









Named Entity Recognition Architecture Combining Contextual and Global Features

Tran Thi Hong Hanh^{1,2,4} , Antoine Doucet² , Nicolas Sidere² ,
Jose G. Moreno³ , and Senja Pollak⁴  

¹ University of Science and Technology in Hanoi, Hanoi, Vietnam

² L3i Laboratory, University of La Rochelle, La Rochelle, France

³ IRIT, University of Toulouse, Toulouse, France

⁴ Jozef Stefan Institute, Ljubljana, Slovenia

senja.pollak@ijs.si

Abstract. Named entity recognition (NER) is an information extraction technique that aims to locate and classify named entities (e.g., organizations, locations, ...) within a document into predefined categories. Correctly identifying these phrases plays a significant role in simplifying information access. However, it remains a difficult task because named entities (NEs) have multiple forms and they are context dependent. While the context can be represented by contextual features, the global relations are often misrepresented by those models. In this paper, we propose the combination of contextual features from XLNet and global features from Graph Convolution Network (GCN) to enhance NER performance. Experiments over a widely-used dataset, CoNLL 2003, show the benefits of our strategy, with results competitive with the state of the art (SOTA).

Keywords: NER · XLNet · GCN · Contextual embeddings · Global embeddings

1 Introduction

The proliferation of large digital libraries has spurred interest in efficient and effective solutions to manage the collections of digital contents (documents, images, videos, etc.) which are available, but not always easy to find. As an alternative to better handle information in digital libraries, named entity recognition (NER) was introduced.

NER is an information extraction technique that aims to locate named entities (NEs) in text and classify them into predefined categories. Correctly identifying entities plays an important role in natural language understanding and numerous applications such as entity linking, question answering, or machine translation, to mention a few.

A crucial component that contributes to the recent success of NER progress is how meaningful information can be captured from original data via the word embeddings, which can be divided into two major types: global features and contextual features (in the scope of this paper, “features” and “embeddings” are interchangeable terms).

- **Global features** [30] capture latent syntactic and semantic similarities. They are first constructed from a global vocabulary (or dictionary) of unique words in the documents. Then, similar representations are learnt based on how frequently the words appear close to each other. The problem of such features is that the words’ meaning in varied contexts is often ignored. That means, given a word, its embedding always stays the same in whichever sentence it occurs. Due to this characteristic, we can also define global features as “*static*”. Some examples are word2vec [4], GloVe [30], and FastText [13].
- **Contextual features** [6] capture word semantics in context to address the polysemous and context-dependent nature of words. By passing the entire sentence to the pretrained model, we assign each word a representation based on its context, then capture the uses of words across different contexts. Thus, given a word, the contextual features are “*dynamically*” generated instead of being static as the global one. Some examples are ELMo [31], BERT [6], and XLNet [40].

In terms of global features, there exist several tokens that are always parts of an entity. The most obvious cases, as an example in the CoNLL 2003 dataset, are the names of countries include U.S. (377 mentions), Germany (143 mentions), Australia (136 mentions), to mention a few. However, it is not true for all tokens in an entity. The token may or may not be part of an entity (e.g., “Jobs said” vs. “Jobs are hard to find”) and may belong to different entity types depending on the context (e.g., “Washington” can be classified as a person or a location). Meanwhile, the contextual features are based on neighboring tokens, as well as the token itself. They aim to represent word semantics in context to solve the problem of using global features, so as to improve the prediction performance (e.g., “Jobs” in “Jobs said” and “Jobs are hard to find” will have different representations).

In this paper, we present a joint architecture to enhance the NER performance simultaneously with static and dynamic embeddings¹. Extensive experiments on CoNLL 2003 dataset suggest that our strategy surpasses the systems with standalone feature representation. The main contributions of this paper are:

- We introduce a new architecture that combines the contextual features from XLNet and the global features from GCN to enhance NER performance.
- We demonstrate that our model outperforms the systems using only contextual or global features alone and has a competitive result compared with SOTAs on CoNLL 2003 dataset.

¹ Link to the code: github.com/honghanhh/ner-combining-contextual-and-global-features.

This paper is organised as follows: Sect. 2 presents the related work, which leads to our approach’s descriptions in Sect. 3 and the corresponding experimental details in Sect. 4. The results are reported in Sect. 5, before we conclude and present future works in Sect. 6.

2 Related Work

2.1 Named Entity Recognition

The term “named entity” (NE) first appeared in the 6th Message Understanding Conference (MUC-6) [9] to define the recognition of the information units. Regarding the surveys on NER techniques [18, 29, 39], we can broadly divide them into four categories: Rule-based, unsupervised learning, feature-based supervised learning, and deep learning based approaches.

Rule-Based Approaches. Rule-based NER is the most traditional technique that does not require annotated data as it relies on manually-crafted rules well-designed by the domain experts (e.g., LTG [26], NetOwl [14]). Despite good performance when the lexicon is exhaustive, such systems often achieve high precision and low recall due to the limitation on domain-specific rules and incomplete dictionaries.

Unsupervised Learning. Another approach that also needs no annotated data is unsupervised learning, typically NE clustering [5]. The key idea is to extract NEs from the clustered groups based on context similarity. The lexical resources, lexical patterns, and statistics are computed on a large corpus and then applied to infer mentions of NEs. Several works proposed the unsupervised systems to extract NEs in diverse domains [8, 28].

Feature-Based Supervised Learning. Given annotated data, features are carefully designed so that the model can learn to recognize similar patterns from unseen data. Several statistical methods have been proposed, notably Markov models, Conditional Random Fields (CRFs), and Support Vector Machines (SVMs). Among them, CRF-based NER has been widely applied to identify entities from texts in various domains [21, 33, 34]. However, these approaches depend heavily on hand-crafted features and domain-specific resources, which results in the difficulty to adapt to new tasks or to transfer to new domains.

Deep Learning. Neural networks offer non-linear transformation so that the models can learn complex features and discover useful representations as well as underlying factors. Neural architectures for NER often make use of either Recurrent Neural Networks (RNNs) or Convolution Neural Networks (CNNs) in conjunction with CRFs [3] to extract information automatically. With further researches on contextual features, RNNs plus LSTM units and CRFs have been proposed [15] to improve the performance. Moreover, the conjunction of bidirectional LSTMs, CNNs, and CRFs [25] is introduced to exploit both word- and character-level representations. The combination of Transformer-based models, LSTMs, and CRFs [20] is also applied to extract knowledge from raw texts and empower the NER performance.

2.2 Embeddings

A key factor that contributes to the success of NER is how we capture meaningful information from original data via word representations, especially global features and contextual features.

Global Features. Global features are context-free word representations that can capture meaningful semantic and syntactic information. It can be represented at different levels such as word-level features [19], lookup features [10], document and corpus features [12]. Recently, the global sentence-level representation [42] has been proposed to capture global features more precisely and it outperforms various sequence labeling tasks. Furthermore, the Graph Neural Network [41] is getting more attention to not only have rich relational structure but also preserve global structure information of a graph in graph embeddings.

Contextual Features. Contextual features are context-aware word representations that can capture word semantics under diverse linguistic contexts. That is, a word can be represented differently and dynamically under particular circumstances. The contextual embeddings are often pretrained on large-scale unlabelled corpora and can be divided into 2 types: unsupervised approaches [16, 17] and supervised approaches [36].

The contextual embeddings succeed in exploring and exploiting the polysemous and context-dependent nature of words, thereby moving beyond global word features and contributing significant improvements in NER. In contrast, the global features are still less-represented.

3 Methodology

In this section, we explain how we extract global as well as contextual features and how to combine them. For global features, we take advantage of GCN [2, 35] to better capture the correlation between NEs and the global semantic information in text, and to avoid the loss of detailed information. For contextual features, we apply XLNet [40], a Transformer-XL pretrained language model that exhibits excellent performance for language tasks by learning from bi-directional context. The details are explained in the following subsections.

3.1 GCN as Global Embeddings

Graph Convolutional Network (GCN) aims to learn a function of signals/features on a graph $G = (V, E)$ with V as Vertices and E as Edges. Given N as number of nodes, D as number of input features, and F as the number of output features per node, GCN takes 2 inputs: (1) An $N \times D$ feature matrix X as feature description; (2) An adjacency matrix A as representative description of the graph; Finally, it returns as output Z , an $N \times F$ feature matrix [7].

Every neural network layer can then be written in the form of a non-linear function:

$$H^{(l+1)} = f(H^{(l)}, A) \quad (1)$$

where $H^{(0)} = X$, $H^{(L)} = Z$, L being the number of layers.

In our specific task, we capture the global features by feeding feature matrix X and adjacent matrix A into a graph using two-layer spectral convolutions in GCN. Raw texts are first transformed into word embeddings using GloVe. Then, universal dependencies are employed so that the input embeddings are converted into graph embeddings where words become nodes and dependencies become edges. After that, two-layer GCN is applied to the generated matrix of nodes feature vectors X and the adjacent matrix A to extract meaningful global features.

Mathematically, given a specific graph-based neural network model $f(X, A)$, spectral GCN follows the layer-wise propagation rule:

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)}) \quad (2)$$

where A is the adjacency matrix, X is the matrix of node feature vectors (given sequence x), D is the degree matrix, $f(\cdot)$ is the neural network like differentiable function, $\tilde{A} = A + I_N$ is the adjacency matrix of the undirected graph G with added self-connections, I_N is the identity matrix of N nodes, $\tilde{D}_i = \sum_j \tilde{A}_{ij}$, $W^{(l)}$ is the layer-specific trainable weight matrix, $\sigma(\cdot)$ is the activation function, and $H^{(l)} \in \mathbb{R}^{(N \times D)}$ is the matrix of activation in the l^{th} layer (representation of the l^{th} layer), $H^{(0)} = X$.

After calculating the normalized adjacency matrix $\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$ in the preprocessing step, the forward model can be expressed as:

$$Z = f(X, A) = softmax(\tilde{A}ReLU(\tilde{A}XW^0)W^1) \quad (3)$$

where $W^{(0)} \in R^{C \times H}$ is the input-to-hidden weight matrix for a hidden layer with H feature maps and $W^{(1)} \in R^{H \times F}$ is the hidden-to-output weight matrix.

$W^{(0)}$ and $W^{(1)}$ are trained using gradient descent. The weights before feeding into Linear layer with Softmax activation function are taken as global features to feed into our combined model. We keep the prediction results of GCN after feeding weights to the last Linear layer to compare the performance and prediction qualities with our proposed architecture's results.

3.2 XLNet as Contextual Embeddings

XLNet is an autoregressive pretraining method based on a novel generalized permutation language modeling objective. Employing Transformer-XL as the backbone model, XLNet exhibits excellent performance for language tasks involving long context by learning from bi-directional context and avoiding the disadvantages in the autoencoding language model.

The contextual features are captured from the sequence using permutation language modeling objective and two-stream self-attention architecture, integrating relative positional encoding scheme and the segment recurrence mechanism from Transformer-XL [40]. Given a sequence x of length T , the permutation language modeling objective can be defined as:

$$\max_{\theta} \mathbb{E}_{\mathbf{z} \sim \mathcal{Z}_T} \left[\sum_{t=1}^T \log p_{\theta} (x_{z_t} \mid \mathbf{x}_{\mathbf{z} < t}) \right] \quad (4)$$

where \mathcal{Z}_T is the set of all possible permutations of the index sequence of length T $[1, 2, \dots, T]$, z_t is the t^{th} element of a permutation $\mathbf{z} \in \mathcal{Z}_T$, $\mathbf{z} < t$ is the first $(t - 1)^{\text{th}}$ elements of a permutation $\mathbf{z} \in \mathcal{Z}_T$, and p_{θ} is the likelihood. θ is the parameter shared across all factorization orders during training so x_t is able to see all $x_i \neq x_t$ possible elements in the sequence.

We also use two-stream self-attention to remove the ambiguity in target predictions. For each self-attention layer $m = 1, \dots, M$, the two streams of representation are updated schematically with a shared set of parameters:

$$\begin{aligned} g_{z_t}^{(m)} &\leftarrow \text{Attention} \left(\text{Q} = g_{z_t}^{(m-1)}, \text{KV} = \mathbf{h}_{\mathbf{z} < t}^{(m-1)}; \theta \right) \\ h_{z_t}^{(m)} &\leftarrow \text{Attention} \left(\text{Q} = h_{z_t}^{(m-1)}, \text{KV} = \mathbf{h}_{\mathbf{z} \leq t}^{(m-1)}; \theta \right) \end{aligned} \quad (5)$$

where $g_{z_t}^{(m)}$ is the query stream that uses z_t but cannot see x_{z_t} , $h_{z_t}^{(m)}$ is the content stream that uses both z_t and x_{z_t} , and K, Q, V are the key, query, value, respectively.

To avoid slow convergence, the objective is customized to maximize the log-likelihood of the target sub-sequence conditioned on the non-target sub-sequence as in Eq. 6.

$$\max_{\theta} \mathbb{E}_{\mathbf{z} \sim \mathcal{Z}_T} \left[\log p_{\theta} (x_{\mathbf{z} > c} \mid \mathbf{x}_{\mathbf{z} \leq c}) \right] = \mathbb{E}_{\mathbf{z} \sim \mathcal{Z}_T} \left[\sum_{t=c, t \neq c+1}^{|\mathbf{z}|} \log p_{\theta} (x_{z_t} \mid \mathbf{x}_{\mathbf{z} < t}) \right] \quad (6)$$

where $\mathbf{z} > c$ is the target sub-sequence, $\mathbf{z} \leq c$ is the non-target one, and c is the cutting point.

Furthermore, we make use of the relative positional encoding scheme and the segment recurrence mechanism from Transformer-XL. While the position encoding ensures the reflection in the positional information of text sequences, the attention mask is applied so the texts are given different attention during the creation of input embedding. Given 2 segments $\mathbf{x} = s_{1:T}$ and $\mathbf{x} = s_{T:2T}$ from a long sequence s , \mathbf{z} and z referring to the permutations of $[1 \dots T]$ and $[T + 1 \dots 2T]$, we process the first segment, and then cache the obtained content representations $\mathbf{h}^{(m)}$ for each layer m . After that, we update the attention for the next segment \mathbf{x} with memory, which can be expressed as in Eq. 7.

$$h_{z_t}^{(m)} \leftarrow \text{Attention} \left(\text{Q} = h_{z_t}^{(m-1)}, \text{KV} = \left[\tilde{\mathbf{h}}^{(m-1)}, \mathbf{h}_{\mathbf{z} \leq t}^{(m-1)} \right]; \theta \right) \quad (7)$$

Similar to global features, we capture the weights before feeding to the last Linear layer and use it as contextual embeddings of our combined model. For the purpose of comparison, we also keep the prediction results of XLNet after feeding weights to the last Linear layer.

3.3 Joint Architecture

Given global and contextual features from GCN and XLNet, respectively, we concatenate and feed them into a Linear layer, which is simplest way to show the most evident impact of these features to the NER task. The proposed approach is presented in Fig. 1.

4 Experimental Setup

In this section, we describe the dataset, the evaluation metrics, as well as present our implementations and experimental configurations on XLNet, GCN, and the joint models in detail.

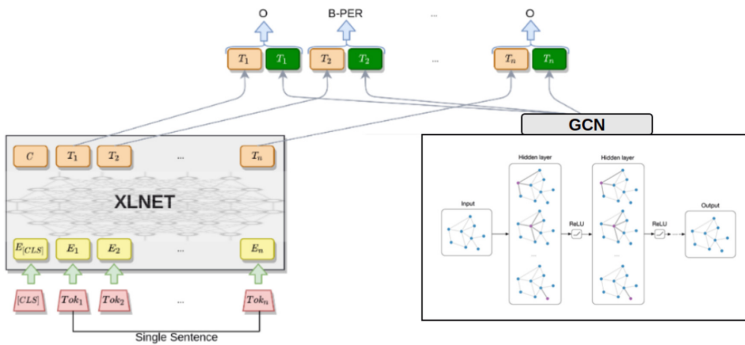


Fig. 1. Visualization of the global architecture of our proposed approach.

4.1 Dataset

We opted for the CoNLL 2003 [38], one of the widely-adopted benchmark datasets for NER tasks. The English version is collected from the Reuters Corpus with news stories between August 1996 and August 1997. The dataset concentrates on 4 types of NEs: persons (PER), locations (LOC), organizations (ORG), and miscellaneous (MISC).

4.2 Implementation Details

Global Embeddings with GCN. The sentences are annotated with universal dependencies from spaCy to create a graph of relations where words become nodes and dependencies become edges. The dataset is then converted into 124 nodes and 44 edges with the training corpus size of approximately 2 billion words, the vocabulary size of 222,496, and the dependency context vocabulary size of 1,253,524. Next, the graph embeddings are fed into 2 Graph Convolution layers with a Dropout of 0.5 after each layer to avoid overfitting. The global features

are captured before the last Linear layer. We perform batch gradient descent using the whole dataset for every training iteration, which is a feasible option as long as the dataset fits in memory. We take advantage of TensorFlow for efficient GPU-based implementation of Eq. 2 using sparse-dense matrix multiplications.

Contextual Embeddings with XLNet. We have investigated on diverse embeddings such as FastText [27]², Flair [1]³, Stanza [32]⁴ and XLNet [40]⁵ pretrained embeddings. Preliminary results suggest that XLNet (XLNet-Base, Cased) outperforms others, therefore, is chosen for our final implementation. The word embedding of size 768 with 12 layers were used for XLNet. Each layer consists of 3 sublayers: XLNet Relative Attention, XLNet Feed Forward, and Dropout layer. The XLNet Relative Attention is a two-stream self-attention mechanism as mentioned in Eq. 7. A Normalization layer with element-wise affine and a Dropout layer are employed around this sub-layer. Meanwhile, XLNet Feed Forward is a fully connected feed-forward network, whose outputs are also of dimension 768, the same as the outputs of the embedding layers. Like the previous sublayers, the Feed Forward layer is surrounded by a Normalization layer and a Dropout layer, however, another 2 Linear layers are added between them. Then, an additional Dropout layer is counted. It is notable that we only take the rate of 0.1 for every Dropout layer inside our model, from sublayers to inside sublayers. After 12 XLNet layers, another Dropout layer is added before the last Linear layer. We capture the intermediary output before the last Linear layer as the contextual features.

Proposed Model. Additional steps were taken to maintain alignments between input tokens and their corresponding labels as well as to match corresponding representations from global features to contextual features in the same sentence. First, we define an attention mask in XLNet as a sequence of 1s and 0s, with 1s for the first sub-word as the whole word embedding after tokenization and 0s for all padding sub-words. Then, in GCN features, we map the corresponding word representation at the position that the XLNet attention mark returns 1s and pad 0 otherwise. Therefore, each sentence has the same vector dimension in both global and contextual embeddings, which simplifies the concatenation.

In our implementation, we used a GPU 2070 Super and a TitanX GPU with 56 CPUs, 128 GB RAM. The hyperparameters were 300 as embedding size, 16 as batch size, 5e-5 as learning rate, 0.5 as dropout rate, 4 for number of epochs.

4.3 Metrics

We choose “relaxed” micro averaged F_1 -score, which regards a prediction as the correct one as long as a part of NE is correctly identified. This evaluation metric has been used in several related publications, journals, and papers on NER [11, 15, 25, 37].

² <https://fasttext.cc/>.

³ <https://github.com/flairNLP/flair>.

⁴ <https://github.com/stanfordnlp/stanza>.

⁵ <https://github.com/zihangdai/xlnet>.

5 Results

We conducted multiple experiments to investigate the impact of global and contextual features on NER. Specifically, we implemented the architecture with only global features, only contextual features, and then the proposed joint architecture combining both features.

As shown in Table 1a, the proposed model achieves **93.82%** in F_1 -score, which outperforms the two variants using global or contextual features alone. In terms of recognition of specific entity types, the details are provided in Table 1b, showing that PER is the category where the best results are achieved, while the lowest results are with the MISC, that is, the category of all NEs that do not belong to any of the three other predefined categories. Note that using only training data and publicly available word embeddings (GloVe), our proposed model has competitive results without the need of adding any extra complex encoder-decoder layers.

Table 1. Evaluation on the prediction results of our proposed model.

(a) Results of the proposed joint architecture compared to only contextual or only global features.

Embeddings	F_1 scores
Global features	88.63
Contextual features	93.28
Global + contextual features	93.82

(b) Performance evaluation per entity type.

Entity types	Precision	Recall	F_1 -score
LOC	94.15	93.53	93.83
MISC	81.33	81.89	81.62
ORG	88.97	92.29	90.60
PER	96.67	97.09	96.88

Furthermore, the benefit of the joint architecture is illustrated in Fig. 2. While contextual features (XLNet), which is used in the majority of recent SOTA approaches, misclassifies the entity, the prediction from GCN and the combined model correctly tags “MACEDONIA” as the name of a location, confirming our hypothesis on the effect of global features.

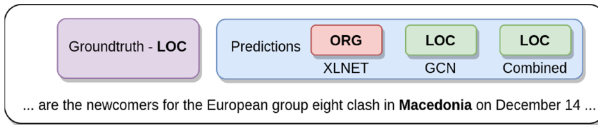


Fig. 2. XLNet, GCN, and the combined model’s prediction on CoNLL 2003’s example.

In Fig. 3, we compare our results with reported SOTA results on the same dataset from 2017 up to now. It can be observed that our results are competitive compared with SOTA approaches as the difference is by a small margin (the current benchmark is 94.3% F_1 -score, compared to 93.82% achieved by our approach). Moreover, we notice that NER performance can be boosted with external

knowledge (i.e. leveraging pretrained embeddings), as proven in our approach as well as in top benchmarks [22–24]. More importantly, complex decoder layers (CRF, Semi-CRF, ...) do not always lead to better performance in comparison with softmax classification when we take advantage of contextualized language model embeddings.

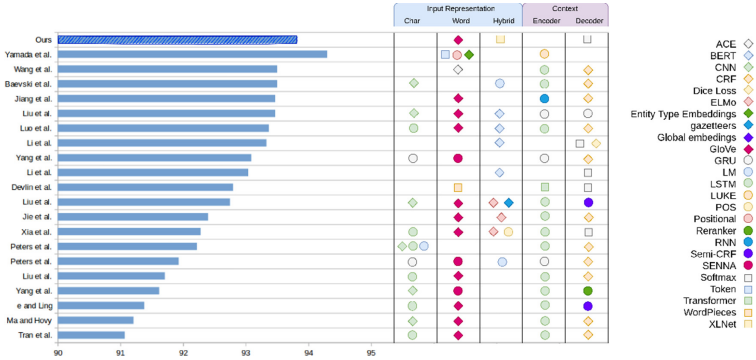


Fig. 3. Comparison of our proposal against SOTA techniques on the CoNLL 2003 dataset in terms of F_1 -score. Values were taken from original papers and sorted by descending order.

6 Conclusion and Future Work

We propose a novel hierarchical neural model for NER that uses both global features captured via graph representation and contextual features at the sentence level via XLNet pretrained model. The combination of global and contextual embeddings is proven to have a significant effect on the performance of NER tasks. Empirical studies on the CoNLL 2003 English dataset suggest that our approach outperforms systems using only global or contextual features, and is competitive with SOTA methods. Given the promising results in English, our future work will consist of adapting the method to other languages, as well to a cross-lingual experimental setting. In addition, we will consider further developing the method by also incorporating background knowledge from knowledge graphs and ontologies.

Acknowledgements. This work has been supported by the European Union’s Horizon 2020 research and innovation program under grants 770299 (NewsEye) and 825153 (EMBEDDIA). The work of S. P. has also received financial support from the Slovenian Research Agency for research core funding for the Knowledge Technologies programme (No. P2-0103) and the project CANDAS (No. J6-2581).

References

1. Akbik, A., Bergmann, T., Blythe, D., Rasul, K., Schweter, S., Vollgraf, R.: FLAIR: an easy-to-use framework for state-of-the-art NLP. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations), pp. 54–59 (2019)
2. Cetoli, A., Bragaglia, S., O’Harney, A., Sloan, M.: Graph convolutional networks for named entity recognition. In: Proceedings of the 16th International Workshop on Treebanks and Linguistic Theories, pp. 37–45 (2017)
3. Chiu, J.P., Nichols, E.: Named entity recognition with bidirectional LSTM-CNNs. *Trans. Assoc. Comput. Linguist.* **4**, 357–370 (2016)
4. Church, K.W.: Word2vec. *Nat. Lang. Eng.* **23**(1), 155–162 (2017)
5. Collins, M., Singer, Y.: Unsupervised models for named entity classification. In: 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (1999)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (1) (2019)
7. Duvenaud, D.K., et al.: Convolutional networks on graphs for learning molecular fingerprints. In: Advances in Neural Information Processing Systems, pp. 2224–2232 (2015)
8. Etzioni, O., et al.: Unsupervised named-entity extraction from the web: an experimental study. *Artif. Intell.* **165**(1), 91–134 (2005)
9. Grishman, R., Sundheim, B.M.: Message understanding conference-6: a brief history. In: COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics (1996)
10. Hoffart, J., et al.: Robust disambiguation of named entities in text. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pp. 782–792 (2011)
11. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. [arXiv:1508.01991](https://arxiv.org/abs/1508.01991) (2015)
12. Ji, Z., Sun, A., Cong, G., Han, J.: Joint recognition and linking of fine-grained locations from tweets. In: Proceedings of the 25th International Conference on World Wide Web, pp. 1271–1281 (2016)
13. Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., Mikolov, T.: Fast-Text.zip: compressing text classification models. [arXiv preprint arXiv:1612.03651](https://arxiv.org/abs/1612.03651) (2016)
14. Krupka, G., IsoQuest, K.: Description of the NEROWL extractor system as used for MUC-7. In: Proceedings of the 7th Message Understanding Conference, Virginia, pp. 21–28 (2005)
15. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 260–270 (2016)
16. Lample, G., Conneau, A.: Cross-lingual language model pretraining. [arXiv:1901.07291](https://arxiv.org/abs/1901.07291) (2019)
17. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: ALBERT: a lite BERT for self-supervised learning of language representations. In: International Conference on Learning Representations (2019)
18. Li, J., Sun, A., Han, J., Li, C.: A survey on deep learning for named entity recognition. *IEEE Trans. Knowl. Data Eng.* (2020)

19. Liao, W., Veeramachaneni, S.: A simple semi-supervised algorithm for named entity recognition. In: Proceedings of the NAACL HLT 2009 Workshop on Semi-supervised Learning for Natural Language Processing, pp. 58–65 (2009)
20. Liu, L., Shang, J., Ren, X., Xu, F.F., Gui, H., Peng, J., Han, J.: Empower sequence labeling with task-aware neural language model. In: 32nd AAAI Conference on Artificial Intelligence, AAAI 2018, pp. 5253–5260. AAAI Press (2018)
21. Liu, S., Sun, Y., Li, B., Wang, W., Zhao, X.: HAMNER: headword amplified multi-span distantly supervised method for domain specific named entity recognition. In: AAAI, pp. 8401–8408 (2020)
22. Liu, T., Yao, J.G., Lin, C.Y.: Towards improving neural named entity recognition with gazetteers. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 5301–5307 (2019)
23. Liu, Y., Meng, F., Zhang, J., Xu, J., Chen, Y., Zhou, J.: GCDT: a global context enhanced deep transition architecture for sequence labeling. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 2431–2441 (2019)
24. Luo, Y., Xiao, F., Zhao, H.: Hierarchical contextualized representation for named entity recognition. In: AAAI, pp. 8441–8448 (2020)
25. Ma, X., Hovy, E.H.: End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In: ACL (1) (2016)
26. Mikheev, A., Moens, M., Grover, C.: Named entity recognition without gazetteers. In: Ninth Conference of the European Chapter of the Association for Computational Linguistics (1999)
27. Mikolov, T., Grave, E., Bojanowski, P., Puhersch, C., Joulin, A.: Advances in pre-training distributed word representations. In: Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018) (2018)
28. Nadeau, D., Turney, P.D., Matwin, S.: Unsupervised named-entity recognition: generating gazetteers and resolving ambiguity. In: Lamontagne, L., Marchand, M. (eds.) AI 2006. LNCS (LNAI), vol. 4013, pp. 266–277. Springer, Heidelberg (2006). https://doi.org/10.1007/11766247_23
29. Palshikar, G.K.: Techniques for named entity recognition: a survey. In: Bioinformatics: Concepts, Methodologies, Tools, and Applications, pp. 400–426. IGI Global (2013)
30. Pennington, J., Socher, R., Manning, C.D.: GloVe: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
31. Peters, M.E., et al.: Deep contextualized word representations. In: Proceedings of NAACL-HLT, pp. 2227–2237 (2018)
32. Qi, P., Zhang, Y., Zhang, Y., Bolton, J., Manning, C.D.: Stanza: a python natural language processing toolkit for many human languages. arXiv preprint [arXiv:2003.07082](https://arxiv.org/abs/2003.07082) (2020)
33. Ritter, A., Clark, S., Etzioni, O., et al.: Named entity recognition in tweets: an experimental study. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pp. 1524–1534 (2011)
34. Rocktäschel, T., Weidlich, M., Leser, U.: ChemSpot: a hybrid system for chemical named entity recognition. *Bioinformatics* **28**(12), 1633–1640 (2012)
35. Seti, X., Wumaier, A., Yibulayin, T., Paerhati, D., Wang, L., Saimaiti, A.: Named entity recognition in sports field based on a character-level graph convolutional network. *Information* **11**(1), 30 (2020)

36. Subramanian, S., Trischler, A., Bengio, Y., Pal, C.J.: Learning general purpose distributed sentence representations via large scale multi-task learning. In: International Conference on Learning Representations (2018)
37. Takeuchi, K., Collier, N.: Use of support vector machines in extended named entity recognition. In: COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002) (2002)
38. Tjong Kim Sang, E.F., De Meulder, F.: Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In: Daelemans, W., Osborne, M. (eds.) Proceedings of CoNLL-2003, Edmonton, Canada, pp. 142–147 (2003)
39. Yadav, V., Bethard, S.: A survey on recent advances in named entity recognition from deep learning models. In: Proceedings of the 27th International Conference on Computational Linguistics, pp. 2145–2158 (2018)
40. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: XLNet: generalized autoregressive pretraining for language understanding. In: Advances in Neural Information Processing Systems, pp. 5753–5763 (2019)
41. Yao, L., Mao, C., Luo, Y.: Graph convolutional networks for text classification. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 7370–7377 (2019)
42. Zhang, Y., Liu, Q., Song, L.: Sentence-state LSTM for text representation. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 317–327 (2018)