Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

Advancing Context-Aware Representations for Named Entity Recognition Using Hierarchical Architectures

Dattatray S. Shingate[1], Dr. Shyamrao V. Gumaste[2]

[1]Research Scholar, Department of Computer Engineering Mumbai Education Trust's Institute of Engineering, Nashik, India
Email: dattatrays_ioe@bkc.met.edu
[2]Research Supervisor, Professor, Department of Computer Engineering Mumbai Education Trust's Institute of Engineering, Nashik, India
Email: shyamraog_ioe@bkc.met.edu

Named Entity Recognition (NER) is a core task in the field of natural language processing (NLP). Hierarchical context-aware-based models are more efficient for name, entity, and relation extraction. In this approach, A model was constructed in this algorithm with the contextual representation of information at the sentence and document levels. In sentence-level representation, the sentences are individually processed to generate the name, entity, and relation. To do that, the information from given sentences can be extracted using a model of BiLSTM. The disadvantage of this approach is that it processes the entire sentences of documents. Not every sentence needs to have an entity and a relationship. Our method creates the filtered sentences by building a sentence classifier. At least one entity and its associated relationship with our entities set will be present in filtered sentences. We can improve overall system performance by reducing overhead calculation. In document-level representation or entity extraction, a memory network is employed.Hierarchical context-aware-based models are more efficient for name, entity, and relation extraction. In this approach, A model was constructed in this algorithm with the contextual representation of information at the sentence and document levels. In sentence-level representation, the sentences are individually processed to generate the name, entity, and relation. To do that, the information from given sentences can be extracted using a model of BiLSTM. The disadvantage of this approach is that it processes the entire sentences of documents. Not every sentence needs to have an entity and a relationship. Our method creates the filtered sentences by building a sentence classifier. At least one entity and its associated relationship with our entities set will be present in filtered sentences. We can improve overall system performance by reducing overhead calculation. In document-level representation or entity extraction, a memory network is employed.

KEYWORDS: NLP, NER, LSTM, BiLSTM, Word Embedding, maxpooling, deep learning, classifier, context-aware, sentence-classifier, document-classifier, ontology, semantic

1. Introduction

A typical data preprocessing activity is known as entity recognition (NER). Identification of the text's most important details and classification into several predetermined categories are involved. A constant subject of discussion or reference in the text will be referred to as an entity. The two steps that make up the basic structure of the model are as follows:

- Identifying items in the text
- Putting them into various categories

Some of the categories that make up NER's most significant architecture include: Person, Organization, Place/ location.

Modern deep learning-based NER systems significantly outperform traditional methods due to their ability to contextualize and assemble words effectively. This is because it utilized an approach called word embedding, which is capable of comprehending the semantic and syntactic relationships between diverse words. It can also analyze topic-specific and high-level terms automatically. As a result, deep learning NER can be used to execute numerous jobs. Deep learning can automate much of the repetitious work, allowing academics to make better use of their time. For the NER extraction using deep learning various algorithms had been proposed LSTM (Hochreiter and Schmidhuber 1997), DCNN - Diluted CNN Weston et al.2019, Bi-LSTM+CRF with noise Guha et al 2021, Multi-Granularity model (MGM) Da San Martino et al 2019, SC-NER Wang et. al 2019.

In this paper, To improve NER modeling, we propose a hierarchical contextualized representation architecture in this study. By considering base Shingate et.al. [34] extending system with the entity's sentence level representation and the entity's document level representation. We are attempting to improve the sentence-level representation for entity extraction in our research. To accomplish this, we will create a Sentence Classifier that will filter the sentences. Sentences with no entity or relations are removed. The advantage of this additional layer over the previous one is that it can reduce model overheads by analyzing non-essential words. This indirectly enhances the overall system performance. We use a memory-based model in document representation to extract the entity and relation and tie it to local and global context documents and information.

2. OUR SYSTEM

For the implementation of the system, we used the IntNet-LSTM-CRF model proposed model proposed by (Xin et al. 2018) as our baseline. The tiers of the encoder are Sequence Labeling Encoder, Sentence Level Encoder, and Document Level Encoder. The sentence classifier is used during the input preprocessing stage.

I. Sentence Classifier.

The primary goal of our research is to implement an improved version of the NER

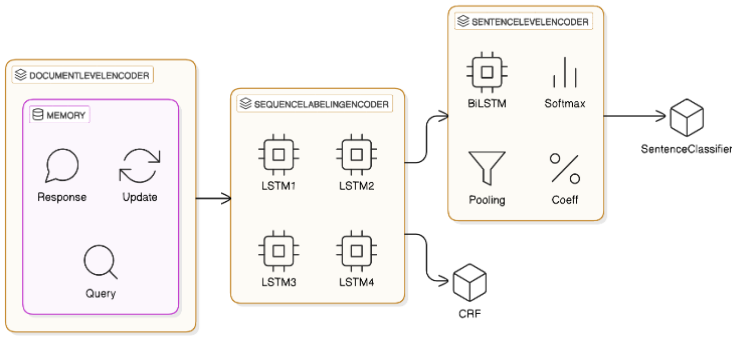


Figure 1: Architecture Diagram of our System implementation

sentence classifier to obtain better results. The ontology and word embedding models are used by the Sentence Classifier. The basic goal of the sentence classifier is to discover outliers by semantically analyzing terms in sentences using ontology and word embedding. So, after utilizing the pruning procedure to remove the outlier.

It operates in two stages. First, the parser examines the input and detects the relevant words. In the following stage, it employs part of speech and word embedding to discover promising terms. Finally, it combines and improves the outcomes of these two modules by integrating appropriate regions. Figure 2 depicts the block diagram of the sentence classifier.

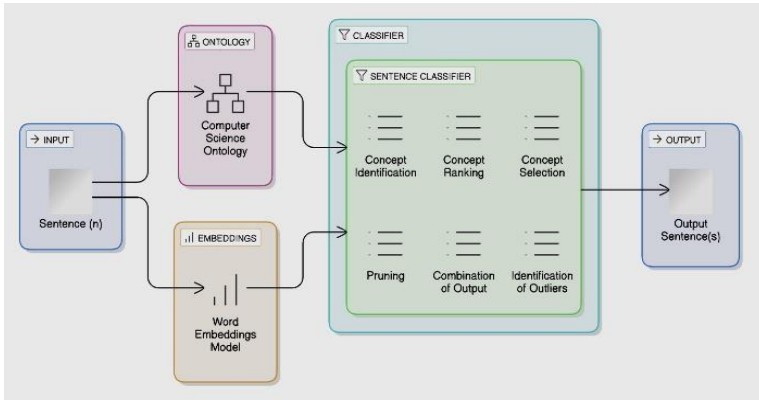


Figure 2: Sentence Classifier

Word Vector was utilized to better depict words. terms that appear close to each other do not necessarily have comparable meanings, but by evaluating the frequency of terms that occur close together, we find that words with similar meanings are found together. GloVE was utilized for the implementation of word vectors.

The classifier then goes through the extracted chunks and utilizes the word vectors to identify the relevant words. Part of speech is used to extract the chunks from the input text. We used the grammar-based chunk parser for a portion of the speech.

<JJ;>*<NN.*>+

where JJ indicates adj and NN indicates noun

In the final stage, the extracted relevant words are ranked using n-gram algorithms. The benefit of this is that we may prioritize the relevant words based on their frequency of occurrence. Pruning is the final stage. The pruning is performed using the average of the ranking values. Words with lower rankings than the average are removed.

Token Representation

For the given input text

S_n ={$word_1$,$word_2$, ….. $Word_n$ } (1)

where n is no of words in a given text. The generated tokens can be expressed as

$X_n=\{x_1,x_2,.....x_n\}$ (2)

And each token is

$X_i =[w_i, c_i]$ (3)

As a result, the token may also be represented as

$X_n=\{[w_1c_1],[w_2,c_2],...[w_nc_n]\}$ (4)

Where

w_i=pre-trained word embedding

c_i=character embedding

Sequence Labelling Encoder

$x_i = [w_i; c_i]$ is then passed into the sequence labeling BiLSTM.

$h_i = [\vec{h};h]$ (5)

Where

$\vec{h} =LSTM (x_i, \vec{h}_{i-1}, \vec{\theta})$ (6)

$\vec{h} =LSTM (x_i, \vec{h}_{i-1}, \vec{\theta})$ (7)

Where $\vec{\theta}$ and $\vec{\theta}$ are trainable parameters

Decoder

The decoder includes a Conditional Random Field. The Viterbi method is used during decoding to find the label sequence with the highest probability

For y = {y_1, ..., y_N} being a predicted sequence of labels with the same length as x. We define its score as

$sc(x,y)=\sum_{i=0}^{N-1} Tr_{yi,yi+1} +\sum_{i=0}^{N} P_{i,yi}$ (8)

Where,

$Tr_{yi,yi+1}$ Represent the transmission score from y_i to $y_{i+1.}$ The CRF model defines a family of conditional probability p(y|x) over all possible tag sequences y:

$$p(y|x)= \frac{\exp^{ect(x,y)}}{\sum_{\underline{y} \epsilon y} \exp^{sc(x,\underline{y})}} \quad (9)$$

During training, we take into account the maximum log-likelihood of the correct tag sequence. During decoding, we

look for the label sequence with the highest score.

$$y^* = \arg\max_{y \in y} sc(x, \underline{y}) \quad (10)$$

II. Sentence Level Representation

Incorporating sentence-level information has proven to be highly beneficial for enhancing the performance of model sequences[18][35]. We use a separate BiLSTM to produce contextualized features. Using word representation $x_i = [w_i; c_i]$ as input, the hidden states of the BiLSTM are denoted as $v \in R^{N \times ds}$, where N represents the sequence length and ds is the concealed size. Because words contribute differently to sentence-level representation, we use label embedding attention (Wang et al. 2018). to gain an attention score for the full sentence and then translate the hidden states $v \in R^{N \times ds}$ into a fixed-sized sentence-level representation $s \in Rds$. The cosine similarity $e(x_i, l_j)$ between the word and label embeddings x_i and l_j can be used to calculate the confidence score of this word-label combination.

$$e(x_i,l_j)=\frac{x_i^T \, l_j}{||\, x_i||\, ||l_j||} \quad (11)$$

Convolutional neural networks (CNN) are used to capture the relative spatial information between consecutive words in a phrase. A max-pooling technique is used to acquire the highest confidence score m_i RP between the i-th word and all labels:

$$m_i = max(W^T \begin{bmatrix} e(i-\frac{k-1}{2},:) \\ \cdots \\ e(i+\frac{k-1}{2},:) \end{bmatrix} + b) \quad (12)$$

where $W \in R^k$ and $b \in R^p$ are trainable parameters, where k is the kernel size, and max denotes max pooling. The attention (confidence) score $\beta \in R^N$ for the entire sentence is

$$\beta = \text{softmax}(m) \quad (13)$$

The sentence-level representation $s \in R^{ds}$ can be simply obtained by averaging the hidden states $v \in R^{N \times ds}$, weighted by the attention score calculated above:

$$s = \sum_{i=1}^{N} \beta_i v_i \quad (14)$$

The sentence-level representation $s \in R^{ds}$ is then concatenated with the word representation $x'_i = [x_i; s]$ and fed to the sequence labeling encoder. Note that the training. We use the label embeddings $l \in R^{P \times dw}$ of all labels, not the ground-truth label embedding. Words with higher confidence scores should contribute more to the sentence-level contextualized representation.

III. Document Level Representation

To memorize document-level contextualized representation, we use the key-value memory component m. Each Memory slot is represented by a pair of vectors (k_1, v_1),..., (k_m, v_m). For each token in the training data, the key represents the word embedding wiw_iwi, while the value corresponds to the associated hidden state hih_ihi generated by the sequence labeling encoder. Due to the dynamic nature of embeddings and contextual representations, the same word can appear in multiple distinct slots based on its context.

To compute the weight of document-level representation, the attention operation is invoked. The memory key $k_j = [k_{sub1};...; k_{subT}]$ is used as the attention key for the unique word, while the memory value vj $[v_{sub1};...; v_{subT}]$ is used as the attention value. Then, the searched word's embedding w_{qi} serves as the attention query q_i. In this section, we look at three compatibility

(1) dot-product attention

$$o_1(q_i, k_j) = q_i k_j^T$$

(2) scaled dot-product attention

$$o_2(q_i, k_j) = \frac{q_i k_j^T}{\sqrt{d_w}} \quad (15)$$

(16)functions $u_{ij} = o(q_i, k_j)$: (17)

The numbers in parentheses indicate the slot index of tokens in memory M. In training instances, the memory slots of these bold tokens are retrieved based on their inverted index and cosine similarity.

$$o_3(q_i, k_j) = \frac{q_i k_j^T}{\|q_i\| \|k_j\|} \quad (18)$$

where denotes the dimension of word embeddings. Memory Reaction The document-level representation is rated as follows

$$\alpha_{ij} = \frac{\exp(u_{ij})}{\sum_{z=1}^{T} \exp(u_{iz})} \quad (19)$$

$$r_i = \sum_{j=1}^{T} \alpha_{ij} v_j \quad (20)$$

The CRF layer then receives the fusion representation g_i R^{dh} of the original hidden representation and this document-level representation, where dh is the hidden side of the sequence labeling encoder:

$$g_i = \lambda h_i + (1 - \lambda) r_i \quad (21)$$

where is a hyperparameter that indicates how much document-aware information is used, 0 for only document-level representation, and 1 for discarding all document-level information

3. ALGORITHMS

Algorithm 1 Method: OurMethod() Input: CoNLL2003 Output: Loss, accuracy parameters

1.	I := ReadDataSet(file)	
2.	$S=\{S_1,S_2,....S_n\}$	eq(1)
3.	for S_i in S	
4.	X_i=Tokens(S_i)	eq(2)
5.	$X=X_i$ U X	`
6.	w_i= Words(X_i)	eq(3)
7.	c_i = Character(X_i)	eq(4)
8.	Words = Words U w_i	
9.	Chars = Chars U c_i	
10.	end	
11.	model = BuildSequencModel(Words,Chars)	
12.	model.encode()	
13.	cs =Calculate_Confident(X,Words,Chars)	eq(11)
14.	cs_matrix= Calculate_ConfidentMatrix(cs)	eq(12)
15.	res=softmax(cs_matrix)	eq(13)
16.	Weigjhts=CalculateWeights()	eq(14,15,16)
17.	Ind=RetrivalIndex_Similarity()	eq(18)
18.	model.embed(X, Words,Chars,Ind)	
19.	model.optimized()	
20.	model. encode()	
21.	loss,acc=model.train()	
22.	model.backward()	
23.	predict=model.decode()	eq(8,9)
24.	acc= Calculate_accuracy()	
25.	return loss, accuracy	

Algorithm 2

Method: BuildSequencModel()

Inputs: X, Words, Chars, Index

Output: model

1. WEmd=Build_WordEmbedding(Words)
2. CharEmd=BuildCharacterEmbedding(Chars)
3. Char_Rep=BiLSTM+CRF(Chars.CharEmd) eq(5,6,7)
4. Word_Rep=BiLSTM+CharRep()
5. model =BiLSTM(Word_Rep)
6. return model

4. EXPERIMENT

Dataset

CoNLL2003 (Sang and De Meulder 2003) is based on the NER Standard.

Word Embeddings

That Have Been Trained, we use the publically available pre-trained 100D GloVe embedding for this. The target and background data are fed into our algorithm. This ensures that the entity and relation representations for the target and context are learned.

Hyper-parameters

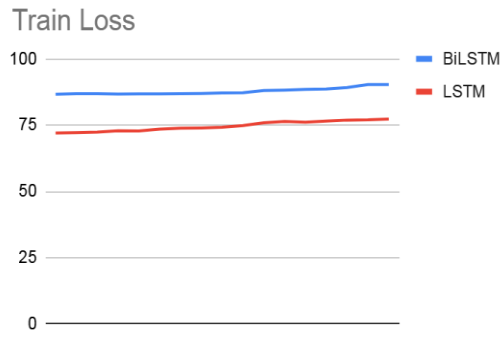
The table below displays the hyperparameters we utilized to optimize our model. Both models make use of these hyperparameters. The BiLSTM flag is used when creating a model with BiLSTM.

Table 1: Hyper-parameter set values

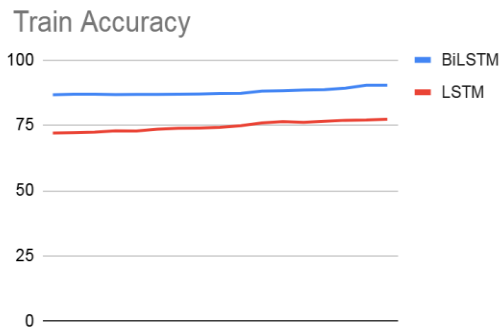
Learning Rate	0.05
momemtum	0.0
l2	1e-8
Hidden Dimension	256
dropout	0.5
rnn_dropout	0.5
LSTM Layer	1
BiLSTM	True/False
GPU	True/False
BatchSize	128
MaxLen	250
MaxError	0.5

5. RESULTS

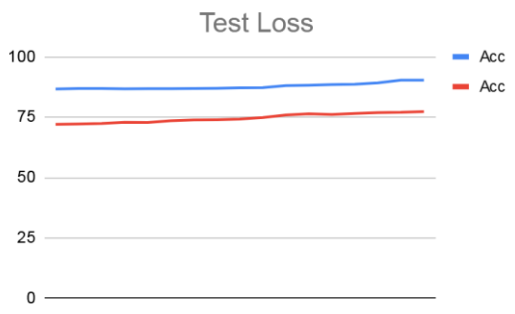
Results are calculated independently of train and test phase of the model. The results are graphically represented in the following graphs



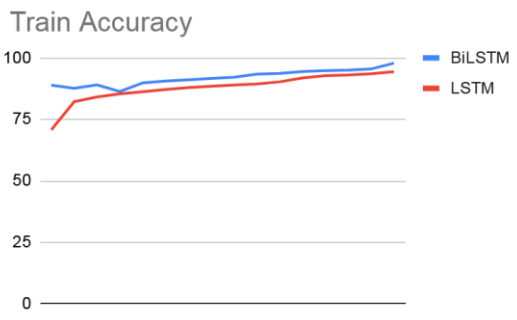
Graph 1: Loss occurance during training



Graph 2: Train Accuracy



Graph 3: Loss occurred in Testing



Graph 4: Test Accuracy

6. CONCLUSION

We modified the hierarchical contextualized technique for NER in this paper. By integrating sentence-level representation with document-level representation, our model fully utilizes the

training instances and spatial information of the embedding space. We take into account the relevance of words in sentences and weight their contributions with label embedding attention for sentence-level representation. The use of sentence classifiers in sentence-level ner information extraction has been effective, improving performance by 2%-3%. The CoNLL-2003 dataset is used to calculate the performance and empirical outcomes.

References

1. Akbik, Bergmann, and Vollgraf 2019] Akbik, A.;Bergmann, T.; and Vollgraf, R. 2019. Pooled contextualized embeddings for named entity recognition. In NAACL
2. [Akbik, Blythe, and Vollgraf 2018] Akbik, A.; Blythe, D.; and Vollgraf, R. 2018. Contextual string embeddings for sequence labeling. In COLING.
3. [Chen et al. 2019] Chen, H.; Lin, Z.; Ding, G.; Lou, J.;Zhang, Y.; and Karlsson, B. 2019. GRN: Gated relation network to enhance convolutional neural network for named entity recognition. In AAAI.
4. [Chiu and Nichols 2016] Chiu, J. P., and Nichols, E. 2016.Named entity recognition with bidirectional LSTM-CNNs.TACL.
5. [Clark et al. 2018] Clark, K.; Luong, M.-T.; Manning, C. D.; and Le, Q. V. 2018. Semi-supervised sequence modeling with cross-view training. In EMNLP.
6. [Devlin et al. 2019] Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In NAACL.
7. [Durrett and Klein 2014] Durrett, G., and Klein, D. 2014. A joint model for entity analysis: Coreference, typing, and linking. TACL.
8. [Ghaddar and Langlais 2018] Ghaddar, A., and Langlais, P. 2018. Robust lexical features for improved neural network named entity recognition. In COLING.
9. [Gillick et al. 2015] Gillick, D.; Brunk, C.; Vinyals, O.; and Subramanya, A. 2015. Multilingual language processing from bytes. Computer Science. [He et al. 2018] He, S.; Li, Z.; Zhao, H.; and Bai, H. 2018.
10. Syntax for semantic role labeling, to be, or not to be. In ACL.
11. [He, Li, and Zhao 2019] He, S.; Li, Z.; and Zhao, H. 2019. Syntax-aware multilingual semantic role labeling. In EMNLP.
12. [Hochreiter and Schmidhuber 1997] Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. Neural computation.
13. [Huang, Xu, and Yu 2015] Huang, Z.; Xu, W.; and Yu, K. 2015. Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint arXiv:1508.01991.
14. [Lafferty, Mccallum, and Pereira 2001] Lafferty, J. D.; Mc-callum, A.; and Pereira, F. C. N. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In ICML.
15. [Lample et al. 2016] Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; and Dyer, C. 2016. Neural architectures for named entity recognition. In NAACL.
16. [Liu et al. 2018] Liu, L.; Shang, J.; Ren, X.; Xu, F. F.; Gui, H.; Peng, J.; and Han, J. 2018. Empower sequence labeling with a task-aware neural language model. In AAAI.
17. [Liu et al. 2019a] Liu, P.; Chang, S.; Huang, X.; Tang, J.; and Cheung, J. C. K. 2019a. Contextualized non-local neural networks for sequence learning. In AAAI.
18. [Liu et al. 2019b] Liu, Y.; Meng, F.; Zhang, J.; Xu, J.; Chen, Y.; and Zhou, J. 2019b. GCDT: A global context enhanced deep transition architecture for sequence labeling. In ACL.
19. Ma and Hovy 2016] Ma, X., and Hovy, E. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In ACL
20. [Miller et al. 2016] Miller, A.; Fisch, A.; Dodge, J.; Karimi, A.-H.; Bordes, A.; and Weston, J. 2016. Key-value memory networks for directly reading documents. In EMNLP.
21. [Pennington, Socher, and Manning 2014] Pennington, J.; Socher, R.; and Manning, C. 2014. GloVe: Global vectors for word representation. In EMNLP.
22. [Peters et al. 2018] Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and

Zettlemoyer, L. 2018.Deep contextualized word representations. In NAACL.
23. [Pradhan et al. 2013] Pradhan, S.; Moschitti, A.; Xue, N.;Ng, H. T.; Bj¨orkelund, A.; Uryupina, O.; Zhang, Y.; and Zhong, Z. 2013. Towards robust linguistic analysis using onto notes. In CoNLL.
24. [Qian et al. 2019] Qian, Y.; Santos, E.; Jin, Z.; Guo, J.; and Barzilay, R. 2019. GraphIE: A graph-based framework for information extraction. In NAACL.
25. [Sang and De Meulder 2003] Sang, E. F., and De Meulder, F.2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In CoNLL.
26. [Shen et al. 2018] Shen, Y.; Yun, H.; Lipton, Z. C.; Kronrod,Y.; and Anandkumar, A. 2018. Deep active learning for named entity recognition. In ICLR.
27. [Strubell et al. 2017] Strubell, E.; Verga, P.; Belanger, D.;and McCallum, A. 2017. Fast and accurate entity recognition with iterated dilated convolutions. In EMNLP.
28. [Tjong Kim Sang 2002] Tjong Kim Sang, E. F. 2002. Introduction to the conll-2002 shared task: Language-independent named entity recognition. In CoNLL.
29. [Tran, MacKinlay, and Jimeno Yepes 2017] Tran, Q.; MacKinlay, A.; and Jimeno Yepes, A. 2017. Named entity recognition with stack residual LSTM and trainable bias decoding. In IJCNLP.
30. [Vaswani et al. 2017] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In NIPS.
31. [Wang et al. 2018] Wang, G.; Li, C.; Wang, W.; Zhang, Y.; Shen, D.; Zhang, X.; Henao, R.; and Carin, L. 2018. Joint embedding of words and labels for text classification. In
32. ACL. [Weston, Chopra, and Bordes 2014] Weston, J.; Chopra, S.; and Bordes, A. 2014. Memory networks. arXiv preprint arXiv:1410.3916.
33. [Xiao et al. 2019] Xiao, F.; Li, J.; Zhao, H.; Wang, R.; and Chen, K. 2019. Lattice-based transformer encoder for neural machine translation. In ACL.
34. M. D. S. Shingate and D. S. V. Gumaste, "Improving Ensemble Classifier for Natural Language Situational Information Analysis," 2024 International Conference on Emerging Innovations and Advanced Computing (INNOCOMP), Sonipat, India, 2024, pp. 305-311, doi: 10.1109/INNOCOMP63224.2024.00057. keywords: {Technological innovation;Social networking (online);Event detection;Natural languages;Blogs;Predictive models;Vectors;Disasters;RoBERTa model;LSTM;Bi-LSTM;Lemmatization;POS tag;NLP;situational information;Event detection
35. [Xin et al. 2018] Xin, Y.; Hart, E.; Mahajan, V.; and Ruvini, J.-D. 2018. Learning better internal structure of words for sequence labeling. In EMNLP.
36. [Yang and Zhang 2018] Yang, J., and Zhang, Y. 2018. NCRF++: An open-source neural sequence labeling toolkit.In ACL.
37. [Yang, Salakhutdinov, and Cohen 2017] Yang, Z.; Salakhut- dinov, R.; and Cohen, W. W. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. In ICLR.
38. [Yang, Zhang, and Dong 2017] Yang, J.; Zhang, Y.; and Dong, F. 2017. Neural reranking for named entity recognition. In RANLP.
39. [Zhang et al. 2020a] Zhang, S.; Zhao, H.; Wu, Y.; Zhang, Z.; Zhou, X.; and Zhou, X. 2020a. DCMN+: Dual co-matching network for multi-choice reading comprehension. In AAAI.
40. [Zhang et al. 2020b] Zhang, Z.; Wu, Y.; Zhao, H.; Li, Z.; Zhang, S.; Zhou, X.; and Zhou, X. 2020b. Semantics-aware BERT for language understanding. In AAAI.
41. [Zhang, Liu, and Song 2018] Zhang, Y.; Liu, Q.; and Song, L. 2018. Sentence-state LSTM for text representation. In ACL.
42. [Zhou and Zhao 2019] Zhou, J., and Zhao, H. 2019. Head-driven phrase structure grammar parsing on Penn Treebank.In ACL