Tseng, and C. P. Chen, "Recurrent neural network-based language models with variation in net topology, language, and granularity," in *Proceedings of the 2016 International Conference on Asian Language Processing, IALP 2016*, Nov. 2017, pp. 71–74, doi: 10.1109/IALP.2016.7875937.

[23] N. Iqbal and N. Iqbal, "Imatinib: a breakthrough of targeted therapy in cancer," *Chemotherapy Research and Practice*, vol. 2014, pp. 1–9, May 2014, doi: 10.1155/2014/357027.

[24] T. Luo and H. Yang, "Two-layer neural networks for partial differential equations: optimization and generalization theory," Jun. 2020. *ArXiv:2006.15733*.

[25] P. Lauren, G. Qu, G.-B. Huang, P. Watta, and A. Lendasse, "A low-dimensional vector representation for words using an extreme learning machine," in *2017 International Joint Conference on Neural Networks (IJCNN)*, May 2017, pp. 1817–1822, doi: 10.1109/IJCNN.2017.7966071.

[26] M. Gridach, "Character-level neural network for biomedical named entity recognition," *Journal of Biomedical Informatics*, vol. 70, pp. 85–91, Jun. 2017, doi: 10.1016/j.jbi.2017.05.002.

[27] Y. Wang, "A new concept using LSTM neural networks for dynamic system identification," in *Proceedings of the American Control Conference*, May 2017, pp. 5324–5329, doi: 10.23919/ACC.2017.7963782.

[28] Y. Wang, Y. Li, Y. Song, and X. Rong, "The influence of the activation function in a convolution neural network model of facial expression recognition," *Applied Sciences*, vol. 10, no. 5, Mar. 2020, doi: 10.3390/app10051897.

[29] K. Hara and K. Nakayamma, "Comparison of activation functions in multilayer neural network for pattern classification," in *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, 1994, vol. 5, pp. 2997–3002, doi: 10.1109/ICNN.1994.374710.

[30] L. Tanabe, N. Xie, L. H. Thom, W. Matten, and W. J. Wilbur, "GENETAG: a tagged corpus for gene/protein named entity recognition," *BMC Bioinformatics*, vol. 6, May 2005, doi: 10.1186/1471-2105-6-S1-S3.
[31] R. Siddalingappa and S. Kanagaraj, "Anomaly detection on medical images using autoencoder and convolutional neural network," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 7, pp. 148–156, 2021, doi: 10.14569/IJACSA.2021.0120717.
[32] S. Yadav and S. Shukla, "Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification," in *Proceedings - 6th International Advanced Computing Conference, IACC 2016*, Feb. 2016, pp. 78–83, doi: 10.1109/IACC.2016.25.
[33] I. Bilbao and J. Bilbao, "Overfitting problem and the over-training in the era of data: particularly for artificial neural networks," in *2017 IEEE 8th International Conference on Intelligent Computing and Information Systems, ICICIS 2017*, Dec. 2017, vol. 2018-Janua, pp. 173–177, doi: 10.1109/INTELCIS.2017.8260032.
[34] I. Nusrat and S. B. Jang, "A comparison of regularization techniques in deep neural networks," *Symmetry*, vol. 10, no. 11, Nov. 2018, doi: 10.3390/sym10110648.
[35] A. Berger and S. Guda, "Threshold optimization for F measure of macro-averaged precision and recall," *Pattern Recognition*, vol. 102, Jun. 2020, doi: 10.1016/j.patcog.2020.107250.
[36] L. Yao, H. Liu, Y. Liu, X. Li, and M. W. Anwar, "Biomedical named entity recognition based on deep neutral network," *International Journal of Hybrid Information Technology*, vol. 8, no. 8, pp. 279–288, Aug. 2015, doi: 10.14257/ijhit.2015.8.8.29.
[37] P. Mishra, S. Biswas, and S. Dash, "Deep learning based biomedical named entity recognition systems," *Springer International Publishing*, 2020.
[38] M. Lobo, A. Lamurias, and F. M. Couto, "Identifying human phenotype terms by combining machine learning and validation rules," *BioMed Research International*, vol. 2017, pp. 1–8, 2017, doi: 10.1155/2017/8565739.
[39] I. El Bazi and N. Laachfoubi, "Arabic named entity recognition using deep learning approach," *International Journal of Electrical and Computer Engineering*, vol. 9, no. 3, pp. 2025–2032, Jun. 2019, doi: 10.11591/ijece.v9i3.pp2025-2032.

BIOGRAPHIES OF AUTHORS

Rashmi Siddalingappa received B.Sc. and M.Sc. (Computer Science) and Ph.D. degree for phonetics and semantic analysis for natural language processing using data mining techniques from Bangalore University, Bengaluru, Karnataka, India. She is currently working as a national postdoctoral research fellow funded by SERB, India, in the computational and data sciences department at the Indian Institute of Science, Bengaluru, India. Her research interests include data mining, natural language processing, artificial intelligence, machine learning, and neural networks. She can be contacted at email: drrashmis64@gmail.com.

Kanagaraj Sekar received an M.Sc. and a Ph.D. in biophysics and crystallography from the University of Madras, India. His postdoctoral work pertained to protein crystallography, which he pursued at the Indian Institute of Science until 1992 and later at the Ohio State University from 1995. Since returning to India in 1998, he has been working at the Indian Institute of Science, Bengaluru, as a senior scientific officer of structural biology and bio-computing at its Bioinformatics Centre. During this period, he has held various positions, including those of a principal research scientist (2004–10) and an associate professor at the Computational and Data Sciences (CDS) department. Since 2016, he has been a professor at the CDS department and heads the Laboratory for Structural Biology and Bio-computing. He has been awarded the junior research fellowships of the Council of Scientific and Industrial Research (1984–88), the University Grants Commission of India (1988–89), and the senior research fellow of the CSIR (1989–92). In addition, the Department of Biotechnology of the Government of India awarded him the National Bioscience Award for Career Development, one of the highest Indian science awards, in 2004. His research in bioinformatics has covered protein crystallography, crystallographic and internet computing, and the development of value-added knowledge bases and algorithms. His studies have been documented by way of several articles, and Google Scholar has listed 211 of them. In addition, he has delivered keynote or plenary speeches at international seminars and conferences and has mentored many doctoral and postdoctoral scholars. He is also a member of the International Union of Crystallography (IUCr). He can be contacted at email: sekar@iisc.ac.in.