FN-Bert: A few-shot Bert-based Model for Chinese Nested Medical NER

Zhang Le (2030026202) Zhang Siruna (2030026215) Li Xiaole (2030026077)

Deng Shenhua (2030026026) Luo Hengyi (2030026102)

***Abstract***—When it comes to Nested Named Entity Recognition (NER), we face a challenging task in natural language processing. In this task, entities can have hierarchical or nested structures. Traditional NER methods struggle with identifying nested entities because a token can have multiple entity types. Additionally, another major challenge in nested NER is the lack of annotated training data. Annotating nested entities requires more labeling effort than traditional NER, which often leads to limited availability of labeled data. The scarcity of training data makes it difficult to train accurate and robust nested NER models. To address these challenges, we thoroughly investigated traditional approaches in the field of nested entities and few-shot learning. We proposed a model called FN-Bert, which is based on Bert-CRF, to tackle the challenges of nested entities and the scarcity of training samples. In our proposed model, we combined a multi-layer pointer network to accelerate training speed and enable the model to recognize nested entities by using a semi-pointer-semi-label entity annotation structure. For few-shot learning, we utilized a general information extraction model called UIE, a large language model, as a teacher model to distill the required knowledge through self-training. In terms of results, we compared our model with baseline models for the NER task on a dataset of nested entities in Traditional Chinese Medicine instructions. Our model demonstrated significant improvements in both running efficiency and accuracy.

***Index Terms***—Nested Named Entity Recognition, Few-shot learning, multi-layer span pointer network, UIE.

# INTRODUCTION

## Background and Importance

A

s early as the 1970s, experts and scholars first tried to introduce AI technology into traditional Chinese medicine (TCM), but with limited success. The Development Plan for the New Generation of artificial Intelligence Industry issued by The State Council in 2017 proposed innovative applications that deeply integrate medical treatment and artificial intelligence. The use of modern information technologies such as AI to boost the development of TCM has also been written into national guidelines.

Ma Jianpeng, dean of the Multi-scale Institute for Complex Systems at Fudan University, said that the outstanding ability of AI in data mining and deep learning can help establish objectified standards and evaluation systems for TCM diagnosis and treatment, broaden TCM Internet application scenarios, and better inherit and develop TCM culture. Therefore, the combination of artificial intelligence and TCM is the key to improve the universality of TCM and promote its development in the future.

## Description of Knowledge Gap

The combination of TCM and AI should not stop at the simple superposition of elements or the appropriation of paradigms. Dean Ma believes that only being on the premise of a thorough understanding of the basic characteristics of traditional Chinese medicine, finding the meeting point of modern technology and traditional culture, and strengthening the adaptive transformation and utilization of technology, can make artificial intelligence our own.

According to Dean Ma, the recently popular ChatGPT conversation on traditional Chinese medicine found that although the emerging artificial intelligence program can answer the objective basic knowledge of traditional Chinese medicine fluently, it still has great shortcomings in clinical diagnosis and is prone to medical ethical problems, and there is still a long way for it to go in the future.

## Topic of Research Paper

The research topic of this paper is a QA Chatbot based on traditional Chinese medicine. Among that, the information extraction part of TCM text is the core part of the construction of TCM knowledge map, which lays the foundation for the upper application, such as the construction of Clinical Decision Support System (CDSS). This NER challenge requires extracting the key information in the instructions of traditional Chinese medicine, including 13 entities such as drugs, drug ingredients, diseases, symptoms, and syndromes, to build a knowledge base of traditional Chinese medicine.

## The Core Approach and Principal Findings

The main research methods are named entity recognition and entity relation extraction through the deep learning model of TCM text information. We chose it because the text information in the field of TCM is relatively complex and professional, which is difficult for traditional rule-based methods to deal with. The deep learning model can automatically discover features and rules through the training of large amounts of data, so as to improve the accuracy of entity recognition and relationship extraction. The findings show that the TCM named entity recognition and entity relationship extraction system based on deep learning model has high accuracy and practicability, and can provide effective support for the research and practice of TCM.

## The Core Approach and Principal Findings

In the field of TCM, the application of natural language processing technology is still in its initial stage, and there may be some errors in the diagnosis and prescription of the disease. Therefore, this project aims to explore how to improve the accuracy and humanization of the QA Chatbot of traditional Chinese medicine text information in clinical diagnosis, so as to further improve the CDSS of traditional Chinese medicine and the effect of clinical diagnosis and treatment.

In order to achieve this goal, our study will be carried out from the following aspects:

1. Analyze the shortcomings of Chatbot in clinical diagnosis.
2. Explore how to use NLP, deep learning and other related technologies to improve the accuracy and humanization of Chatbot in clinical diagnosis and treatment.
3. By analyzing the data in the practical application of Chatbot in TCM, explore the solutions to its problems.

4. Based on the above research results, an improved model and algorithm for clinical diagnosis of QA Chatbot, which is suitable for TCM text information, is proposed, and its effectiveness and feasibility are further verified.

Through the above measures, we hope to provide a more accurate, more convenient and more humanized solution for TCM clinical diagnosis and treatment, and make more positive contributions to improving the effect of clinical diagnosis and treatment.

# Literature Review

# **[[1]](#footnote-0) Overview of NER**

Named Entity Recognition (NER) refers to the recognition of text fragments belonging to predefined categories from free text. It mainly covers two tasks: 1) Edge Detection 2) Type Recognition.

The NER task was first proposed by the Sixth Message Understanding Conference [13], when only a few generic entity categories were defined, such as places, institutions, people and such coarse-grained entity types. At present, named entity recognition task has penetrated into various vertical fields and focus more on fine-grained entity type. Therefore, in the perspective of the number of entity types, NER can be classified into: coarse-grained and fine-grained tasks. But according to entity type, Named Entity Recognition also can be classified into two categories: generic and domain-specific.

Looking through the development of NER, there’re four mainly stages:

### Rule-based Approaches

The rules-based NER tasks rely on manual rules. Rules can be designed based on domain-specific dictionary and syntactic-lexical patterns[14]. But Flaws of it are relatively obvious. It works well when dictionaries are exhaustive. However, high accuracy and low recall rates are often observed from such systems due to domain-specific rules and incomplete dictionaries. What’s more, it’s hard to apply this model to other fields.

### Unsupervised Learning Approaches

The key idea of this method is clustering[15]. The commonly used features or auxiliary information include lexical resources, corpus statistics (TF-IDF), shallow semantic information (block NP-chunking), which are calculated on large corpora. Based on it, people extract named entities from clustering groups according to context similarity.

### Feature-based Supervised Learning Approaches

When supervised learning is applied, it can be regarded as multi-classification or sequential labeling. It takes advantage of machine learning algorithms to learn model and recognize similar patterns.

Feature engineering is extremely important in supervised NER tasks. Common feature representation: 1) Word-level features[16] (POS or morphology) 2) list lookup features (Wikipedia gazetteer or DBpedia gazetteer) 3) Document and Corpus features (local syntax or multiple occurrences).

Classical machine learning algorithms include Hidden Markov model (HMM), decision tree (DT), support vector machine (SVM) and conditional random field (CRF).

### Supervised Learning method based on deep learning

The core advantages of deep learning are the non-linear mapping ability of DL from input to output. It enables DL to learn high-dimension potential semantic information. In addition, it saves the workload of designing NER features which requires domain knowledges. The model can effectively learn representation and potential factors from raw data. Finally, it can train an end-to-end model, provide chances for us to design complex NER tasks.

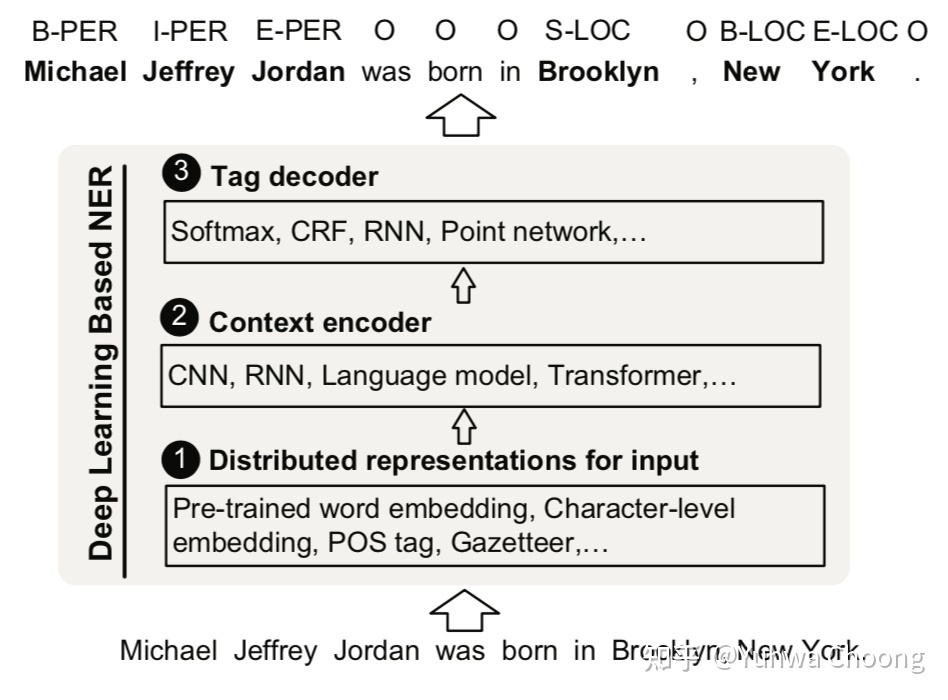
Supervised Learning method based on deep learning often contain three stages, as shown below:

FIG. 1. Distributed Representation for Input

### (a) Word-level representation

### Mikolov first came up with word2vec (two frames, CBOW and skip-gram), then is Stanford's Glove[17], then is Facebook's fasttext and SENNA at last. Using these word embedding methods, some research works use different corpus for training, such as PubMed and NYT in biomedical field.

### (b) Character-level representation

### Character level usually refers to the open embedding of English or other languages with natural separators. In Chinese, character level embedding can mainly reduce OOV rate. In this paper, two common embedding methods of character level are given, which are CNN and RNN.

### (c) Hybrid representation

### In addition to word-level representation and character-level representation, some researches have also embedded some other semantic information, such as lexical similarity, POS tagging, segmentation, semantic dependence, Chinese character parallelism, Chinese pinyin, etc. In addition, some studies start from multimodal learning and embed visual features through modal attention mechanisms. The paper also puts BERT in this category, and regards location embedding, token embedding and segment embedding as mixed information representations.

*B. Context Encoder*

### (a) CNN

### A word is represented as an N-dimensional vector through the embedding layer[18]. Subsequently, the whole sentence is encoded by convolution (usually one-dimensional convolution) to obtain the local features of each word. It can also be sent to the decoding layer together with local feature vectors.

### (b) RNN

### The commonly used cyclic neural networks include LSTM and GRU, and bidirectional network BiRNN is often used in NLP to extract problem features from left to right and right to left. A new network structure, ConvLSTM, which is not included in the paper, is a better combination of CNN and RNN, but the samples need to be reconstructed.

### (c) Recursive Neural Networks

### Compared with cyclic neural network, recursive neural network has tree hierarchy structure. A good feature of cyclic neural network is to process variable length sequences through the neuronal cyclic structure, while it is difficult to model the data with tree or graph structure (such as syntactic parse tree). Besides, its training algorithm is different from conventional Back Propagation algorithm, but adopts BPTS (Back Propagation Through Structure). Although recursive neural networks feel good in theory, they are not so good in practice and are difficult to train. In contrast, the research of tree LSTM in recent years has been frequently mentioned and has many applications in relation extraction and other tasks.

### (d) Transformer

### An article called Attention is all you need in Google has pushed attention mechanism to the top of the new wave[19]. Meanwhile, transformer, a pure stacked self-attention, dot product and feedforward neural network structure that does not rely on CNN and RNN structure, is also known. Since then, transformer has proven to be better at long distance text dependencies than RNN.

C Tag Encoder

It inputs context and generates tags which correspond to input sequences.

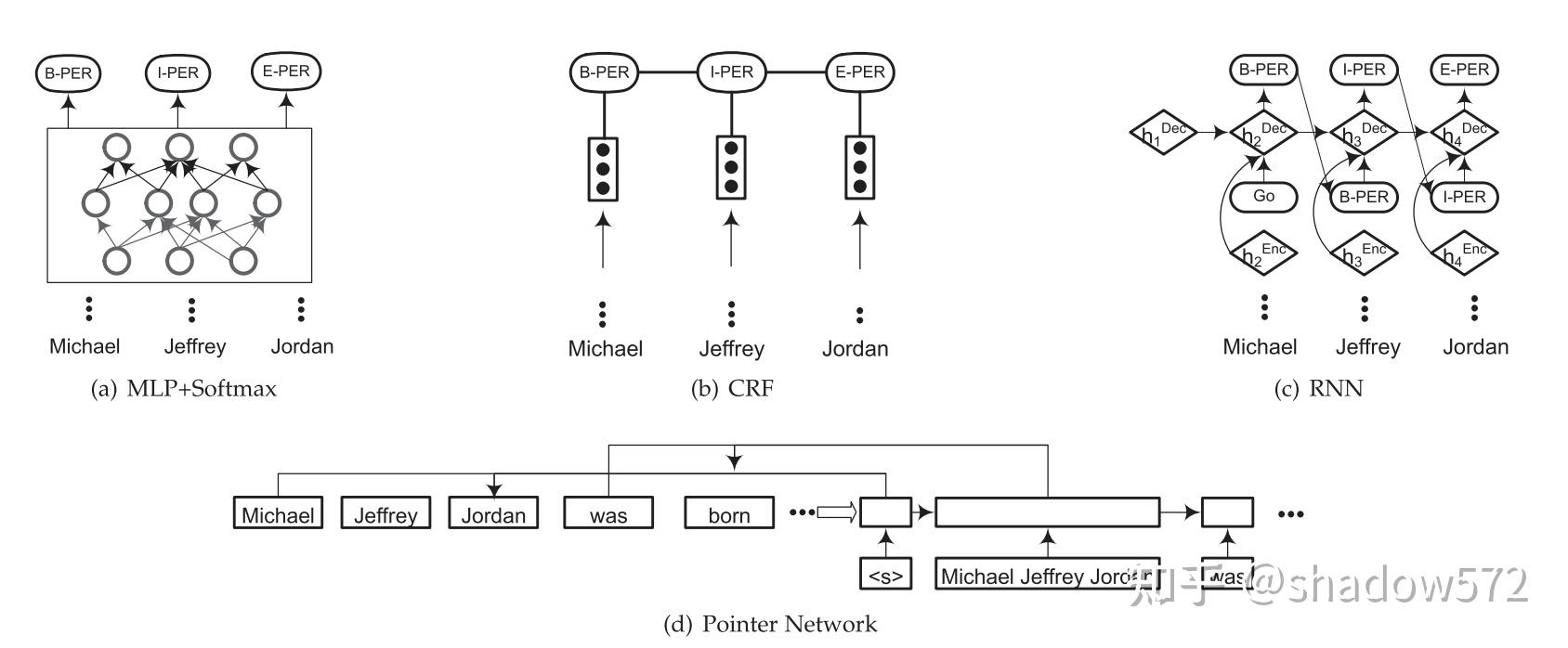


Fig. 2. Tag Encoder

(a) MLP+Softmax

It can transform Tag labeling task to multi-classification. Every word’s tags are predicted according to context and mutually independent, ignoring their neighbors.

(b) CRF

CRF is the most common choice for tag decoders. It’s a random field with observation sequence as global condition and has been widely used in a feature-based approach as well as in deep learning based NER models. It often appears above the bidirectional LSTM layer and CNN layer.

However, CRF cannot take full advantage of segment-level information because the internal properties of segments cannot be full encoded at the word level.

(c) RNN

It’s commonly accepted that RNN tag decoders are superior to CRF and it trains faster when the number of entity types is large.

(d) Pointer Network

Pointer networks apply RNN to learn conditional probabilities of output sequences, where elements are discrete markers that correspond to positions in input sequences. It represents variable-length dictionaries by using the softmax probability distribution as a "pointer". Zhai et al. firstly generate sequence labels using a pointer network. As shown in Figure, the pointer network first identifies a block (or segment) and then marks it. Repeat this until we have processed all the words in the input sequence.

Main evaluation methods: 1) Exact-match Evaluation 2) Relaxed-match Evaluation. When we do NER tasks, there’re always special cases: discontinuous entity and nested entity. Current method is to modify the structure of model, such as layered-based model[20], region-based model[21]. Other methods like machine reading comprehension or finding anchors works in some cases.

For NER task, the opportunities and future research directions of NER tasks are multiclass entity, nested entity, entity identification and entity link federated task, Informal Text NER based on deep learning using auxiliary resources (in addition, knowledge graph direction), NER model compression and deep transfer learning for NER.

To sum up, the existing research of NER is relatively rich, and it is crucial to choose a novel combination if one wants to blossom again.

# **Overview of Nested Named Entity**

In this section, we will introduce the problems of traditional NER in solving nested NER tasks, and how to deconstruct NER tasks to solve problems from different perspectives so that the model can recognize nested NER, and introduce representative solutions in the field of nested NER in recent years.

There is no way to solve the nesting case with a Flat NER, because in a nested NER, a token might have two different types. For example, "North" in "Peking University" belongs to both B-Location and B-Organization; "Beijing" also has two tags: I-Location and I-Organization.

# **How to deconstruct NER tasks**

## Change the target of the classification task from single tag to multi-tag

An easy solution to this problem is to change the output from a single class to a multiclass with no Schema change and no model change: that is, at the end of the classification, from a single class to all classes that meet a specified threshold. More specifically, the following two schemes exist:

[1] Does not change the Schema at all, only when the training set is input, the labels in the training set change from the original one-hot encoding form to a uniform distribution of a specified category; Change the loss function to BCE or KL-divergence in training; When reasoning, given a hard threshold, all categories above this threshold will be predicted as the token class. (Katiyar & Cardie, 2018)

[2] To modify the Schema, common in all categories will be two combination, produce new labels (such as: B - the Location and B - Organization together, constructing a new tag B - Loc | Org); The advantage of this is that the final classification task is still a single classification because all possible classification targets are covered in the Schema. (Straková, et al., 2019)

We believe that during the years of exploration, this scheme has been studied by scholars, because it is simple and easy to do, with little changes; However, except for two articles in NAACL18 and ACL19 that explore these solutions in detail, we have rarely seen a paper that uses this approach to solve the problem. Because it has some obvious problems: only for the first realization way, it is too difficult to set the goal of model learning, and the definition of threshold value is subjective and difficult to generalize. Only for the second implementation, the label is increased exponentially, resulting in too sparse distribution, difficult to learn; For multilevel nesting, a lot of composite tags need to be defined; And the original problem: the modified Schema predicts results that are no longer unique when restored to entities.

## Decode process for modifying models

In this context, Decode refers to the process of classifying tokens based on token representation output from the model. FFN + Softmax/CRF + Argmax is a set of operations in Sequence Labeling. The first implementation of A is also, strictly speaking, a naive modification of the Decode process, but we discuss some more efficient solutions here. It is worth noting that the Decoder modification is designed to ensure that multiple categories can be assigned to a token at the same time, so we still regard the following scheme as a task of Sequence Labeling (although the final output label list length may be different from the number of tokens, But this is because of the original single classification into multiple classification inevitably caused).

Since direct use of FFN mapping for single classification cannot solve the nesting problem, and multi-classification is not easy to do work, can we consider using generative methods, such as Decoder in seq2seq, to generate each token tag one by one? The Decoder is able to unbind the number of incoming tokens from the number of output categories, allowing more than one token to be tagged -- but unlike the original generation method, in addition to using a special character [EOS](end of sentence) to mark the completion of the generation process, We need to introduce a special character [EOW](end of word) to indicate that the tag generated next belongs to the next token. (Straková, et al., 2019)

It is also a very interesting scheme to use hierarchical method to predict token representation: if a classification fails to solve the problem of entity nesting, it will continue to classify the result of the first classification and iterate accordingly until the maximum number of iterations is reached or no new entities are generated. The problem with this solution is that it requires a high level of Decoder learning, and if there is a misjudgment in the previous iteration, this problem may be passed on to the subsequent iteration.(Straková, et al., 2019)

Compared with common multi-label classification, these methods are designed from the perspective of the task itself. Through horizontal (Sequence generation) and vertical (hierarchical Labeling), the original Sequence Labeling model is modified to enforce the binding form of input token and output label.

## Discard Sequence Labeling

As we have mentioned several times above, the sequence tagging task does not naturally support assigning multiple tags to a token, although we have modified it at several levels to enable it to be applied to multiple tag classifications. But since it is not as effective or fungible when applied to a Nested task, why not just discard the task form and try another solution?

We can mitigate these problems with artificial rules or Settings, such as: Model training is difficult: training a classifier separately for each type; Space-time complexity: Assuming that the longest entity length is n, only enumerate all cases with length less than n when fully enumerating subsequences; A large number of negative samples: before performing classification, one or more general filters should be trained, or a batch of negative samples should be screened first by manual rules; Sample negative samples during training instead of using all of them.

In fact, almost all current solutions that eliminate Sequence Labeling are looking for a good 2-stage model and working to mitigate these problems. Typically, we can easily build the model described above using existing model skeletons (Bi-LSTM, ELMo, etc.). (Xia, Congying, et al., 2019) Seven bi-LstMs and one ELMo were used to extract context information at different levels, and a Self-Attention was used to enhance the representation of each subsequence by context.

The scheme proposed in this paper is close to the naive proposal above except for the constraint of full enumeration with length R, so it will be more difficult to work with it. In this paper, the author introduces multiple Bi-LSTM models in order to learn different levels of information, which essentially introduces more parameters and external information to solve this problem.

After the token obtains global information through 3 Bi-LSTM+1 ELMo, the author maps the output si of token to a 2R space through a fully connected network, and then represents: The first R values of this vector represent the probability that the R subsequence before and after the token is a reasonable entity candidate, and the last R values of this space represent the probability that the R subsequence before and after the token is not a reasonable entity candidate. Comparing these two probabilities, we are able to filter a total of R candidate entities that contain this token.

Straková, et al. (2019) gives a solution similar in idea to Xia, Congying, et al. (2019). However, instead of full enumeration, Fisher, Joseph, and Andreas Vlachos (2019) chooses to obtain candidate entities from another Angle: to predict the probability of connection between two tokens. If the probability of connection between two tokens is high (reflected in the paper as the value tends to 0), it is considered that they are more likely to be in the same entity at the current level. Multiple layers of nested information can then be identified by iteratively updating the representation of each token in the entity.

Fig. 3.

Straková, et al. (2019) puts forward a hypothesis: in an entity, there will always be one or more tokens that are the anchor point of the entity (that is, if the token appears, there is a fair probability that the token is within an entity. The author takes the department in The department of [xxx] as an example to explain that it is very likely to appear in an entity of type Organization). From this, we can turn the task of identifying nested entities into the task of finding an anchor point: first finding an anchor point and deciding what category it represents, then finding the boundary of the entity in which the anchor point resides.

We regard the task of finding entities as a Machine Reading Comprehension (MRC) task, that is, we inquire whether there is an answer to the specified question in the sentence. Here the question represents a query of a specified category (e.g., Is there any Location in this sentence?). As the answer to the question, the model outputs the start and end positions of all corresponding entities in the sentence. (Li, Xiaoya, et al, 2019)

Inspired by the process of constructing a statement parse tree, the identification of nested entities is regarded as a process of constructing a parse tree: At each time step, based on the current state, the model decides whether to assign a tag to the specified token, to assign a higher level tag to the two entities that have already been tagged (thus achieving tag nesting), or to skip the tokens currently being processed. Such an operation is more like an RNN unit with an internal conditional statement, which can handle a token in different ways depending on the current situation: label it or leave it unprocessed. (Wang, Bailin, et al., 2019)

# **Representative solutions in the NER field**

## Token-by-token resolution: A transition-based approach

Wang, Bailin, et al., (2019) State transition-based approaches occupy a place in the Nested NER solutions proposed in the last three years. This approach is somewhat like the finite automaton of compilation principle, and is very similar in that they do parse some input. If we can parse a sentence in the form of a state transition to extract all the nested entities from it, the first thing we need to work out is how to put the previous information in the current state as well. Happily, there is one structure that naturally has this property: the time series model.

Inspired by the shift-reduce parsing in constituency parsing, the authors devised a system for parsing the above sentences. Throughout this process, we need to maintain three structures: the Stack, and the top element of the stack is the one we are currently working with. Depending on the context and current state, we need to either ignore the element, label it, or compound it with a previous element and label it at a higher level. The queue (Buffer). In the queue are the remaining tokens in the sentence to be processed. And the Action. The operator is essentially a model that determines which parsing operation to perform based on the current system state.

There are three types of resolution operations: SHIFT: Removes a token from the head of the queue and moves it to the stack. It is important to note that this does not mean that we are skipping the processing of the current token. Given the stack's functional definition, the purpose of this step is to actually start processing the token; Unary-X: Pops the top element, labels it [X], and pushes it back onto the stack; Reduce-X: Pop top elements twice, label them [X], and push them back onto the stack.

As you can easily see from the above, only the Reduce operation gives entities a higher level, namely nested labels. Assuming the current input sentence: Indonesian leaders visited him, a correct sequence of parsing operations would look like the Action column in the following figure:

In summary, this paper regards the text content to be processed, the text result to be processed, and the Action operation to be performed as a system, Embed it into a vector to represent the current overall state, and then use the classifier to judge the next operation based on this state. Each of these steps takes place at the Embed level.

## Method based on Hypergraph

Now let's revisit the sentence: “... that his fellow pilot Dabid William had ...”. And nested annotation results (where L=Last=End=E, U=Unit=Single=S) :

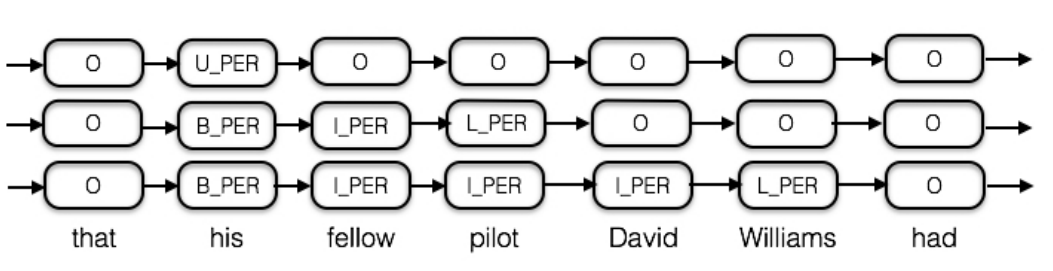


Fig. 4. Visualization on Hypergraph input

As you can see, tokens like his, although nested within three entities, will only have two values. We combine the same values of each token to get the Hypergraph shown below:

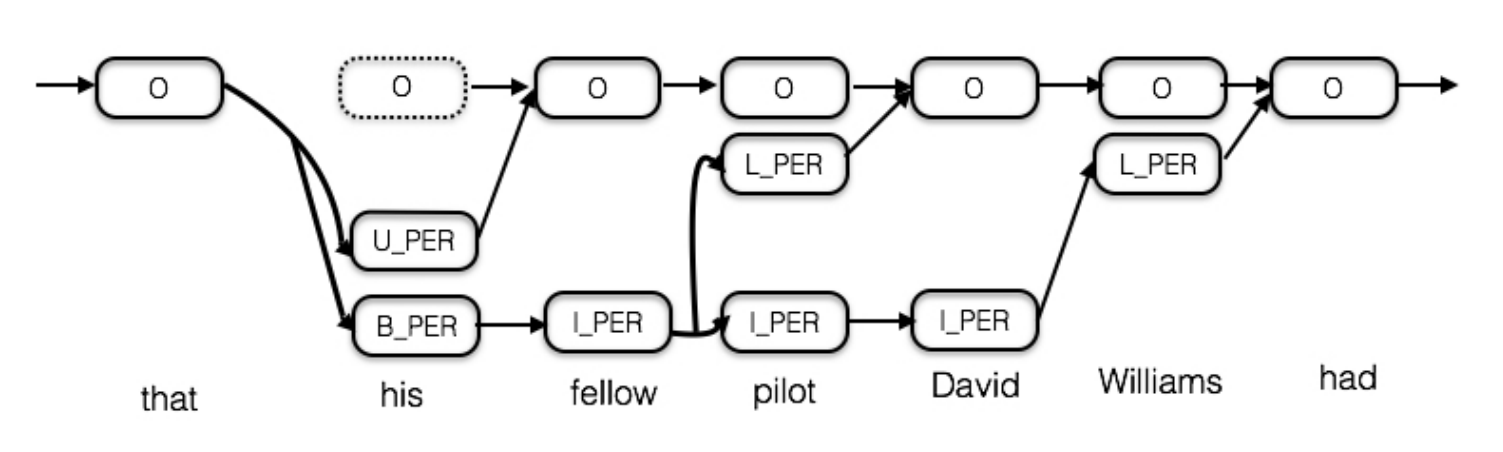


Fig. 5. Visualization on Hypergraph processing

It is important to note that we need to ensure that each token has at least one O tag. The purpose of this is to effectively model the probability of starting each new entity. As shown in the figure above, if a token does not have an O tag, we need to add a void O to it.

Next, we expect the trained model to be able to give an input token and output all its possible tags. For example, the input his model should output UPER and BPER (O is not directly predicted here, but we will discuss how the added void O should be added to the training process later). We can use a Decoder to implement this process. This Decoder uses the LSTM Unit.

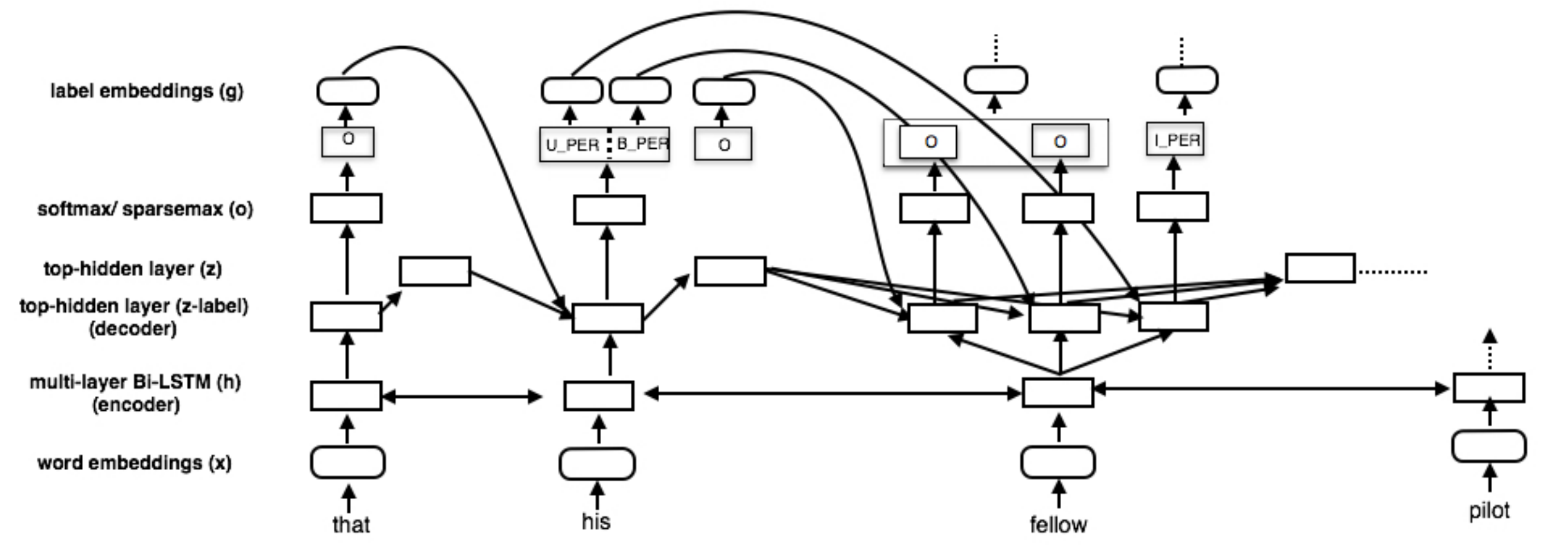


Fig. 6. workflow of Hypergraph

sparse softmax and softmax were also compared in this paper. sparse softmax was used because the author believed that most output tags in each time step would be sparse, so the use of sparse Softmax would produce better results. sparse softmax improved ACE2004 and ACE2005 by more than 5 points compared with normal softmax in the following experiments.

## Methods based on reading comprehension

Shannon.AI put forward the idea of using NER task as MRC around the middle of last year [7]. This method is currently located in SOTA in ACE04&05, GENIA and KBP2017 nested entity recognition datasets, especially in the latter two datasets. In addition, the author also conducted tests on the Flat NER task in both Chinese and English, and obtained SOTA results. MRC tasks can generally be formalized as: given a piece of information (passage) and given a question (question), the model needs to find the phrase (span) from the information to answer this question. From a task understanding point of view, the solution seems to be very similar to NER: we just need to think of each label X as a question: what is the entity of the X label in this sentence? Then enter the sentence of the entity to be extracted and ask, and then use the phrase output by the model as the entity of the X label in the sentence. BERT naturally allowed two different sentences to be entered into the model at the same time, due to the task design during pre-training. We enter questions and information into BERT in the following form:

[CLS] Question [SEP] Passage [SEP]

Are used to identify the beginning and interval of an input sentence, respectively. And just wait for the model to give us the answer. Three classifiers are trained to predict several problems after model output, which are: whether the current token is the beginning position of an entity; Whether the current token is the end location of an entity; Whether the two positions in the first two classifiers match (if the beginning position i matches the end position j, we assume that the positions from xi, xi+1... xj minus 1. xj is an entity. As a result of the design of the above tasks, the loss function of the whole model is determined by the sum of the cross entropy losses of the initial position recognition, the end position recognition and the position matching, namely:

## 

An article posted on arxiv on April 26, 2023 was the first article our group saw in which large-scale Language Models (LLM) was used to solve NER task. In our opinion, LLM is the optimal solution to NER problem, especially in small sample scenes, LLM with rich prior knowledge always amazes me with its emergence ability.

For Neurolinguistic model, the paper says the language model does one thing: Determine if the language is reasonable. Its development history: .

Existing neural language models: word2vec, Glove, fasttext, ELMO, BERT, GPT, GPT2, XLNET, ALBERT and RoBERTa.

# **Overview of Few-shot**

As a pivotal task of information extraction, NER is essential for a wide range of technologies [7][12][30][31]. Nowadays, many computer scientists have invented different methods of NER datasets. For instance, CoNLL’03 [10] is regarded as one of the most popular datasets, which is curated from Reuters News and includes 4 coarse-grained entity types.

1. *Self-training*

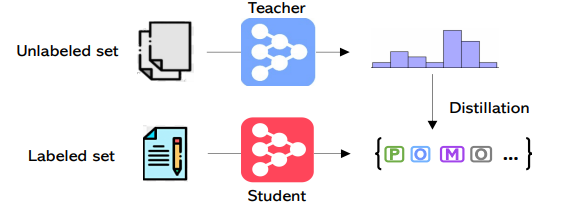


Fig. 7. Self-training: An NER system (teacher model) trained on a small labeled dataset used to predict soft labels for sentences in a large unlabeled dataset.

Self-training [21] is a strategy to leverage unlabled in-domain data [9]. It is one of the earliest semi-supervise methods. What is more, in the field of visual object detection, neural machine translation and so on, this vary approach also can achieve high performance quality.

The stronger data augmentation and more labeled data can diminish the value of pre-training, when self-training is always helpful in both low-data and high-data regimes [9]. Now, we focus on its performance in a few-shot named entity recognition. In this strategy, through self-training, unlabeled in-domin data was used to predict the soft labels. Also, few-shot learning can be improved by self-training, if the ratio of the unlabeled data is relatively high.

1. *Proto NER*

There are already some solutions to the few-shot NER. One is semi-supervised learning, another solution is in an unsupervised way. However, both of the methods can only provide general solutions, which means the solutions they provide cannot be appropriate, new methods that are able to collect more task-specific information.

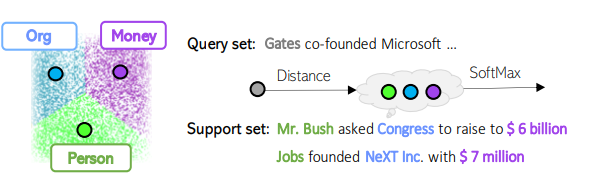


Fig. 8. A prototype set is constructed via averaging features of all tokens belonging to a given entity type in the support set (e.g., the prototype for Person is an average of three tokens: Mr., Bush and Jobs).

A prominent approach to the task of learning from only few examples is metric learning[11], which has some features of semi-supervised learning. Because in this method, they use the old knowledge learned from the common data to label the new uncommon ones without seeing from examples first. In this approach, even zero-shot NER can be achieved.

Prototypical network is a research conducted by Snell et al., it can be used when labelled samples are coarse to classify settings, which can classify objects into task-specific clusters by mapping objects into vectors.

In this method, there are mainly two architectures — the commonly used RNN

baseline and a prototypical network adapted for the NER task (Fritzler, A., Logacheva, V., & Kretov, M. , 2018). In the prototypical network, the main change is that they map the hidden states to M-dimensional spaces in the feed-forward layer, which originally map to logits corresponding to labels.

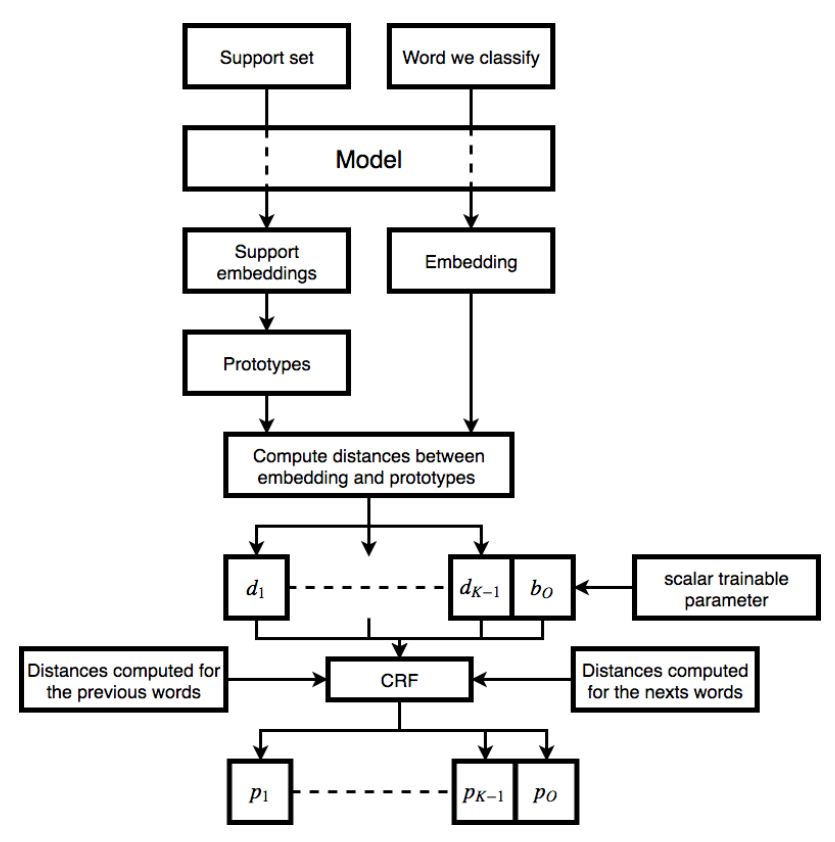
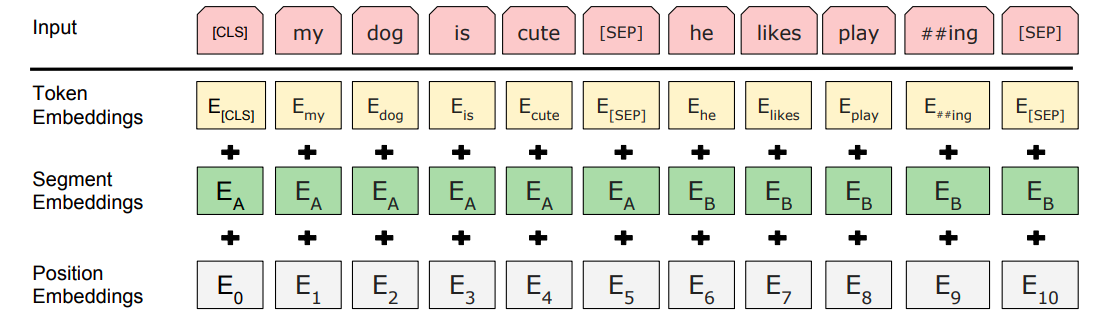


Fig. 9. The architecture of the prototypical network model

Because the majority of entities contain only one word, the quality of predicting entities can be not precise, if there are only small number of instances for training or too few entities( like 20 entities) in total.

This very method achieves F1 scores of over 60 in almost all categories, however this approach mainly focuses on predicting entities for one class. Perhaps, more efforts should be paid in training data from all classes.

1. *Noisy Supervised Pre-training*

This method can greatly improve NER accuracy. Now, a large-scale web has been applied in noisy supervised pre-training (NSP). Though introducing inevitable noises , this automatic annotation procedure is highly scalable and affordable (J et al., 2020). Proposed NSPs classify entities by learning representations, preventing over-fitting by the prior experience learnt from extracted entities.

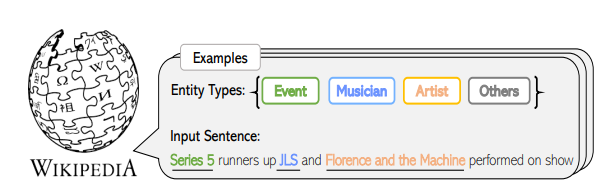


Fig. 10. The Wikipedia dataset is employed for supervised pre-training, whose entity types are related but different

# Methodology

**Baseline**

## Baseline model: BERT-CRF

BERT[28] is designed to pre-train the deep bidirectional representation of unlabelled text on the transformation of all layers of bidirectional context. The pre-trained BERT model can be fine-tuned with only one additional output layer to suit for a variety of tasks, such as problem solving and language inference, thereby reducing the need for elaborate specific architectures for NLP tasks. BERT is the first fine-tuning based model to achieve state-of-the-art performance on a range of sentence-based and character-level tasks, outperforming many task-specific architectures.

BERT has three types of word embedding here. Before fed into Input, transform words into the following format: token embedding + segment embedding +position embedding.

Fig. 11 Self-Attention Mechanism

##### (1) Token Embedding: It is necessary to use the Tokenization tool of BERT for word segmentation before it’s fed into the model. For example, after word segmentation, "Playing" is converted to "Play" + "# #ing". And then feed into the Input.

(2) Segment embedding: In some tasks, two sentences will be fed into the Input together. Therefore, [SEP] is a marker to decide whether to separate two sentences. "[CLS]" is used to classify whether the two sentences in the input are dependent.

(3) Position embedding: Because there is no RNN or LSTM in the network structure, it cannot obtain the position information of the sequence, so it is necessary to construct a position embedding. BERT initializes a position embedding and learns it through training.

## CRF

Conditional random field (CRF)[29] was proposed by Lafferty et al in 2001. It is an undirected graph model combining the characteristics of maximum entropy model and hidden Markov model. CRF is often used in named entity recognition, part-of-speech tagging, gene prediction, noise reduction, and object detection problems.

CRF is mainly used for sequence labeling problems so that each frame in the sequential data is classified. Since it is classified, it requires encoding this sequence with CNN or RNN and then activate it with softmax in a full-connection layer. When we design labels, the target output sequence involves some context correlation. Therefore, CRF separates the correlation at the output level, avoiding putting these associations at the coding level, which hopes that the model will learn them on its own.

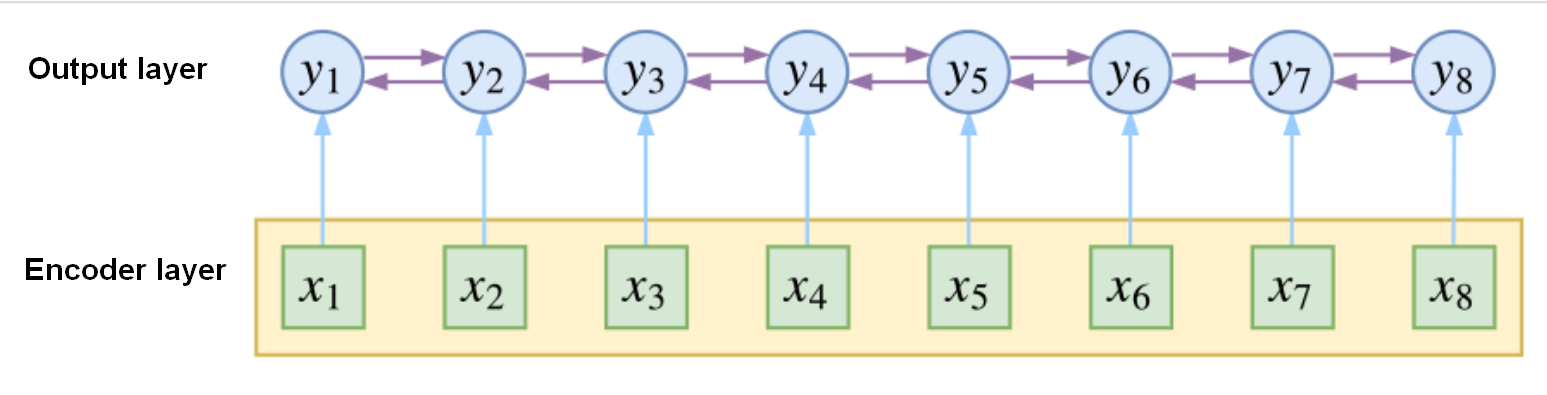


Fig. 12. CRF explicitly considers context at the output layer

The really neat thing about CRF is that it's based on the probability of the path and the path itself. Let's say an input has frames, and each frame is labeled with a (a possibility), then theoretically we have a different output (). It can be visualized as the following network diagram. In the figure below, each point represents the possibility of a label, the lines between the points represent the association between the labels, and each labeling result corresponds to a complete path on the graph.

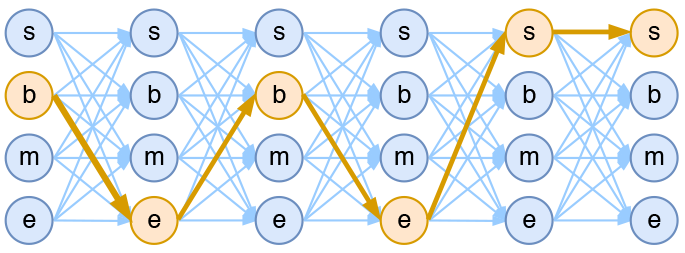


Fig. 13. Output network diagram in the 4-tag word segmentation model

Finally, to train the CRF model, it uses the maximum likelihood method as loss function, which is equal to:

In the CRF model, we can recursively calculate the normalization factor because it only considers the relation of the adjacent label (Markov hypothesis), which reduces the computational amount of the original exponential level to the linear level. As the picture shows:

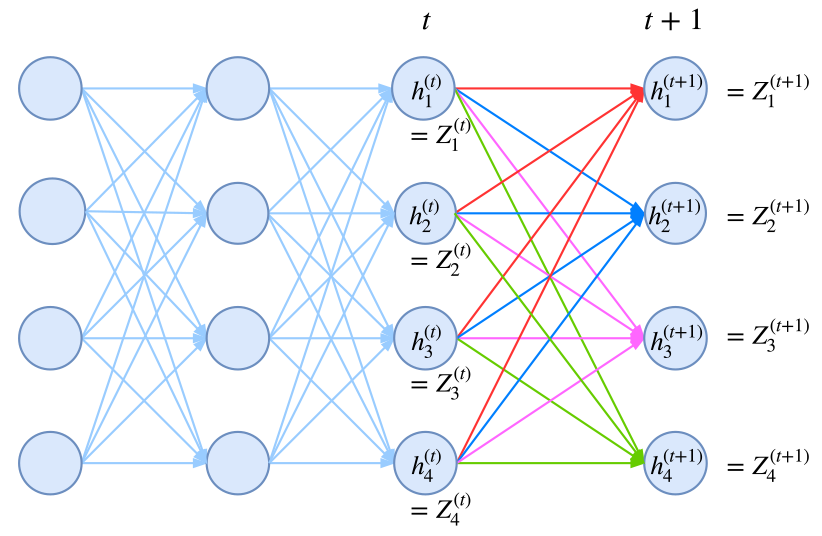


Fig. 14. Graph of recursive computation of normalized factors

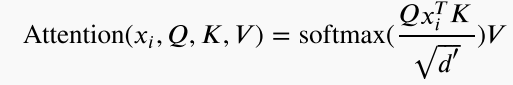
**BERT-Span model**

Here is the mathematical derivation of the BERT-Span model:

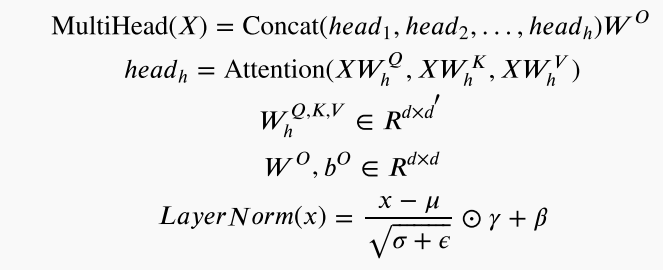
Firstly, the input of the BERT model is an input sequence consisting of tokens, where the representation of each token is denoted as ( i indicates the position of the token in the sequence) and is d-dimensional vector.

Next is the Transformer Encoder of BERT, which consists of layers, each with attention heads, and the dimension of each head is =*d/h*.

The Multi-Head Attention part is calculated as follows:

For a given input , the attention value of it in the *l*-th layer and *h*-th head is computed. *Q, K* and *V* are Query, Key and Value matrices, respectively, all with dimensions of × *n*. The matrices *Q* and *K* are obtained by a linear transformation of the input matrix *X*, and is the same as *X*.

The results of Multi-Head Attention are then concatenated, linearly transformed, and added to a residual connection, followed by Layer Normalization:



Where are the parameters of linear transformations, , , γ, β and σ are μ parameters of LayerNorm, and are the mean and variance of the input , respectively, and ⊙is the element-wise multiplication.

Then, the output of the Multi-Head Attention is fed into a feed-forward network:



Where , , , are the parameters of the two linear transformations in the feed-forward network.:

Finally, a residual connection and Layer Normalization are applied:



This completes the Transformer Encoder of BERT. Next is the computation of the BERT-Span model.

First, a linear transformation is performed on the output of the last layer of the Transformer Encoder:



Where and are the parameters of the linear transformation.

Then, for a given input and at positions i and j (i < j), the BERT-Span model's classifier computes

the probability of the span consisting of the two tokens:



Where is the representation of the span [i,j] [⋅, ⋅] denotes the slicing operation, and f is a combination of linear transformations and a feed-forward network.

Note that the specific form of the function can be adjusted according to the actual situation. A commonly used form is as follows:



Where , , , are the parameters of the two linear transformations in the feed-forward network.

**FEW-SHOT**

We firstly adapt the pseudo labeling, which is a kind of semi-supervised learning to improve model performances by unlabeled data. In this case, we use 5-shot data to adjust to the model and predict by the schema of 13 self-defined classes.

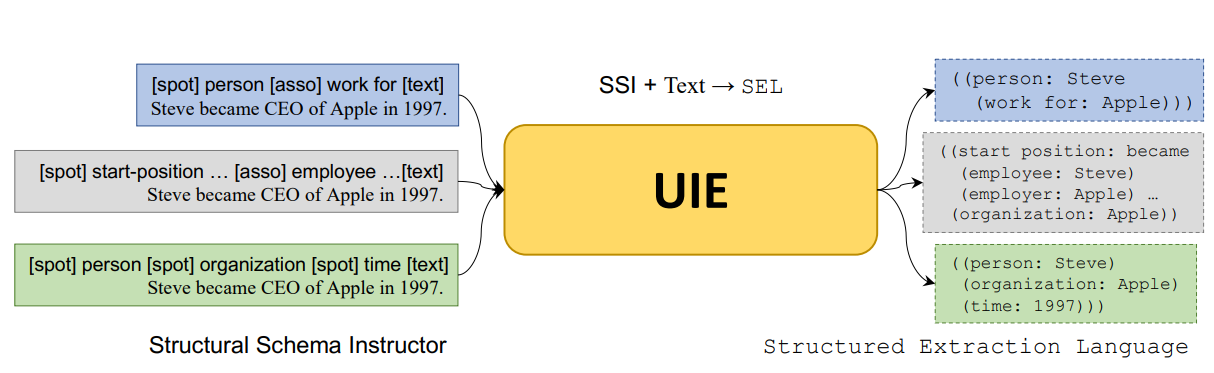
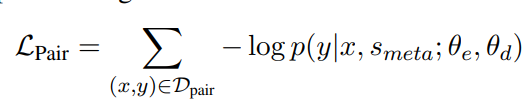
**

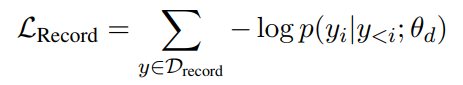
Fig. 15. The overall framework of UIE

However, as a man-machine-language, it is hard to apply a large amount of data.Luckily, as a large scale model, a valid prediction can be achieved t by using prompt mechanism in few-shot NER. We use it to print pseudo labels on unlabeled data through auto training.

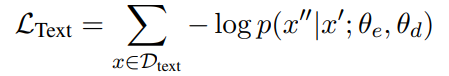
*Text-to-Structure Pre-training using* . We pre-train Universal Information Extraction (UIE) to reach the goal of capturing the text-to structure mapping ability ={(x,y)}. Then, we define spot type, associating type, positive schema as , and + for each parallel pair (x,y).To learn general mapping ability, we also automatically construct negative schemas for each pair, for example, we first sample negative spots and negative association set , then concatenate meta-schema = ∪ ∪ , and construct the final extraction target(Lu et al,. ). Finally, the objective of text-to-structure pre-training is:



*Structure Generation Pre-training with ,* which is an expression of structured extraction language (SEL).

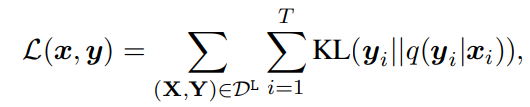


*Retrofitting Semantic Representation using* . Pre-training UIE model also with the masked language model tasks (Raffel et al., 2020) on to retrofit semantic representations of UIE.

*Final Pre-training Criteria.*The final objective is the combine of the above tasks:

*On-Demand Fine-tuning**.* Given the pre-trained UIE model, we can quickly adapt it to different IE tasks and settings through model fine-tuning. Given a labeled corpus Dtask = {(s, x, y)}, we fine-tune the UIE model using teacher-forcing cross-entropy loss[24].

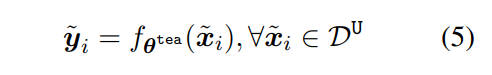
In the target domain, the cost of manually labeling entities is high, however, collecting large amounts of unlabled data becomes relatively easy. Hence, with the help of limited labled data , we use leveraging unlabled data to improve the performance.

1. Through cross-entropy (1), learn the teacher model with .

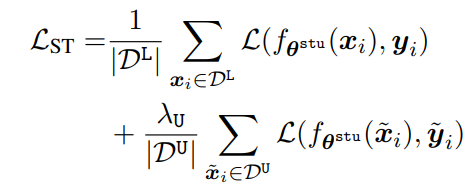
(1)

1)

1. Create soft labels via a teacher model on unlabled tokens:

 (2)

1. Through cross-entropy (1), learn the student model on both labeled and unlabeled tokens.

(3)

VI. Implementation and Experimental Results

This section shows FN-Bert performance for named entity recognition (NER) on a Chinese nested medical data [26].

## Experimental Setup

### Dataset

Artificial intelligence is continuously helping the inheritance and innovation of traditional Chinese medicine under the catalysis of the epidemic. The precipitation and mining of the knowledge system of traditional Chinese medicine is a basic work. By mining the instructions for Chinese medicine to construct a knowledge graph of rational use of Chinese medicine, it will lay a better foundation for standardized diagnosis and treatment of traditional Chinese medicine. Extracting key information from the instructions for Chinese medicine can help expand the knowledge base of traditional Chinese medicine. Therefore, the dataset Entity Recognition of Traditional Chinese Medicine’s Manual was published.

This dataset has 1000 data points for training, 496 points for testing, 500 unlabeled examples and is annotated according to the entities defined in 13 categories.

|  |
| --- |
| Dataset Structure |
| 1. {“id”: 1, // int, document id |
| 1. “text”: “xxx”, // string, original content of drug instructions 2. “annotations”: [ // list, all entity annotations in text   {   1. “entity”: “新生化颗粒”, // string, entity content 2. “label”: “药品”, // string, entity category 3. “start\_offset”: 12, // int, starting index of the entity in text 4. “end\_offset”: 17 // int, ending index of the entity in text (left-closed and right-open)   },   1. … |

It is noted that the data distribution is unbalanced, *Symptom* has the biggest number and *drug\_group* has only few examples.

图表, 条形图

描述已自动生成

Fig. 16. Distribution of the dataset.

It is also important that there exist many nested entities that some sub entity occurs in super entity.

图示

描述已自动生成

Fig. 17. An example of nested entity

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Methods | Precision | Recall | F1 | Speed |
| Bert-base-CRF | 0.8193 | 0.8283 | 0.8237 | 4 |
| Roberta-large- CRF | 0.7822 | 0.8727 | 0.8249 | 17 |
| Roberta-large- CRF-Auto | 0.8136 | 0.8492 | 0.8310 | 17 |
| UER-large-CRF | 0.8065 | 0.8677 | 0.8360 | 17 |
| UER-large-CRF-Auto | 0.8248 | 0.8643 | 0.8440 | 17 |
| UER-large-CRF-Auto-UIE | 0.8726 | 0.8769 | **0.8747** | 17 |
| UER-large-Span-Auto-UIE | 0.8124 | 0.8482 | 0.8299 | **11** |

### Data Preprocessing

The dataset in FN-Bert model contains both plain text and structured data. As it is Bert-based, the Bert tokenizer is used in preprocessing. The steps are follows: -

Firstly, using the first layer of Bert to convert the text to embedding vectors. It will remove all punctuations and the upper case is transformed to lower case. We use the same vocabulary file as the Bert was pretrained with mass data which has the term frequency in particular order and we didn’t amend Bert tokenizer layer. It is noted that we use the char level tokenizer that help to avoid drifting.

Secondly, the former span offset is converted to 4 kinds of labels that the ‘S’ prefix is used to the entity that has one token, the ‘B” prefix is used to the begin of the entity and “E” for ending, the ‘I’ indicate the interval span of the entity.

Thirdly, to use Auto-training, we fitted a base model with the origin data and use it to predict the pseudo data on the unlabeled data and use UIE, a MML for information extraction task, to predict on the unlabeled examples to constitute the pseudo data with length of 1000.

Before training, training data is split into Training and Validation set. The ratio is 8:2.

### Training Environment

The hardware environment is Intel(R) Xeon(R) Gold 6330 CPU with 2.00GHz and NVIDIA GeForce RTX 3090 with 24G memory. Using 3.8.1 python with Torch 2.0.0. The paddlenlp edition is 2.5.2.

## Model Performance

We choose Bert-CRF as base model as it is the most common solution for NER task. The FN-Bert model we come up with will compare with the base model with various parameters and backbone.

For the baseline model, precision and recall are balanced, run time is short. Changing the backbone model, there will be some improvement due to the larger scale model. However, after change to larger scale model, the precision and recall become unbalanced.

The best model is the combination of UER [27], CRF layer and auto-training method. It is obvious that with the auto-training, the evaluation index become balanced again and therefore get the best score.

From nested entity perspective, UER with span method has lower F1 but can handle nested entity well. We believe we can use model ensemble method to gather the advantage of CRF

and Span method.

From time perspective, larger model cost more time, but with span pointer network, the model not only can speed up, but also can detect the nested entity.

## Evaluation

This dataset uses strict F1-score as evaluation method, which means, only when text id, start, interval, end and entity are all the same, then the predict is equal to gold truth. F1 is calculated by these equations.

Table 1. Evaluation with four indexs for models.

V. Conclusion

In this project, FN-Bert model is proposed for nested Chinese medical named entity recognition and compared with state-of-art models for NER.

FN-Bert is a composited neural network based on Bert model with CRF layer and Span layer. Due to the feature of small number of training set and complex nested entity of the dataset. We come up with two methods to overcome it. From few-shot perspective, we use auto-training to overcome the small number of training set and use UIE as the teacher model. Our FN-Bert model can achieve the distillation from teacher model, which means it can inherit the ability of teacher model on this dataset with 10 times faster prediction speed. From nested entity perspective, we use span pointer network to figure the problem. The span method not only can figure out the nested problem but also speed up the running time.

In this project, we overcome the problem of small dataset and nested problem. In next step, we will try other method to do better distillation in order to achieve better few-shot performance with low utilization and ensemble CRF model and Span model to predict well both on nested entities and flat entities.

References

1. Katiyar, A., & Cardie, C. (2018). Nested Named Entity Recognition Revisited. <https://doi.org/10.18653/v1/n18-1079>
2. Straková, Jana, Milan Straka, and Jan Hajič. "Neural architectures for nested NER through linearization." arXiv preprint arXiv:1908.06926 (2019).
3. Shibuya, Takashi, and Eduard Hovy. "Nested Named Entity Recognition via Second-best Sequence Learning and Decoding." arXiv preprint arXiv:1909.02250 (2019).
4. Xia, Congying, et al. "Multi-Grained Named Entity Recognition." arXiv preprint arXiv:1906.08449 (2019).
5. Fisher, Joseph, and Andreas Vlachos. "Merge and Label: A novel neural network architecture for nested NER." arXiv preprint arXiv:1907.00464 (2019).
6. Lin, Hongyu, et al. "Sequence-to-nuggets: Nested entity mention detection via anchor-region networks." arXiv preprint arXiv:1906.03783 (2019).
7. Li, Xiaoya, et al. "A Unified MRC Framework for Named Entity Recognition." arXiv preprint arXiv:1910.11476 (2019).
8. Wang, Bailin, et al. "A neural transition-based model for nested mention recognition." arXiv preprint arXiv:1810.01808 (2018).
9. Huang, J., Li, C., Subudhi, K., Jose, D., Balakrishnan, S., Chen, W., Peng, B., Gao, J., & Han, J. (2020). Few-Shot Named Entity Recognition: A Comprehensive Study. arXiv.org.
10. Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002).
11. Fritzler, A., Logacheva, V., & Kretov, M. (2018). Few-shot classification in named entity recognition task. Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, 993–1000. https://doi.org/10.1145/3297280.3297378
12. Wanyun Cui, Yanghua Xiao, Haixun Wang, Yangqiu Song, Seung-won Hwang, and Wei Wang. 2017. Kbqa: learning question answering over qa corpora and knowledge bases. In Proceedings of 43rd Very Large Data Base Conference Endowment, volume 10.
13. Grishman R , Sundheim B . Message Understanding Conference 6: A Brief History[C]// Proceedings of the 16th conference on Computational linguistics - Volume 1. Association for Computational Linguistics, 1996.
14. S. Zhang and N. Elhadad, “Unsupervised biomedical named en- tity recognition: Experiments with clinical and biological texts,” J. Biomed. Inform., vol. 46, no. 6, pp. 1088–1098, 2013.
15. D. Nadeau, P. D. Turney, and S. Matwin, “Unsupervised named- entity recognition: Generating gazetteers and resolving ambigu- ity,” in CSCSI, 2006, pp. 266–277.
16. J.-H. Kim and P. C. Woodland, “A rule-based named entity recognition system for speech input,” in ICSLP, 2000.
17. Mikolov T , Sutskever I , Chen K , et al. Distributed Representations of Words and Phrases and their Compositionality[J]. Advances in neural information processing systems, 2013, 26. <https://arxiv.org/pdf/1310.4546.pdf>
18. R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, “Natural language processing (almost) from scratch,” J. Mach. Learn. Res., vol. 12, no. Aug, pp. 2493–2537, 2011.
19. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in NIPS, 2017, pp. 5998–6008.
20. Z. Huang, W. Xu, and K. Yu, “Bidirectional lstm-crf models for sequence tagging,” arXiv preprint arXiv:1508.01991, 2015.
21. H Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. IEEE Transactions on Information Theory.
22. Ju M, Miwa M, Ananiadou S. A neural layered model for nested named entity recognition[C]//Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). 2018: 1446-1459.
23. Sohrab M G, Miwa M. Deep exhaustive model for nested named entity recognition[C]//Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. 2018: 2843-2849.
24. Lu, Y., Liu, Q., Dai, D., Xiao, X., Lin, H., Han, X., Sun, L., & Wu, H. (2022). Unified Structure Generation for Universal Information Extraction. arXiv.org.
25. Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-totext transformer. Journal of Machine Learning Research, 21(140):1–67.
26. Tianchi, “Entity Recognition of Traditional Chinese Medicine's Manual.” 2020. [Online] Available: <https://tianchi.aliyun.com/dataset/dataDetail?dataId=86819>
27. Zhao, et.al, “UER: An Open-Source Toolkit for Pre-training Models,” arXiv preprint arXiv: 1909.05658.
28. Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova.BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding[J/OL].https: //arxiv. org/abs/1810.04805,2018-10-11.
29. S. Zheng et al., "Conditional Random Fields as Recurrent Neural Networks," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 2015, pp. 1529-1537, doi: 10.1109/ICCV.2015.179.
30. Ning Ding, Ziran Li, Zhiyuan Liu, Haitao Zheng, and Zibo Lin. 2019. Event detection with triggeraware lattice neural 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), pages 347–356, Hong Kong, China. Association for Computational Linguistics.
31. Ying Shen, Ning Ding, Hai-Tao Zheng, Yaliang Li, and Min Yang. 2020. Modeling relation paths for knowledge graph completion. IEEE Transactions on Knowledge and Data Engineering.

1. . [↑](#footnote-ref-0)