Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.034439
Article

A Federated Named Entity Recognition Model with Explicit Relation for Power Grid

Jingtang Luo[1], Shiying Yao[1], Changming Zhao[2,*], Jie Xu[3] and Jim Feng[4]

[1]State Grid Sichuan Economic Research Institute, Chengdu, China
[2]School of Computer Science, Chengdu University of Information Technology, Chengdu, China
[3]School of Information and Communication Engineering University of Electronic Science and Technology of China, Chengdu, China
[4]Amphenol Global Interconnect Systems, San Jose, CA 95131, USA
*Corresponding Author: Changming Zhao. Email: zcm84@cuit.edu.cn
Received: 17 July 2022; Accepted: 23 November 2022

Abstract: The power grid operation process is complex, and many operation process data involve national security, business secrets, and user privacy. Meanwhile, labeled datasets may exist in many different operation platforms, but they cannot be directly shared since power grid data is highly privacy-sensitive. How to use these multi-source heterogeneous data as much as possible to build a power grid knowledge map under the premise of protecting privacy security has become an urgent problem in developing smart grid. Therefore, this paper proposes federated learning named entity recognition method for the power grid field, aiming to solve the problem of building a named entity recognition model covering the entire power grid process training by data with different security requirements. We decompose the named entity recognition (NER) model FLAT (Chinese NER Using Flat-Lattice Transformer) in each platform into a global part and a local part. The local part is used to capture the characteristics of the local data in each platform and is updated using locally labeled data. The global part is learned across different operation platforms to capture the shared NER knowledge. Its local gradients from different platforms are aggregated to update the global model, which is further delivered to each platform to update their global part. Experiments on two publicly available Chinese datasets and one power grid dataset validate the effectiveness of our method.

Keywords: Power grid; named entity recognition; federal learning

1 Introduction

With the development of an intelligent power grid, the construction of a knowledge graph of the power grid is an important task that needs to be completed urgently. As the basis of knowledge graph construction, entity recognition needs to complete the entity recognition of power grid text data. However, due to the existence of a large number of unstructured data in power grid text data, when

applying existing entity recognition methods, a large amount of manual pre-processing is required. The labeled data is used for training, but the manual labeling of entities is cumbersome and tedious, resulting in high labor consumption and labor costs. Power grid knowledge involves power equipment, transmission network, safety production rules, operational data, e-commerce, and other fields. If a separate named entity recognition model is established for each domain, the time and labor costs will increase dramatically.

In recent years, deep learning has emerged as a powerful strategy for learning feature representations directly from data and has led to remarkable breakthroughs in Natural Language Processing (NLP). When applied to named entity recognition (NER) task, deep learning can learn intricate hidden representations without complex feature engineering and rich domain knowledge. So deep learning-based methods have overwhelmingly surpassed traditional rule-based methods and statistical-based methods in NER. Sufficient labeled data is critical for these methods to train accurate NER models. However, power grid data annotated in a single operating platform such as a substation is usually limited. Annotating enough labeled data for power grid NER is expensive and time-consuming and requires a lot of expertise in the power grid field. Although many operation platforms may have some annotated power grid NER datasets, because power grid data are rich in user and company information and are highly sensitive to privacy and safety, they cannot be directly shared to train power grid NER models. Moreover, there are more and more applications of the power grid system, and the knowledge is becoming more and more complex. Most engineers only understand the domain knowledge in their field but do not know the other related fields. Therefore, a power grid knowledge graph needs to extract knowledge from many platforms, and these documents often have certain confidentiality properties.

Therefore, it is necessary to develop a federated learning (FL) named entity recognition model with encrypted data from each platform. Federated learning has become the standard computing paradigm for distributed machine learning, allowing geographically distributed data sources to efficiently train machine learning models while their original data remains in its original location. Compared to traditional machine learning algorithms, which require all training data to be centralized in one centralized location, federated learning relaxes the need for raw data migration and instead pushes down the training of machine learning models to each source. During federated training, only locally trained model parameters are shared with the outside world, and a centralized server is responsible for aggregating these parameters to compute the federated model.

In this paper, we propose an FL-based neural model for the power grid NER task, which is enhanced by explicit relative position encoding (ERP) in a distributed environment. The ERP mechanism is designed to explicitly capture contextual information on relative positions between character and word. Experimental results on a simulated environment with isolated data from five domains demonstrate the effectiveness of our approach, where the proposed model outperforms different baselines and boosts the performance of power grid NER by leveraging the labeled data on different platforms for model training in a collaborative way, at the same time remove the need to directly exchange raw data among different platforms for better privacy protection.

The main contributions of this paper are summarized as follows:

We propose a framework based on federated learning and the FLAT model to learn a more accurate power grid NER model from the labeled data of multiple operating platforms without the need to directly exchange the raw privacy-sensitive data among different platforms.

We propose an ERP modular to learn the relative positions of different categories of entities and to emphasize the model's ability to learn local information.

We conduct extensive experiments on different benchmark datasets to demonstrate the effectiveness of the proposed ERP-FL method.

The rest of the paper is organized as follows. Section 2 briefly reviews the previous works about NER and federal learning; Section 3 demonstrates the proposed approach in detail; In Section 4, we discuss the experiment settings and results; The final section shows the conclusion and future direction.

2 Related Works

We have investigated the previous NER task, the most successful neural network was the Bi-LSTM (Bi-directional Long Short Term Memory) [1] +CRF (Conditional Random Field) [2] model which achieved the SOTA performance. Yang et al. [3] proposed a neural re-ranking model for NER, where a convolutional layer with a fixed window size is used on top of a character embedding layer. Zhai et al. [4] first applied pointer networks to produce sequence tags. The pointer networks first identify a chunk (or a segment) and then label it. This operation is repeated until all the words in the input sequence are processed. Recently, large transformer-based models such as BERT (Bidirectional Encoder Representation from Transformers) [5] and XLM (Cross-lingual Language Model Pretraining) [6]. In order to analyze the performance of federated training on the NER task, we apply an FL model inspired by [7]. Google originally introduced federated learning to construct an input method model by using user data in private mobile phone [8]. They proposed the Federated Average method (FedAvg) which is the first synchronous FL algorithm. Many researchers have proposed other federated learning algorithms for larger-scale platforms inspired by this FL approach [9].

3 Approach

3.1 FLAT Model with Explicit Relative Position Encoding

Following the existing works [10], we adopt the FLAT model as the client model. The FLAT model considers the NER task as a sequence labeling task. FLAT is a Flat-Lattice Transformer for Chinese NER, which converts the lattice structure into a flat structure consisting of spans. Each span in the original lattice denotes a character or latent word and its position. This structure contributes to ignoring the impact of word segmentation errors and achieving the highest performance on many Chinese NER datasets.

The NER model is able to output a tag sentence "O O B I I I E O B I I E O O O O O O" given an input sentence. Each word is manually assigned a label for training [11–13]. We use a semantic label of BIEO format in which tags 'B', 'I', 'E', and 'O' represent accordingly begin, inside, end, and outside to label all the words.

However, when the FLAT model is dealing with the task of intelligent extraction of Chinese text entities, due to the ambiguity of Chinese word segmentation, there is still a problem of poor entity recognition effect. The relative position relationship matrix is constructed in an implicit way by analyzing the model. In order to improve the performance of the algorithm, it is necessary to explicitly construct the relative position relationship matrix.

In the original FLAT algorithm, the final relative position matrix is obtained by calculating four different relative position distances, then the self-attention matrix is calculated, and the class label corresponding to each position is obtained by outputting the Transformer structure to the CRF network [14–16]. The original FLAT only considers the different calculation methods of relative positions, and there is no obvious feature of highlighting different types of relationships. However,

the named entity model not only needs to learn long-distance dependencies but also capture the surrounding local information [17]. In order to improve the model's ability to learn the relative positions of different categories and to emphasize the model's ability to learn local information, we propose a novelty calculation method of the relative position matrix, the explicit relative position matrix (ERP), based on the original calculation process, as shown in Fig. 1.

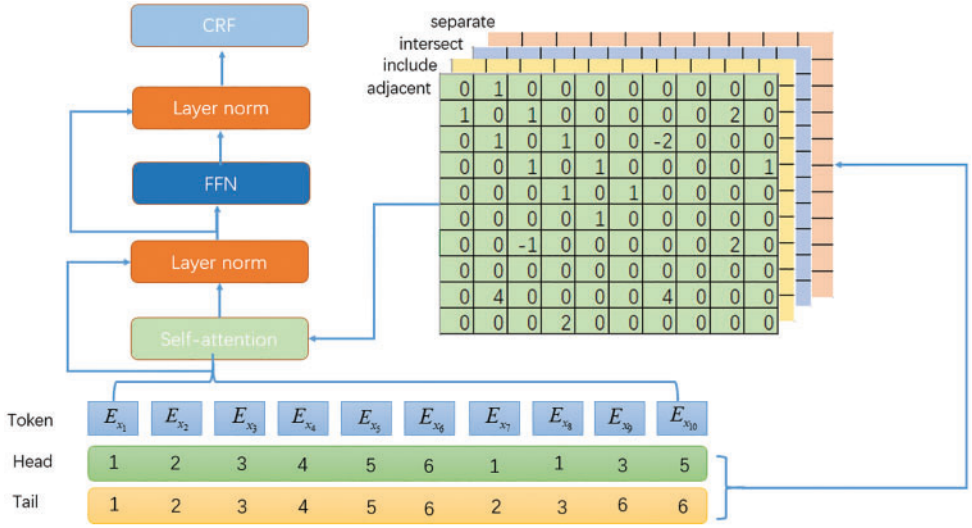

Figure 1: Overall of the FLAT with explicit relative position matrix

Through the explicit relative position matrix, the four relative positional relationships between each character and character, character and word, and word and word in a sentence are explicitly constructed. By judging the relationship between the head and tail positions, all tokens are classified into four relationships: intersection, adjacent, inclusion, and separation. Then, according to the pool of each relationship, an explicit relative position relationship matrix (as shown in Fig. 2) is constructed, and the multi-head self-attention matrix can be calculated, and then the final output category label can be obtained through the standard Transformer block structure and CRF layer.

We will describe the specific calculation process of the ERP module in detail as Fig. 1. We maintain a dictionary of technical terms collected from various platforms. Then according to the above dictionary, we obtain a potential words sequence in input sentence. The final flat input sequence includes a sequence of characters and potential words. Firstly, the input sequence $[w_1, w_2, \cdots, w_k]$ is converted into a token sequence by looking up a vocabulary table. Then the token sequence passes into an embedding layer to output an embedded sequence $\left[E_{x_1}, E_{x_2}, \cdots, E_{x_N}\right]$. We classify the relationship between characters and words into four types, including intersecting, adjacent, and separating. Assuming two tokens are $e_{p:q}$ and $e_{k:l}$, the relative relationship between them is determined by the formula (1):

$$\begin{cases} q = k-1 \ \ or \ \ l = p-1 \rightarrow \text{adjacent} \\ p < k \leq q < l \ \ or \ \ k < p \leq l < q \rightarrow \text{intersect} \\ p \leq k \leq q \leq l \ \ or \ \ k \leq p \leq l \leq q \rightarrow \text{include} \\ q < k-1 \ \ or \ \ l > p-1 \rightarrow \text{separate} \end{cases} \tag{1}$$

0	1	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	2	0
0	1	0	1	0	0	-2	0	0	0
0	0	1	0	1	0	0	0	0	1
0	0	0	1	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	0
0	0	-1	0	0	0	0	0	2	0
0	0	0	0	0	0	0	0	0	0
0	4	0	0	0	0	4	0	0	0
0	0	0	2	0	0	0	0	0	0

Figure 2: Explicit relative position matrix (adjacency relationship)

According to the obtained four span pools, four relative position matrices are calculated, which are intersection, adjacent, inclusion, and separation matrices. The elements of each matrix are calculated by the formula:

$$d_{ij}^{adjacent} = \begin{cases} H_i - H_j & i < j \\ T_i - T_j & i > j \\ 0 \; i = j \end{cases} \tag{2}$$

$$d_{ij}^{include} = \begin{cases} H_i - H_j & i < j \\ T_i - T_j & i > j \\ 1 \; i = j \end{cases} \tag{3}$$

$$d_{ij}^{\text{intersect/separate}} = \begin{cases} T_i - H_j & i < j \\ H_j - T_i & i > j \\ 0 \; i = j \end{cases} \tag{4}$$

where H and T denote the head and tail position of span. Then we calculate the final relative position relationship matrix R based on the obtained four relative relationship matrices, where each element is calculated as:

$$R_{ij} = \text{ReLU}\left(W_r\left(d_{ij}^{\text{adjacent}} \oplus d_{ij}^{\text{include}} \oplus d_{ij}^{\text{intersect}} \oplus d_{ij}^{\text{separate}}\right)\right) \tag{5}$$

where W_r are the learnable parameters and ReLU is the nonlinear function.

We use the explicit relative position matrix to compute the multi-head self-attention matrix Att.

$$\begin{aligned} &\text{Att}\,(Q, K, V) = softmax\,(A)\,V \\ &A_{ij} = E_{x_i} W_Q W_K^\top E_{x_j}^\top + E_{x_i} W_Q W_K^\top R_{ij}^\top + u W_Q W_K^\top E_{x_j}^\top + v W_Q W_K^\top R_{ij}^\top \\ &V = E_x W_v \end{aligned} \tag{6}$$

where $W_Q, W_K, W_V \in \mathbb{R}^{d_{\text{model}} \times d_{\text{head}}}$ are the learnable parameters.

The following calculation is the same as the vanilla Transformer. After FLAT with ERP, we only take the character representation into the output layer, followed by a Conditional Random Field (CRF).

3.2 Framework

We propose our privacy-preserving approach for power grid NER in Fig. 3. The client server in each platform is responsible for local model updating and global model sharing. We train the different local models with privately labeled data. After all global models share respective gradient, the central server aggregates each gradient and update the global model. It then distributes the updated parameters of the shared model to various platforms for the next round of model training.

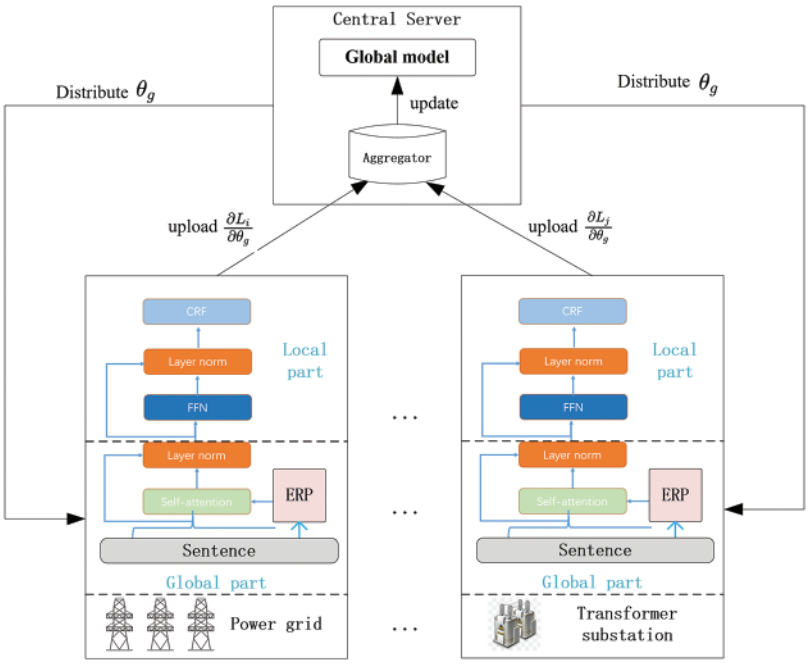

Figure 3: The overall framework of FLAT-ERP federal learning

However, traditional federated learning has the following two problems in the power grid NER task. 1) In practical scenarios, the labeling data of different platforms have the specific vocabulary and labeling forms, so if all platforms use the same model parameters, it will cause incompatibility. 2)Each platform focus on a different entity type. For example, some platforms prefer to find scheduling instructions, while others may obtain electricity measurement units, grid equipment names, and fault names. Therefore, we split the FLAT model into two parts, global and local part, inspired by [18–20]. The global part consists of the embedding layer, self-attention layer, and layer norm, which is responsible for capturing the semantic information in the sentence. This global part is trained by large-scale and well-annotated corpus over all platforms instead of insufficient domain-specific data for enhancing generalization ability. The local part contains the FFN layer, layer norm, and CRF, which is responsible for learning domain-specific context representations and label decoding strategies. We train the private module only with local data and exchange neither its parameters nor gradients. This framework facilitates model training by leveraging the useful information of labeled data in different platforms and protecting private data. In each platform P_i, wd denote the training dataset as S_i and loss function as L_i. For each iteration, the platform P_i selects a mini-batch training dataset $M_i = \frac{S_i}{\sum_{j \in P} S_i} M$, where M is the global model batch size. We utilize the dataset M_i in platform P_i to calculate the

gradients which parameters are $\frac{\partial \mathcal{L}_i}{\partial \theta_l^i}$ and $\frac{\partial \mathcal{L}_i}{\partial \theta_g^i}$ in local part and global part of FLAT-ERP model respectively. We transmit the global gradients $\frac{\partial \mathcal{L}_i}{\partial \theta_g^i}$ to the central server to calculate the aggregation gradient of global model. Meanwhile, the parameters of the local part are updated though the formula $\theta_l^i = \theta_l^i - \alpha \frac{\partial \mathcal{L}_i}{\partial \theta_l^i}$ and the hyper parameter α is the learning rate. The advantage of this mechanism is that there is no need to share private data between client servers, which ensures data security. At the same time, the model of each platform only synchronizes the representation ability of sentences and retains its independent sequence labeling ability, which makes the model of each platform more suitable for its own labeling tasks.

The central server has a global model to share with each platform and an aggregator to aggregate the local gradients from all platforms. Thus, the aggregated gradients $\frac{\partial \mathcal{L}}{\partial \theta_g}$ are calculated by the weighted average of all uploaded gradients.

$$\frac{\partial \mathcal{L}}{\partial \theta_g} = \sum_{i \in \mathcal{P}} \frac{|\mathcal{S}_i|}{\sum_{j \in \mathcal{P}} |\mathcal{S}_j|} \frac{\partial \mathcal{L}_i}{\partial \theta_g} \tag{7}$$

Since the client server only uploads part of the gradient, it is impossible to infer the training data of each platform, which is to well protect privacy and safety. The parameters of global model in the central server are updated by $\theta_g = \theta_g - \alpha \frac{\partial \mathcal{L}}{\partial \theta_g}$. After the update of the global model, the latest parameter will be distributed to each platform to update their global part of the FLAT-ERP model.

Overall, in the proposed federal learning framework, the FLAT-ERP model is able to capture the representation feature from all operation platforms and learn the specific labeling output in each operation platform. Meanwhile, the framework contributes to avoiding the exchange of private data and the possibility of data exposure.

4 Experiments

4.1 Settings and Datasets

For both client FLAT-ERP models, we use the default setting (i.e., 160-dimensional hidden vectors, 480 FFN size, eight multi-head attentions). SGD is chosen as the optimizer with an initialized learning rate of 1e-3 and weight decay of 5e-2. The momentum is 0.9, and the warmup epoch is 10. We use the dropout strategy to mitigate overfitting, and the dropout rate is set to 0.5 in the embedding layer and 0.3 in the output layer. The global model in the central server adopts 64 aggregated numbers of gradients M. The metrics used to evaluate correct tag (entity) predictions are F1. All models were evaluated on the same original test dataset 10 times.

We choose two public NER datasets, MSRA [21] and Ontonotes v4, for evaluating the general ability of our proposed model. Then we tested the performance of the framework on the self-built Chinese power grid dataset. The dataset contains 250,000 annotated corpora with 32 entity types. We randomly split each dataset as 80% training data and 20% testing data. The training data is assigned to 8 client servers equally.

4.2 Overall Results

Table 1 illustrates the experimental results of our FLAT-ERP federal learning framework and several baseline NER models, including BLSTM-CNN-CRF, LSTMCRF, CNN-CLSTM-CRF, lattice LSTM, CGN, original FLAT and single platform version FLAT-ERP.

Table 1: Experimental results (F1 scores) of different models

Models	MSRA	Ontonotes	Power grid
LSTM-CRF	90.94	69.78	52.19
BLSTM-CNN-CRF	91.67	70.23	52.93
Lattice LSTM,	93.18	73.88	58.71
CGN	93.47	74.79	59.02
Original FLAT	94.12	**76.45**	61.29
FLAT-ERP (single platform)	**95.07**	76.12	62.34
FLAT-ERP (federal learning)	93.78	75.54	**63.88**

There are several observations from the test set results. First, the FLAT-ERP model has achieved better performance than other baseline models in a single platform. It shows that explicit relative position encoding has improved the ability to capture the relationship between characters and lexicons. Second, on the premise of ensuring the safe use of data, FLAT-ERP (federal learning) achieves slightly weaker performance than the original FLAT model. At the same time, on the power grid entity named entity dataset, our federated learning framework continues to improve performance on a single platform. This is because the NER models of different platforms share their corpus representation capabilities and retain their own domain. Bias for output label predictions.

4.3 Influence of Clients' Number

To evaluate the effect of the number of clients, we combined the training and validation datasets from the original dataset and divided them into roughly equal-sized partitions for 8, 16, and 32 clients, such that each partition (client end) have its own unique label types. Table 2 shows the final learning performance of each model for different clients' numbers. It can be seen that when the number of clients for federated learning starts to increase, the performance of our proposed framework begins to drop, which is also consistent with the experimental results of most federated learning frameworks. But as the number of clients reaches 8, federated learning outperforms the single-platform model. Since then, as the number of clients increased, the performance of federated learning has decreased again.

Table 2: Experimental results (F1 scores) of different clients

Number of clients	F1-score
Centralized	62.34
4-clients	59.18
8-clients	**63.88**
16-clients	61.25
32-clients	60.93

The results show that with the current dataset size and number of entity types, our framework is able to surpass the performance of the centralized model for a certain number of clients.

5 Conclusions

In this paper, we propose an improved FLAT model with explicit relative position encoding based on the features of the grid entity recognition task and extend the model to a federated learning framework. We divide the FLAT-ERP model into a global part that can share gradients and a local part that can only update parameters locally, which removes the need to directly exchange the privacy-sensitive power grid data among different platforms for better privacy protection. Experiments on three benchmark datasets show that our method can effectively improve the performance of NER by leveraging useful information from multiple operating platforms in a privacy-preserving demand. In the future, we will focus on extending the proposed model to more fields and improving the performance of the open-end dataset.

Funding Statement: This work was supported by State Grid Science and Technology Research Program (SGSCJY00NYJS2200026).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[2] J. D. Lafferty, A. McCallum and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. ICML*, Morgan, Kaufmann, pp. 282–289, 2001.
[3] J. Yang, Y. Zhang and F. Dong, "Neural reranking for named entity recognition," in *Proc. RANLP*, Varna, Bulgaria, pp. 784–792, 2017.
[4] F. Zhai, S. Potdar, B. Xiang and B. Zhou, "Neural models for sequence chunking," in *Proc. AAAI, 2017*, San Francisco, California, USA, pp. 3365–3371, 2017.
[5] J. Devlin, M. W. Chang and K. Lee, "Bert: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv: 1810.04805, 2018.
[6] A. Conneau and G. Lample, "Cross-lingual language model pretraining," in *Proc. Advances in Neural Information Processing Systems*, Vancouver, BC, Canada, pp. 7057–7067, 2019.
[7] J. Mathew, D. Stripelis and J. L. Ambite, "Federated named entity recognition," arXiv e-prints arrive: 2203.15101, 2022.
[8] B. McMahan, E. Moore and D. Ramage, "Communication-efficient learning of deep networks from decentralized data," *Artificial Intelligence and Statistics*, vol. 54, pp. 1273–1282, 2017.
[9] P. Kairouz, H. B. McMahan, B. Avent and A. Bellet, "Advances and open problems in federated learning," *Foundations and Trends in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.
[10] X. Li, H. Yan and X. P. Qiu, "FLAT: Chinese NER using flat-lattice transformer," arXiv preprint arXiv: 2004.11795, 2020.
[11] G. Lample, M. Ballesteros, S. Subramanian and K. Kawakami, "Neural architectures for named entity recognition," in *Proc. the 2016 Conf. of the North American Chapter of the Association for Computational Linguistics*, San Diego, USA, pp. 260–270, 2016.
[12] Y. Zhang, H. S. Chen and Y. H. Zhao, "Learning tag dependencies for sequence tagging," in *Proc. 27th Int. Joint Conf. on Artificial Intelligence*, Stockholm, Sweden, pp. 4581–4587, 2018.

[13] E. Strubell, P. Verga and D. Belanger, "Fast and accurate entity recognition with iterated dilated convolutions," in *Proc. Conf. on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, pp. 2670–2680, 2017.
[14] M. Lewis, Y. H. Liu and N. Goyal, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proc. the 58th Annual Meeting of the Association for Computational Linguistics*, Online, pp. 7871–7880, 2020.
[15] J. P. C. Chiu and E. Nichols, "Named entity recognition with bidirectional LSTM-CNN," *Transactions of the Association for Computational Linguistics*, vol. 4, no. 1, pp. 357–370, 2016.
[16] N. Alsaaran and M. Alrabiah, "Arabic named entity recognition: A bert-bgru approach," *Computers, Materials & Continua*, vol. 68, no. 1, pp. 471–485, 2021.
[17] X. Z. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," in *Proc. the 54th Annual Meeting of the Association for Computational Linguistics*, Berlin, Germany, pp. 1064–1074, 2016.
[18] D. Jiang, Y. Song, Y. Tong and X. Wu, "Federated topic modeling," in *Proc. the Int. Conf. on Knowledge Management*, Durham, NC, USA, pp. pp 1071–1080, 2020.
[19] S. Ge, F. Z. Wu, C. H. Wu and Tao Qi, "Fedner: Privacy-preserving medical named entity recognition with federated learning," arXiv preprint arXiv: 2003.09288, 2021.
[20] B. Y. Lin, C. He, Z. Zeng and H. Wang, "Fednlp: Benchmarking federated learning methods for natural language processing tasks," arXiv preprint arXiv: 2104.08815, 2021.
[21] G. Levow, "The third international Chinese language processing bakeoff: Word segmentation and named entity recognition," in *Proc. the Fifth SIGHAN Workshop on Chinese Language Processing*, Sydney, Australia, pp. 108–117, 2006.