A Survey on Named Entity Recognition

Authors: Rao Jiansheng

Date: 2025-07-16T00:00:00+00:00

Abstract

Named Entity Recognition (NER, NamedEntityRecognition) is a critical component in natural language processing systems, widely applied in tasks such as question answering, information retrieval, and relation extraction. Although NER systems have undergone decades of research and development, named entity recognition systems utilizing deep neural networks (NN) have only been introduced in recent years. In this survey report on neural network-based named entity recognition, we will provide a comprehensive overview of the application of deep neural network architectures in NER, and compare them with traditional feature engineering-based NER methods as well as other supervised or semi-supervised learning algorithms. Furthermore, we will elaborate on neural network models and architectures that have been frequently employed in named entity recognition tasks in recent years, including domain-specific NER models such as LEBERT, SpanKL, MFME-NER, BERT-CRF, and FLAT.

Full Text

A Survey Report on Named Entity Recognition

Rao Jiansheng
School of Computer Science, Sun Yat-sen University, Guangzhou

Abstract

Named Entity Recognition (NER) is a critical component in natural language processing systems, widely applied in question answering, information retrieval, and relation extraction tasks. While NER systems have undergone decades of research and development, neural network-based approaches have only been introduced in recent years. This survey report on neural network-based NER provides a comprehensive overview of deep neural network architectures for NER, comparing them with traditional feature-engineering-based methods and other supervised or semi-supervised learning algorithms. Additionally, we elaborate

on recently prominent neural network models and architectures for NER tasks, including LEBERT, SpanKL, MFME-NER, BERT-CRF, and FLAT.

Keywords: Natural Language Processing; Named Entity Recognition; Deep Neural Networks; Feature Engineering; Neural Network Models

Named Entity Recognition (NER) is a fundamental task in natural language processing that aims to identify and annotate words with specific meanings from text. These entities are typically distinctive nouns such as person names, locations, organizations, times, monetary amounts, etc. Early NER tasks primarily employed traditional methods like Hidden Markov Models (HMM) and Conditional Random Fields (CRF), which were based on probabilistic distribution models. In recent years, NER model architectures have predominantly been based on deep neural networks, achieving more advanced performance with minimal feature engineering. The concept of "Named Entity Recognition" was first introduced by Grishman and Sundheim at MUC 1996 [1]. Since then, numerous NER tasks have been proposed [2-7]. Early NER systems relied on hand-crafted rules, dictionaries, orthographic features, and ontologies, similar to lexical analyzers in modern compilers. These systems also employed probabilistic statistical models such as HMM and CRF. Following rule-based systems, machine learning-based NER systems utilizing feature engineering emerged [8]. Beginning with Collobert et al. (2011) [9], neural network-based NER systems adopted minimal feature engineering approaches and gradually gained popularity in academia. Researchers have proposed various neural network architectures for NER, primarily based on Recurrent Neural Networks (RNN) that operate on character, subword, and/or word embeddings.

In Section 2 of this survey, we summarize key techniques for NER, starting with early traditional methods (HMM, CRF). Section 3 transitions to modern deep learning-based NER methods. Section 4 provides detailed descriptions of specific deep learning-based NER approaches, including LEBERT, SpanKL, MFME-NER, BERT-CRF, and FLAT. Section 5 introduces performance evaluation and benchmark datasets for NER tasks. Section 6 concludes the survey.

2 Traditional Methods

2.1 Hidden Markov Model

The Hidden Markov Model (HMM), proposed by Leonard E. Baum et al. (1966) [10], is a statistical model that describes the process of generating observable observation sequences from a hidden state sequence. The core idea of HMM is that, given the current hidden state, observations are generated through specific probability distributions. HMM consists of two components: a hidden state sequence and an observation sequence. In NLP tasks, hidden states typically represent entity categories, while observation states represent characters or words in text.

In HMM-based NER, we use HMM to handle sequence labeling tasks in text,

aiming to identify specific entities (e.g., person names, locations, organization names) and annotate them with different categories. HMM models the relationship between hidden states (entity categories) and observation states (characters), assuming each character or word is an observation state corresponding to a hidden state representing the entity category. Common entity tags include B-ORG (beginning of organization), I-ORG (inside organization), B-PER (beginning of person), I-PER (inside person), B-LOC (beginning of location), I-LOC (inside location), and O (non-entity). These tag choices depend on contextual hidden states and transition probabilities.

HMM derives the most likely hidden state sequence through three main probabilistic models: (1) Initial state probability distribution: $P(q_1)$, representing the probability of the system being in hidden state q_1 at time $t = 1$. (2) State transition probability: $P(q_t|q_{t-1})$, representing the probability of transitioning from hidden state q_{t-1} to q_t. (3) Observation probability: $P(o_t|q_t)$, representing the probability of observing o_t given hidden state q_t. As shown in [Figure 1: see original paper], the hidden state sequence transitions via state transition probabilities, while observations are generated from hidden states via emission probabilities. HMM' s goal is to infer the most likely hidden state sequence from known observation sequences.

Therefore, HMM aims to infer hidden state sequences from observation sequences. For this purpose, HMM uses the Viterbi algorithm based on dynamic programming to compute the most likely hidden state sequence, with the joint probability expressed as:

$$P(Q, O|\lambda) = P(q_1)P(q_t|q_{t-1})P(o_t|q_t)$$

where Q represents the hidden state sequence, O represents the observation sequence, and λ denotes model parameters (state transition probabilities, observation probabilities, and initial state probabilities). HMM' s core idea is to generate observable sequences through transition probabilities between hidden states and emission probabilities between observation and hidden states, and to infer the most likely hidden state sequence through these relationships. In NER tasks, HMM demonstrates good performance, particularly in recognizing and annotating multi-word entities.

HMM has two fundamental assumptions: (1) The first hidden state (entity tag) depends only on the previous hidden state (entity tag) and is independent of other hidden states except adjacent ones. (2) Observation independence assumption: Observations depend only on the current hidden state and not on hidden states at other times.

2.2 Conditional Random Field (CRF)

Conditional Random Field (CRF), proposed by John D. Lafferty et al. (2001) [11], is a probabilistic graphical model for sequence labeling tasks, particularly

suitable for tasks with dependencies in annotation and prediction. Unlike HMM, CRF can more flexibly handle global dependencies between input sequences and output labels, rather than just local neighboring states.

In CRF-based NER, the model computes the conditional probability of each possible label sequence by modeling global context of the input sequence. Unlike HMM' s separation of hidden state transition probabilities and observation probabilities, CRF models the relationship between input sequences and output labels through joint probability. CRF aims to maximize the conditional probability:

$$P(Y|X) = \frac{\exp(\sum_k \lambda_k f_k(x, y))}{Z(X)}$$

where $X = \{x_1, x_2, ..., x_T\}$ is the input sequence, $Y = \{y_1, y_2, ..., y_T\}$ is the corresponding label sequence, $f_k(x, y)$ are feature functions, λ_k are weights, and $Z(X)$ is the normalization factor ensuring the conditional probabilities sum to 1.

CRF' s advantage lies in its ability to capture long-range dependencies through global feature modeling. For example, in NER tasks, CRF can utilize entire context information to determine whether an entity is complete, avoiding HMM' s locality issues. CRF models can simultaneously consider multiple features such as word form features, contextual features, and part-of-speech features.

In NER tasks, CRF tags typically represent entity categories like person names (B-PER), locations (B-LOC), and organization names (B-ORG), similar to HMM. However, CRF differs in its ability to simultaneously utilize multiple features rather than relying solely on transition probabilities between neighboring states.

For example, in the sentence "I live in Shanghai," CRF can correctly annotate "Shanghai" as a location (B-LOC) by modeling contextual features between "live in" and "Shanghai."

CRF model training typically employs maximum likelihood estimation, optimizing parameters λ_k to maximize the conditional probability $P(Y|X)$ given input sequence X. To improve training efficiency, CRF is usually trained using optimization algorithms such as gradient descent or quasi-Newton methods.

As shown in [Figure 2: see original paper], CRF models dependencies between inputs and outputs through global features, unlike HMM which only models through local state transitions. This makes CRF more flexible and accurate in handling NER tasks.

2.3 Rule-Based and Feature Engineering Methods

Rule-based and feature engineering methods have been widely applied in NER, particularly dominating early NER systems. These methods rely on manually

designed features and rules to identify entities through analysis of input text. The core of these methods lies in utilizing expert knowledge to handcraft features and rules for recognizing named entities. Common features include word form features (e.g., prefixes, suffixes), contextual features (e.g., surrounding words), and part-of-speech features.

In traditional rule-based NER systems, predefined rule sets are typically used to match entities in text. For example, rules might specify "if a word starts with 'Mr.' , it is likely a person name" or "if a word appears in a specific location list, it may be a place name." These rules can be implemented through regular expressions or dictionary matching.

However, rule-based methods have certain limitations: rules struggle to cover all possible cases and lack generalization capability for new domains or entity types. To overcome this issue, feature engineering methods emerged. Feature engineering automatically learns model parameters by extracting numerous features from text and feeding them into machine learning algorithms. Feature selection is a crucial step in this process, determining which information is most critical for NER tasks.

Rule-based and feature engineering methods are typically combined with machine learning models for training, such as decision trees and Support Vector Machines (SVM). These models learn relationships between extracted features and labels to classify and recognize named entities. Although these methods can achieve good performance on specific tasks and domains, they usually depend heavily on manual feature design and domain knowledge, showing poor adaptability to large-scale, diverse datasets and tasks.

With the rise of deep learning and end-to-end models, rule-based and feature engineering methods have gradually been replaced by more automated deep learning approaches, though they still have important applications in some tasks, particularly when annotated data is insufficient.

3 Deep Learning Methods

3.1.1 Recurrent Neural Network (RNN)

Recurrent Neural Network (RNN) is a class of deep learning models capable of processing sequential data, widely applied in sequence labeling tasks such as NER. Unlike traditional neural networks, RNN introduces recurrent connections to transfer information between time steps, effectively capturing contextual dependencies in sequences.

In NER tasks, RNN utilizes contextual information to annotate each word or character. In RNN, the hidden state depends not only on the current input but also on the previous hidden state. Specifically, the basic RNN structure is:

$$h_t = \sigma(W_{hh}h_{t-1} + W_{xh}x_t + b_h)$$

$$y_t = W_{hy}h_t + b_y$$

where h_t is the hidden state at time t, x_t is the current input (e.g., character or word vector representation), y_t is the output label (e.g., entity category), σ is the activation function (e.g., tanh or ReLU), W_{hh}, W_{xh}, and W_{hy} are weight matrices, and b_h and b_y are bias terms.

RNN' s key characteristic is its recurrent connections, enabling it to retain valuable contextual information when processing long sequences. However, standard RNNs may encounter gradient vanishing or exploding problems in long sequence training, limiting their performance on long-range dependencies. To address this issue, improved RNNs such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) emerged, which effectively mitigate gradient vanishing through gating mechanisms. LSTM controls information flow through three gates (input gate, forget gate, and output gate), thereby maintaining long-term dependency information.

Although RNN performs well in processing sequential data, its computational efficiency is relatively low, and it still faces challenges when processing long sequences. In NER tasks, RNN is often combined with CRF to further improve sequence labeling accuracy, particularly when strong dependencies exist between labels.

[Figure 3: see original paper] illustrates how RNN-T (Recurrent Neural Network-Transducer) aligns the input sequence "pain in my back" to the label "sym: back_{pain}," where end-markers indicate the end of a span. Horizontal lines denote time steps in the input sequence (with blank output symbols), while vertical lines indicate generating the next label in the target sequence.

3.1.2 BiLSTM-CRF Architecture

The combination of Bidirectional Long Short-Term Memory (BiLSTM) and Conditional Random Field (CRF), known as BiLSTM-CRF, is a deep learning architecture that has achieved significant progress in NER and other sequence labeling tasks, proposed by Huang et al. (2015) [12]. BiLSTM captures bidirectional contextual information in sequence data by combining forward and backward LSTM networks, while CRF models dependencies between labels on top of BiLSTM output, further improving sequence labeling accuracy.

In the BiLSTM-CRF architecture, the input sequence is first encoded by bidirectional LSTM. LSTM networks effectively handle long-range dependency information through their inherent recurrent structure, overcoming the common gradient vanishing problem in traditional RNNs when training on long sequences. Bidirectional LSTM extracts contextual information from both forward and backward directions simultaneously, providing more comprehensive contextual representations for each word. Specifically, given input sequence $X = \{x_1, x_2, ..., x_T\}$, the BiLSTM output h_t is the concatenation of forward and backward LSTM results:

$$h_t = \text{BiLSTM}(x_t)$$

where x_t represents the word vector at time step t in the input sequence, and h_t represents the contextual representation of that word, containing bidirectional information from both past and future contexts.

Subsequently, the CRF layer models dependencies between labels based on BiLSTM output. Unlike traditional classification methods, CRF can simultaneously consider global dependencies across the entire label sequence rather than predicting individual labels independently. The CRF conditional probability is expressed as:

$$P(Y|X) = \frac{\prod_{i,j} A_{ij} \cdot h_i}{Z(X)}$$

where $Y = \{y_1, y_2, ..., y_T\}$ is the label sequence, $X = \{x_1, x_2, ..., x_T\}$ is the input sequence, A_{ij} is the state transition matrix, h_i is the feature representation for label i at time step t generated by BiLSTM, and $Z(X)$ is the normalization factor ensuring the conditional probabilities sum to 1. Through this approach, CRF effectively captures global dependencies between labels, avoiding errors that may occur with independent label prediction.

The BiLSTM-CRF architecture' s advantage lies in its ability to simultaneously leverage bidirectional contextual information and label dependencies. This makes it perform excellently in NER tasks, particularly in accurately identifying the beginning and end of multi-word entities, avoiding boundary errors in simple label prediction methods. Compared to using BiLSTM or CRF alone, BiLSTM-CRF achieves significant improvements in entity recognition accuracy by combining both strengths.

[Figure 4: see original paper] shows a schematic diagram of the BiLSTM-CRF model. Additionally, the BiLSTM-CRF architecture has achieved excellent performance on multiple standard NER datasets, demonstrating its potential for various sequence labeling tasks. By capturing sequence context information through bidirectional LSTM and modeling global label dependencies through CRF, the BiLSTM-CRF architecture enables efficient and accurate named entity recognition, providing a powerful solution for NER and other sequence labeling tasks.

3.2 Pre-trained Language Models

Pre-trained language models have achieved remarkable success in sequence labeling tasks such as NER, particularly Transformer-based pre-trained models like BERT (Bidirectional Encoder Representations from Transformers) proposed by Devlin et al. (2019) [13] and RoBERTa proposed by Liu et al. (2019) [14].

BERT encodes input sequences through bidirectional context information, significantly improving upon traditional language models. In BERT, text representations are encoded through bidirectional Transformers, enabling the model to understand context from both left and right directions simultaneously. BERT employs Masked Language Model (MLM) for training, where random words in the input sequence are masked and the model predicts these masked words based on context. BERT' s training objective is to maximize the conditional probability $P(\hat{x}_i|\hat{X})$, where $\hat{x}_i$ is the word at the masked position and $\hat{X}$ is the context sequence. Specifically, BERT' s training objective is:

$$P(\hat{X}) = P(\hat{x}_i|\hat{X}_{\setminus i})$$

where $\hat{X}_{\setminus i}$ represents the context with the i-th word removed, and the model utilizes context information for prediction through this approach.

Compared to BERT, RoBERTa (A Robustly Optimized BERT Pretraining Approach) optimizes BERT by using larger training corpora, longer training times, and removing BERT's "Next Sentence Prediction"(NSP) task. RoBERTa further improves pre-trained model effectiveness, particularly in NER tasks, achieving better performance through larger corpora and more optimized training strategies.

Although BERT performs excellently in NER, it still faces challenges in modeling label dependencies for sequence labeling tasks. To address this issue, Deriu et al. (2020) combined BERT with CRF to form the BERT+CRF architecture [15]. In this architecture, BERT generates context-rich word vector representations, while CRF further models dependencies between labels. CRF optimizes sequence label prediction by jointly modeling transition probabilities between labels, thereby improving NER accuracy. The joint probability of BERT+CRF is expressed as:

$$P(Y|X) = \frac{\prod_{i,j} A_{ij} \cdot h_i}{Z(X)}$$

where $Y = \{y_1, y_2, ..., y_T\}$ is the label sequence, $X = \{x_1, x_2, ..., x_T\}$ is the input sequence, A_{ij} is the transition matrix between labels, h_i is the feature representation for label i at time step t generated by BERT, and $Z(X)$ is the normalization factor. CRF avoids potential errors from independent label prediction by globally modeling dependencies between labels.

[Figure 5: see original paper] illustrates the BERT-CRF model workflow. Given an input document, the text is first tokenized using WordPiece (Wu et al., 2016) [16], then divided into overlapping spans with a defined stride (3 in this case). The maximum context tokens for each span are shown in bold. These spans are then fed into the BERT model, followed by a classification layer that generates label score sequences for each span. Subword entries (starting with

##) are removed from the spans, and the remaining tokens are passed to the CRF layer. Maximum context tokens are selected and concatenated to form the final predicted labels.

The architecture combining BERT and RoBERTa with CRF provides strong performance support for NER and other sequence labeling tasks, particularly in handling multi-word entities and complex entity boundaries, significantly improving annotation accuracy.

3.3 Transformer Architecture

The Transformer architecture, proposed by Vaswani et al. (2017), is a self-attention mechanism-based approach [17] designed to solve long-term dependency problems in sequence-to-sequence tasks. Unlike traditional RNNs and LSTMs, Transformer relies entirely on self-attention mechanisms without using recurrent structures. Self-attention captures global context information by computing relationships between all positions in the input sequence, enabling parallel sequence processing and significantly improving computational efficiency. In Transformer, input sequences are processed through multiple encoder and decoder layers, each containing two main components: Multi-Head Self-Attention and Feed-Forward Neural Network. Specifically, Transformer' s core principle can be expressed as:

For input sequence $X = \{x_1, x_2, ..., x_T\}$, each input vector x_t is transformed through self-attention to generate weighted context representations:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where Q, K, V represent Query, Key, and Value matrices respectively, and d_k is the dimension of key vectors. Multi-head attention maps queries, keys, and values to multiple different spaces for processing, concatenates the results, and obtains the final output through linear transformation:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O$$

where h is the number of heads and W^O is the output projection matrix. Through this approach, Transformer can compute in parallel and capture relationships between all positions in the sequence.

[Figure 6: see original paper] shows the Transformer model architecture. Based on Transformer, researchers have proposed many variant models such as GPT, T5, and XLNet. GPT (Generative Pretrained Transformer), proposed by OpenAI (2018), uses a unidirectional autoregressive model trained by maximizing word prediction probability given context [18]. Specifically, GPT' s training objective is:

$$P(x_1, x_2, ..., x_T) = \prod_t P(x_t | x_1, x_2, ..., x_{t-1})$$

This autoregressive training approach enables GPT to perform excellently in generation tasks. T5 (Text-to-Text Transfer Transformer), proposed by Raffel et al. (2020), converts all NLP tasks into a unified text-to-text framework [19], whether classification, translation, or generation tasks. T5' s objective function is similar to other Transformer models but provides a more general solution by treating all tasks as text generation problems. XLNet, proposed by Yang et al. (2020), combines advantages of autoregressive and autoencoding models [20], training the model by maximizing the joint probability of all possible permutations to further enhance context modeling capability.

The Transformer architecture offers significant advantages in NER tasks. First, Transformer can capture global context information through self-attention mechanisms, unlike traditional RNNs that are limited to local dependencies. Second, Transformer enables parallel computation, avoiding computational bottlenecks in RNNs when processing long sequences. Especially in NER tasks, Transformer' s powerful context modeling capability can effectively identify entity boundaries and improve multi-word entity recognition accuracy. Additionally, pre-trained Transformer models (such as BERT and GPT) leverage large-scale corpora pre-training, enabling models to fully utilize existing linguistic knowledge in downstream tasks, thereby significantly improving performance on NER and other sequence labeling tasks.

4 Specific Deep Learning Models

4.1 BERT-CRF (BERT with Conditional Random Fields)

BERT-CRF, proposed by Hu et al. (ICIS2022) [21], is a strong baseline model widely adopted in Chinese NER tasks in recent years. This method combines the advantages of the pre-trained language model BERT and the classic sequence labeling model CRF to enhance entity recognition accuracy and robustness.

4.1.1 Research Motivation Traditional Chinese NER methods mostly rely on manually constructed rules and feature templates, suffering from low accuracy, high dependence on domain knowledge for feature design, poor generalization capability, and inability to handle polysemous words in context, making it difficult to cope with complex unstructured data.

In recent years, deep learning methods have been widely applied to NER tasks, achieving excellent results without extensive manual feature engineering. Particularly, the BERT pre-trained language model can effectively capture word ambiguity information and deep semantic features through bidirectional context encoding, demonstrating outstanding performance in natural language processing. However, BERT's native output labels are independent of each other, failing

to adequately model sequential relationships between labels, which cannot meet the contextual correlation requirements of sequence labeling tasks.

To address these issues, this paper proposes a Chinese NER method combining BERT and CRF. BERT serves as a powerful tool for contextual information representation, automatically extracting rich word-level and semantic features, while the CRF layer enhances sequence labeling consistency by modeling transition constraints between labels, thereby further improving model effectiveness in named entity recognition. Experiments show that this method achieves an F1 score of 94.5% on the People' s Daily dataset, significantly outperforming traditional methods and validating the effectiveness of BERT-CRF in Chinese NER tasks.

4.1.2 Research Method Model Overview: The BERT-CRF model consists of two main components: the BERT language model and the CRF sequence labeling layer. First, BERT pre-trains on input text using Masked Language Model (MLM) and Next Sentence Prediction (NSP), thereby obtaining rich contextual semantic representations at both word and sentence levels, automatically extracting lexical and semantic features to generate dynamic word vectors. Through this process, BERT can effectively model context information and semantic relationships in long texts, providing strong feature support for downstream tasks.

Subsequently, the model stacks a Conditional Random Field (CRF) layer on top of BERT encoding. The CRF layer models label transition relationships in sequence labeling tasks, improving entity boundary recognition accuracy and label sequence consistency through explicit constraints on label dependencies. Finally, the CRF layer outputs the optimal label sequence through a global optimal solution algorithm.

The model structure is shown in [Figure 7: see original paper]. Input is encoded by the BERT model to obtain contextual representations for each token, which are then jointly modeled by the CRF layer for label transition relationships and label prediction, ultimately outputting entity recognition results.

BERT Layer: BERT' s core adopts a bidirectional Transformer neural network as the encoder and leverages open-domain corpora for pre-training. Through this mechanism, the model can predict the next word by combining bidirectional context information, thereby obtaining richer semantic representations. During pre-training, BERT uses the Masked Language Model (MLM), randomly masking 15% of words in sentences and predicting them through context, thereby learning word-level and sentence-level contextual semantic features, generating dynamic word vectors and automatically extracting numerous lexical and semantic features.

BERT model training consists of two stages. The first stage is pre-training, which trains the language model on unlabeled corpora; the second stage is fine-tuning, which adjusts the pre-trained language model based on downstream

tasks. In this study, BERT is applied to the NER task, with single sentences as input and embedded labels for each token in the sentence as output. BERT' s pre-training stage is shown in [Figure 8: see original paper], where input consists of masked sentences; the fine-tuning stage is shown in [Figure 9: see original paper], where for entity recognition tasks, the model takes a single sentence as input and outputs its corresponding label sequence.

When processing input text sequences, tokenization is first performed to obtain the tokenized text sequence. Then, some words are randomly masked, and special tokens [CLS] and [SEP] are added. At this point, each word's embedding consists of three parts: Token Embedding (word vector), Segment Embedding (sentence segment vector), and Position Embedding (position vector). Finally, this sequence of vectors is input into the bidirectional Transformer for feature extraction, obtaining sequence representations rich in semantic features.

CRF Layer: The two commonly used methods in sequence labeling modules are Conditional Random Field (CRF) model and Softmax classification. While the Softmax function can output the label with the highest probability for each word, labels are independent without considering contextual relationships, leading to decreased accuracy. Therefore, this paper adopts CRF for label prediction, fully considering text contextual correlation and adding reasonable constraints during label prediction to enhance the validity and rationality of prediction results.

Let the input sequence be:

$$x = \{x_1, x_2, \cdots, x_n\}$$

Corresponding output label sequence:

$$y = \{y_1, y_2, \cdots, y_n\}$$

where P_{i,y_i} represents the probability of the i-th word being labeled as y_i, and transition matrix $A_{y_i,y_{i+1}}$ represents the transition probability from label y_i to y_{i+1}. The scoring function for label sequence y is:

$$\text{score}(x, y) = \sum P_{i,y_i} + \sum A_{y_i,y_{i+1}}$$

By normalizing scores across all possible paths, the probability of predicted sequence y can be calculated:

$$P(y|x) = \frac{\exp(\text{score}(x, y))}{\sum_{\hat{y} \in Y_x} \exp(\text{score}(x, \hat{y}))}$$

where $\hat{y}$ represents the true label sequence and Y_x represents the set of all possible label sequences.

Taking the logarithm of the above equation yields the likelihood function of the label sequence:

$$\log P(y|x) = \text{score}(x, y) - \log \sum_{\hat{y} \in Y_x} \exp(\text{score}(x, \hat{y}))$$

In the final decoding stage, the sequence with the highest score is selected as the output:

$$y^* = \arg\max_{\hat{y} \in Y_x} \text{score}(x, \hat{y})$$

4.1.3 Model Performance In this study, both the BERT-CRF model and the Fine-tune BERT-CRF model were compared on the People' s Daily 1998 first-half corpus. This dataset uses BIO annotation and contains six entity labels: B-PER, I-PER, B-LOC, I-LOC, B-ORG, and O, focusing on recognition of person names, locations, and organization names.

For evaluation metrics, Precision (P), Recall (R), and F1-score are used as model performance criteria, calculated as shown in equation (7):

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \times 100\%$$

where T_p represents the number of correctly predicted entities, F_p represents the number of incorrectly predicted entities, and F_N represents the number of missed entities.

Experimental results show that the BERT-CRF model finally converged at epoch 87, with Precision of 0.911, Recall of 0.864, and F1 value of 0.884. In contrast, the Fine-tune BERT-CRF model converged faster, reaching optimal performance at only epoch 15, with Precision of 0.955, Recall of 0.938, and F1 value improved to 0.945. The comparison demonstrates that the Fine-tune BERT-CRF model shows significant improvements in both accuracy and convergence speed compared to the original BERT-CRF model, proving its superior effectiveness and training efficiency in NER tasks.

4.2 LEBERT (Lexicon Enhanced BERT)

LEBERT (Lexicon Enhanced BERT), proposed by Liu et al. (ACL2021) [22], focuses on Chinese sequence labeling tasks and introduces a new method for integrating lexical knowledge into the BERT framework.

This method addresses the problem that traditional BERT Chinese representation relies solely on character granularity and lacks explicit lexical information modeling. By designing a Lexicon Adapter module, external dictionary information is injected into BERT' s internal Transformer layers, achieving deep fusion of word and character information, thereby enhancing the model' s ability to model entity boundaries and word-level semantics. Experiments show

that LEBERT achieves superior results compared to existing mainstream methods on multiple standard Chinese NER datasets such as MSRA and OntoNotes, demonstrating stronger robustness particularly in recognizing out-of-vocabulary and low-frequency entities, providing an effective paradigm and important reference for subsequent Chinese NER research.

4.2.1 Research Motivation Although BERT and its variants have achieved remarkable results in Chinese NER tasks, standard BERT models are based solely on character-level modeling without explicitly integrating word-level information, making it difficult to fully capture implicit lexical knowledge in Chinese. Existing studies have attempted to introduce shallow dictionary features (such as concatenating word vectors or adding LSTM layers) on top of BERT representations to enhance the model, but this model-level fusion approach has two limitations: First, dictionary information fails to deeply integrate into the pretrained model' s interior, preventing deep interaction with BERT' s hierarchical representations; second, external features are separated from BERT representations, failing to fully unleash BERT' s representation capability. To address this, LEBERT proposes introducing a Lexicon Adapter module inside BERT, injecting dictionary information directly into Transformer layers through layer-wise injection, achieving deep interaction between external lexical knowledge and contextual semantic information, aiming to improve modeling capabilities for word boundaries and entity boundaries in Chinese sequence labeling. This method aims to overcome the limitations of traditional shallow fusion, achieving organic unification of lexical features and deep semantic representations from the model' s bottom layers, thereby improving entity recognition accuracy and robustness.

4.2.2 Research Method The overall architecture of the proposed LEBERT model is shown in Figure 2. Compared to original BERT, LEBERT has two main differences. First, LEBERT simultaneously introduces character features and dictionary features as input. Second, LEBERT introduces Lexicon Adapter modules between Transformer layers, enabling more effective integration of dictionary knowledge into the BERT model.

This section introduces three core components: (1) Char-words Pair Sequence, which naturally integrates word-level information into character sequences; (2) Lexicon Adapter, which explicitly injects external dictionary features into BERT; (3) Lexicon Enhanced BERT, which achieves deep fusion of external dictionary knowledge and model representations by applying Lexicon Adapter across BERT layers.

Char-words Pair Sequence: To effectively introduce dictionary information to compensate for traditional BERT' s neglect of Chinese word granularity information, LEBERT extends the original character-based input sequence into a Char-Words Pair sequence. This method explicitly associates each character with its potential corresponding words, providing the model with richer word-

level knowledge.

Specifically, given a Chinese dictionary D and a Chinese sentence $s_c = \{c_1, c_2, ..., c_n\}$ containing n characters, a prefix tree (Trie) is first built based on dictionary D. Then all character subsequences in the sentence are enumerated and matched in the Trie to retrieve all potential words in the sentence. Taking "美国人民" (American People) as an example, matched words include "美国" (America), "美国人" (American), "国人" (Compatriot), and "人民" (People). Each matched word is then mapped back to every character it covers, forming the set of words associated with that character. The matching process is shown in [Figure 11: see original paper].

Finally, the authors construct the char-words pair sequence $s_{cw} = \{(c_1, ws_1), (c_2, ws_2), ..., (c_n, ws_n)\}$, where c_i represents the i-th character in the sentence and ws_i represents the set of words matching c_i. This sequence retains original character-level information while introducing higher-level lexical knowledge, providing richer and more structured input features for subsequent models.

Lexicon Adapter: The Lexicon Adapter module is designed to integrate dictionary features into BERT. For each character c_i and its associated word set ws_i, the adapter first encodes the word set through a word embedding layer to obtain word representations x_{ws_i}. These representations are then fused with the original character representations through a bilinear attention mechanism:

$$\tilde{h}_i = h_i^c + z^w$$

where h_i^c is the character representation and z^w is the weighted word representation. This process is followed by standard modules such as Dropout and LayerNorm, ultimately forming context representations fused with lexical information.

Lexicon Enhanced BERT: Lexicon Enhanced BERT (LEBERT) achieves the goal of injecting external lexical knowledge inside BERT layers by combining Lexicon Adapter (LA) with BERT. Specifically, LA is inserted between Transformer layers at a specific layer k of BERT, as shown in [Figure 10: see original paper], allowing lexical information to directly interact with contextual representations.

Given a Chinese sentence $s_c = \{c_1, c_2, ..., c_n\}$ of length n, LEBERT first constructs the char-words pair sequence $s_{cw} = \{(c_1, ws_1), (c_2, ws_2), ..., (c_n, ws_n)\}$ as described in Section 3.1. Then, the character sequence $\{c_1, c_2, ..., c_n\}$ is fed into BERT' s input embedding layer, processed through Token, Segment, and Position Embeddings to obtain representation sequence $E = \{e_1, e_2, ..., e_n\}$.

This representation sequence E is input into the Transformer encoder, with each layer updated according to:

$$G = \text{LN}(H^{l-1} + \text{MHAttn}(H^{l-1}))$$
$$H^l = \text{LN}(G + \text{FFN}(G))$$

where $H^l = \{h_1^l, h_2^l, ..., h_n^l\}$ represents the output of layer l, $H^0 = E$, LN is LayerNorm, MHAttn is multi-head attention mechanism, and FFN is the feed-forward network with ReLU activation.

When injecting lexical information between layer k and layer $(k+1)$, the output of layer k is first obtained as $H^k = \{h_1^k, h_2^k, ..., h_n^k\}$. Then each character with its word pair (h_i^k, x_{ws_i}) is fed into the Lexicon Adapter to generate the updated representation $\tilde{h}_i^k$. Since BERT typically contains $L = 12$ Transformer layers, the updated representation $\tilde{H}^k = \{\tilde{h}_1^k, \tilde{h}_2^k, ..., \tilde{h}_n^k\}$ is continued through the remaining $(L-k)$ layers for subsequent computation, ultimately obtaining the output H^L for the sequence labeling task.

4.2.3 Model Performance The authors conducted comprehensive experiments on LEBERT across three tasks: Chinese NER, Chinese Word Segmentation (CWS), and Chinese Part-of-Speech (POS) tagging, using ten public datasets and comparing it with baseline methods including directly fine-tuned BERT, BERT+Word fusion models, and lexicon-pretrained models like ERNIE and ZEN, with standard F1 score as the evaluation metric. Experimental results show that LEBERT achieved state-of-the-art results on all datasets, further validating the effectiveness and superiority of fusing lexical information within BERT layers.

In Chinese NER tasks, LEBERT achieved F1 scores of 70.75%, 82.08%, 95.70%, and 96.08% on the Weibo, OntoNotes, MSRA, and Resume datasets respectively, comprehensively surpassing BERT+Word and existing best methods, demonstrating strong generalization capabilities across different domains. In Chinese word segmentation tasks, LEBERT achieved F1 scores of 96.91%, 98.69%, and 97.52% on the PKU, MSR, and CTB6 datasets, further outperforming various fusion and pre-training methods, indicating that deep fusion of lexical information with BERT is particularly effective for boundary recognition. In Chinese POS tagging tasks, LEBERT achieved F1 scores of 97.14%, 95.18%, 96.06%, and 95.74% on the CTB5, CTB6, UD1, and UD2 datasets, again exceeding existing best BERT-based models, fully validating the method' s robustness and effectiveness across different syntactic labeling tasks.

Further analysis shows that LEBERT brings significant relative error reduction compared to the BERT baseline across all datasets. For example, on the MSRA NER dataset, LEBERT reduces relative error by 18.71% compared to BERT, and by 23.79% on the UD1 POS dataset. Additionally, compared to the BERT+Word fusion model, LEBERT achieves better results on both Span F1 and Type Accuracy metrics, indicating that injecting lexical information inside

BERT layers better improves boundary detection and category discrimination capabilities, showing stronger robustness especially in long sentences.

Ablation studies further explore the impact of the number of Lexicon Adapter insertion layers, finding that shallow injection (e.g., at layer 1) achieves the best effect, while deep or multi-layer injection 反而 leads to performance degradation, possibly due to overfitting or representation disturbance. Experiments also verify the importance of fine-tuning BERT parameters; without fine-tuning BERT, F1 scores drop by 7.03 points on OntoNotes and 3.75 points on UD1, further demonstrating that lexical information combined with pre-trained representations still requires appropriate parameter adaptation capabilities.

In summary, LEBERT achieves significant and stable performance improvements across multiple Chinese sequence labeling tasks, fully validating the effectiveness of its lexical enhancement approach.

4.3 SpanKL (Span-based Knowledge Learning)

SpanKL (Span-based Knowledge Learning for Continual NER), proposed by Zhang et al. (ACL 2023) [23], focuses on the incremental NER (CL-NER) scenario and proposes a new span modeling approach for continuous learning of entity types.

This method addresses the problem that existing CL-NER methods struggle to balance learning of old and new categories and are prone to forgetting learned knowledge. Based on span representations, it designs a task-decoupled modeling framework that constructs independent span representation spaces for different entity types, combining Bernoulli KL divergence and multi-label BCE loss to distill knowledge for historical entity categories and learn current categories respectively, thereby effectively alleviating multi-task conflicts and forgetting issues, and improving the robustness and generalization ability of new and old category recognition. Experiments show that SpanKL achieves superior performance compared to existing mainstream methods on multiple typical incremental NER datasets, demonstrating good continuous learning capability and entity representation transfer ability, providing a new paradigm and 思路 for subsequent CL-NER research.

4.3.1 Research Motivation In real-world applications, NER tasks often face the problem of continuously growing entity categories, such as personal assistants that need to continuously adapt to recognition of new entity types. However, current mainstream sequence labeling methods struggle to effectively address this challenge under the continual learning (CL) background, suffering from catastrophic forgetting problems. Additionally, traditional sequence labeling methods typically assume that "non-entity" labels (O tags) in the current task mean that the segment will not become an entity in the future, leading to annotation conflicts between tasks and causing continuous model retraining and forgetting.

To solve these problems, the SpanKL model proposes using Span as the basic modeling unit, explicitly representing and classifying continuous character fragments, and converting the NER task into a multi-label classification problem at the span level. This modeling approach naturally adapts to continual learning scenarios: on one hand, it retains existing knowledge through knowledge distillation to avoid forgetting historical entity types; on the other hand, the multi-label modeling approach ensures that newly introduced entity types in the future will not conflict with current non-entity labels. Furthermore, compared to traditional sequence labeling, SpanKL' s independent modeling of spans and entity categories provides stronger knowledge transfer and continuous learning capabilities, particularly naturally supporting nested entity continual learning. Therefore, SpanKL provides a more compatible and forward-adaptive solution for CL-NER, better adapting to learning and generalization of new knowledge while maintaining old knowledge.

4.3.2 Research Method Problem Definition: The authors adopt the mainstream setting in recent continual learning NER (CL-NER) research, formalizing the task as a Class-Incremental Learning (CIL) problem. Specifically, assume there is a sequence of tasks arriving in order $T_1, T_2, ..., T_l$, where each task T_l contains its own unique set of entity categories $E_l = \{e_{l1}, e_{l2}, ...\}$, and only entities in this set are annotated. Entity categories are non-overlapping between tasks; for example, if the ORG category is learned in T_1, it will not appear in subsequent tasks. However, text fragments can repeat across tasks, allowing an entity fragment to be assigned different labels in different tasks, not limited to nested entities and other special cases.

In the first stage ($l = 1$), model M_1 is trained from scratch on dataset D_1 to recognize entity set E_1. From the second stage onwards ($l > 1$), model M_l continues training on the current stage dataset D_l based on the previously learned model M_{l-1}, with the goal of cumulatively recognizing all entity categories involved in current and historical stages, i.e., $\bigcup_{i=1}^{l} E_i$. [Figure 13: see original paper] shows the overall architecture of the SpanKL model.

Contextual Encoder: This component uses a contextual encoder to model input text and capture contextual dependencies between tokens in the sequence. This module can be implemented by CNN, RNN, or pre-trained language models. Specifically, the input sequence first obtains word vector representations $E \in \mathbb{R}^{n \times d_e}$ through an embedding layer, then passes through the contextual encoder to obtain context-aware hidden states $H \in \mathbb{R}^{n \times d_h}$. This process can be briefly expressed as:

$$E = \text{Embed}(X), \quad H = \text{CtxEnc}(E)$$

where Embed is the word embedding layer and CtxEnc is the contextual encoder. This encoder is shared across all tasks for unified contextual representation acquisition.

Span Representation Layer: For entity span modeling, SpanKL designs a specialized span representation layer that obtains representations of different spans by modeling contextual representations of the input sequence, and then classifies each span to predict its entity category. Specifically, the span representation layer first constructs span representations h_{sij} based on input contextual representation $H = [h_1, h_2, ..., h_n]$, typically by concatenating representations of the span' s start and end tokens, i.e., $h_{sij} = \text{SpanRep}(h_i, h_{i+1}, ..., h_j)$.

Subsequently, SpanKL employs a dual feed-forward network (FFN) to model the start and end points of spans separately, obtaining final span representations through dot product interaction. Specifically, for each entity type, SpanKL designs dedicated feed-forward networks for discriminative modeling:

$$h_{sij}^k = \text{FFN}_{s,k}(h_i)^\top \text{FFN}_{e,k}(h_j) \times (d_o)^{-0.5}$$

where k represents the entity type, s, e represent the start and end positions of the span respectively, and d_o is the output dimension normalization term. This design allows each entity type to have independent modeling parameters, enhancing representation capability and task decoupling. When the number of tasks increases, SpanKL only needs to add corresponding FFN layers for seamless expansion, maintaining discriminative modeling for different tasks and entity categories. Additionally, for unified modeling, SpanKL proposes a Span matrix that organizes all span representations h_{sij}^k related to the k-th entity category into an upper triangular matrix $M_k \in \mathbb{R}^{n \times n}$ to enhance the model' s perception and utilization of span information.

Multi-Label Loss Layer: To ensure good comparability and extensibility during forward inference, SpanKL ultimately formulates span classification as a multi-label prediction problem. Specifically, the authors compute loss using Binary Cross Entropy (BCE) on span matrix prediction logits after sigmoid activation, aligning them with gold labels. Unlike common multi-class methods (i.e., softmax activation with Cross Entropy (CE) loss), BCE loss effectively avoids interference between different entity types during logit normalization, making discrimination of each entity type more independent, particularly suitable for mixed single-task and multi-task learning scenarios.

For each entity type, independent binary classification is performed, with BCE loss calculated as:

$$\hat{p}(k|s_{ij}) = \text{sigmoid}(h_{sij}^k)$$

$$L_{BCE} = -\sum [p(k|s_{ij}) \log \hat{p}(k|s_{ij}) + (1 - p(k|s_{ij})) \log(1 - \hat{p}(k|s_{ij}))]$$

where $p(k|s_{ij})$ represents the gold label and $\hat{p}(k|s_{ij})$ is the model prediction. The above L_{BCE} loss is calculated only based on span matrices corresponding to entity types in the current task.

Knowledge Distillation: To ensure the model' s retention capability for old entity categories during incremental learning, SpanKL adopts a Knowledge Distillation (KD) mechanism to prevent forgetting previously learned entity categories. In the $l > 1$ incremental stage, the previously trained model M_{l-1} (teacher) first performs one-time forward inference on the current stage training set D_l to obtain Bernoulli distribution results for all old entity category spans as soft labels $e_p(k|s_{ij})$. Then, Bernoulli KL divergence loss is calculated with the current model M_l (student) predictions:

$$L_{KD} = \sum [e_p(k|s_{ij})(\log e_p(k|s_{ij}) - \log \hat{p}(k|s_{ij})) + (1-e_p(k|s_{ij}))(\log(1-e_p(k|s_{ij})) - \log(1-\hat{p}(k|s_{ij})))]$$

where $e_p(k|s_{ij})$ is the soft label to be fitted and $\hat{p}(k|s_{ij})$ is the current model' s prediction. This loss is calculated only on span matrices related to old categories.

The final loss is obtained through weighted summation of BCE loss and KD loss across multiple training stages:

$$L = \alpha L_{BCE} + \beta L_{KD}$$

where α and β are weight hyperparameters for the two loss terms.

4.3.3 Model Performance SpanKL demonstrates superior continual learning performance on both OntoNotes and Few-NERD datasets. On OntoNotes under three mainstream settings (Split-All, Split-Filter, Filter-Filter), SpanKL improves F1 by 3.95%, 3.78%, and 2.25% respectively compared to the second-best model AddNER, with the final step gap to the non-CL upper bound reduced to -0.76, -0.65, and -3.83, significantly outperforming existing mainstream methods. In contrast, while AddNER performs close to ExtendNER on Split-All, it shows more robustness on Filter-based settings, whereas ExtendNER degrades severely under complex settings. Additionally, in Few-NERD where multiple categories need to be learned incrementally at each step, SpanKL still improves F1 by 2.83% over AddNER, further validating its adaptability to complex scenarios.

Further investigation shows that SpanKL can more stably maintain performance on learned entities, with higher and smoother per-category curves, less forgetting, and more robust cross-task transfer. Meanwhile, its separated representation mechanism under multi-task settings effectively alleviates entity interference, demonstrating stronger entity generalization and anti-forgetting capabilities, significantly outperforming AddNER and ExtendNER based on label concatenation modeling, validating its potential as an effective baseline for the CL-NER field.

4.4 MFME-NER (Multi-Feature Memory Encoding for NER)

MFME-NER (Multi-Feature Memory Encoding for NER), proposed by Liu et al. (2025) [24], focuses on Chinese NER tasks in the psychological medicine domain and proposes a new sequence labeling method that fuses multi-level and multi-granularity feature information.

This method addresses the difficulties in entity recognition caused by multi-granularity information (including characters, pinyin, radicals, etc.) and scarce domain knowledge in psychological medicine texts. It designs a multi-level feature fusion mechanism that effectively encodes and integrates multi-source features through improved MFE-BERT combined with GA-FNN Attention modules, thereby enhancing the model' s modeling capability and generalization ability for domain-specific entities in psychological medicine. Experiments show that MFME-NER achieves superior results compared to existing mainstream methods on domain tasks such as the CCKS-2019 mental health dataset, demonstrating stronger robustness and generalization particularly in fine-grained entity and domain-specific entity recognition, providing an effective paradigm and important reference for Chinese NER research in the medical health domain.

4.4.1 Research Motivation Chinese NER tasks in the psychological medicine domain face challenges such as complex domain text structures, scattered knowledge, and insufficient generalization capability of existing models, making it difficult for general pre-trained models to effectively recognize domain-specific entities. Additionally, Chinese text contains multi-granularity information such as characters, pinyin, and radicals, which are particularly important for entity recognition in the psychological medicine domain but have not been fully exploited and fused by existing methods.

To address this, MFME-NER proposes a model that fuses multi-level and multi-granularity features, combining improved MFE-BERT with GA-FNN Attention mechanisms to enhance model robustness and recognition performance for Chinese entities in the psychological medicine domain.

4.4.2 Research Method [Figure 14: see original paper] shows the MFME-NER model architecture.

Character-Level Feature Extraction: For psychological medicine domain long texts where entities consist of many Chinese characters and contextual meanings are variable, MFME-NER adopts character-level modeling to improve accuracy in entity boundary recognition and semantic modeling. Input text is treated as a character sequence, and an improved pre-trained model MFE-BERT maps characters to feature vectors, avoiding noise from word segmentation.

MFE-BERT builds upon standard BERT by fusing information from multi-layer Transformer encoder outputs, concatenating contextual semantic features from different layers to enhance representation richness and semantic completeness.

Specifically, the model concatenates attention outputs from each layer and applies linear transformation to obtain feature vectors containing both shallow lexical information and deep semantic information, alleviating semantic dilution during single-layer information propagation. Finally, character-level feature representations are output through fully connected dimensionality reduction, improving representation effectiveness and understanding of psychological medicine terminology.

Subsequently, the model uses BiLSTM to further model contextual relationships in character sequences, combining forward and backward state sequences to capture long-distance dependencies and richer contextual information, improving entity recognition effectiveness under long and complex syntactic structures. BiLSTM effectively enhances model robustness for character-level psychological medicine entities, with output features more accurately representing long-distance and bidirectional semantic relationships, boosting downstream task recognition performance.

Structure and Pinyin Granularity Feature Modeling: To enhance the model' s ability to model fine-grained semantic information of Chinese characters in the psychological medicine domain, MFME-NER introduces a dual-granularity feature fusion mechanism for structure and pinyin. For structure granularity, the model uses Chinese dictionary tools to decompose characters into components such as radicals, obtains the structure sequence, and maps it to structure embeddings x_{rd}^i through Word2Vec training:

$$x_{rd}^i = e_{rd}(f_{radical}(s_i))$$

where $f_{radical}$ maps character s_i to structural information and e_{rd} is the structure lookup table. Then Convolutional Neural Network (CNN) extracts local and global features, with final structure feature representation obtained through max pooling:

$$h_{max}^{(rd)} = \max\text{pool}(h^{(rd)})$$

For pinyin granularity, the model obtains text pinyin sequences using pinyin tools and gets pinyin embeddings x_{py}^i through Word2Vec training:

$$x_{py}^i = e_{py}(f_{pinyin}(s_i))$$

where f_{pinyin} is the pinyin mapping function and e_{py} is the pinyin lookup table. Similarly, CNN extracts local features and max pooling yields pinyin granularity features:

$$h_{max}^{(pinyin)} = \max\text{pool}(h^{(pinyin)})$$

To fuse the above dual-granularity information, MFME-NER concatenates structure and pinyin features and completes final fusion through a fully connected layer:

$$x_{rd\&pinyin} = \text{Concat}(h_{max}^{(rd)}, h_{max}^{(pinyin)})W + b$$

where W is the weight matrix and b is the bias term. This mechanism effectively integrates structural and phonetic feature information, improving the model' s ability to recognize entities with multiple meanings, pronunciations, and structures in the psychological medicine domain.

GA-FNN Attention Model Design: To address the difficulty of modeling long-distance dependencies in long psychological medicine texts for entity recognition, MFME-NER proposes a multi-modal multi-granularity feature fusion method based on Gated Feed-Forward Neural Network Attention (GA-FNN Attention).

This method focuses on three types of semantic features: character, pinyin, and structure, which are modeled separately and then fused through a gating mechanism, thereby enhancing the model' s modeling capability and robustness for complex texts.

GA-FNN Attention consists of two parts. First, the FNNAttention mechanism performs weighted aggregation of multi-granularity features to obtain global context information. Taking character granularity as an example, its global features are calculated as:

$$\hat{h}_c^t = \sum_{i=1}^{n} \frac{\exp(u(h_c^i))}{\sum_{j=1}^{n} \exp(u(h_c^j))} h_c^i$$

where $u(\cdot)$ represents the feed-forward neural network and h_c^i is the character granularity state sequence output by BiLSTM. Local features for pinyin and structure granularity are extracted through CNN layers, with similar calculations:

$$\hat{h}_{pr}^t = \sum_{i=1}^{n} \frac{\exp(u(h_{pr}^i))}{\sum_{j=1}^{n} \exp(u(h_{pr}^j))} h_{pr}^i$$

Subsequently, different granularity information is adaptively fused through a gating mechanism:

$$\hat{h}_c^t = \tanh(W_{\hat{c}} h_{\hat{c}}^t + b_{\hat{c}})$$
$$\hat{h}_{pr}^t = \tanh(W_{\hat{pr}} h_{\hat{pr}}^t + b_{\hat{pr}})$$

$$g_t = \sigma(W_{gt}(\hat{h}_c^t \oplus \hat{h}_{pr}^t))$$

$$z_t = g_t \hat{h}_c^t + (1 - g_t)\hat{h}_{pr}^t$$

where $W_{\hat{c}}$, $W_{\hat{pr}}$, and W_{gt} are weight matrices, $b_{\hat{c}}$ and $b_{\hat{pr}}$ are bias terms, σ is the Sigmoid activation, $\oplus$ denotes concatenation, g_t is the gating coefficient, and z_t is the final fused representation serving as input to the CRF layer. This mechanism effectively alleviates feature dilution and noise accumulation in long texts, improving model stability and generalization capability.

CRF Label Prediction and Loss Function Optimization: To further improve label dependency modeling for entity recognition, MFME-NER introduces a Conditional Random Field (CRF) layer in the final stage to optimize label prediction results through sequential modeling. Compared to traditional per-character independent prediction methods, CRF can effectively capture dependencies between labels, combining the aforementioned multi-granularity feature fusion results to improve label prediction rationality and consistency. This structure is shown in [Figure 15: see original paper].

The final model loss function is designed based on the CRF layer, with regularization terms introduced to avoid overfitting:

$$L = -S_t - \log(e^{S_1} + e^{S_2} + \cdots + e^{S_N}) + \alpha\|\theta\|^2$$

where S_t represents the score of the true annotation path, $P_{total} = \sum e^{S_i}$ is the total score of all possible paths, θ represents trainable model parameters, and α is a hyperparameter determined through cross-validation to penalize model parameters and suppress overfitting.

4.4.3 Model Performance To validate the effectiveness of the proposed MFME-NER model in psychological medicine entity recognition tasks, the authors conducted extensive comparative experiments on a self-constructed psychological medicine dataset (PsyDataset) and the public CBLUE biomedical text dataset. On PsyDataset, the model recognizes seven entity types: disease, susceptible population, symptom, alias, affected body part, examination, and department. The dataset contains 3,927 manually annotated entities, with symptom entities being the most numerous at 924, and other categories such as disease (627) and examination (313) being reasonably distributed, reflecting characteristics of real psychological medicine texts.

Experiments use Precision, Recall, and F1 as evaluation metrics. Results show that MFME-NER achieves an F1 score of 85.35% on the PsyDataset character-level test set and 80.78% on the CBLUE dataset character-level test set, both significantly outperforming comparison methods and demonstrating strong entity recognition performance and generalization capability. Furthermore, MFME-NER also achieves excellent performance at the entity level, with F1 scores of 83.00% on PsyDataset and 79.95% on CBLUE, both higher than mainstream

baseline models, indicating that the proposed method can effectively alleviate polysemy and homophone issues in psychological medicine texts and improve entity boundary and semantic modeling capabilities.

Additionally, the authors analyzed recognition effectiveness across different label categories, with results showing that MFME-NER has stronger recognition robustness for longer entities such as symptoms and diseases, benefiting from effective modeling of long-distance dependency information through multi-granularity feature fusion and gating mechanisms. Comprehensive results demonstrate that MFME-NER possesses good adaptability and practical value in psychological medicine and related domains.

4.5 FLAT (Flat-Lattice Transformer)

FLAT (Flat-Lattice Transformer), proposed by Li et al. (ACL 2020) [25], aims to solve efficiency and effectiveness problems in Chinese NER tasks when fusing lexical information.

4.5.1 Research Motivation Chinese NER is more challenging than English due to its lack of explicit word boundaries, typically requiring dictionary-based construction of character-word lattice structures to introduce potential word information. Numerous studies have shown that lattice structures can effectively improve NER performance, but their complex and dynamic graph structures make it difficult for existing models (such as lattice LSTM) to fully utilize GPU parallel computing, resulting in low inference speed. Additionally, some work attempts to convert lattice to Graph Neural Network (GNN) modeling, but such methods still require additional RNN integration to preserve sequential information, increasing model complexity and inference cost.

To address these issues, FLAT innovatively proposes converting lattice structures into flattened span sets and leveraging Transformer' s powerful long-distance modeling capability and parallel computing advantages. Through carefully designed head-tail position encoding, FLAT significantly improves model efficiency while retaining lattice lexical information. This method explicitly annotates each span (character or word) with its start and end positions in the original sequence, allowing characters to directly interact with words containing them through Transformer's global self-attention mechanism, thereby enhancing the model' s representation capability for lexical boundaries and entity recognition. Experiments show that FLAT achieves performance and inference efficiency surpassing existing dictionary-enhanced methods on multiple standard Chinese NER datasets such as OntoNotes and MSRA, providing an efficient and feasible new paradigm for Chinese NER. [Figure 16: see original paper] shows the FLAT model workflow.

4.5.2 Research Method Lattice to Flat Structure Conversion: After obtaining the character-level lattice structure based on a dictionary, it can be further converted to a flat structure. Flat-Lattice can be defined as a set of

spans, where each span corresponds to a token and its start and end positions, i.e., head and tail, as shown in Figure 16: see original paper. Tokens can be characters or words. Head and tail represent the position indices of the start and end characters of the token in the original character sequence, determining the token' s specific position in the lattice. For character-level tokens, head and tail positions are identical.

The process of recovering Flat-Lattice to the original lattice structure is straightforward: first select tokens where head and tail coincide to recover the original character sequence; then use other tokens (words) to construct skip-paths based on their head-tail information. Since this conversion process is reversible, Flat-Lattice is considered capable of preserving the original lattice structure information.

Span Relative Position Encoding: FLAT' s flat-lattice structure consists of spans of different lengths. To model interactions between spans, a relative position encoding method is proposed. For any two spans x_i and x_j, based on their start and end positions, there are three relationships: intersection, containment, and separation. Unlike directly encoding relationships discretely, FLAT transforms head and tail information into dense vectors through continuous transformation to express richer structural relationship information, improving modeling of fine-grained relationships between characters and words. Define the start and end positions of x_i as $head[i], tail[i]$, and calculate four relative distances:

$$d_{ij}^{(hh)} = head[i] - head[j], \quad d_{ij}^{(ht)} = head[i] - tail[j]$$
$$d_{ij}^{(th)} = tail[i] - head[j], \quad d_{ij}^{(tt)} = tail[i] - tail[j]$$

Finally, the four distance encodings are fused into a relation representation:

$$R_{ij} = \text{ReLU}(W_r[p_{d^{(hh)}} \oplus p_{d^{(ht)}} \oplus p_{d^{(th)}} \oplus p_{d^{(tt)}}])$$

where W_r is a learnable parameter, $\oplus$ denotes concatenation, and p_d calculation follows [17]:

$$p_{(2k)}(d) = \sin(d/10000^{2k/d_{model}}), \quad p_{(2k+1)}(d) = \cos(d/10000^{2k/d_{model}})$$

Next, FLAT introduces this relation representation into the Transformer attention mechanism:

$$A_{i,j}^* = W_q^\top x_i W_{k,E}^\top x_j + W_q^\top x_i W_{k,R}^\top R_{ij} + u^\top x_j W_{k,E} + v^\top R_{ij} W_{k,R}$$

where $W_q, W_{k,E}, W_{k,R} \in \mathbb{R}^{d_{model} \times d_{head}}$, and $u, v \in \mathbb{R}^{d_{head}}$ are learnable parameters. Finally, A is replaced with A^*, and other calculations remain the same as vanilla Transformer.

The model ultimately retains only character-level representations for output to the CRF layer for labeling [11].

4.5.3 Model Performance Experimental results on four Chinese NER datasets (OntoNotes, MSRA, Resume, Weibo) show that FLAT improves F1 by an average of 1.72 points compared to TENER [26] without lexical information, and by an average of 1.51 points compared to Lattice LSTM-based models. Compared with other lexicon-enhanced models (CGN [27]), FLAT achieves an average F1 improvement of 0.73, demonstrating superior performance in Chinese NER tasks. The improvements are particularly significant on large datasets such as OntoNotes and MSRA, while relatively smaller on small datasets like Resume and Weibo.

Further experiments show that FLAT' s fully connected self-attention mechanism brings two significant advantages over Lattice LSTM: (1) it can explicitly model interactions between characters and their corresponding words, and (2) it effectively captures long-distance dependencies. Masking self-matching word attention leads to significant F1 drops, validating the importance of this mechanism.

Additionally, in terms of inference efficiency, FLAT leverages Transformer's high parallelization characteristics, achieving significant speedup compared to Lattice LSTM and GNN methods. At batch size 16, FLAT achieves approximately 4.97x speedup over single-sample inference, while Lattice LSTM only achieves 2.1x.

To investigate the source of model improvements, the authors introduce two fine-grained evaluation metrics: Span F and Type Acc. Results show that FLAT improves more significantly on Span F, indicating more accurate entity boundary localization, while word-level representations also enhance type classification capability.

Finally, combining pre-trained BERT word representations, FLAT+BERT further improves performance. On large-scale datasets like OntoNotes and MSRA, FLAT+BERT improves F1 by 1.68 and 1.14 points respectively compared to BERT+CRF, while improvements are relatively smaller on small datasets like Resume and Weibo.

5 Evaluation and Datasets

5.1 NER Evaluation Metrics

Grishman and Sundheim (1996) proposed a scoring method for NER performance based on two dimensions: type and text [28]. Type scoring focuses on whether the model' s predicted labels are correct regardless of entity boundary accuracy, while text scoring focuses on whether the model' s predicted entity boundaries are accurate regardless of label correctness. Under each scoring dimension, precision is defined as the ratio of correctly predicted entities to total predicted entities; recall is defined as the ratio of correctly predicted entities to

total human-annotated entities; and (micro-averaged) F1 score is defined as the harmonic mean of precision and recall.

CoNLL evaluation (Tjong Kim Sang and De Meulder, 2003; Tjong Kim Sang, 2002) proposed strict matching evaluation metrics [2]: a prediction is considered correct only when both the entity label and its boundaries match the gold standard exactly. CoNLL also adopts micro-averaged F1 score, measuring model performance through the harmonic mean of precision and recall.

Additionally, relaxed F1 and strict F1 evaluation metrics are widely used in many NER domain shared tasks [5-6, 29]. Relaxed F1 considers a named entity correctly recognized as long as part of it is identified correctly, while strict F1 requires character-level boundary consistency between model predictions and human annotations. In some datasets (e.g., as described in Liu et al., 2015) [30], unlike CoNLL standards, word-level boundary annotations are not provided, so relaxed F1 metrics are designed to alleviate comparison issues caused by different segmentation across systems, enabling comparability across systems with different segmentation granularities.

The commonly used NER evaluation metrics are defined below.

Precision: Precision measures how many of the entities predicted as positive by the model are truly entities, defined as the ratio of correctly predicted entities to total predicted entities. Let TP be the number of entities predicted and truly being entities, and FP be the number of non-entities incorrectly predicted as entities. The precision calculation formula is:

$$\text{Precision} = \frac{TP}{TP + FP} \times 100\%$$

Recall: Recall measures the model' s coverage of true entities, defined as the ratio of correctly identified entities to all true entities. Let FN be the number of true entities incorrectly predicted as non-entities. The recall calculation formula is:

$$\text{Recall} = \frac{TP}{TP + FN} \times 100\%$$

F1 Score: F1 score is the harmonic mean of precision and recall, used to comprehensively evaluate the trade-off between precision and coverage. Its calculation formula is:

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \times 100\%$$

In NER tasks, the above metrics are typically calculated at the entity level rather than word level, meaning a prediction is considered correct only when the model correctly identifies both the start/end boundaries and category of an

entity. This evaluation approach better aligns with the essential requirements of NER tasks and more accurately reflects the model' s ability to recognize named entities. These three metrics together provide a comprehensive performance evaluation perspective for NER models, corresponding to different dimensions of "accuracy," "completeness," and "overall performance," serving as core criteria for measuring NER algorithm quality.

5.2 Common NER Datasets

Since Grishman and Sundheim (1996) proposed the first NER shared task [1], the research community has constructed numerous shared tasks and datasets for NER. CoNLL 2002 (Tjong Kim Sang, 2002) and CoNLL 2003 (Tjong Kim Sang and De Meulder, 2003) [2] are based on news corpora, covering four languages: Spanish, Dutch, English, and German, focusing on four named entity types: PER (person), LOC (location), ORG (organization), and MISC (miscellaneous, including all entities that cannot be classified into the first three categories). In addition, NER shared tasks for multiple languages have been conducted subsequently, including Indian languages [33], Arabic [31], German [32], and Slavic languages [4]. Named entity categories vary significantly across different datasets and languages. For example, the Southeast Asian language dataset proposed by Rajeev Sangal and Singh (2008) [33] includes categories such as person names, positions, time expressions, abbreviations, numbers, and brands. The dataset constructed by Benikova et al. (2014) based on German Wikipedia and online news maintains the same entity categories as CoNLL 2002 and 2003, covering person names (PER), organization names (ORG), location names (LOC), and others (OTH) [32]. The shared task organized by Piskorski et al. (2017) covers seven Slavic languages (Croatian, Czech, Polish, Russian, Slovak, Slovenian, and Ukrainian), with entity categories also including person names, location names, organization names, and others [4].

In the biomedical domain, Kim et al. (2004) organized the BioNER task based on MedLine abstracts, focusing on entity categories such as proteins, DNA, RNA, and cell attributes [34]. Uzuner et al. (2007) proposed a clinical note de-identification task requiring NER models to identify patient personal information phrases that need anonymization [35]. The 2010 I2B2 NER task (Uzuner et al., 2011) also focused on clinical data, covering three entity types: clinical problems, examinations, and treatments [35]. Segura Bedmar et al. (2013) organized a drug NER shared task in SemEval 2013 Task 9, with entity categories including drugs, brands, drug categories, and unapproved or new drugs (drug_n) [5]. Krallinger et al. (2015) proposed the CHEMDNER task, further focusing on chemical and drug domain entities such as common names, systematic names, abbreviations, chemical formulas, families, and identifiers [29]. Biological and microbiological NER datasets [6, 36] mostly originate from PubMed and biology websites, with entity categories focusing on bacteria, habitats, and geographic locations. In biomedical NER systems, segmentation of clinical and drug-related entities is considered particularly challenging due to complex writing forms of

named entities [30].

Additionally, social media (such as Twitter) has become an important domain for NER research, where traditional NER systems experience significant performance degradation due to spelling variations and incomplete syntactic structures [37]. Entity categories in Twitter are more diverse, covering person names, companies, facilities, bands, sports teams, movies, TV shows, etc., mostly based on user behavior habits. While most named entity annotations adopt flat structures, some datasets introduce more complex structures. For example, Ohta et al. (2002) constructed a dataset containing nested named entities where one entity can contain other entities inside [38]. Strassel et al. (2003) annotated entities and their head phrases [39]. In chemical and clinical NER datasets, discontinuous entities are common [29]. Eltyeb and Salim (2014) surveyed various NER systems proposed for the aforementioned different NER datasets, focusing on research progress in chemical domain NER [40].

6 Summary and Conclusions

This paper systematically reviews and analyzes research progress in the NER field, from traditional methods to deep learning approaches. In the traditional methods section, we detailed HMM, CRF, and rule-based and feature engineering methods, summarizing their basic principles, application methods, and limitations. In the deep learning methods section, we focused on analyzing the application of sequence models such as RNN and BiLSTM-CRF in NER, further 梳理了 BERT 及其衍生模型在 NER 任务中的核心思路与优势, and provided detailed introductions and performance analyses of recently emerging representative models (LEBERT, SpanKL, MFME-NER, BERT-CRF, FLAT).

Overall, deep learning methods have become the mainstream direction in current NER research, especially approaches combining pre-trained language models, which significantly improve entity recognition effectiveness and robustness, possessing stronger cross-domain generalization capabilities. Meanwhile, recent research has not only focused on model accuracy improvements but also explored multi-granularity information fusion, multi-task adaptation, and continual learning, driving NER technology to better meet practical application requirements. Experimental results demonstrate that strategies such as lexical enhancement, explicit modeling of context and entity structure, and multi-granularity feature fusion have clear effects on improving NER task performance.

Although existing methods have achieved excellent performance on public datasets, there remains room for improvement in areas such as domain adaptation, long text modeling, low-resource environments, and complex entity structures. Additionally, different task settings (such as cross-task continual learning and multilingual NER) pose higher requirements for model generalization and adaptation capabilities.

In summary, NER tasks have evolved from early rule-based and statistical models to method systems centered on deep learning and pre-training, with research

focus gradually expanding from performance improvement to more challenging practical application problems. Future research should continue to focus on enhancing model adaptability to complex scenarios, reducing dependence on annotated data, and exploring more interpretable and generalizable modeling methods to further promote the practical deployment and application of NER technology across various domains.

References

[1] R. Grishman and B. Sundheim, "Message understanding conference-6: a brief history," in *Proceedings of the 16th Conference on Computational Linguistics - Volume 1*, ser. COLING ' 96. USA: Association for Computational Linguistics, 1996, p. 466–471. [Online]. Available: https://doi.org/10.3115/992628.992709

[2] E. F. Tjong Kim Sang and F. De Meulder, "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition," in *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, 2003, pp. 142–147. [Online]. Available: https://aclanthology.org/W03-0419/

[3] E. F. Tjong Kim Sang, "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)*, 2002. [Online]. Available: https://aclanthology.org/W02-2024/

[4] J. Piskorski, L. Pivovarova, J. Šnajder, J. Steinberger, and R. Yangarber, "The first cross-lingual challenge on recognition, normalization, and matching of named entities in Slavic languages," in *Proceedings of the 6th Workshop on Balto-Slavic Natural Language Processing*, T. Erjavec, J. Piskorski, L. Pivovarova, J. Šnajder, J. Steinberger, and R. Yangarber, Eds. Valencia, Spain: Association for Computational Linguistics, Apr. 2017, pp. 76–85. [Online]. Available: https://aclanthology.org/W17-1412/

[5] I. Segura-Bedmar, P. Martínez, and M. Herrero-Zazo, "SemEval-2013 task 9: Extraction of drug-drug interactions from biomedical texts (DDIExtraction 2013)," in *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, S. Manandhar and D. Yuret, Eds. Atlanta, Georgia, USA: Association for Computational Linguistics, Jun. 2013, pp. 341–350. [Online]. Available: https://aclanthology.org/S13-2056/

[6] R. Bossy, W. Golik, Z. Ratkovic, P. Bessières, and C. Nédellec, "BioNLP shared task 2013 –an overview of the bacteria biotope task," in *Proceedings of the BioNLP Shared Task 2013 Workshop*, C. Nédellec, R. Bossy, J.-D. Kim, J.-j. Kim, T. Ohta, S. Pyysalo, and P. Zweigenbaum, Eds. Sofia, Bulgaria: Association for Computational Linguistics, Aug. 2013, pp. 161–169. [Online]. Available: https://aclanthology.org/W13-2024/

[7] Ö. Uzuner, B. R. South, S. Shen, and S. L. DuVall, "2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text," *Journal of the American*

Medical Informatics Association: JAMIA, vol. 18, no. 5, pp. 552–556, 2011.

[8] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, 08 2007.

[9] 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. null, p. 2493–2537, Nov. 2011.

[10] L. E. Baum and T. Petrie, "Statistical Inference for Probabilistic Functions of Finite State Markov Chains," *The Annals of Mathematical Statistics*, vol. 37, no. 6, pp. 1554 –1563, 1966. [Online]. Available: https://doi.org/10.1214/aoms/1177699147

[11] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the Eighteenth International Conference on Machine Learning*, ser. ICML ' 01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, p. 282–289.

[12] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," 2015. [Online]. Available: https://arxiv.org/abs/1508.01991

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423/

[14] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019. [Online]. Available: https://arxiv.org/abs/1907.11692

[15] J. Deriu, K. Mlynchyk, P. Schläpfer, A. Rodrigo, D. von Grünigen, N. Kaiser, K. Stockinger, E. Agirre, and M. Cieliebak, "A methodology for creating question answering corpora using inverse data annotation," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, Eds. Online: Association for Computational Linguistics, Jul. 2020, pp. 897–911. [Online]. Available: https://aclanthology.org/2020.acl-main.84/

[16] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Łukasz Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean, "Google' s neural machine translation system: Bridging the gap between human and machine translation," 2016. [Online]. Available: https://arxiv.org/abs/1609.08144

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023. [Online]. Available: https://arxiv.org/abs/1706.03762

[18] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," OpenAI Blog, 2018. [Online]. Available: https://cdn.openai.com/research-covers/language-unsupervised/language_{{{understanding}}}_{{paper}}}.pdf

[19] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020. [Online]. Available: https://arxiv.org/abs/1910.10683

[20] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," 2020. [Online]. Available: https://arxiv.org/abs/1906.08237

[21] S. Hu, H. Zhang, X. Hu, and J. Du, "Chinese named entity recognition based on bert-crf model," in *2022 IEEE/ACIS 22nd International Conference on Computer and Information Science (ICIS)*, 2022, pp. 105–108.

[22] W. Liu, X. Fu, Y. Zhang, and W. Xiao, "Lexicon enhanced chinese sequence labeling using bert adapter," 2021. [Online]. Available: https://arxiv.org/abs/2105.07148

[23] Y. Zhang and Q. Chen, "A neural span-based continual named entity recognition model," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 11, p. 13993–14001, Jun. 2023. [Online]. Available: http://dx.doi.org/10.1609/aaai.v37i11.26638

[24] Z. Liu, G. Zhang, and Y. Shen, "Psychomedical named entity recognition method based on multi-level feature extraction and multi-granularity embedding fusion," *Scientific Reports*, vol. 15, no. 1, p. 16927, May 2025, publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41598-025-90939-8

[25] X. Li, H. Yan, X. Qiu, and X. Huang, "FLAT: Chinese NER using flat-lattice transformer," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, Eds. Online: Association for Computational Linguistics, Jul. 2020, pp. 6836–6842. [Online]. Available: https://aclanthology.org/2020.acl-main.611/

[26] H. Yan, B. Deng, X. Li, and X. Qiu, "Tener: Adapting transformer encoder for named entity recognition," 2019. [Online]. Available: https://arxiv.org/abs/1911.04474

[27] S. Li, Z. Zhao, R. Hu, W. Li, T. Liu, and X. Du, "Analogical reasoning on Chinese morphological and semantic relations,"in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, I. Gurevych and Y. Miyao, Eds. Melbourne, Australia: Association

for Computational Linguistics, Jul. 2018, pp. 138–143. [Online]. Available: https://aclanthology.org/P18-2023/

[28] R. Grishman and B. Sundheim, "Message Understanding Conference-6: A brief history," in *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, 1996. [Online]. Available: https://aclanthology.org/C96-1079/

[29] M. Krallinger, O. Rabal, F. Leitner, M. Vazquez, D. Salgado, Z. Lu, R. Leaman, Y. Lu, D. Ji, D. M. Lowe, R. A. Sayle, R. T. Batista-Navarro, R. Rak, T. Huber, T. Rocktäschel, S. Matos, D. Campos, B. Tang, H. Xu, T. Munkhdalai, K. H. Ryu, S. Ramanan, S. Nathan, S. Žitnik, M. Bajec, L. Weber, M. Irmer, S. A. Akhondi, J. A. Kors, S. Xu, X. An, U. K. Sikdar, A. Ekbal, M. Yoshioka, T. M. Dieb, M. Choi, K. Verspoor, M. Khabsa, C. L. Giles, H. Liu, K. E. Ravikumar, A. Lamurias, F. M. Couto, H.-J. Dai, R. T.-H. Tsai, C. Ata, T. Can, A. Usié, R. Alves, I. Segura-Bedmar, P. Martínez, J. Oyarzabal, and A. Valencia, "The CHEMDNER corpus of chemicals and drugs and its annotation principles," *Journal of Cheminformatics*, vol. 7, no. 1, p. S2, Jan. 2015. [Online]. Available: https://doi.org/10.1186/1758-2946-7-S1-S2

[30] S. Liu, B. Tang, Q. Chen, and X. Wang, "Drug-Drug Interaction Extraction via Convolutional Neural Networks," *Computational and Mathematical Methods in Medicine*, vol. 2016, no. 1, p. 6918381, 2016, _{eprint}: https://onlinelibrary.wiley.com/doi/pdf/10.1155/2016/6918381. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1155/2016/6918381

[31] K. Shaalan, "A survey of Arabic named entity recognition and classification," *Computational Linguistics*, vol. 40, no. 2, pp. 469–510, Jun. 2014. [Online]. Available: https://aclanthology.org/J14-2008/

[32] D. Benikova, C. Biemann, and M. Reznicek, "NoSta-D named entity annotation for German: Guidelines and dataset," in *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC' 14)*, N. Calzolari, K. Choukri, T. Declerck, H. Loftsson, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis, Eds. Reykjavik, Iceland: European Language Resources Association (ELRA), May 2014, pp. 2524–2531. [Online]. Available: https://aclanthology.org/L14-1251/

[33] A. K. Singh, "Named entity recognition for south and south east asian languages: Taking stock,"in *International Joint Conference on Natural Language Processing*, [Online]. Available: https://api.semanticscholar.org/CorpusID:15441841

[34] N. Collier, T. Ohta, Y. Tsuruoka, Y. Tateisi, and J.-D. Kim, "Introduction to the bio-entity recognition task at JNLPBA," in *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA/BioNLP)*, N. Collier, P. Ruch, and A. Nazarenko, Eds. Geneva, Switzerland: COLING, Aug. 28th and 29th 2004, pp. 73–78. [Online]. Available: https://aclanthology.org/W04-1213/

[35] Ö. Uzuner, Y. Luo, and P. Szolovits, "Viewpoint paper: Evaluating the state-of-the-art in automatic de-identification," *J. Am. Medical Informatics Assoc.*, vol. 14, pp. 550–563, 2007. [Online]. Available: https://api.semanticscholar.org/CorpusID:8746850

[36] L. Hirschman, A. Yeh, C. Blaschke, and A. Valencia, "Overview of biocreative: Critical assessment of information extraction for biology," *Bioinformatics*, vol. 21, no. 15, pp. 3011–3016, 2005.

[37] T. Baldwin, M. C. de Marneffe, B. Han, Y.-B. Kim, A. Ritter, and W. Xu, "Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition," in *Proceedings of the Workshop on Noisy User-generated Text*, W. Xu, B. Han, and A. Ritter, Eds. Beijing, China: Association for Computational Linguistics, Jul. 2015, pp. 126–135. [Online]. Available: https://aclanthology.org/W15-4319/

[38] T. Ohta, Y. Tateisi, and J.-D. Kim, "The genia corpus: an annotated research abstract corpus in molecular biology domain," in *Proceedings of the Second International Conference on Human Language Technology Research*, ser. HLT ' 02. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002, p. 82–86.

[39] S. Strassel and A. Mitchell, "Multilingual resources for entity extraction," in *Proceedings of the ACL 2003 Workshop on Multilingual and Mixed-language Named Entity Recognition*. Sapporo, Japan: Association for Computational Linguistics, Jul. 2003, pp. 49–56. [Online]. Available: https://aclanthology.org/W03-1507/

[40] S. Eltyeb and N. Salim, "Chemical named entities recognition: A review on approaches and applications," *Journal of cheminformatics*, vol. 6, p. 17, 04 2014.

Note: Figure translations are in progress. See original paper for figures.