International Journal of Knowledge and Language Processing
Volume 13, Number 1, Dec **2022**

AKLP International ©2010 ISSN 2191-2734
pp. 1—**19**

A Review of Leveraging External Knowledge into Named Entity Recognition

Feng Liu
Peking University
No.5, Yiheyuan Road, Haidian District, Beijing, China
liufengpku@stu.pku.edu.cn

Received October 2022; revised October 2022

ABSTRACT. *When performing NER tasks on Chinese texts, the small-scale data, the diverse Chinese word constructions, and linguistic informality are often hindrances to the accuracy of the model. Compared to English, Chinese NER tasks often require additional feature engineering to get better results. Fortunately, it is possible to leverage external knowledge into NER models to improve performance. In this paper, we investigate and introduce common methods of infusing external knowledge into NER models. For each method, we describe the core idea as well as the key components of the model. Finally, we briefly introduce some future trends of NER.*

Keywords: Named Entity Recognition, Knowledge Graph, Self-Attention, Sequence Labelling

1. **Introduction.** Named entity recognition (NER) is a sub-task within the domain of information extraction. The goal of the task is to identify and classify relevant entities, such as people, places, organizations, etc. from the given unstructured text. Accordingly, Chinese named entity recognition is the named entity recognition task performed on Chinese text. NER is the fundamental task for many downstream tasks, such as information retrieval, relation extraction, reading comprehension, question answering systems, knowledge graphs, machine translation, etc. [27,32]

Since the term "named entity recognition" was proposed at the Sixth Message Understanding Conferences (MUC-6) in 1995 [8], corresponding developments have emerged. Many representative methods are proposed, including rule-based methods, unsupervised methods, supervised methods, statistical learning methods, and deep learning methods. However, the research around NER is still dominated by English, and the direct migration of these methods to Chinese text does not yield good results. The reason is that Chinese has more complex language features compared to English. Compared with English, Chinese does not have obvious spaces as word separators, so it is more difficult to determine word boundaries; moreover, modern Chinese texts are often written in mixed Chinese and English, which requires NER models to recognize not only Chinese entities

but also entities with mixed languages; in addition, Chinese entities themselves have features such as ambiguity and word formation flexibility that are hard to tackle, which add to the difficulty of named entities recognition tasks [32].

Therefore, when performing Chinese NER tasks, additional feature engineering is often needed to get better results. One of the common means is to leverage external knowledge to provide prior information to the task. This article investigates common methods to introduce external knowledge into Chinese NER tasks.

2. **NER-related background knowledge.** Before diving into detailed methods for infusing external knowledge, this section provides the background knowledge related to the NER task. Specifically, the definition of NER, commonly used methods, NER datasets, and evaluation criteria will be introduced.

2.1 **Name entity recognition.** The term “named entity (NE)” was first introduced at the Sixth MUC Conference in 1995. However, MUC-6 and the subsequent MUC-7 did not discuss and define what named entity is, but only stated that the entities to be labeled are "unique identifiers of entities". The conferences also specified the three major categories (named entities, time expressions, and quantitative expressions) and seven subcategories of entities to be identified in NER evaluation, among which named entities are divided into three categories: human names, organization names, and place names. After that, most conferences basically follow the definition and classification proposed at MUC, but the actual task is mainly to identify person names, place names, organization names, and other named entities. In addition to the mainstream NER review conferences, some scholars also consider named entities as proper names, as names of someone or something; or as objects of our interest that can be used to solve specific problems, etc. [8] Throughout the history of NER research, there is no official or universally accepted definition of named entities. Most NER tasks have focused on the recognition of names of people, places, organizations, and specific entities related to the task, so the specific scope of named entities can be customized according to the specific task. In this paper, we define named entity recognition as a task that aims to identify and classify components of text that represent named entities.

2.2 **Methods.** Methods of NER can be classified into three main categories based on its development process: rule-based methods, statistical machine learning-based methods, and deep learning-based methods.

(1) Rule-based methods

Early NER mainly used rule-based methods, where a limited number of rules were constructed manually and then strings matching these rules were found in the text. Some algorithms for automatic rule discovery and generation with the help of machines emerged at that time, such as the DLCoTrain algorithm which pre-defines the seed rule set like Decision List, and then performs unsupervised training on the rules based on the corpus to get more rules, and eventually, the algorithm has an accuracy rate of more than 91% on the

recognition of people, places, and organizations. Similarly, there are algorithms that use Bootstrapping for automatic rule generation. Rule-based algorithms can achieve high recognition results on a specific corpus, but they require human effort to develop a large number of rules, have an extreme dependence on domain knowledge, and cannot cope with the variability of named entities [8].

(2) Statistical machine learning-based methods

The statistical machine learning-based methods regard NER as a sequential labeling task. For each token in the text, there are several candidate labels, and the labels indicate the boundary and category of the corresponding entity which the word lies in. For instance, in the BIOES labeling system, B stands for a token that begins an entity, I stands for a token that is inside an entity, O stands for a token outside of any entity, E stands for a token that ends an entity, and S stands for a token that gets a single entity. By looking at NER tasks as sequential labeling tasks, we obtain entity boundaries and categories. Sequential labeling is currently the most common and effective NER method. Classical machine learning models such as HMM, ME, CRF, and SVM have been successfully applied to the NER task [8].

(3) Deep learning-based methods

In recent years, more and more NER tasks are turning to deep learning-based models. The use of word vectors, for instance, on one hand solves the problem of data sparsity brought by high-dimensional space, and on the other hand provides more semantic information about the word and context. The use of word vectors facilitates the infusion of other heterogeneous information under a uniform vector space [8], which brings more external knowledge to the NER task and yields significant improvements to the effectiveness of NER models. Common models include RNN, CNN, LSTM, BiLSTM-CRF, Self-Attention, and their combinations or variants.

2.3 **Datasets.** Datasets play an important role in model training and data evaluation process, and their data quality has a significant impact on the effectiveness of the models. TABLE 1 lists some of the most commonly used datasets in Chinese NER tasks, including their names, resource, entity types, etc.

MSRA was derived from news, and Weibo was derived from social media (Weibo) [33]. The OntoNotes datasets come from many domains and have different versions [34]. The Resume dataset is from Sina Finance and includes the resumes of executives of several Chinese listed companies [35]. The PKU dataset originated from the second SIGHAN bakeoff on Chinese word segmentation and named entity recognition. The E-commerce dataset is obtained by manually annotating text in Taobao, the largest e-commerce platform in China [36]. And the CLUENER dataset is a well-defined fine-grained dataset for named entity recognition in Chinese from CLUE organization (CLUENER2020).

TABLE 1. Common datasets used in NER tasks

Names	Resources	Entities	Websites

MSRA	News	3	/
Weibo	Social media	4	https://github.com/quincyliang/nlp-public-dataset/tree/master/ner-data/weibo
Resume	Sina finance	8	https://github.com/jiesutd/LatticeLSTM
OntoNotes	Magazines, news, webs	18	https://catalog.ldc.upenn.edu/LDC2013T19
PKU	competition	3	http://sighan.cs.uchicago.edu/bakeoff2005/
E-commerce	Taobao categories	2	https://github.com/PhantomGrapes/MultiDigraphNER
CLUENER	News	10	https://github.com/CLUEbenchmark/CLUENER2020

2.4 **Evaluation metrics.** NER tasks are evaluated mainly in two ways, exact match or relaxed match. Exact match metrics introduced by CoNLL consider a prediction to be correct only when the predicted boundary and category are both correct, simultaneously [38]. Relaxed match metrics consider a prediction to be correct as long as part of the named entity is identified correctly [7]. Since NER can be seen as mainly two subtasks: boundary detection and type recognition, exact match is more applicable.

In most NER tasks, precision, recall, and F1-score are used to evaluate model performance. Precision is defined as the number of entities a system predicted correctly (true positive) divided by the number that the system predicted (true positive and false positive). Recall was defined as the number of entities a system predicted correctly divided by the number of the ground truth (true positive and false negative). And the F1-score was defined as the harmonic mean of precision and recall [7]:

$$Precision = \frac{T_P}{T_p + F_p} \times 100\%$$

$$Recall = \frac{T_P}{T_p + F_N} \times 100\%$$

$$F1_{score} = \frac{2 \times Precision \times Recall}{Precision + Recall} \times 100\% \#(1)$$

3. **Common ways to leverage external knowledge into NER tasks.** The following introduces five common ways to infuse external knowledge: learning from Chinese character information; learning from Chinese word segmentation information; learning from context; learning from knowledge graph; and treating NER tasks as machine reading comprehension tasks.

3.1 **Learning from character information.** Chinese characters are pictographs, and the composition of a word (character), or the composition of a character (radical) often contains certain information, and such information often leads to greater utilization when training

data is sparse, or when the corpus is colloquial. Many deep learning models use character information for NER tasks. One of the representative models is ME-CNER, a simple yet effective neural framework that derives character-level embeddings with rich semantic information harnessed at multiple granularities ranging from radical, character to word levels [12].

The overall structure of ME-CNER is shown in FIGURE 1. The model uses three embeddings, namely character embedding, word embedding, and radical embedding. The embeddings are concatenated as the final representation of the characters, and then sent to a BiGRU-CRF model to label the sentence.

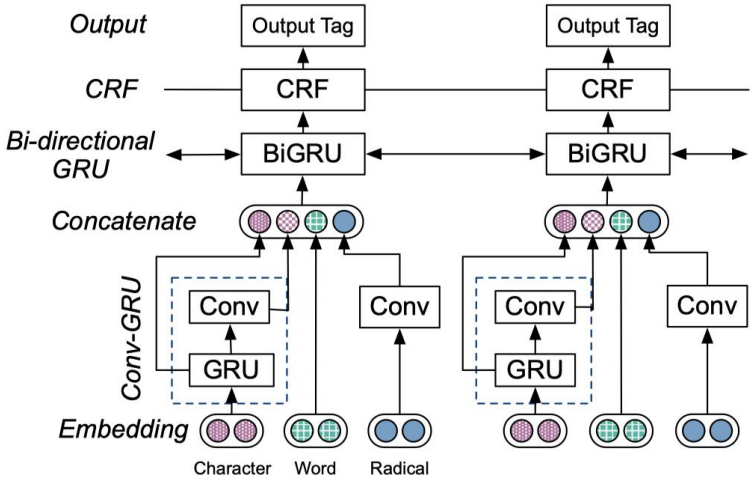

FIGURE 1. The structure of ME-CNER [12].

(1) Radical embedding

Radicals are the base component of Chinese characters, and radicals often carry certain implications, even when they're not in the same character [39]. By analyzing the radicals of Chinese characters, additional knowledge can often be obtained. For example, characters with radicals like "疒" are often related to disease and "尸" with death. These characters are rarely used in a context that has good meaning. On the contrary, characters with radicals including "钅", "木", "氵" and "火" are often related to metal, wood, water, or fire, respectively, and they appear very common in a person's name to meet the Wu-Xing theory (a Chinese folk belief).

ME-CNER uses a CNN network to extract local context features of the radical sequence, which also enables the model to better infer semantics of the characters that only appear in the test set. In other words, model tends to generalize well for rare characters correctly.

(2) Character embedding

First, the character embedding C_i is fed into the GRU layer. Then the output $\mathrm{X} = [x_1, \ldots, x_l]$ is fed into a convolutional layer padded to the same length as the input. Finally, the output of the convolutional layer is concatenated with the output of the GRU layer to form the final representation for each character.

$$X_i = GRU(C_1, \ldots, C_i)$$
$$Y = Conv(X)$$
$$Z_i = X_i \oplus Y_i \#(2)$$

In this way, semantic knowledge from both local context and long-term dependency are combined.

(3) Word embedding

Word embedding is the most common way to exploit high-level semantics. As a Chinese word is often a multi-characters, ME-CNER duplicates the word embedding for its constituent characters. For example, as an example shows in Figure 1, both component character "班" and "长" are aligned with a shared word embedding of "班长". Embedding is initialized with random values if the target word is not in the vocabulary. FIGURE 2 shows a live example of how the model works.

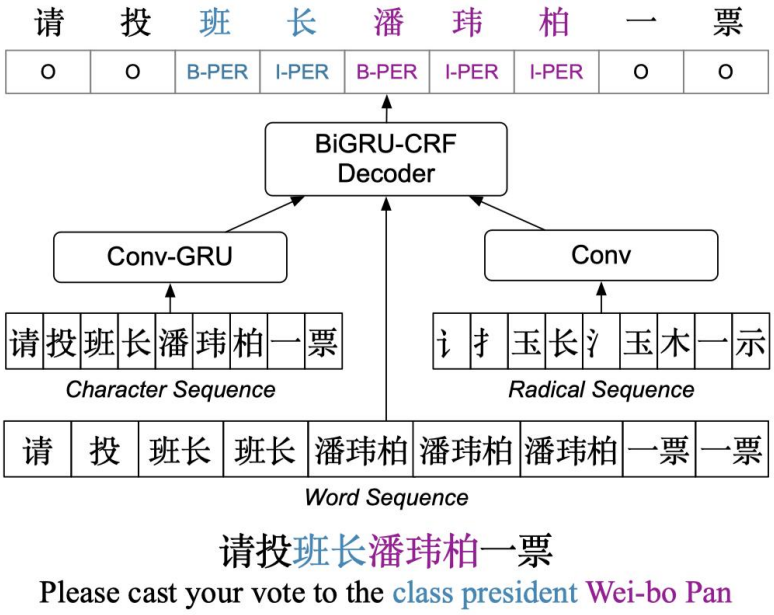

FIGURE 2. An example of the labeling process. [12].

3.2 **Learning from word segmentation information.** Chinese word segmentation (CWS) aims to identify the boundaries of words in a sentence. The CWS task has more annotated data compared to NER, and there is some similarity between the two tasks. As shown in FIGURE 3, the two tasks often share the same word boundaries.

Task	Hilton ↑ 希尔顿			leaves ↑ 离开		Houston ↑ 休斯顿			Airport ↑ 机场	
Chinese NER	希 B-PER	尔 I-PER	顿 I-PER	离 O	开 O	休 B-LOC	斯 I-LOC	顿 I-LOC	机 I-LOC	场 I-LOC
CWS	希 B	尔 I	顿 E	离 S	开 S	休 B	斯 I	顿 E	机 B	场 E

FIGURE 3. Chinese NER and CWS tasks often share the same word boundaries [10].

Cao [10] proposed an adversarial migration model to train both NER and CWS models to mine the shared information between the two tasks, to achieve the effect of using a large amount of CWS data to improve the NER task. As shown in FIGURE 4, the model consists of five parts: embedding layer, shared-private feature extractor, self-attention, task-specify CRF, and task discriminator. Detailed information of each component is described below.

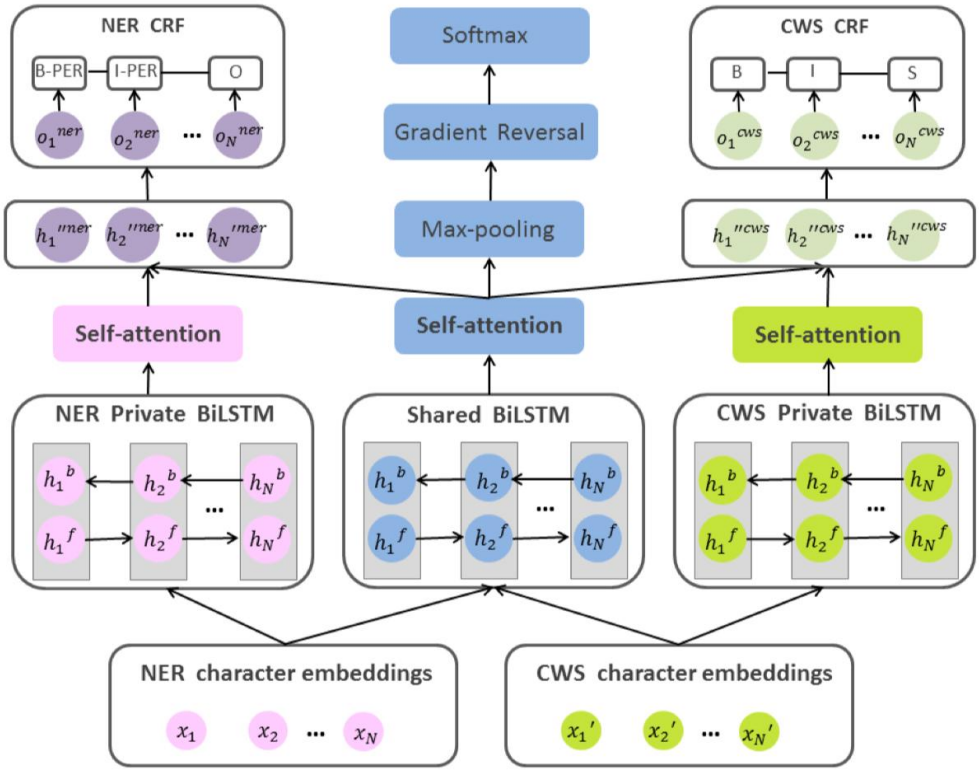

FIGURE 4. The overall structure of the model [10].

(1) Embedding layer

For a given Chinese sentence $X = \{C_1, \ldots, C_N\}$, embedding vector is retrieved from pre-trained embedding matrix for each character C_i.

(2) Shared-private feature extractor

The model uses BiLSTM to extract features. The hidden state of BiLSTM could be

modeled as follows:

$$\vec{h_i} = \vec{LSTM}(\vec{h_{i-1}}, x_i)$$
$$\overleftarrow{h_i} = \overleftarrow{LSTM}(\overleftarrow{h_{i+1}}, x_i)$$
$$h_i = \vec{h_i} \oplus \overleftarrow{h_i} \#(3)$$

As shown in FIGURE 4, the model proposed a shared-private feature extractor along with private feature extractors. The private feature extractor in both CWS and NER tasks is used to extract task-specific features, while the shared BiLSTM layer is used to learn task-shared word boundaries. Specifically, for any sentence in CWS or NER task, the hidden states of shared and private BiLSTM layer can be obtained as follows:

$$s_i^k = BiLSTM(x_i^k, s_{i-1}^k; \theta_s)$$
$$h_i^k = BiLSTM(x_i^k, h_{i-1}^k; \theta_k) \#(4)$$

where θ_s and θ_k are the shared parameters and private parameters respectively.

(3) Self-attention

The model utilizes self-attention mechanism to learn the dependencies between characters in a sentence and to capture inner structure information. Specifically, the model uses multi-head self-attention. Let $H = \{h_1, h_2, ..., h_N\}$ denote the output of private BiLSTM and $S = \{s_1, s_2, ..., s_N\}$ denote the output of shared BiLSTM. The scaled dot-product attention can be described as follows:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d}})V \#(5)$$

where Q, K, V are query matrix, keys matrix, and value matrix, respectively.

Accordingly, the multi-head attention can be described as follows:

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$
$$H' = (head_i \oplus ... \oplus head_h)W_o \#(6)$$

where W_i^Q, W_i^K and W_i^V are trainable projection parameters. W_o is also trainable parameter.

(4) Task-specific CRF

The final representation of a sentence is obtained by concatenating the representations from private space and shared space after self-attention layer:

$$H'^k = H'^k \oplus S'^k \#(7)$$

where H'^k and S'^k are the outputs of private self-attention and shared self-attention structure, respectively.

To model dependencies between successive labels, the author exploits CRF [42] to infer tags instead of using h^{ii} directly. Each task is assigned a specific CRF layer. Given a sentence x with a predicted tag sequence y, the CRF tagging procedure can be described

as follows:

$$o_i = W_s h_i'' + b_s$$

$$s(x, y) = \sum_{i=1}^{N} (o_{i,y_i} + T_{y_{i-1}, y_i})$$

$$\bar{y} = \underset{y \in Y_x}{\operatorname{argmax}}\, s(x, y) \quad \#(8)$$

where $W_s \in \mathbb{R}^{|T| \times 4d_h}$ and $b_s \in \mathbb{R}^T$ are trainable parameters. $|T|$ denotes the number of output labels. o_{i,y_i} represents the score of the y_i-th tag of the character c_i. T is a transition score matrix. Y_x represents all candidate tag sequences for a given sentence. Viterbi algorithm is used in the decoding process to get the predicted tag sequence $\bar{y}$.

For training, the author uses negative log-likelihood object as the loss function and gradient back-propagation method to minimize the loss function. The procedure is described as follows:

$$p(\hat{y}|x) = \frac{e^{s(x,\hat{y})}}{\sum_{\tilde{y} \in Y_x} e^{s(x,\tilde{y})}}$$

$$L_{\text{Task}} = -\sum_{i=1}^{T} \log p(\hat{y}^{(i)}|\boldsymbol{x}^{(i)}) \quad \#(9)$$

where $\hat{y}$ is the ground-truth label sequence. $(x^{(i)}; \hat{y}^{(i)})$ is a pair of training examples. T is the number of training examples.

(5) Task Discriminator

To guarantee that specific task features do not exist in share space, the model incorporates adversarial training. Inspired by adversarial networks[43], the author proposed a task discriminator to determine which task the sentence comes from. The procedure can be expressed as follows:

$$s'^{\text{k}} = Maxpooling(S'^{\text{k}})$$

$$D(s'; \theta_d) = softmax(W_d s'^{\text{k}} + b_d) \quad \#(10)$$

where θ_d is the parameters. $W_d \in \mathbb{R}^{K \times 2d_h}$ and $b_d \in \mathbb{R}^K$ are trainable parameters. And K is the number of tasks.

Besides the task loss L_{Task}, the model introduced an adversarial loss L_{Adv} to prevent specific features of CWS task from leaking into share space. The adversarial loss can be described as follows:

$$L_{Adv} = \min_{\theta_s} (\max_{\theta_d} \sum_{k=1}^{K} \sum_{i=1}^{T_k} \log D(E_s(\boldsymbol{x}_{\boldsymbol{k}}^{(\boldsymbol{i})}))) \quad \#(11)$$

where θ_s is the trainable parameters of shared BiLSTM. E_s denotes the shared feature

extractor. T_k is the number of training examples. $x_{k^{(i)}}$ is the i-th example of task.

To address the minimax optimization problem, a gradient reversal layer [44] is added below the softmax layer. More detail can be found in [10,44].

3.3 **Learning from context.** In scenarios such as chat, search, and social media, the text is characterized by randomness and informality. The extensive use of abbreviations and verbal vocabularies in scattered texts often brings more challenges to the NER task. Jung [3] refers to the text generated by users in social scenarios as microtext, and proposes that when performing NER on a microtext, the microtext cluster of related texts can be used to introduce more contextual information and thus improve performance. The key to the method is the evaluation of semantic relations between microtexts and the generation of microtext clusters. More details are described below.

(1) Preliminaries

Microtext. A microtext t_i is a text that has three features: (i) word frequencies; (ii) timestamps; (iii) a set of neighbors (who have published microtexts related to current microtext).

Once a microtext is given, word features of the microtext can be represented by:

$$TF(t_i) = [w_1, w_2, \ldots, w_N]^T \#(12)$$

Contextual association. Given two microtexts t_i and t_j, the contextual association between them can be represented as:

$$C(t_i, t_j) = \sum_{e \in E} \alpha_e C_e(t_i, t_j)$$
$$\sum_{e \in E} \alpha_e = 1 \#(13)$$

where E is the set of any possible heuristics, and α_e is the normalizing coefficient.

Social network. A social network S describes the relationship between a set of users $\mathcal{E}$:

$$S = < \mathcal{E}, \mathcal{N} > \#(14)$$

where $\mathcal{E}$ represents a set of unique IDs, and $\mathcal{N}$ represents a set of relations between two entities in $\mathcal{E}$.

To discover contextual relationships between microtexts, the author proposed three heuristics, described as follows:

Semantic association. Semantic association C_O measures the similarity between two sets of word features. When using word frequency as feature, the process can be expressed as:

$$C_O(t_i, t_j) = \frac{|TF(t_i) \cap TF(t_j)|}{\max(TF(t_i), TF(t_j))} \#(15)$$

where the intersection operation can be done using string matching.

Temporal association. Temporal association C_T describes how close two microtexts

are to time:

$$\mathcal{C}_T(t_i, t_j) = \frac{\kappa_T}{\exp(|\pi_j - \pi_i|)} \#(16)$$

where κ_T is a delaying coefficient.

Social association. Social association $\mathcal{C}_S$ describes the frequency of communication between relevant users:

$$\mathcal{C}_S(t_i, t_j) = \frac{\kappa_S}{\exp(path(u_i, u_j))} \#(17)$$

where κ_T is a delaying coefficient, and the path is a function that returns the length of the shortest path between relevant users from social network described earlier.

(2) Forming microtext cluster

Given a microtext t_i, a microtext cluster $\mathcal{T}$ is a set of microtexts that are contextually associated with t_i. The formula can be expressed as follows:

$$\mathcal{T}(t_i) = \{t_j | \mathcal{C}(t_i, t_j) \geq k\} \#(18)$$

where k is the threshold.

Once the microtext clusters are built, the NER model can be trained on them instead of independent texts. A case study on Twitter shows that the use of microtexts improved the accuracy by 5.4%.

3.4 **Learning from knowledge graph.** When performing domain-specific NER tasks, the scale of labeled data often limits model performance. Since structured knowledge often exists in a particular field, scholars tend to infuse this external knowledge into NER model to improve accuracy. One common approach is to use knowledge graphs. With the development of knowledge graph embedding technology [6], it is common to integrate knowledge graph embeddings with deep learning models. In 2020, He et al. [5] proposed a method to combine knowledge graph embedding with self-attention mechanism when doing NER tasks on Chinese marine text. As shown in FIGURE 5, the model consists of three layers: a word-level embedding layer to encode the original data; a context sequence encoding layer to incorporate context information and knowledge graph information; and a CRF decoding layer to decode the information. More details are described below.

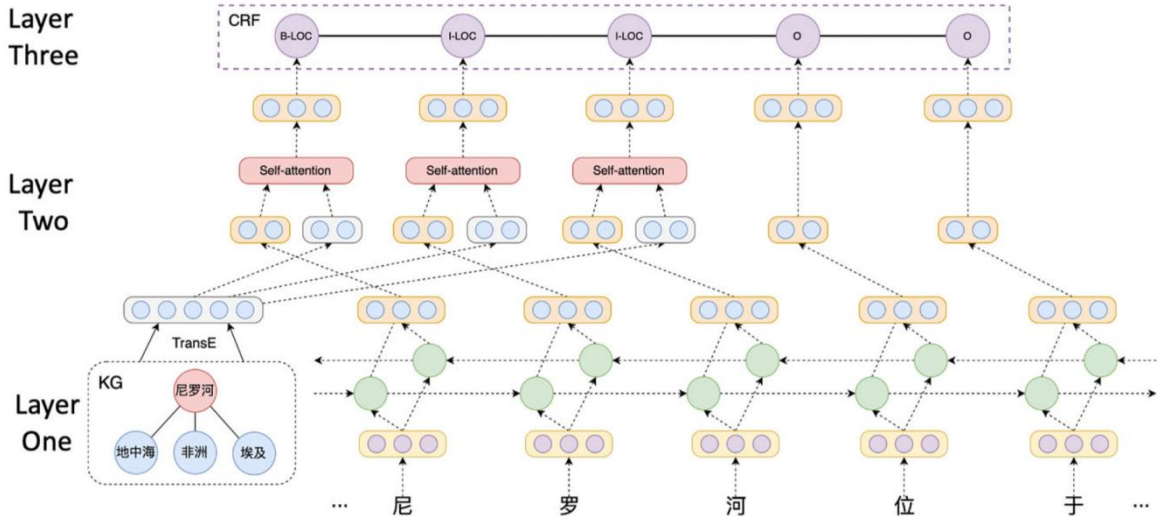

FIGURE 5. Incorporating knowledge graph embeddings and self-attention with BiLSTM-CRF model [5].

(1) Word-level embedding layer

The first layer of the model is a word embedding layer. The original data is transformed into a sequence of vectors by looking up the word vector table. The author pre-trained the vector table on the Chinese Wikipedia corpus using Glove [45]. For out-of-vocabulary words, a fixed dimension vector with random values is used.

(2) Context sequence encoding layer

Once the word vectors are extracted by word embedding layer, the output is sent to a Bi-LSTM[47] layer to encode context information. LSTM[46] is a variant of RNN which tackles the problem of exploding gradient and vanishing gradient of traditional RNN, and can better model long-distance relationships. Bi-LSTM consists of a forward LSTM and a backward LSTM. By combining two LSTM layers from both directions, Bi-LSTM introduces information from both past and present at each time step and therefore better models contextual information.

The LSTM introduces a gating mechanism to control the memory process, specifically: 1) a forget gate f to forget some information about the past; 2) an input gate i to remember some information at the current time step; 3) an output gate o to output the final information obtained after previous changes. The detailed calculation process of each unit at each time t can be expressed as follows:

$$f_t = \sigma\left(W_f\left[h_{t-1}, x_t\right] + b_f\right)$$
$$i_t = \sigma\left(W_i\left[h_{t-1}, x_t\right] + b_i\right)$$
$$\tilde{C}_t = \tan\left(W_c\left[h_{t-1}, x_t\right] + b_C\right)$$
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$
$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tan(C_t) \#(19)$$

where x_t is the input information at time t and h_t is the output of the LSTM. W_f, W_i, W_o and b_f, b_i, b_o are the weight matrixes and biases of the forget gate, input gate, and output gate, respectively. σ represents the sigmoid function and $*$ represents the element-wise product.

To incorporate auxiliary information for the task, the entity embeddings of knowledge graph are employed. Given a set of triplets $S = (h, l, t)$, where $h, t \in E$ are entities and $l \in L$ is a relationship, following an energy-based framework, the goal of TransE is to minimize the margin-based ranking criterion:

$$L = \sum_{(h,l,t) \in S} \sum_{(h',l,t') \in S'_{(h,l,t)}} [\gamma + d(h + l, t) - d(h' + l, t')]_+ \#(20)$$

where d is a dissimilarity measuring function which is either L_1 or the L_2-norm. $\gamma > 0$ is a margin hyperparameter. $[x]_+$ denotes the positive part of x. $S'_{(h,l,t)} = \{(h', l, t) | h' \in E\} \cup \{(h, l, t') | t' \in E\}$. After the training process, embeddings will have this feature: the closer two entities on the knowledge graph are, the closer the embedding distance is.

After character vector is obtained by Bi-LSTM and TransE, the Self-Attention mechanism is used to further extract text features. Specifically, the model used multi-head attention to extract features at different levels: three same outputs from the concatenation of Bi-LSTM hidden layer and TransE embedding are sent to attention structure and then the information vector y_i is obtained. Before decoding, h_i and y_i are added according to certain weights to form the information vector z_i, which combines the context features with its own structure features. The process is described as follows:

$$h_i = h_{fw} // h_{bw} // h_{kg}$$
$$y_i = \text{multi\_head}(h_i, h_i, h_i)$$
$$z_i = \lambda \cdot h_i + (1 - \lambda) \cdot y_i \#(21)$$

where $h_{fw}, h_{bw}, h_{hg}, h_i$ and $y_i \in R^{d_h}$, d_h is the dimensionality of the embeddings, and the symbol $//$ represents the splicing of two vectors. The h_{fw} and h_{bw} represent the outputs of forward LSTM and backward LSTM, respectively. The h_{kg} represents the knowledge graph embedding of entity generated by TransE.

(3) CRF decoding layer

After information vector is generated by Bi-LSTM and self-attention, it is sent to a CRF layer to decode the information. The CRF tagging process is the same as described in 3.2, thus it will not be repeated here.

3.5 **NER as MRC.** In 2020, Li et al. [14] proposed a unified NER framework. Instead of

treating the task of NER as a sequence labeling problem, the team proposed to formulate it as a machine reading comprehension (MRC) task. For example, extracting entities with the label PERSON is formulated as extracting the answer span to the question "which person is mentioned in the text". This transformation naturally tackles the nested entity issue, in which more than one entity covers the same token (FIGURE 6 shows an example). Since sequence labeling model only predicts one label for each token, it's impossible to predict both/all correct entities. MRC approach easily solves this problem by answering two questions. Additionally, since the query carries prior information about the entity to extract, this strategy facilitates the extraction process, leading to better performance at NER tasks.

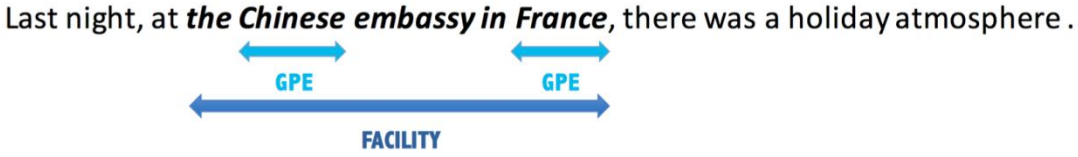

FIGURE 6. An example of nested entities [14].

The key parts of the method are described below.

(1) Task formalization

Given a sequence $X = x_1, \ldots, x_n$, where n is the length of the sequence, the goal is to find every entity in X, and then assign a label $y \in Y$ to it, where Y is the set of all possible entity types.

To solve NER problem in MRC style, the dataset needs to be transformed first. For each annotated entity in the original dataset, we need to obtain $(q_y, x_{start,end}, X)$, where $q_y = q_1, \ldots, q_m$ is the natural question associated with entity type y. $x_{start,end} = x_{start}, x_{start+1}, \ldots, x_{end-1}, x_{end}$ is a substring of X. And X is the input sequence. Through this transformation, $(QUESTION, ANSWER, CONTEXT)$ triplets that the MRC tasks need can be obtained.

(2) Query generation

The question-generation procedure is vital since the query contains prior information about the context and label. There are different ways to generate questions. Li [40] utilizes a template-based procedure in question generation to capture semantic information. Li [14] takes human annotation guidelines as references to generate queries. Annotation guidelines are notes provided by dataset builders, which are usually descriptions of tag catteries. They're explicit and concise so that annotators can annotate the corpus without any ambiguity. TABLE 2 shows an example of designed questions.

TABLE 2. An example of designed questions.

Entity	Question
Location	Find locations in the text, including nongeographical locations, mountain ranges

	and bodies of water.
Facility	Find facilities in the text, including buildings, airports, highways and bridges.
Organization	Find organizations in the text, including companies, agencies and institutions.

(3) Model details

The model utilizes BERT[41] as the backbone. The question q_y and the passage X are concatenated to format the input string $\{[CLS], q_1, q_2 \ldots, q_m, [SEP], x_1, x_2, \ldots, x_n\}$, the string is then sent to BERT and a context representation matrix $E \in \mathbb{R}^{n \times d}$ is generated.

For span selection, the MRC model uses two binary classifiers, one to predict whether each token is the start index or not, and the other to predict whether each token is the end token or not. This strategy has the potential to extract all possible entities in a context according to q_y. Specifically, given the representation matrix E from BERT, the model predicts the probability of each token being a start index as follows:

$$P_{start} = softmax_{each\,row}(E \bullet T_{start}) \in \mathbb{R}^{n \times 2} \#(22)$$

where $T_{start} \in \mathbb{R}^{d \times 2}$ is the weights to learn, and each row of P_{start} denotes the probability of each index being the start position of an entity according to the given query. The end index prediction uses the same procedure, except that another weights $T_{end} \in \mathbb{R}^{d \times 2}$ is learned to predict probability matrix P_{end}.

A problem brought by the two binary classifiers strategy is that in a given context, there might be multiple entities of the desired target, which share the same category. This means that multiple start indexes and end indexes might be found in a context. An instinct is to match a start index with its nearest end index; however, this does not work since overlap entities exist. The author uses a binary classification model to predict whether a start index i_{start} and an end index i_{end} should be matched:

$$P_{i_{start}, i_{end}} = sigmoid\,(m \cdot concat\,(E_{i_{start}}, E_{i_{end}})) \#(23)$$

where $m \in \mathbb{R}^{1 \times 2d}$ is the weights.

4 **Future trends.** Although the introduction of additional knowledge can improve the performance of the model, the NER task still faces many problems.

(1) Scarce and imbalanced labeling data

In real-world tasks, annotated data is often sparse and imbalanced. Domain-specific data often requires experts to annotate, which is costly and labor-intensive. In addition, there are no clear standards for how to label corpus, which makes it more difficult to obtain labeled data. Furthermore, imbalanced labeling data is common in many fields, such as e-commerce, where there are often only a few positive samples and a large amount of unlabeled data. In scenarios like this, methods like zero-shot learning, few-shot learning,

positive and unlabeled learning, and prompt learning, etc. are often adopted.

(2) The high cost of training a model

The introduction of large-scale pre-trained language models, such as BERT and GPT, tends to significantly improve the performance of NER tasks. However, their training process is very resource-intensive, and few users or institutions could afford the cost of training such models. Optimization of the complexity of such pre-trained models can be considered in the future to introduce lightweight and less resource-consuming models.

(3) Domain-shared or task-shared information is not yet fully utilized

Chinese NER has performed well in specific domains, but the model trained on one dataset often does not work well when applied to another [48]. Further research is needed on how to introduce transfer learning into NER tasks and improve model's generalization ability. In addition, NER tasks can be learned jointly with other tasks [32], for example, Name Entity Linking, Word Segmentation, Relation Extraction, etc. This is often regarded as multi-task learning, which is also an important research direction.

5 **Conclusion.** When labeling entities in Chinese corpus, the accuracy of the model can be improved by introducing additional knowledge in the face of sparse data, diverse Chinese word constructions, and linguistic informality. In this paper, we present the background knowledge of NER and introduce five common ways to leverage external knowledge into Chinese NER tasks. We also describe some future trends in the NER field. We hope this paper will help readers understand common ways to leverage external knowledge into NER tasks and inspire more excellent work.

REFERENCES

[1] Singh, V., Vijay, D., Akhtar, S. S., & Shrivastava, M. (2018, July). Named entity recognition for hindi-english code-mixed social media text. In *Proceedings of the seventh named entities workshop* (pp. 27-35).

[2] Liu, A., Du, J., & Stoyanov, V. (2019). Knowledge-augmented language model and its application to unsupervised named-entity recognition. *arXiv preprint arXiv:1904.04458*.

[3] Jung, J. J. (2012). Online named entity recognition method for microtexts in social networking services: A case study of twitter. *Expert Systems with Applications*, 39(9), 8066-8070.

[4] Dadas, S. (2019, June). Combining neural and knowledge-based approaches to named entity recognition in polish. In *International Conference on Artificial Intelligence and Soft Computing* (pp. 39-50). Springer, Cham.

[5] He, S., Sun, D., & Wang, Z. (2022). Named entity recognition for Chinese marine text with knowledge-based self-attention. *Multimedia Tools and Applications*, 81(14), 19135-19149.

[6] Wang, Q., Mao, Z., Wang, B., & Guo, L. (2017). Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12), 2724-2743.

[7] Yadav, V., & Bethard, S. (2019). A survey on recent advances in named entity recognition from deep learning models. *arXiv preprint arXiv:1910.11470*.

[8] Liu, W. (2018). A Review on Named Entity Recognition. *Journal of the China Society for Scientific and*

Technical Information, 37(3), 329-340.

[9] Su, F., Sun, C., & Jing, N. (2022). A Context-Fusion Method for Entity Extraction Based on Residual Gated Convolution Neural Network. *Beijing Da Xue Xue Bao*, 58(1), 69-76.

[10] Cao, P., Chen, Y., Liu, K., Zhao, J., & Liu, S. (2018). Adversarial transfer learning for Chinese named entity recognition with self-attention mechanism. In *Proceedings of the 2018 conference on empirical methods in natural language processing* (pp. 182-192).

[11] Sui, D., Chen, Y., Liu, K., Zhao, J., & Liu, S. (2019, November). Leverage lexical knowledge for Chinese named entity recognition via collaborative graph network. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (pp. 3830-3840).

[12] Xu, C., Wang, F., Han, J., & Li, C. (2019, November). Exploiting multiple embeddings for chinese named entity recognition. In *Proceedings of the 28th ACM international conference on information and knowledge management* (pp. 2269-2272).

[13] Ding, N., Xu, G., Chen, Y., Wang, X., Han, X., Xie, P., ... & Liu, Z. (2021). Few-nerd: A few-shot named entity recognition dataset. *arXiv preprint arXiv:2105.07464*.

[14] Li, X., Feng, J., Meng, Y., Han, Q., Wu, F., & Li, J. (2019). A unified MRC framework for named entity recognition. *arXiv preprint arXiv:1910.11476*.

[15] Fritzler, A., Logacheva, V., & Kretov, M. (2019, April). Few-shot classification in named entity recognition task. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing* (pp. 993-1000).

[16] Tong, M., Wang, S., Xu, B., Cao, Y., Liu, M., Hou, L., & Li, J. (2021). Learning from miscellaneous other-class words for few-shot named entity recognition. *arXiv preprint arXiv:2106.15167*.

[17] Akiti, C. (2021). Efficient Few-Shot Learning For Named Entity Recognition.

[18] Tu, Y., Liu, & S. (2017). Network representation learning: an overview (in Chinese). *Sci Sin Inform, 47: 980–996*, doi: 10.1360/N112017-00145.

[19] Zhou, Z., Li. (2019). Survey on representation learning methods oriented to heterogeneous information network[J]. *Journal of Frontiers of Computer Science and Technology, 13(7)*: 1081-1093.

[20] Yin, J., Huang., R. (2019). Research and development of network representation learning[J]. *Chinese Journal of Network and Information Security, 5(2)*: 77-87.

[21] Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*.

[22] Daiber, J., Jakob, M., Hokamp, C., & Mendes, P. N. (2013, September). Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th international conference on semantic systems* (pp. 121-124).

[23] Mendes, P. N., Jakob, M., García-Silva, A., & Bizer, C. (2011, September). DBpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th international conference on semantic systems* (pp. 1-8).

[24] Liu, L., & Yu. (2019). Named Entity Recognition Using Linked Data. *Journal of the China Society for Scientific and Technical Information*, 38(2), 191-200.

[25] Li, L., & Lin. (2021). Review of Transfer Learning for Named Entity Recognition. *Journal of Frontiers of Computer Science and Technology*, 15(2), 206-218.

[26] Nebhi, K. (2012, December). Ontology-based information extraction from twitter. In *Proceedings of the Workshop on Information Extraction and Entity Analytics on Social Media Data* (pp. 17-22).

[27] Zhao, L., & Cai. (2022). Survey of Chinese Named Entity Recognition. *Journal of Frontiers of Computer Science and Technology*, 16(2), 296.

[28] Sheng. (2019). Transfer learning in named entity recognition. (Master's thesis, Harbin Institute of Technology).

[29] Zhang. (2019). Research on the key technology of scientific briefing automatic generation based on network representation learning. (Master's thesis, Peking University).

[30] Zhao. (2021). Generating math word problems based on deep learning. (Master's thesis, Peking University).

[31] Wang, D., Cui, P., & Zhu, W. (2016, August). Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 1225-1234).

[32] Zhang, D., Wang, & G. (2022). Recent advances of Chinese named entity recognition based on deep learning. *Journal of Chinese information processing*, 36(6), 20-35.

[33] Peng, N., & Dredze, M. (2015, September). Named entity recognition for chinese social media with jointly trained embeddings. In *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 548-554).

[34] Weischedel, R., Pradhan, S., Ramshaw, L., Palmer, M., Xue, N., Marcus, M., ... & Houston, A. (2011). Ontonotes release 4.0. *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*.

[35] Zhang, Y., & Yang, J. (2018). Chinese NER using lattice LSTM. *arXiv preprint arXiv:1805.02023*.

[36] Ding, R., Xie, P., Zhang, X., Lu, W., Li, L., & Si, L. (2019, July). A neural multi-digraph model for Chinese NER with gazetteers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (pp. 1462-1467).

[37] Xu, L., Dong, Q., Liao, Y., Yu, C., Tian, Y., Liu, W., ... & Zhang, X. (2020). CLUENER2020: fine-grained named entity recognition dataset and benchmark for chinese. *arXiv preprint arXiv:2001.04351*.

[38] Sang, E. F., & De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

[39] Ho, C. S. H., Ng, T. T., & Ng, W. K. (2003). A "radical" approach to reading development in Chinese: The role of semantic radicals and phonetic radicals. *Journal of literacy research, 35*(3), 849-878.

[40] Li, X., Yin, F., Sun, Z., Li, X., Yuan, A., Chai, D., ... & Li, J. (2019). Entity-relation extraction as multi-turn question answering. *arXiv preprint arXiv:1905.05529*.

[41] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[42] Lafferty, J., McCallum, A., & Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

[43] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM, 63*(11), 139-144.

[44] Ganin, Y., & Lempitsky, V. (2015, June). Unsupervised domain adaptation by backpropagation. In *International conference on machine learning* (pp. 1180-1189). PMLR.

[45] Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543).

[46] Hochreiter, S., & Schmidhuber, J. (1996). LSTM can solve hard long time lag problems. *Advances in neural information processing systems, 9.*

[47] Huang, Z., Xu, W., & Yu, K. (2015). Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

[48] Kang, S., Zhu, & L. (2022). A survey on Chinese named entity recognition with deep learning. *Journal of Huazhong University of Science and Technology(Natural Science Edition)*.