Jiang *et al*. *Intell Robot* 2023;3(3):337-54
**DOI:** 10.20517/ir.2023.21

Intelligence & Robotics

Research Article

Open Access

# Discrete sequence rearrangement based self-supervised chinese named entity recognition for robot instruction parsing

**Cong Jiang[1], Qingyang Xu[1], Yong Song[1], Xianfeng Yuan[1], Bao Pang[1], Yibin Li[2]**

[1]School of Mechanical, Electrical & Information Engineering, Shandong University, Weihai 264209, Shandong, China.
[2]School of Control Science and Engineering, Shandong University, Jinan 250061, Shandong, China.

**Correspondence to:** Dr. Qingyang Xu, School of Mechanical, Electrical & Information Engineering, Shandong University, Weihai Wenhua Xilu No.180, Weihai 264209, Shandong, China. E-mail: qingyangxu@sdu.edu.cn

**How to cite this article:** Jiang C, Xu Q, Song Y, Yuan X, Pang B, Li Y. Discrete sequence rearrangement based self-supervised chinese named entity recognition for robot instruction parsing. *Intell Robot* 2023;3(3):337-54. http://dx.doi.org/10.20517/ir.2023.21

**Received:** 25 Apr 2023 **First Decision:** 25 Jun 2023 **Revised:** 30 Jun 2023 **Accepted:** 1 Jul 2023 **Published:** 25 Jul 2023

**Academic Editor:** Simon X. Yang **Copy Editor:** Yanbin Bai **Production Editor:** Yanbin Bai

## Abstract

Named entity recognition (NER) plays an important role in information extraction tasks, but most models rely on large-scale labeled data. Getting the model to move away from large-scale labeled datasets is challenging. In this paper, a SCNER (Self-Supervised NER) model is proposed. The BiLSTM (Bidirectional LSTM) is adopted as the named entity extractor, and an Instruction Generation Subsystem (IGS) is proposed to generate "Retelling Instructions", which analyzes the similarities between the input instructions and "Retelling Instructions" as the losses for model training. A series of rules based on traditional learning rules have been proposed for discrete forward computation and error backpropagation. It mimics language learning in human infants and constructs a SCNER model. This model is used for robot instruction understanding and can be trained on unlabeled datasets to extract named entities from instructions. Experimental results show that the proposed model is competitive with the supervised BiLSTM-CRF and BERT-NER models. In addition, the model is applied to a real robot, which verifies the practicality of SCNER.

**Keywords:** Chinese named entity recognition, self-supervised, robotics, discrete sequence rearrangement

## 1. INTRODUCTION

One of the core tasks of robot control is to parse instructions, extract useful information, and then drive the robot to move accordingly[1]. Named entity recognition (NER) is one of the main text information extraction methods[2]. Named entity is defined as the attribute name of a real object[3], such as a Person, Organization, Location, etc. The accurate extraction of named entities from texts is fundamental to understanding the intrinsic meaning of a text. As a basic task, it is the foundation of many natural language processing (NLP) applications[4], such as relationship extraction[5], text understanding[6], information extraction[7], machine translation[8], entity corpus construction[9], etc. There are mainly three types of NER models: rule-based learning methods[10], unsupervised learning methods[11], and feature-based supervised learning methods[12]. Currently, supervised models dominate the NER task, but most of these models rely on large-scale, manually labeled datasets for training, which is often costly for dataset aggregation. Unsupervised or self-supervised based NER does not require labeled datasets and only relies on a small amount of labeled data. However, the main challenge lies in how to provide accurate learning direction or classification basis for model training.

In the early unsupervised model studies of NER, some researchers have attempted to address the above issues. There are two main solutions proposed by them: One is to build a common dictionary with a small amount of known data, and these data are used as the clustering center to provide a classification basis for the model[13]. He *et al.* adopt the knowledge-based self-attention for Chinese NER[14]. The other aims to construct "seed" rules as the classification standard of words and provide the basis of clustering for the model. The "seed" rules contain prior information, such as grammatical information or special prompt words. After determining the clustering centers or classification bases based on prior information, these two types of models are often used to extract named entities from unlabeled data by computing the similarity of lexical contexts to analyze the data structure and distribution characteristics. Wang *et al.* adopt enhanced dictionary semantic knowledge for Chinese NER[15]. The pre-trained language models are also used for Chinese NER[16]. It is worth noting that no matter which method is used, the core step is mostly the Coarse-Grained information extraction of named entities using retrieval or pattern matching[17]. Current mainstream unsupervised NER methods can be divided into discriminative and generative models. The discriminative model is based on the traditional method and constructs more reasonable rules to extract named entities[18]. The generative model aims to achieve the optimal subdivision of the generated entity category with the highest probability through model design.

Thanks to the efforts of researchers, several breakthroughs have been achieved in unsupervised NER. However, in the task of Chinese named entity extraction, the development is relatively slow because sentences do not have obvious word boundaries. The generative model SLMs (Segmental Language Models) proposed by Sun *et al.* achieves about 75% extraction accuracy on multiple datasets[11,19], which is a breakthrough in the research of Chinese NER. Moreover, since unsupervised models need to incorporate sufficient contextual information, it is often not possible to use unsupervised word segmentation in some applications, such as robotic language parsing, due to the limited contextual information of concise instructions. Therefore, learning and understanding languages as efficiently and accurately as humans is a challenge for robots.

In this paper, we propose a Self-Supervised NER (SCNER) for robot instruction parsing. The aim is to free the model from complex parameter training and feature presetting, rule construction, and reliance on large-scale, manually labeled datasets. As shown in Figure 1, after the robot gets the input instructions, the word segmentation is extracted based on the NER model, and these word segmentations are used to construct "Retelling Instructions". Since the model is untrained, the named entities extracted by the model at the beginning are mostly incorrect, and the "Retelling Instructions" constructed based on these named entities are also far from the input instructions. However, the model can "understand" the input instruction after simple training according to the Loss; that is, the model can extract the named entity correctly and construct the "Retelling Instructions" consistent with the content of the input instruction. Based on these correctly named entities, we

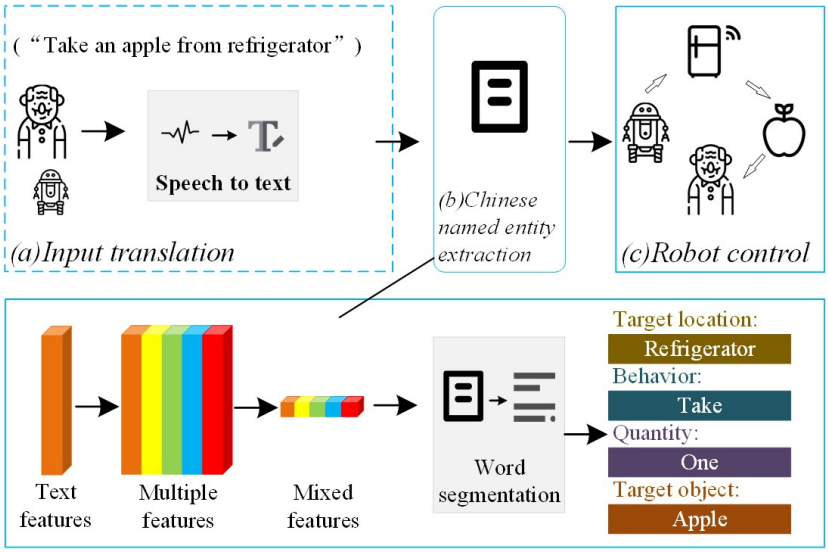



**Figure 1.** Self-supervised named entity recognition model for robot behavior control. (a) is an input conversion module that converts speech input to text information, which is not necessary when the input is text. (b) refers to the extraction of text features, feature enhancement, and fusion (generally including Pinyin, partial, parts of speech, and other information), and the extraction of named entities based on the fusion features in a self-supervised manner. (c) shows the scene of driving the robot movement based on the extracted named entities.

can drive the robot according to our requirements.

The contributions of this paper can be summarized as follows: (1) A Self-Supervised Learning (SSL) mechanism for Chinese NER is proposed, which completely weans our network from relying on manually labeled datasets; (2) A new learning rule is defined that enables the model to propagate back in the discrete computation process; (3) In the Instruction Generation Subsystem (IGS), a position information matrix construction rule that is independent of the static graph is adopted, rather than learning approximation according to the target, as in the Gumbel-Sinkhorn network[20], which theoretically makes our model simpler and faster; (4) The object detection network is adopted for practical robot object detection and capturing, which verifies the practicability of the SCNER model.

## 2. RELATED WORKS

NER has always been a trendy research topic in NLP. Most of the early NER models use the manual dictionary or construction rules to extract named entities by retrieval or pattern matching[21]. The construction of these models is often time-consuming and laborious. With the development of deep learning, researchers gradually begin to use DNN (Deep Neural Networks) to automatically extract advanced features and named entities[22]. Nguyen *et al.* used the Word2Vc toolkit to embed word vectors and achieve better results compared to traditional methods in named entity extraction on the cyclic neural network. Chiu and Ma *et al.*[23,24] used Convolutional Neural Networks (CNN) to extract character-level representative features of words based on the Bidirectional LSTM (BiLSTM)[25], and these feature sequences are converted into word vectors as input of the encoder. Based on this, Tran *et al.*[26] introduced the idea of a Residual Network[27] into the NER model, which overcame the degradation problem in the network stacking to a certain extent. Due to the particularity of Chinese grammatical structures, the CWS (Chinese Word Segmentation) is often different from other NER models. This is reflected in the extracted named entity, which usually consists of multiple words rather than a single word. Yue *et al.*[28] designed a statistical method based on the Chinese language, the perceptron algorithm[29] was used for word-level discrimination training, and the beam search was used as a decoder. Based on the traditional word segmentation model, Ma *et al.*[30] proposed an embedded matching method

for CWS, which combined characters into words by setting separators and combinators and encoded them as features, together with segmentation actions for named entity extraction. Deng *et al.*[31] further introduced the 4-tag method. Compared to the two-tag method, the four-tag method adds the M-flag and S-flag and improves the CWS method by adding the embedding feature of adjacent characters. Zhang *et al.*[32] extended BiLSTM-CRF to model the CWS task as a character-level sequence labeling problem. In addition, Sun *et al.*[11] and Ye *et al.*[33] adopted the word embedding method and generative network for unsupervised CWS and achieved considerable results. With the development of large language models, Tang *et al.*[34] proposed a BERT-BiLSTM-AM-CRF model that used BERT to extract the dynamic word vector combined with context information and input the results into the CRF layer for decoding after further training through the BiLSTM module. Huang *et al.*[35] proposed a domain adaptive segmenter based on BERT for introducing open-domain knowledge. Private and shared projection layers were proposed to capture domain-specific knowledge and common knowledge, respectively. Tian *et al.*[36] proposed a memory network to incorporate wordhood information with several popular encoder-decoder combinations for CWS. However, the need for computing power in large models is huge.

Recently, the unsupervised and self-supervised algorithms have developed rapidly. Liu *et al.*[37] proposed Knowledge-Augmented Language Model (KALM), which makes unsupervised NER achieve a similar performance as a supervised model in some aspects by adding a gating mechanism to the traditional unsupervised NER model. SSL is a typical unsupervised algorithm. Yann LeCun defined SSL as "the machine parts of its input for any observed part" at the AAAI2020 and expressed affirms the development prospect of SSL, which has attracted wide attention. SSL differs from unsupervised learning slightly; the latter focuses on detecting specific data patterns, whereas the former aims to reproduce them[40]. In other words, SSL is still a supervised learning paradigm, and tags are still needed in the learning process, but these tags are derived from the data themselves rather than by a manual method.

Recently, there has been a lot of research on SSL in CV (computer vision)[38], GNN (graph neural networks)[39], NLP[40], and other latest topics. Because the end-to-end training is easy to make the network fall into the local minimum, the training process adopts a greedy approach[41]. The early unsupervised models train the stacked autoencoders[42] or deep belief networks[43] layer by layer without tags, and then fine-tuning is done. With the proposition of residual structure and batch regularization[44] and the cleverer activation function[45], the model can realize end-to-end learning, and the traditional training methods are gradually improved. In a recent study in the field of NLP, Giorgi *et al.*[46] realized SSL by comparing the distance between randomly sampled text fragments and the target of a pre-trained embedded encoder based on the universal sentence. Fang *et al.* proposed CBERT (conditional BERT) based on BERT[47,48] and realized self-supervised training by comparing the training mode of constructing sentences with the original sentences. The mechanism of SSL is similar to the CBERT of the machine translation model.

Therefore, the SSL approach frees the model from its dependence on labeled data. Based on the mechanism of SSL, a discrete sequence rearrangement-based self-supervised Chinese NER model is proposed. The self-supervised process is consistent with the rules of human language organization, which is a problem of discrete sequence rearrangement. Yang *et al.*[49] proposed the KBLSTM (Knowledge-aware BiLSTM) model that achieved high accuracy on the ACE2005 dataset, and the external knowledge is adopted to encode knowledge into discrete index features and combined with BiLSTM. The backbone of the proposed model is the BiLSTM model. Since there is no supervised learning behavior in the training process, the CRF module in the inference layer is replaced by a winner-take-all approach, and a "Retelling Instructions" generation subsystem is added for self-supervised training of the model. For the proposed model, the 4-tag method is adopted, which is consistent with the method of Deng, and based on this, several features are introduced, such as Pinyin, partial, and parts of speech, to enhance word vector embedding, which enables the model to gather sufficient feature information to overcome the problem without enough context-relevant features for short instructions. For

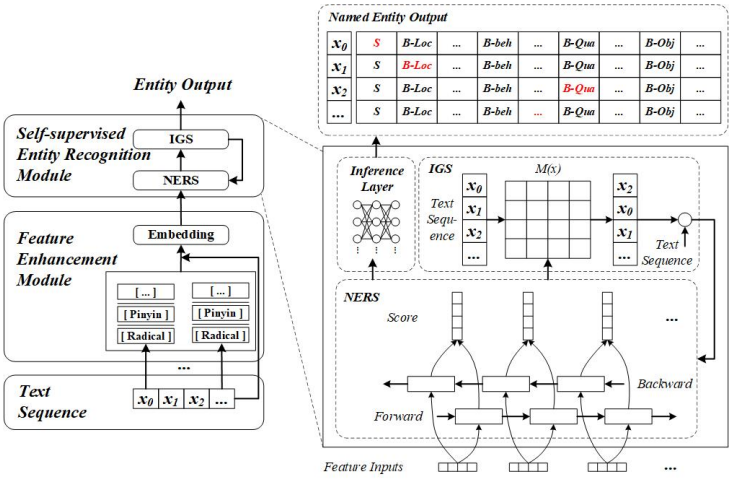



**Figure 2.** The general architecture of the SCNER models. SCNER: Self-supervised named entity recognition.

discrete sequence rearrangement, error backpropagation is a challenge, which is different from conventional neural network training. A special mechanism is constructed for training, and also the proposed model does not require large-scale labeled data to pre-train the model as BERT.

## 3. DISCRETE SEQUENCE REARRANGEMENT-BASED SELF-SUPERVISED CHINESE NAMED ENTITY RECOGNITION

In this section, the overall framework of SCNER is introduced, and then the feature enhancement module and Self-supervised Entity Recognition Module are introduced. Finally, we explain the basic learning rules of the model.

### 3.1. Overview

The SCNER model (as shown in Figure 2) consists of a feature enhancement module and a Self-supervised Entity Recognition Module, which are used for pre-processing and SCNER, respectively. The Self-supervised Entity Recognition Module includes a NERS (Named Entity Recognition Subsystem) and an IGS. A NERS generates scores for each Chinese character corresponding to each named entity category, and these scores form a raw score matrix. The generation method of "Retelling Instructions" is to reorder the input instructions according to the information in the matrix and the grammar rules. However, this cannot be achieved by using the raw scoring matrix without processing, as the scattered energy in the matrix causes the different characters of the input instructions to interfere with each other when reordering. In this paper, we construct a Reposition Matrix in the form of a permutation matrix to eliminate the interaction between Chinese characters, the implementation of which is given below. Based on this, the model can independently apply the position information of the raw score matrix to each Chinese character in the process of creating the "Retelling Instructions", and also, the error backpropagation for SSL of the model can be carried out. The methods of generating "Retelling Instructions" are given in the description of IGS and learning rules in Part C. After the completion of the SSL through multiple interactions by the model, the raw score matrix is directly used for entity category reasoning through the inference layer, and the named entity category attributes of each Chinese character can be gathered.

### 3.2. Feature enhancement module

In the NER model, an important hidden feature is context information learning. However, for robot linguistic instructions, such contextual information gathered by large-scale pre-training are scarce. Compared with other

languages, Chinese sentences have unique grammatical rules, and Chinese characters also have a variety of descriptive attributes, such as Pinyin and partial. Feature enhancement can be achieved through the creation of feature sequences of this unique language structure, and the connection between contexts can be established to a certain extent.

In this paper, $\mathcal{X}_c = [\mathcal{X}_{c0}, \cdots, \mathcal{X}_{cn}]$ represents a sentence, where $\mathcal{X}_{ci}$ represents the $i^{th}$ Chinese character and $\mathcal{X}_F$ represents the feature sequence corresponding to the sentence. Given a sentence $x_c$, the purpose of the feature enhancement operation is to build a feature sequence set $\mathcal{X}_F$ by integrating other attributes of Chinese characters and mapping them to a feature vector set $\psi_F$. In this paper, five-dimensional attributes are used for feature enhancement, so the feature sequence set of a sentence contains five elements: $\{\mathcal{X}_c, \mathcal{X}_p, \mathcal{X}_r, \mathcal{X}_f, \mathcal{X}_b\}$. Where $\mathcal{X}_c$ is the sequence of Chinese characters, $\mathcal{X}_p$ is the Pinyin sequence corresponding to the sentence, $\mathcal{X}_r$ is the partial sequence, $\mathcal{X}_f$ is the part of speech sequence, and $\mathcal{X}_b$ is the word boundary sequence. The method of word boundary dividing is similar to the 4-tag method proposed by Deng *et al.*[31], in which Chinese words are divided according to the position by [B(Begin), M(Middle), E(End), S(Single)]. The difference is that in the process of feature embedding, we do not use any pre-trained word vector embedding model. Our feature embedding method can be expressed as follows:

$$\psi_i = \sum W_i^l \mathcal{M}(\mathcal{X}_i) + b_i^l \tag{1}$$

where $\mathcal{X}_i \in \mathbb{R}^d$ represents the original input sequence containing all the feature information of a Chinese character, and d represents the dimension of the feature. Function $\mathcal{M}$ represents an encoding mapping based on a statistical dictionary. $W^{(l)}$ is the weight matrix of the linear transformation, and $b^{(l)}$ is the deviation vector. Both of them are part of self-supervised closed-loop learning and are generated in end-to-end training without pre-training.

### 3.3. Self-supervised entity recognition module

As shown in Figure 2, a Self-supervised Entity Recognition Module, including a NERS and an IGS, is designed. Based on a traditional BiLSTM model, which temporarily shields the inference layer, we process the input feature sequence and feed it to IGS to generate retelling instructions. The inference layer is reactivated after the completion of the SSL process. Next, we introduce the NERS and IGS.

#### *3.3.1. Named entity recognition subsystem*

One of the key tasks of statement parsing is the extraction of named entities. The traditional method usually uses the BiLSTM model as the feature extraction model and then gathers the strong context information in the form of a state transition matrix through the conditional random field in the supervised learning process and carries on the inference through the Viterbi decoding module[20,24,47,50]. The structure of the NERS is similar, but the CRF module has been removed from the NERS due to our self-supervised training using completely untagged data. The one-dimensional feature sequence $\psi_i' = \left[\varphi_{i_c}', \cdots, \varphi_{i_p}', \cdots \varphi_{i_r}', \cdots\right]$ obtained by flattening the feature vector $\psi_i$ is entered into the NERS model to obtain the score sequence $\mathcal{L}_i$, which is implemented as follows:

$$\mathcal{L}_i = BiLSTM(\psi_i') \tag{2}$$

Where $BiLSTM()$ represents the traditional bidirectional network operation without an inference layer.

#### *3.3.2. Instruction generation subsystem*

In this paper, we design a special construction rule to realize the input instruction independent transfer to "Retelling Instructions" according to the Reposition Matrix. In detail, IGS first processes the Raw Score Matrix from the NERS output to obtain a Reposition Matrix of One-Hot encoding type. The model then reorders the Input Instructions using the Reposition Matrix as the permutation matrix to generate Retelling Instructions. The Loss is calculated based on the difference between Retelling Instructions and Input Instructions and is used

for model training. But it should be noted that the reordering of instructions and the generation of Reposition Matrix are discrete, and how to implement backpropagation in it is a difficult problem that we need to solve.

Based on the idea of Gumbel-Sinkhorn [16], our method aims to construct an ideal positional information matrix. The difference is that the Gumbel-Sinkhorn algorithm uses supervised learning to generate the Reposition Matrix, to solve the non-differentiable problem in discrete operation by continuous approximation method. In this paper, the Reposition Matrix is extracted through discrete operation, "$\mathcal{K}operation$", which is independent of the network. The complete implementation is as follows.

Based on the NERS, the model calculates the score $\mathcal{L}_i = [\ell_{i0}, \ell_{i1}, \ldots, \ell_{id}]$ of the correlation between each Chinese character and the named entity through calculation, where $0 \leq i \leq n$,$d$ represents the number of named entity categories. The set of score sequences constitutes the raw score matrix $\mathcal{L} = [\mathcal{L}_0, \mathcal{L}_1, \ldots, \mathcal{L}_n]$ for all inputs corresponding to the named entity class. The process of getting the Reposition Matrix $\mathcal{L}^{(M)}$ derived from the raw score matrix is expressed as:

$$\mathcal{L}^{(M)} = \mathcal{K}\left([\mathcal{L}_0, \mathcal{L}_1, \ldots, \mathcal{L}_n]^T - \mathcal{L}' + \mathcal{L}^{(N)}\right) + \mathcal{L}' - \mathcal{L}^{(N)} \tag{3}$$

Where $\mathcal{K}$ represents the "$\mathcal{K}operation$", which filters the original score matrix to obtain the Reposition Matrix, and the implementation process is expressed as follows:

$$\mathcal{K}(X_{n\times d}) = onehot(argmax(X_{n\times d})) \tag{4}$$

Where $argmax()$ compresses the matrix $X_{n\times d}$ into an n-dimensional vector, $onehot()$ expands the n-dimensional vector into a one-hot-matrix with $n \times d$ dimensions.

$\mathcal{L}'$ and $\mathcal{L}^{(N)}$ represent the constraint rule and the balance rule, respectively. Without constraint [Figure 3A] and balance rules [Figure 3B], the model degrades rapidly during training. Their implementation is expressed as follows:

$$\mathcal{L}' = [\mathcal{L}_0, \mathcal{L}_1, \ldots, \mathcal{L}_n] \bullet \mathcal{K}([\mathcal{L}_0, \mathcal{L}_1, \ldots, \mathcal{L}_n]) \tag{5}$$

$$\mathcal{L}^{(N)} = \alpha \bullet \mathcal{R}\left(-\left|\mathcal{L}^{(M)} - \mathcal{N}_{n\times d}(1)\right|\right) \bullet \mathcal{L} \tag{6}$$

Where $\alpha$ is the equilibrium factor, and $\mathcal{R}$ is the linear rectification unit. $\mathcal{N}_{n\times d}(1)$ represents a constant matrix with $n \times d$dimensions and all elements of 1.

As shown in Figure 4, the energy distribution in the Reposition Matrix is more concentrated than in the raw score matrix, which helps to reorder the characters independently during the generation of the Retelling Instructions. However, due to the limitation of learning rules, the position information in the original fractional matrix generated by the model is usually not ideal, which means the model cannot converge. In the next section, we discuss why the traditional learning rules do not apply to SCNER and propose our solution.

#### *3.3.3. Learning rules of the SCNER model*

As we discussed above, the Reposition Matrix can avoid the interference of characters in the process of Retelling Instructions, but the learning method of the model is the key problem since the Loss of the SCNER is caused by Retelling Instructions and input instructions, which means that the model cannot be learned according to the traditional back propagation method. For example, the encoding of the Input Instruction at the character position of $L_1$ is $x_1$, and the character in the same position in Retelling Instruction is encoded as $x_2$. If the value of $x_1 - x_2$ is negative, this does not mean that the model should increase the weight to increase the value of the output at the $L_1$ position but that it should reduce the likelihood of putting $x_2$ in the $L_1$ position by changing the information in the reposition matrix. In order to solve this problem, the value function of the SCNER model is designed as follows:

$$cost = mean((R_s - \mathcal{S}(I_s - 2 \bullet relu(I_s - R_s)))^2) \tag{7}$$

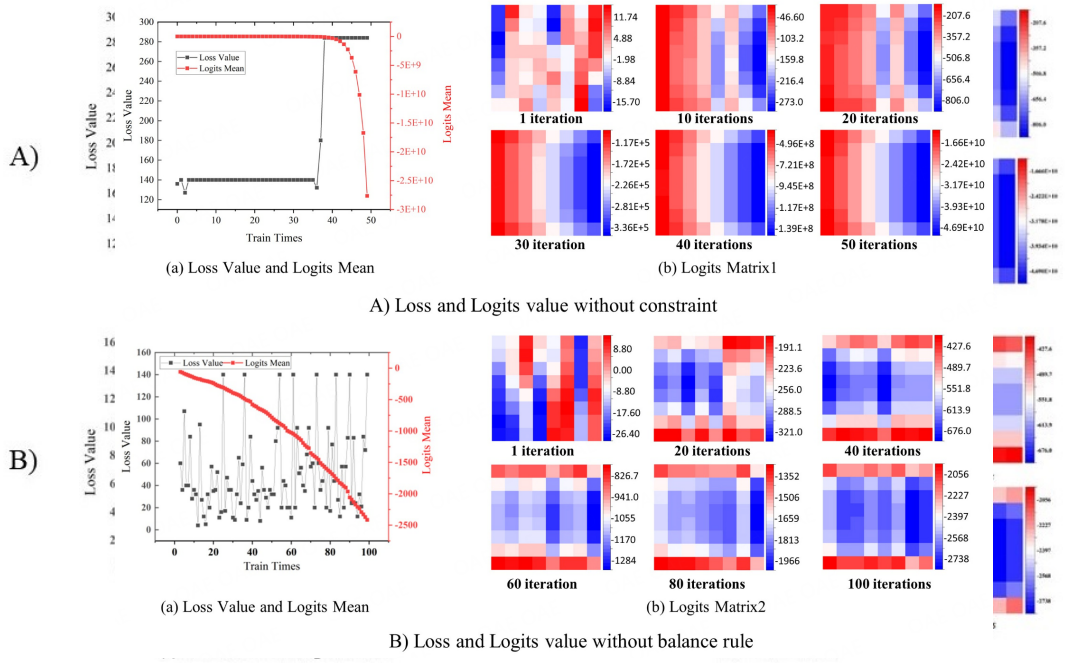



**Figure 3.** The general architecture of the SCNER models. SCNER: Self-supervised named entity recognition.

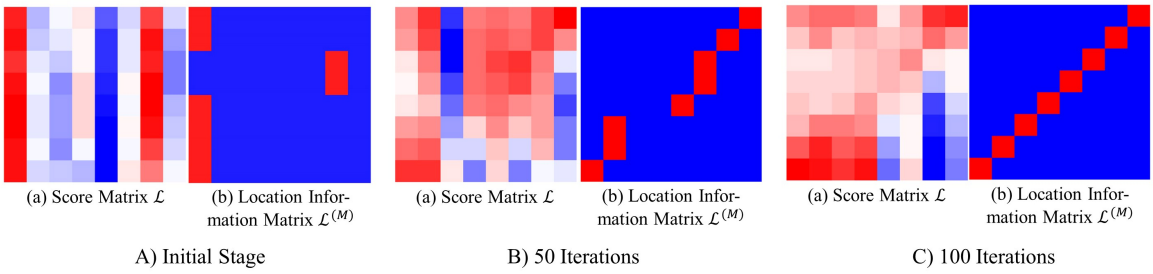



**Figure 4.** The general architecture of the SCNER models. SCNER: Self-supervised named entity recognition.

Where $R_s$ stands for Retelling Instructions, $I_s$ for input instructions, and $\mathcal{S}$ for masking operations during error backpropagation. $R_s = I_s \bullet \mathcal{L}^{(N)}$ stands for Retelling Instructions, $I_s$ for input instructions, and $\mathcal{S}$ for masking operations during error back propagation. The details of the $R_s$ and $\mathcal{S}$ are described in subsection 3.3.2.

## 4. EXPERIMENTS

### 4.1. Details

To test the performance of the model, we constructed a robotic command dataset for the named entity extraction experiment. As shown in Table 1, the robot instruction contains five entity categories: Location, Object, Quantity, Behavior, and Name, each of which is labeled in B, I, E, and O format. Among them, B is the start position, I is the middle position, E is the end position, and O is the other. Instructions are created as logical combinations of different physical elements, for example, "Take an Apple from the Refrigerator". Unreasonable combinations, such as "Take a Sweater from the Refrigerator", are not allowed. In the end, we chose a representative set of 1,500 instructions to make up our dataset, all of which are grouped into three syntactic structures: "Take an apple from the refrigerator", "Clean the living room", and "Play the movie Gone with the

**Table 1. Details of the robot instruction dataset**

| Entity | | Instructions | Entity annotation type |
|---|---|---|---|
| **Category** | **Element** | | |
| Location | Supermarket<br>Living Room<br>Storage Room<br>Bedroom<br>Refrigerator..... | Take an apple from the refrigerator<br>Take a coat from the bedroom<br>Go downstairs and throw the trash | |
| Object | Apple Beer<br>Sweater Express<br>Delivery Dictionary<br>Movie..... | Get a dictionary from the study<br>Take a lemon from the kitchen.....<br>Play the Song "Little Apple"<br>Play the song Hotel California | O: Other<br>B: Beginning |
| Quantity | One A Piece of<br>A Bottle of<br>A Box of<br>Two Pounds..... | Play the song Edelweiss<br>Play the movie Transformers<br>Play the movie Gone with the Wind<br>Playing the French Open | I: Intermediate<br>E: End |
| Behavior | Take Play<br>Buy Borrow<br>Take out Open..... | Turn on the news channel.....<br>Clean the living room<br>Tidy up the study | |
| Name | Hotel California<br>Gone with the Wind<br>Doctor Strange<br>..... | Open the door<br>Do the dishes<br>Take out the trash..... | |

Wind".

To obtain more contextual information, the model will perform information enhancement for each input instruction. In detail, the model uses the combined encoding sequence of five features of Word, Pinyin, Radical, Part of Speech, and Word Boundary as input to alleviate the problem of sparse context information. The enhanced feature sequence is mapped into word vectors of 100 dimensions (each feature is mapped to 20 dimensions). The embedded module of the word vector used in the model is not pre-trained but a mapping based on simple statistics of input instructions.

In the experiment, the two-layer BiLSTM network is adopted in our word segmentation model for feature extraction. Each layer of the LSTM network contains 1000 LSTM units, and global feature information is used as output in the bottom layer. For each layer of the encoder, the model concatenates the input processing results of all memory units to obtain the full-time sequence of features and use them as the input of the next layer. In our experiment, the Loss function of the model is set as the mean square error, the learning rate is 1e-4, the balance factor is 0.12, and the number of iterations of the self-supervised training is set as 200.

Our experiments include an SSL process experiment, a self-supervised online learning experiment, and a self-supervised batch learning experiment. It is important to note that in all SSL, the input data to the model is unlabeled. In the first experiment, we recorded the learning process of the model to a single instruction. The model starts SSL after getting the input instruction, and the end condition is that the model can accurately identify all named entities in the instruction. This experiment aims to demonstrate the process and results of the SSL of the model, which is the basis for the following experiments. The results presented consist of four parts: (1) The accuracy of entity extraction during the self-supervised training process, which demonstrates the process of exploring the syntactic combination patterns of the model during the training process; (2) Visualization of entity extraction, which visualizes the variation of the model output during training; (3) Loss curve, which shows the convergence and learning speed of the model; and (4) Logits Matrix, which describes the difference between the paraphrased instructions and the input instructions constructed by the model, and there should be a clear correspondence between them.

In the self-supervised online learning experiment, the model is fed multiple instructions one by one. Each instruction is different from the previous instruction, either with different syntax or with different content.

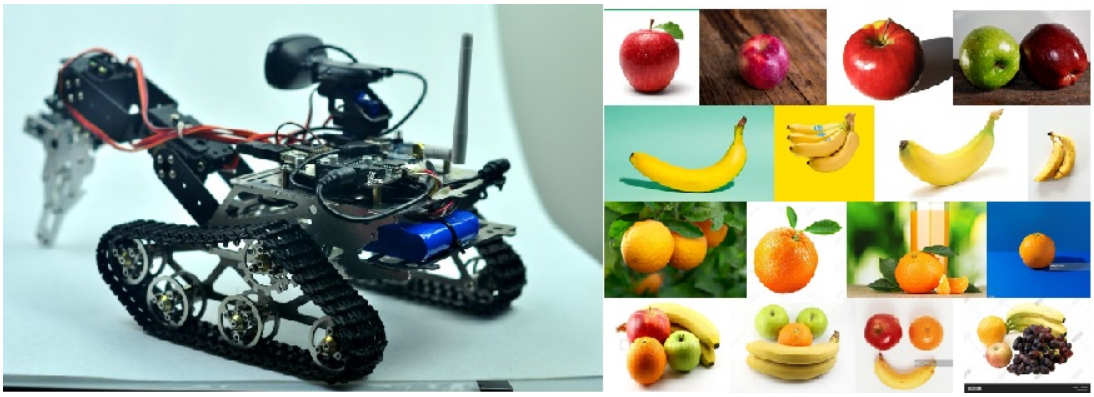


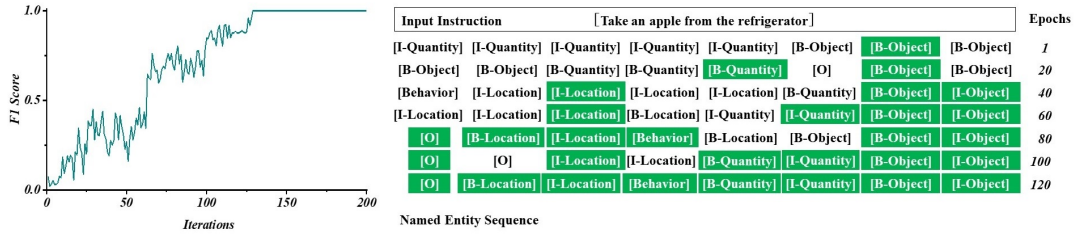
**Figure 5.** Tracked robot and dataset.

The condition for the end of each instruction processing is still that all named entities in this instruction can be accurately extracted. It aims to demonstrate the online learning capability of the proposed model, i.e., how the model learns new knowledge in a self-supervised way without forgetting the old knowledge when it is finished training and deployed to the robot. This simulates a scenario where the knowledge of the robot is updated online in daily use. The results presented include (1) online learning of many instructions with the same grammar rules. The hypothetical scenario for this experiment is to update the database of a given entity on the robot, e.g., a newly released movie, "Gone with the Wind", to be added to the Object database. The experiment demonstrates that the model can quickly acquire new knowledge "Play the movie "Gone with the Wind" based on existing knowledge, such as "Play the movie "Forrest Gump". (2) Online learning of many instructions with different syntax rules. The hypothetical scenario for this experiment is that the developer wants to add some instructions to the robot that differ significantly from the instructions used for training. The experiment verifies the ability of the model to learn such knowledge while overcoming the forgetting catastrophe of old knowledge.

In the process of self-supervised batch learning, we divide the instruction into many small batches and feed them to the network in a way similar to supervised batch learning. In this learning process, the model only performs an SSL process for each batch instruction instead of implementing accurate NER as in the previous two experiments. This experiment aims to demonstrate the effect of batch training on the proposed model. The previous two experiments have shown that an SSL approach on each instruction is practical, but this leads to a significant time overhead in the pre-training phase of the model. Therefore, a more efficient solution is pre-training using the classical batch training approach and updating knowledge using the self-supervised online learning approach.

We also use the tracked robot, as shown in Figure 5a, to conduct behavioral control verification experiments for fetching the apples and oranges, and it communicates with the server through Wi-Fi. For the object detection model, we migrate the YOLO-V4 model that has been pre-trained on the VOC2007 dataset as the object detection model. It should be fine-tuned on the fruit data set shown in Figure 5b, which is derived from Baidu AI Studio and contains 300 images of four categories: oranges, apples, bananas, and mixtures. In this experiment, language parsing and object detection are all run on the server, and the robot is responsible for behavior implementation and environment perception. It should be noted that the behavior pattern of the robot and the path information in the environment are known by default, and this part can be further improved by constructing a semantic map and other technologies.

### 4.2. Results and analysis




(a) F1 Score (b) Named Entities Sequences

**Figure 6.** Training results. (a) is the F1 score curve, with the horizontal axis representing the number of iterations of SSL and the vertical axis representing the F1 Score of the model for identifying named entities. During all iteration cycles, there is only one input instruction for the model. (b) is the sampling of the NER results of the model in the training process, which is used to record the named entity sequences generated by SCNER under different training cycles. In the figure, the label of a named entity with green background is identified correctly, whereas the label without processing is identified wrongly, and its display content is consistent with the F1 Score curve.

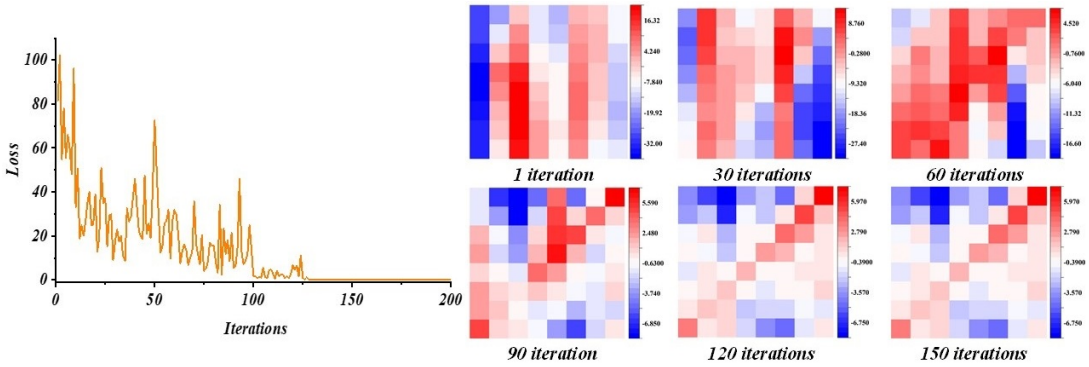



**Figure 7.** Training process.

### *4.2.1. Self-supervised learning process experiment*

The algorithm for the SCNER model used in the self-supervised learning process experiment is given in Algorithm 1.

Algorithm 1 Self-supervised learning process experiment

Input: Input Instructions
Output: Named entity sequence, F1 score, and Loss
1: initialize: Create the SCNER model, Loss=1
2: while Loss ≥0 do
3: Named entity sequence, Loss, F1 Score=SCNER(Input instruction)
4: end while

As shown in Figure 6, the model accepts only one instruction at a time as an input, and the state of the model is randomly initialized when receiving an instruction, so the model cannot accurately name the entity for the input instruction at the beginning. After about 130 cycles of SSL, the model can converge to extract the named entity in the input instruction accurately and maintain stability [Figure 7]. Similarly, the raw score matrix of the model also gradually transitions from a completely disordered state to an ordered state. Although there is still some influence in the non-target named entity category, the influence is almost negligible. After the completion of the SSL process for the first instruction, the model saves the node and uses it for the extraction of named entities for subsequent instructions of the same syntax. If the latter input instruction has a different syntax structure, the model is initialized before learning. The judgment is based on the parts-of-speech sequence in the input feature sequence.

*4.2.2. Self-supervised online learning experiment*

The algorithm for the SCNER model used in the self-supervised online learning experiment is given in Algorithm 2.

Algorithm 2 Self-supervised online learning process experiment

```
Input: Input Instructions
Output: Named entity sequence, Epochs
initialize: Create the SCNER model
for i = 1, 2, ··· , len(Input Instructions) do
    Loss=1, Epochs=0
    while Loss ≥0 do
        Named entity sequence=SCNER(Input Instruction[i])
        Epochs+=1
    end while
end for
```

As shown in Figure 8a, we use six-sentence instructions with three grammatical structures for SSL. Each instruction appears only once in each stage, and the model performs self-supervised training repeatedly until it can accurately identify all named entities. When this is achieved, the learning process ends, and the number of iterations of SSL is recorded. It can be seen that, even if the instructions are not identical, the existing learning experience is available for later instruction learning of the same grammatical structure. And when the learning of the later instruction makes the model forget the previous instruction, the model can also converge rapidly through a few learning cycles when the previous instruction is re-input. Finally, the model can accurately extract named entities from all instructions.

It should be noted that in this section, the number of iterations of the model refers to the minimum number of iterations required by the model to achieve the correct NER (that is, the accuracy rate is 1) of an input instruction. This means that the number of iterations is a judgment index of the learning speed rather than the complexity of the model.

To verify the performance of the model on a large amount of data, we feed 300 syntax-structured instructions into the model repeatedly for named entity extraction. As shown in Figure 8b, the convergence rate of the model is slowed down due to the increase in instructions. However, the model can still accumulate learning experience to speed up the parsing of instructions with the same grammatical structure. As the training goes on, the number of iterations of SSL for the correctly named entity extraction is rapidly reduced when the same instructions are re-inputted. Finally, the model can realize the NER of all instructions without SSL.

*4.2.3. Self-supervised batch learning experiment*

The algorithm for the SCNER model used in the self-supervised batch learning experiment is given in Algorithm 3. The experiment uses all the datasets for training and randomly selects 100 instructions from them as the verification set and the test set. It should be noted that the data fed to the BERT-NER model and the BiLSTM-CRF model as a training set are labeled, while the data fed to the SCNER model are not.

As shown in Figure 9, the BERT-NER model quickly achieves accurate prediction of all instructions with a few iterations, and the SCNER model also has this capability with about 100 iterations. The BiLSTM-CRF model did not perform as well as expected, with an additional 2,000 iterations to achieve an F1 score of 0.856 ± 0.007. These results show that the SCNER model is competitive with the traditional supervised NER model. Although the BERT-NER model performs better, the learning method of the SCNER model is more distinctive and does not require labeled data, so its performance is satisfactory. It should also be emphasized that during this experiment, the instructions used for supervised BERT-NER model testing were randomly selected from

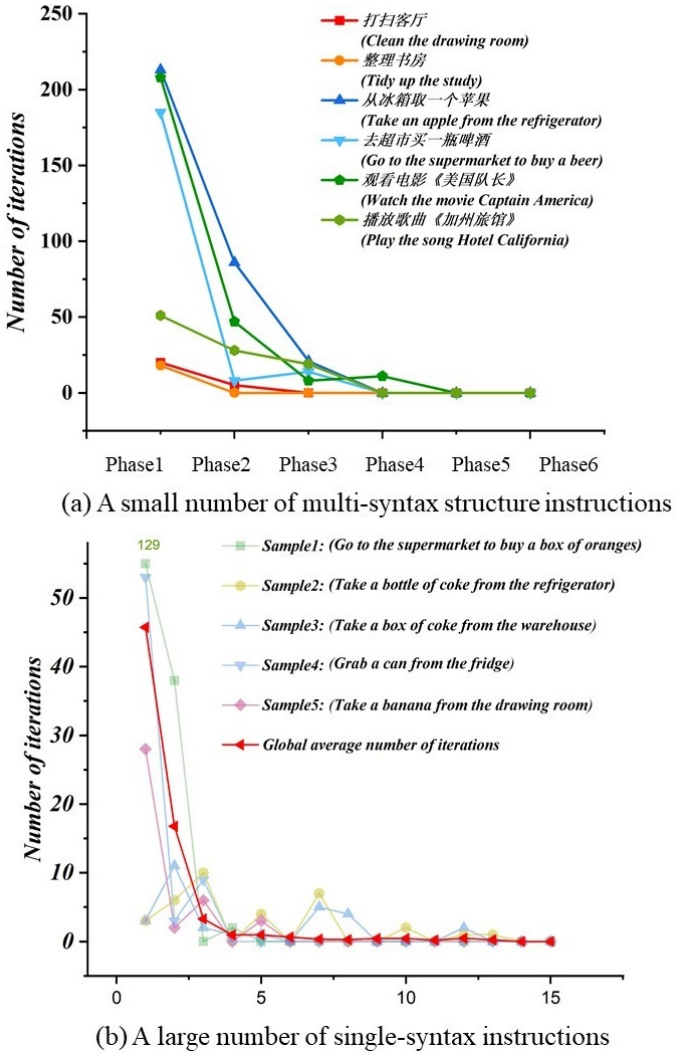



**Figure 8.** Repetitive self-supervised learning process. (a) is the process of learning a small number of input instructions that conform to a variety of grammatical structures. Blue, green, and red correspond to three different grammatical structures, and dark samples are entered before light samples. (b) is the process of learning a large number of instructions that conform to a grammatical structure. The light color line is to track the learning process of five instructions in random sampling, and the dark color line is to average the number of iterations needed to extract the named entity from all samples in one cycle. The horizontal axis of both graphs represents the number of times a sample appears in the model, and the vertical axis represents the number of SSL iterations needed to accurately implement the NER of input instructions.

the training set. In other words, the model has already learned these instructions during the pre-training phase. And if we test the BERT-CRF model with instructions not present in the training set, it is difficult to get correct results.

As shown in Table 2, we create 100 instructions independent of the training set to test the pre-trained model. Before testing, we use a weakly supervised learning method to pre-train the model. In detail, we fine-tune the pre-trained BERT model using our dataset to introduce weak supervision signals and then test the model with newly created unfamiliar instructions.

The 100 instructions used in the test can be divided into three grammatical structures, such as "Take a Lemon from the Refrigerator", "Play the Song Lemon", and "Clean up the Study", and the corresponding number of

Algorithm 3 Self-supervised batch learning process experiment

Input: Input Instructions
Output: Named entity sequence, F1 Score
1: initialize: Create the SCNER model, Loss=1,Batch(Input Instructions)
2: while Loss≥0 do
3: for $i = 1, 2, \cdots,$ len(Input Instructions) do
4: Named entity F1 score=SCNER(Input Instruction[i])
5: end for
6: end while

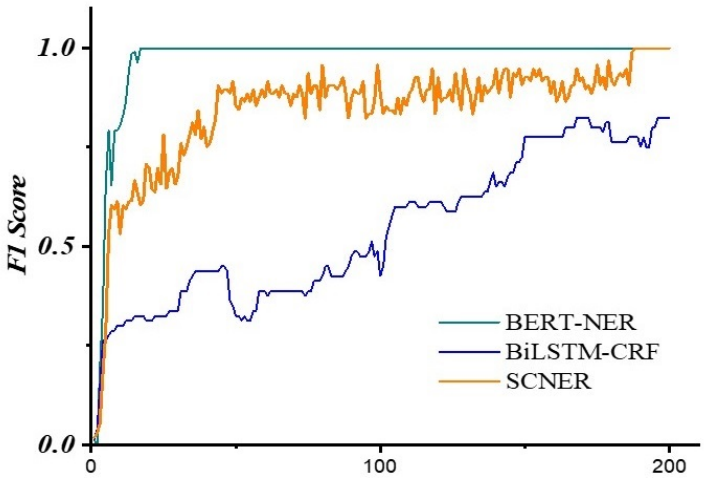



**Figure 9.** Batch learning result.

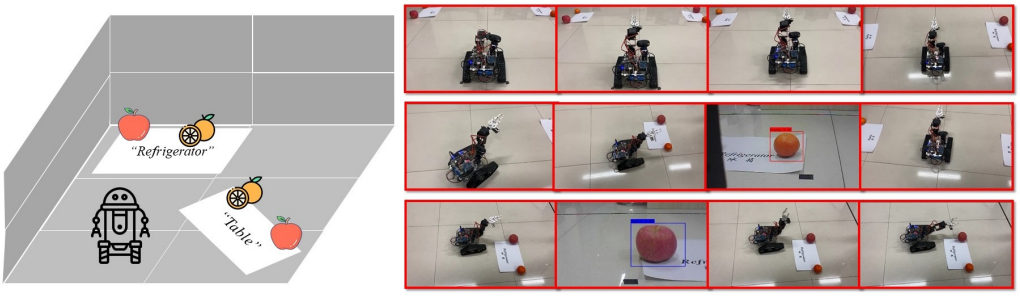
**Table 2. Test results of pre-trained BERT-NER model on unfamiliar instructions**

| Instruction type | Accuracy |
|---|---|
| The instructions for the structure of "Clean up the Study" | 0.6667 |
| The instructions for the structure of "Play the Song Lemon" | 0.8333 |
| The instructions for "Take a Lemon from the Refrigerator" | 0.5 |

instructions are 40, 40, and 20, respectively. It should be noted that we deliberately created some ambiguities to increase the complexity of the test. For example, we use the instruction "Take a Lemon from the Refrigerator" in the training set and label "Lemon" as the "Object" entity class. In the test set, we use instructions such as"Play the Song Lemon", where "Lemon" should be predicted as a song name. It can be seen that although the model has high accuracy for all instructions in pre-training, during the testing process, its prediction accuracy for these three types of instructions decreases; although the grammatical structure of these instructions is the same, only the content is different.

#### *4.2.4. Practical display of robot-SCNER*

After entity extraction, entities, such as "B-location" and "I-object" output by the model, can be used to drive robot motion. As shown in Figure 10a, we built an experimental environment for testing the control effect of the robot. To eliminate the requirements of different shapes, sizes, and usage modes of the "Location" target on robot motion control and simplify the test process, we substituted the real "Location" target with a signboard printed with the name of the location target. Figure 10b shows the motion shot of the robot with the command "Take an apple from the fridge" as an example. As shown in Table 2, the robot uses the YOLO-V4 network to search for an object after entering the "refrigerator" signage area. When there is no"Object" in the field of vision, the robot switches position or posture to detect other positions in the current area. After finding the target




(a) Experimental environment (b) Shots of robot movement

**Figure 10.** Robot motion control test.


**Figure 11.** Shot sampling in application experiments based on a single-arm robot.

object, the robot clamps the object according to the preset motion scheme. In addition, we also conducted experiments on a commercial single-arm robot (As shown in Figure 11), which is also the requirement of the fund on which our research is based.

## 5. CONCLUSIONS

In this work, we propose a self-supervised Chinese NER model SCNER, which can achieve Chinese named entity extraction in specific application scenarios and apply it to robot motion control. SCNER is based on the traditional NER model and takes the human imitation learning mechanism as the paradigm to realize SSL, which can work in an environment without labeled information. An SSL method based on discrete sequence rearrangement is constructed, which enables the model to construct Retelling Instructions based on the entity extraction results and use the distance between Retelling Instructions and input instructions as a loss function for model training. To address the problem that the backpropagation of the discrete operation of the matrix in training is not available, a two-channel discrete matrix generation method makes the output consistent with the discretized matrix while enabling backpropagation in this phase. In the experiment, SCNER can

achieve a stable and accurate NER effect with a very short self-supervised training of about 100 cycles and accurately control the robot movement according to the acquired named entity objects. In the future, we plan to redesign the instruction generation method of the " Retelling Instructions" construction subsystem to increase the generality of the model.

## DECLARATIONS

### Acknowledgments

The authors would like to thank the Editor-in-Chief, the Associate Editor, and the anonymous reviewers for their valuable comments.

### Authors' contributions

Implemented the methodologies presented and wrote the paper: Jiang C, Xu Q
Performed oversight and leadership responsibility for the research activity planning and execution and developed ideas and evolution of overarching research aims: Song Y, Yuan X, Pang B
Performed providing administrative and financial support: Li Y
All authors have revised the text and agreed to the published version of the manuscript.

### Availability of data and materials

Not applicable.

### Financial support and sponsorship

This research was funded by the National Key Research and Development Plan of China under Grant (2020AAA0108900), the National Natural Science Foundation of China under Grants (61573213, 61803227, 61603214, 61673245), and the Natural Science Foundation of Shandong Province (ZR2020MD041, ZR2020MF077).

### Conflicts of interest

All authors declared that there are no conflicts of interest.

### Ethical approval and consent to participate

Not applicable.

### Consent for publication

Not applicable.



## REFERENCES

1. Deuerlein C, Langer M, Seßner J, Heß P, Franke J. Human-robot-interaction using cloud-based speech recognition systems. *Procedia CIRP* 2021;97:130–5. DOI
2. Cheng JR, Liu JX, Xu XB, Xia DW, Liu L, Sheng VS. A review of Chinese named entity recognition. *KSII T Internet Info* 2021;15:2012–30. DOI
3. Yu J, Bohnet B, Poesio M. Named entity recognition as dependency parsing. In: Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics; 2020 Jul 5-10; Online: Association for Computational Linguistics; 2020. pp. 6470–6. DOI
4. Lin H, Lu Y, Tang J, et al. A rigorous study on named entity recognition: can fine-tuning pretrained model lead to the promised land? In: Webber B, Cohn T , He Y , and Liu Y , editors. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing; 2020 Nov 8-12; Online: Association for Computational Linguistics; 2020. pp. 7291–300. DOI
5. Zhou G, Su J, Zhang J, Zhang M. Exploring various knowledge in relation extraction. In: Knight K, Ng HT, Oflazer K, editors. Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics; 2005 Jun 25-30; Ann Arbor, Michigan. Association for Computational Linguistics; 2005. pp. 427–34. DOI

6. Cheng P, Erk K. Attending to entities for better text understanding. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence; 2020 Feb 7-12; New York, USA: AAAI; 2020. pp. 7554–61. DOI
7. Petkova D, Croft WB. Proximity-based document representation for named entity retrieval. In: Mário J. Silva, Alberto A. F. Laender, Ricardo Baeza-Yates, Deborah L. McGuinness, Bjorn Olstad, Øystein Haug Olsen, André O. Falcão, editors. Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management; 2007 Nov 6-10; New York, USA: Association for Computing Machinery; 2007. pp. 731–40. DOI
8. Virga P, Khudanpur S. Transliteration of proper names in cross-lingual information retrieval. In: Hinrichs EW, Roth D, editors. Proceedings of the ACL 2003 Workshop on Multilingual and Mixed-language Named Entity Recognition; 2003 Jul 7-12; Sapporo, Japan: Association for Computational Linguistics; 2003. pp. 57–64. DOI
9. Chen HH, Yang C, Lin Y. Learning formulation and transformation rules for multilingual named entities. In: Hinrichs EW, Roth D, editors. Proceedings of the ACL 2003 Workshop on Multilingual and Mixed-language Named Entity Recognition; 2003 Jul 7-12; Sapporo, Japan: Association for Computational Linguistics; 2003. pp. 1–8. DOI
10. Light M. Corpus processing for lexical acquisition. *J Logic Lang Inf* 1998;7:111–4. DOI
11. Sun Z, Deng Z. Unsupervised neural word segmentation for Chinese via segmental language modeling. In: Riloff E, Chiang D, Hockenmaier J, Tsujii J, editors. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing; 2018 Nov 2-4; Brussels, Belgium: Association for Computational Linguistics; 2018. pp. 4915–20. DOI
12. Shaalan K. A survey of Arabic named entity recognition and classification. *Comput Linguist* 2014;40:469–510. DOI
13. Wang Y, Tong H, Zhu Z, Li Y. Nested named entity recognition: a survey. *ACM T Knowl Discov D* 2022;16:1–29. DOI
14. He S, Sun D, Wang Z. Named entity recognition for Chinese marine text with knowledge-based self-attention. *Multimed Tools Appl* 2022;81:19135–49. DOI
15. Wang TB, Huang RY, Hu N, Wang HS, Chu GH. Chinese named entity recognition method based on dictionary semantic knowledge enhancement. *Ieice T Inf Syst* 2023;E106D:1010–7. DOI
16. Zhang H, Wang XY, Liu JX, Zhang L, Ji LX. Chinese named entity recognition method for the finance domain based on enhanced features and pretrained language models. *Inf Sci* 2023;625:385–400. DOI
17. Goyal A, Gupta V, Kumar M. Recent named entity recognition and classification techniques: a systematic review. *Comput Sci Rev* 2018;29:21–43. DOI
18. Zhu E, Li J. Boundary smoothing for named entity recognition. In: Muresan S, Nakov P, Villavicencio A, editors. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics; 2022 May 22-27; Dublin, Ireland: Association for Computational Linguistics; 2022. pp. 7096–108. DOI
19. Zhang Y, Wang M, Huang Y, Gu Q. Improving Chinese segmentation-free word embedding with unsupervised association measure. [Preprint]. arXiv. July 5, 2020 [accessed 2023 July 21]. Available from: https://doi.org/10.48550/arXiv.2007.02342.
20. Mena G, Belanger D, Linderman S, Snoek J. Learning latent permutations with Gumbel-Sinkhorn networks. In: Yoshua B, Yann LeCun, Tara S, editors. 6th International Conference on Learning Representations; 2018 Apr 30- May 3; Vancouver, BC, Canada: OpenReview.net; 2022. pp. 1–22. Available from: https://openreview.net/forum?id=Byt3oJ-0W.
21. Zhou G, Su J. Named entity recognition using an HMM-based chunk tagger. In: Isabelle P. Proceedings of the 40th Annual Meeting on Association for Computational Linguistics; 2002 Jul 7-12; Philadelphia Pennsylvania: Association for Computational Linguistics; 2002. pp. 473–80. DOI
22. Dos Santos CIC, Guimar A Es V. Boosting named entity recognition with neural character embeddings. In: Duan X, Banchs RE, Zhang M, Li H, Kumaran A, editors. Proceedings of the Fifth Named Entity Workshop; 2015 Jul 7-12; Beijing, China: Association for Computational Linguistics; 2015. pp. 25–33. DOI
23. Chiu J, Nichols E. Named entity recognition with bidirectional LSTM-CNNs. *Trans Assoc Comput Linguist* 2016;4:357–70. DOI
24. Ma X, Hovy E. End-to-end sequence labeling via Bi-directional LSTM-CNNs-CRF. In: Erk K, Smith NA, editors. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics; 2016 Aug 7-12; Berlin, Germany: Association for Computational Linguistics; 2016. pp. 1064–74. DOI
25. Dyer C, Ballesteros M, Ling W, Matthews A, Smith NA. Transition-based dependency parsing with stack long short-term memory. In: Zong C, Strube M, editors. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing; 2015 Jul 26-31; Beijing, China: Association for Computational Linguistics; 2016. pp. 334–43. DOI
26. Tran Q, MacKinlay A, Jimeno Yepes A. Named entity recognition with stack residual LSTM and trainable bias decoding. In: Kondrak G, Watanabe T, editors. Proceedings of the Eighth International Joint Conference on Natural Language; 2017 Nov 27–Dec 1; Taipei, Taiwan: Asian Federation of Natural Language Processing; 2017. pp. 566–75.Available from: https://aclanthology.org/I17-1057.
27. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition; 2016 Jun 27–30; Las Vegas, NV, USA: IEEE; 2016. pp. 770–8. DOI
28. Zhang Y, Clark S. Chinese segmentation with a word-based perceptron algorithm. In: Zaenen A, Bosch A, editors. Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics; 2007 Jun 23–30; Prague, Czech Republic: Association for Computational Linguistics; 2007. pp. 840–7.Available from: https://aclanthology.org/P07-1106.
29. Collins M. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In: Hajic J, Matsumoto Y, editors. Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing; 2002 Jul 6–7; Prague, Czech Republic: Association for Computational Linguistics; 2002. pp. 1–8. DOI
30. Ma J, Hinrichs E. Accurate linear-time Chinese word segmentation via embedding matching. In: Zong C, Strube M, editors. Proceedings of

the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing; 2015 Jul 26–31; Beijing, China: Association for Computational Linguistics; 2015. pp. 1733–43. DOI

31. Deng X, Sun Y. An improved embedding matching model for Chinese word segmentation. In: Wang X, Zhou J, editors. 2018 International Conference on Artificial Intelligence and Big Data; 2018 May 26–28; Chengdu, China: IEEE; 2018. pp. 195–200. DOI
32. Zhang Q, Liu X, Fu J. Neural networks incorporating dictionaries for Chinese word segmentation. In: McIlraith SA, Weinberger KQ, editors. Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence; 2018 Feb 2–7; New Orleans Louisiana USA: AAAI; 2018. pp. 5682–9. DOI
33. Ye Y, Li W, Zhang Y, Qiu L, Sun J. Improving cross-domain Chinese word segmentation with word embeddings. In: Burstein J, Doran C, Solorio T, editors. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies; 2019 Jun 2–7; Minneapolis, Minnesota: Association for Computational Linguistics; 2019. pp. 2726–35. DOI
34. Tang X, Huang Y, Xia M, Long C. A multi-task BERT-BiLSTM-AM-CRF strategy for Chinese named entity recognition. *Neural Process Lett* 2023;55:1209–29. DOI
35. Huang W, Cheng X, Chen K, Wang T, Chu W. Towards fast and accurate neural Chinese word segmentation with multi-criteria learning. In: Scott D, Bel N, Zong C, editors. Proceedings of the 28th International Conference on Computational Linguistics; 2020 Dec 8–13; Barcelona, Spain: International Committee on Computational Linguistics; 2020. pp. 2062–72. DOI
36. Tian Y, Song Y, Xia F, Zhang T, Wang Y. Improving Chinese word segmentation with wordhood memory networks. In: Jurafsky D, Chai J, Schluter N, Tetreault J, editors. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics; 2020 Jul 5-10; Online: Association for Computational Linguistics; 2020. pp. 8274–85. DOI
37. Liu A, Du J, Stoyanov V. Knowledge-augmented language model and its application to unsupervised named-entity recognition. In: Burstein J, Doran C, Solorio T, editors. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies; 2019 Jun 2–7; Minneapolis, Minnesota: Association for Computational Linguistics; 2019. pp. 1142–50. DOI
38. Jing L, Tian Y. Self-supervised visual feature learning with deep neural networks: a survey. *IEEE Trans Pattern Anal Mach Intell* 2021;43:4037–58. DOI
39. Jaiswal A, Babu AR, Zadeh MZ, Banerjee D, Makedon F. A survey on contrastive self-supervised learning. *Technologies* 2021;9:2. DOI
40. Liu X, Zhang F, Hou Z, et al. Self-supervised learning: generative or contrastive. *IEEE Trans Knowl Data Eng* 2023;35:857–76. DOI
41. McDonald D. Large-scale kernel machines. In: Bottou L, Chapelle O, DeCoste D, Weston J, editors. Scaling Learning Algorithms toward AI. Cambridge: MIT Press; 2007. pp. 321–59.Available from:https://ieeexplore.ieee.org/servlet/opac?bknumber=6267226.
42. Bengio Y, Lamblin P, Popovici D, Larochelle H. Greedy layer-wise training of deep networks. In: Schölkopf B, Platt JC, Hoffman T, editors. Proceedings of the 19th International Conference on Neural Information Processing Systems; 2006 Dec 4-7;Canada: MIT Press, 2006. pp. 153–60. Available from: https://ieeexplore.ieee.org/document/6287632.
43. Hinton GE, Osindero S, Teh Y. A fast learning algorithm for deep belief nets. *Neural Comput* 2006;18:1527–54. DOI
44. Ioffe S, Szegedy C. Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Bach F, Blei D, editors. Proceedings of the 32nd International Conference on International Conference on Machine Learning; 2015 Jul 6-11; Lille, France: JMLR.org, 2015. pp. 448–56. DOI
45. Nair V, Hinton GE. Rectified linear units improve restricted boltzmann machines. In: Fürnkranz J, Joachims T, editors. Proceedings of the 27th International Conference on International Conference on Machine Learning; 2010 Jun 21-24; Haifa, Israel: Omnipress, 2010. pp. 807–14. DOI
46. Giorgi J, Nitski O, Wang B, Bader G. DeCLUTR: deep contrastive learning for unsupervised textual representations. In: Zong C, Xia F, Li W, Navigli R, editors. Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing; 2021 Aug 1-8; Online: Association for Computational Linguistics, 2021. pp. 879–95. DOI
47. Devlin J, Chang M, Lee K, Toutanova K. BERT: pre-training of deep bidirectional transformers for language understanding. In: Ammar W, Louis A, Mostafazadeh N, editors. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies; 2019 Jun 1-8; Minneapolis, Minnesota: Association for Computational Linguistics, 2019. pp. 4171–86. DOI
48. Fang H, Wang S, Zhou M, Ding J, Xie P. CERT: contrastive self-supervised learning for language understanding. [Preprint]. arXiv. June 18 2020 [accessed 2023 July 21]. Available from: https://doi.org/10.48550/arXiv.2005.12766.
49. Yang B, Mitchell T. Leveraging knowledge bases in LSTMs for improving machine reading. In: Barzilay R, Kan MY, editors. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics; 2017 Jul 30–Aug 4; Vancouver, Canada: Association for Computational Linguistics; 2017. pp. 1436–46. DOI
50. Lample G, Ballesteros M, Subramanian S, Kawakami K, Dyer C. Neural architectures for named entity recognition. In: Knight K, Nenkova A, Rambow O, editors. Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies; 2016 Jun 12–17; San Diego, California: Association for Computational Linguistics, 2016. pp. 260–70. DOI