Hindawi
International Journal of Intelligent Systems
Volume 2023, Article ID 3322735, 16 pages
https://doi.org/10.1155/2023/3322735

Research Article

A TL_FLAT Model for Chinese Text Datasets of UAV Power Systems: Optimization and Performance

Mingming Shen[1,2,3] **Shaobo Li**[1,2] **Jing Yang**[1,2] **Ansi Zhang**[1,2] **Qiuchen He**[1] **and Ruiqiang Pu**[1]

[1]*School of Mechanical Engineering, Guizhou University, Guiyang 550025, China*
[2]*State Key Laboratory of Public Big Data, Guizhou University, Guiyang 550025, China*
[3]*School of Mechanical & Electrical Engineering, Guizhou Normal University, Guiyang 550025, China*

Correspondence should be addressed to Shaobo Li; lishaobo@gzu.edu.cn

Received 8 November 2022; Revised 14 December 2022; Accepted 17 December 2022; Published 22 February 2023

Academic Editor: B. B. Gupta

The manufacturing processes of unmanned aerial vehicle (UAV) power systems generate large amounts of data and knowledge. The extraction of useful information or patterns from redundant data and knowledge texts has become a challenge in intelligent manufacturing. Unfortunately, graphics processing unit (GPU)-based parallel computing is limited, and the inference speeds of the available named entity recognition (NER) models for Chinese text datasets are low because they are mainly based on the long short-term memory (LSTM) algorithm. Herein, first, the flat-lattice transformer (FLAT) model was optimized by using a stochastic gradient descent with momentum (SGDM) optimizer and adjusting the model hyperparameters. Compared with the existing NER methods, the proposed optimization algorithm achieved better performance on the available dataset. Then, an NER method named the TL_FLAT model based on transfer learning and the abovementioned optimization model was introduced. Finally, a Chinese text dataset from a UAV power system created by the authors was used to validate the proposed method. The F1 score was 76.26%, the precision value was 76.98%, and the recall value was 75.56%, indicating that the TL_FLAT model was suitable for Chinese text entity recognition for UAV power systems.

1. Introduction

Advanced assembly technology determines the manufacturing costs, production cycles, and final quality levels of UAV power systems and is the core indicator of the development level of a country's aviation industry [1, 2]. The textual knowledge of a UAV power system usually exists in the form of unstructured text data, such as technology patents, standards, product manuals, and technical documents. As an important carrier, UAV power system text knowledge contains a large amount of technical information and knowledge generated during the design, production, and manufacturing processes of the aviation industry. However, the extraction of useful information or patterns from redundant data and knowledge texts has become a challenge in intelligent manufacturing.

Natural language processing (NLP) techniques focus on how to mine and use valuable information derived from unstructured data and form an important branch of artificial intelligence. Named entity recognition (NER) is an important direction of NLP that aims to obtain meaningful entities and entity types from large amounts of textual data (e.g., names, locations, and organizations). NER is also one of the key technologies in the knowledge graph for the intelligent assembly of aerospace power systems. NER first appeared at the Sixth Message Understanding Conference (MUC-6), and it essentially extracts various category entities from unstructured input text. Compared with English NER, Chinese NER has no obvious word boundary information, and it is relatively intuitive to use character information for Chinese NER [3]. Many researchers have conducted related studies on Chinese entity recognition methods, such as rule-based methods, dictionary-based methods, deep learning-based methods, and word-information-based methods [4–6]. Wang et al. [7] used sensitive word rule methods to recognize the real conversational intentions of users, solving the error-conduction problem caused by inaccurate word separation outputs and achieving better results. This type of method mainly relies on experts to develop rules, has high corpus requirements, relies strongly on lexicons and rules, and has better recognition effects for entities that exist in the lexicons or are covered in the rules; however, such a method cannot recognize complex entities. Xiang and Shi Xiaodong [8] used a combination of statistics and rules and employed a hidden Markov model for entity recognition, yielding a significant improvement, but their approach could not effectively solve the entity ambiguity problem.

Deep learning, with its ability to autonomously learn features, has been widely used for NER since it was proposed in 2006 [9]. Majtner et al. [10] used a deep learning method to recognize entities belonging to the black tumor category in the medical field, which solved the inefficient manual extraction of features and achieved a better recognition effect, but the method could not effectively solve the complex entity recognition and entity embedding problems. After the bidirectional encoder representations from transformers (BERT) algorithm was proposed [11], Ren et al. [12] used the location vector information in the BERT algorithm to achieve improved recognition of entities such as indicator names, and then they employed a bidirectional long short-term memory (BiLSTM) algorithm to learn the long-sequence semantic features in the quantitative indicators of the standard texts of fisheries. Then, an attention mechanism was fused with the BERT + BiLSTM algorithm to solve the long-sequence semantic dilution problem, and finally, a conditional random field (CRF) layer was used to obtain the predicted sequence labels. However, entity recognition in the field of aviation manufacturing and assembly mainly concerns the recognition of key features such as the algorithm, parts, parameters, materials, functions, structures, and features involved in web pages, documents, patents, technical reports, etc., which are apparently domestic and foreign. Sui et al. [13] proposed a multimodal multitasking algorithm based on their own labeled dataset to explore a multimodal named entity recognition (NER) approach for Chinese textual and auditory content by introducing a speech-to-text alignment assistance task. Zhang et al. [14] proposed a machine reading comprehension framework that integrates adaptive positive untagging techniques into NER and experimentally demonstrated that the framework is effective for datasets containing a large number of untagged entities.

Based on this, deep learning and machine learning were combined to propose an NER method for aviation manufacturing based on transfer learning and a flat-lattice transformer (FLAT) model in this paper. The automatic extraction of entities from a text can lay the foundation for the construction of a knowledge graph for a UAV power system assembly. The main contributions of this paper are as follows:

(1) For the case of data scarcity in the aviation manufacturing industry, a Chinese text dataset was created for the assembly of UAV power system components in a UAV power system.

(2) A model named the TL-FLAT, in which the FLAT model's flat-lattice structure and relative position encoding were combined, making full use of lattice information and possessing the excellent parallelism capability of transformers, was optimized and performed for the Chinese text dataset of the UAV power system.

(3) Transfer learning was used as a guide for achieving effective entity recognition for small-scale datasets with the TL-FLAT algorithm.

2. Related Work

2.1. Overview of Chinese NER. NER has been a research hotspot, and its related methods are mainly based on rules and lexicons, machine learning, deep learning techniques, etc. Gradual lexical enhancement through the introduction of lexical information has become an important means for enhancing Chinese NER indicators.

NER method-based rules and lexicons rely on manually developed rules, which are based on domain-specific gazetteers and syntactic lexical patterns, but such rules have mostly been developed for entity knowledge within a domain and cannot be generalized for all domains. Kim and Woodl [15] used artificial rules for NER in the spoken input text. Alexandra Pomares Quimbaya et al. [16] used an artificial rule-based dictionary to extract named entities from electronic medical record texts, and more well-known systems include LaSIE [17], NetOwl [18], Facile [19], and SAR [20]. When the given lexicon is exhaustive, rule-based systems are good choices. However, in some specific domains, high precision and low recall are often yielded by such systems due to their specific rules and incomplete vocabularies, and these systems are not transferable to other domains.

The traditional machine learning-based NER method takes probability statistics as the essential NER sequence labeling problem. A sequence labeling method extracts entity blocks from marker sequences, and then the extracted entity blocks are grouped together to finally obtain named entities and their categories (composed of several words). There are two main types of methods: supervised and unsupervised methods, and the typical unsupervised learning method is clustering [21]. Collins [22] used only a small amount of seed annotation data and seven entity features, including spelling, the entity context, and the entity itself, for entity recognition. Nadeau et al. [23] proposed an unsupervised system for gazetteer construction and named entity disambiguation; their approach is based on simple and efficient heuristics through the combination of entity extraction and disambiguation. Through supervised learning, NER can be transformed into a multiclass classification or sequence annotation task. Given a data sample with annotations, carefully designed features can be used to represent each training example. Machine learning models can then be used to identify similar patterns from unknown data. Many machine learning models have been applied in supervised NER, such as the hidden Markov model (HMM) [24], the decision tree, the maximum entropy model [25], the support vector machine (SVM) [26], and the conditional random field (CRF) [27].

In recent years, deep learning-based NER models have dominated and achieved some results; compared with feature-based approaches, deep learning helps to automatically discover hidden features. In English NER, neural network-based NER structures are divided into word-level [28, 29] and character-level [30, 31] according to how they represent words in sentences. However, compared with English, Chinese texts have characteristics such as difficulty determining the boundaries of related entities and complex grammatical structures, which make it difficult to name entities in Chinese. Therefore, the Chinese NER task is generally decomposed into two subtasks: Chinese word segmentation and word sequence annotation, which are based on the word sequence information that can enable character-based sequence learning of multiboundary information. However, Chinese word boundary segmentation errors impact entity recognition. To solve this problem, an LSTM model, the Lattice-LSTM model [32], which uses all words in a sentence that are matched by a single character and encodes these words into a directed acyclic graph (DAG), was proposed. Due to its rich lexical information, Lattice-LSTM has achieved good results on various datasets; however, the DAG structure sometimes fails to select the correct path, which may lead to the degradation of the lattice model into a partially word-based model. To solve the abovementioned problem, Liu et al. [33] proposed a new word-character LSTM (WC-LSTM) model to integrate word information into a character-based model, which first represents a Chinese sentence as a series of character-word pairs to integrate word information into each character, thus ensuring that the model does not degenerate into a partially word-based model. They also designed four-word encoding strategies that can encode word information into fixed-size vectors, thus enabling the model to be batch trained and adapted to various application scenarios. Gui et al. [34] proposed a convolutional neural network (CNN)-based Chinese NER model with lexicon rethinking (RCNN), which encodes matched words with different window sizes. The model mainly processes the whole input sentence as well as all potential words in parallel with a CNN and applies a reflection mechanism to handle conflicts between potential words in the lexicon. Gui et al. [35] proposed a lexicon-based graph neural network for Chinese NER (LGN), which treats Chinese NER as a node classification task to achieve better interaction effects and obtain better predictions between characters and words. Sui et al. [36] addressed the lexical matching problem and proposed a character-based collaborative graph network, including an encoding layer, a graph layer, a fusion layer, and a decoding layer. Ding et al. [37] proposed a multidirectional graph model that can effectively solve the conflict matching problem by learning contextual information. However, the abovementioned models all use LSTM as their backbone encoder, and this hybrid encoding technique can make the model complicated because NER is very sensitive to sentence structures. Yan et al. [38] proposed an improved transformer encoder for NER (TENER), which improves the performance of transformer-based models in the NER task by using directional relative position encoding, thus reducing the number of parameters and improving the attention distribution for the Chinese NER task; this approach was shown to perform better than a BiLSTM-based model through experimental validation. Ma et al. [39] proposed implementing a Lattice-LSTM model by combining all matching words for each character into a character-based NER model, which not only encodes lexical information in character representations but also enables a fast inference speed.

2.2. NER in Aviation Manufacturing. The lack of a canonical annotated corpus has been a major difficulty for domain-specific NER; however, the training processes of deep network models usually require a large annotated corpus for training. Otherwise, overfitting tends to occur. Therefore, the direct application of deep network models to specific domains is usually not very useful [40]. One study combined deep learning with active learning and introduced a lightweight NER method, the CNN–CNN-LSTM model, to accelerate this operation and reduce the amount of labeled training data required [41]. Pretrained word embeddings were performed via transfer learning on unlabeled electronic health records for secondary tasks, and the output embeddings were used as the basis for a series of NN architectures [42]. Furthermore, an approach using a combination of active learning and self-learning was proposed to reduce the workload required for the NER task involving tweet streams [43]. Certainly, transfer learning has also been widely used in NER; for example, a mandarin NER module based on a transfer learning system was constructed for collecting and analyzing disaster information in disaster management [44]. To address the lack of labeled data in artifact recognition, [45] proposed a combination of BiLSTM and a CRF for named artifact entity recognition, which is a semisupervised model that uses labeled data to conduct training to achieve effective recognition performance. Some researchers have studied the application of deep learning in the field of aviation. Iacovelli et al. [46–48] applied intelligent reflective surfaces, swarm intelligence, and other strategies to the field of UAV Internet to maximize the reliability and effectiveness of UAV communication. Zhou et al. [49] proposed a distributed graph embedding-based sequence knowledge graph convolutional network model (SKGCN, "sequence knowledge graph convolutional network") for assembly process planning, taking the aero-engine pressurizer rotor as the research object.

3. Applications of the FLAT Used in NER for a UAV Power System Dataset

A UAV power system dataset is characterized by difficulty in determining the boundaries of relevant entities, complex

Chinese grammatical structures, and a small amount of data, so an effective method for improving the effect of NER for UAV power system datasets is urgently needed. This approach should maximize the use of lexical information, avoid the loss of lexical information, be compatible with the lexical architecture, and accelerate the inference speed of the model. Based on this idea, this paper proposes a model that can be used for Chinese text entity recognition for UAV power system parts, the TL_FLAT model. The model takes the relative position relationship information of the entities, words, and characters in a sequence of sentences as its inputs, adopts a transformer model as the encoder and decoder, and then uses a CRF as a classifier to perform sequence labeling on the features output by the transformer to obtain the optimal solution for the labeled sequence.

3.1. Lattice-LSTM Model Architecture. Lattice-LSTM [32] is a Chinese NER model based on an LSTM model that uses Lattice-LSTM to characterize the words in a sentence and integrate the potential word information into a character-based LSTM-CRF, which is a good solution to the NER error problem and the inability to fully utilize the explicit word and word order information contained in the given sentence based on word separation and character-based methods.

Lattice-LSTM can be viewed as adding words as inputs to a character-based NER model, where the lattice is a DAG where each node is a character or a potential word. Suppose that the inputs are a sequence of characters $c_1, c_2, \ldots, c_m$ and a subsequence of all characters matching the words in dictionary D. Lattice-LSTM automatically segments large raw text to construct a dictionary D. $w_{b,e}^{d}$ is used to denote a subsequence in a sentence, where b is the starting subscript of a character, and e is the ending subscript of a character. Figure 1 shows an example with the sentence "(in Chinese) wu ren ji dong li xi tong (UAV power system)" for "(in Chinese) wu ren (nobody)."

The basic structure of Lattice-LSTM is the same as that of character-based LSTM, with c_j representing the j th character of an input with a character sequence $c_1, c_2, \ldots, c_m$ and each character passing through the embedding layer. The word representation can be obtained from the following equation:

$$x_j^c = e^c\left(c_j\right), \tag{1}$$

where e^c is the character embedding matrix.

After the model obtains the character representation through the character embedding matrix, it can obtain the cell state and hidden state through the input gate, forget gate, and output gate. The decoding part of the model uses a CRF layer connected to the top layer of the model, calculates the output probability for each hidden layer, and selects the maximum output probability $P(y \mid s)$. For a labeled sequence $y = (l_1, l_2, \ldots, l_\tau)$, the probability is calculated, as shown in the following equation:

$$P(y \mid s) = \frac{\exp\left(\sum_i\left(W_{CRF}^{l_i} h_i + b_{CRF}^{\left(l_{i-1}, l_i\right)}\right)\right)}{\sum_{y'} \exp\left(\sum_i\left(W_{CRF}^{l_i'} h_i + b_{CRF}^{\left(l_{i-1}', l_i'\right)}\right)\right)}, \tag{2}$$

where h_i is a representation of the hidden layer nodes, y' denotes an arbitrary sequence of labels, $W_{\mathrm{CRF}}^{\mathrm{li}}$ is a model parameter for l_i, $b_{\mathrm{CRF}}^{(l_{i-1}, l_i)}$ is a bias that is specific to l_{i-1} and l_i, $i = 1, 2, \ldots, \tau$, where τ is n for character-based and lattice-based models and m for word-based models.

Although Lattice-LSTM effectively improves the NER performance of the model, its forward and backward lexical information cannot be shared when BiLSTM is utilized due to the recurrent neural network (RNN) characteristics of this network, which do not retain continuous memory with lexical information.

3.2. Model Architecture of the FLAT. A transformer model [50] uses a full-attention structure instead of LSTM, abandoning the traditional encoder-decoder model that must combine the inherent patterns of CNNs or RNNs and using a positional encoding mechanism for data preprocessing, thereby achieving better results while reducing computational effort and improving parallel efficiency.

This model is essentially composed of two structurally similar encoders and decoders, which are stacked by six identical basic layers, each of which consists of two sublayers, multihead attention, and a feedforward network (FFN), as shown in Figure 2.

The principle of the attention mechanism is to mimic the attention of the human brain by selectively allocating its limited attention to the most important information. The encoder-decoder architecture with an attention mechanism is an improvement upon the traditional sequence-to-sequence model architecture, where both the encoder and decoder are RNNs. The encoder accepts an input sequence of tokens and encodes it into a fixed-length vector; the decoder takes a fixed-length vector as input and generates an output sequence in a token-to-token manner.

In the attention layer, the transformer computes the attention of multiple headers independently and stitches the results with a certain weight to obtain the final output; each computation is performed with the following equation:

$$\begin{aligned} A_{ttention}(A, V) &= \operatorname{softmax}(A) V, \\ A_{ij} &= \left(\frac{Q_i K_j^T}{\sqrt{d_k}}\right), \\ |Q, K, V| &= E_x\left|W_q, W_k, W_v\right|, \end{aligned} \tag{3}$$

where Q, K, V is the matrix of input word vectors, d_k is the dimensionality of the input vector, E is the first token embedding layer or the output of the previous layer, $W_q, W_k, W_v \in R^{d_{\text{model}} \times d_{\text{head}}}$ are learnable parameters, and A_{ij} is the attention module that is responsible for automatically learning the attention weights. In the multihead attention layer, K and V come from the output of the encoder, and Q comes from the output of the decoder in the previous layer, so the input of the decoder has the encoder output in addition to the token embeddings and position encodings.

To solve the degradation problem in deep learning, each sublayer of the transformer coding unit is connected with

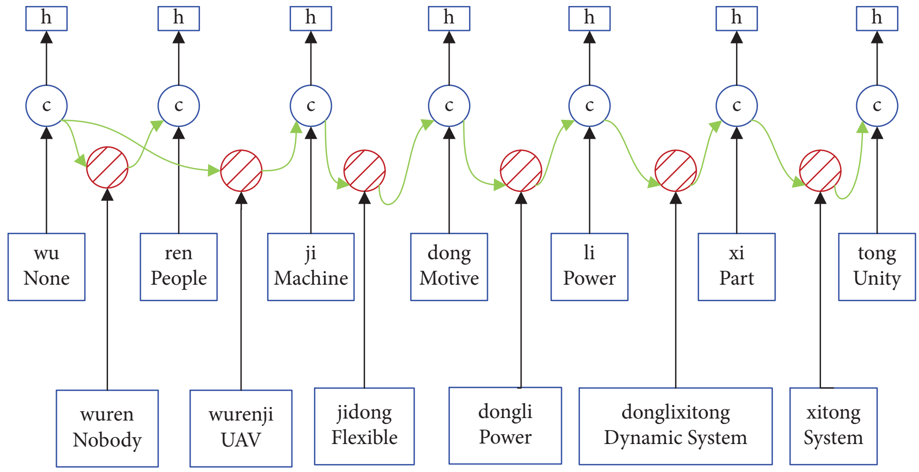

FIGURE 1: Lattice-LSTM architecture.

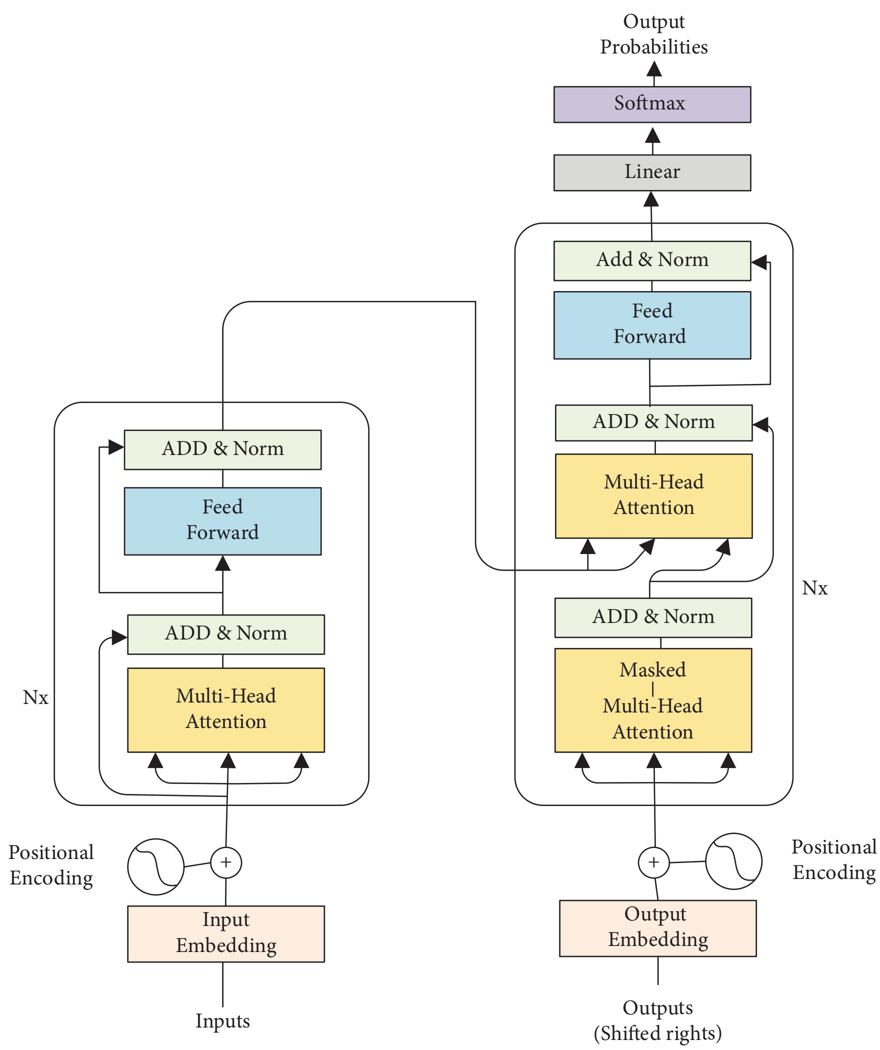

FIGURE 2: Transformer model architecture [50].

a residual connection and layer normalization, which includes two linear transformations and one nonlinear activation function. The calculation principle is shown in the following equation:

$$FFN = \max\left(0, xW_1 + b_1\right)W_2 + b_2. \tag{4}$$

x denotes the input, W_1 and W_2 are the parameters that can be learned in the layers before and after the middle layer, respectively, and b_1 and b_2 are the bias parameters that can be learned in the layers before and after the middle layer, respectively. However, the original transformer, which captures sequence information via absolute position encoding, is not directly applicable to the NER task [38, 51]. The principle of the FLAT [52] is as follows: first, the lattice structure is flattened from a DAG to a flat structure and then combined with the position vector representation idea of the transformer. For each character and vocabulary, head position and tail position encodings are constructed to reconstruct the original lattice structure, where the head and tail of each character are the same and the head and tail of each vocabulary are skipped. As shown in Figure 3, a token is a character or word. The head and tail indicate the position indices of the first and last characters of the token in the original sequence, respectively, and they indicate the position of the token in the lattice. The sequence of characters is first constructed using tokens with the same head and tail, and then skip paths are constructed using the heads and tails of other tags (words).

Since the FLAT structure consists of spans with different lengths, the relative position encodings of spans are used to encode the interactions between spans. For two spans x_i and x_j in a lattice, there are three possible relations: intersection, inclusion, and separation, which are determined by their heads and tails. The relations are calculated by successive transformations of the head and tail information. Let head$[i]$ and tail$[i]$ denote the head and tail positions of the span x_i, respectively. Four relative distances can be used to represent the relationship between x_i and x_j.

$$\begin{aligned} d_{ij}^{(hh)} &= \text{head}[i] - \text{head}[j], \\ d_{ij}^{(ht)} &= \text{head}[i] - \text{tail}[j], \\ d_{ij}^{(th)} &= \text{tail}[i] - \text{head}[j], \\ d_{ij}^{(tt)} &= \text{tail}[i] - \text{tail}[j], \end{aligned} \tag{5}$$

where $d_{ij}^{(hh)}$ denotes the distance between the head of x_i and the tail of x_j, and $d_{ij}^{(ht)}$, $d_{ij}^{(th)}$, and $d_{ij}^{(tt)}$ have similar meanings. To find the optimal algorithm, the final relative position encoding of the span is a nonlinear transformation of the four distances, and we use the rectified linear unit (ReLU) nonlinear activation function, as shown in the following equation:

$$R_{ij} = \text{Re}LU\left(W_{\text{r}}\left(P_{d_{ij}^{(hh)}} \oplus P_{d_{ij}^{(th)}} \oplus P_{d_{ij}^{(ht)}} \oplus P_{d_{ij}^{(tt)}}\right)\right), \tag{6}$$

where W_r is the learnable parameter and $\oplus$ denotes the tandem operator. P_d is the absolute position coding used by the transformer. The main principle is to set a number for each position, and each number corresponds to a vector. Then, by combining the position vector and word vector, a section of position information can be introduced into each word to help the attention mechanism distinguish words in different positions and learn the position information, as shown in the following equation:

$$\begin{aligned} P_d(\text{pos}, 2i) &= \sin\left(\frac{\text{pos}}{10,000^{2i/d_{\text{model}}}}\right), \\ P_d(\text{pos}, 2i+1) &= \cos\left(\frac{\text{pos}}{10,000^{2i/d_{\text{model}}}}\right). \end{aligned} \tag{7}$$

Then, a modified multihead attention mechanism [53] is used to encode the relative span positions, as shown in the following equation:

$$\begin{aligned} A_{i,j}^{*} = W_q^T E_{x_i}^T E_{x_i} W_{k,E} + W_q^T E_{x_i}^T R_{ij} W_{k,R} + u^T E_{x_i} W_{k,E} \\ + v^T R_{ij} W_{k,R}, \end{aligned} \tag{8}$$

where $W_q, W_k, W_v \in R^{d_{\text{model}} \times d_{\text{head}}}$ and $u, v \in R^{d_{\text{head}}}$ are learnable parameters, and A will be replaced by A^* in equation (3).

Finally, the character representation is placed in the output layer, and this layer uses a CRF.

4. Proposal of a TL-FLAT Algorithm Based on FLAT Model Optimization and Transfer Learning

4.1. FLAT Optimization with SGDM and Hyperparameter Adjustment. The chosen model optimization algorithm is directly related to the performance of the final model, and sometimes the poor results obtained may not be caused by features or the model design but rather by the optimization algorithm. The most commonly used optimization models are stochastic gradient descent (SGD) and adaptive moment estimation (Adam). SGD is the most basic and commonly used optimization algorithm for deep learning models, but it possesses problems such as slow descent speed and possible continuous oscillation on both sides of the gully. Therefore, SGD with momentum (SGDM) [54], which introduces first-order momentum on top of SGD to suppress the oscillation of SGD, was introduced. SGDM considers that the gradient descent process can incorporate inertia to increase the descent rate. Kingma et al. [55] describe an adaptive learning rate-based algorithm that stores not only the average of the historical squared gradients of exponential decay as in AdaDelta and root mean square propagation (RMSprop) but also the mean of the exponential decay of historical gradients, and it calculates adaptive learning rates for each parameter.

Therefore, we choose the SGDM and Adam algorithms to optimize the model separately for experiments to derive the optimization algorithm that makes the model perform best.

The selected hyperparameter values [56] play a decisive role in the performance of deep learning models; for example, the learning rate is a very sensitive parameter that determines the step size of the deep learning model weight iterations and is usually determined by using a search method through experiments. The encoder of the model consists of N identical layers that are stacked together, and the number of layers has an impact on the performance of our model. The head_dim is the dimensionality of the

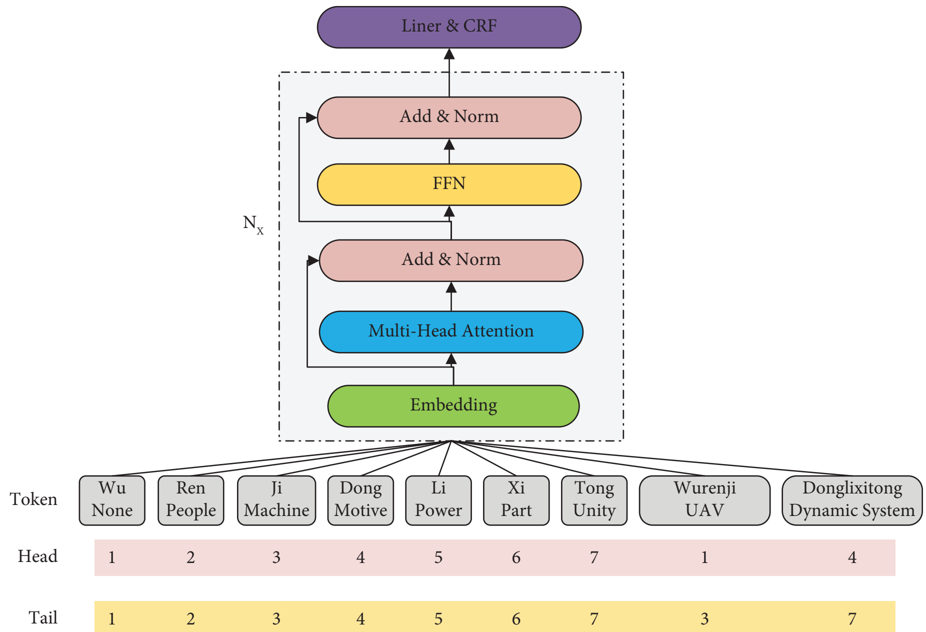

Figure 3: FLAT model architecture.

multihead attention-based hidden layer of the model, which affects the performance and computational power of the model when the number of heads is a fixed value (generally 8). Additionally, we use dropout in the multihead attention layer to reduce the multihead attention weight to reduce overfitting and improve the running speed of the model, and we set the parameter of the dropout function to 0.3–0.5 with reference to the literature [57]. Therefore, to obtain better entity recognition performance from the FLAT model, we conducted exploratory experiments on the optimization algorithm, learning rate (*lr*), number of model encoding layers, and head_dim with the CLUENER2020 dataset.

4.2. Pseudo Code and Complexity Analysis of the Optimized FLAT Model. The pseudocode of the optimized FLAT algorithm is shown in Algorithm 1. The input is a sequence of characters, and the output is a list of entity types corresponding to the characters.

First, two position codes are added to each token/span, indicating the position of the start (head) and end (tail) of the span in the sentence, where head$[i]$ and tail$[i]$ are used to represent the position coordinates of the head and tail of the span, respectively. Equation (7) is used to calculate the relative distances of x_i and x_j from four different angles, and the four distances are stitched together in the ReLU nonlinear transformation. After the four distances are spliced together, a ReLU nonlinear transformation is performed. Equation (8) is used to calculate the position encoding vector. Relative position encoding-based self-attention is performed, followed by Add & Norm, FFN, and Add & Norm of the transformer. Finally, the F1, precision, and recall model evaluation parameters are the output of the liner and CRF layers. Computational complexity refers to the time and space (memory) resources required to execute the model at runtime. It mainly involves two aspects of time complexity and space complexity, and the representation method is commonly used for the asymptotic representation of a large O value. The optimized FLAT model is mainly computed from encoding locations, relative location distances, and attention, with the complexity expressed as $O(N^2)$. Based on the previous theoretical analysis, the numerical results are as shown in Section 5.3.

4.3. TL_FLAT Model Integrating Transfer Learning with the Optimized FLAT. Transfer learning is essentially a machine learning approach that applies knowledge or patterns learned on a domain or task to a different but related domain or problem [58]. Transfer learning does not require the training data to be independent and identically distributed to the test data and does not need to train the model in the target domain from scratch, which can significantly reduce the need for training data and the training time in the target domain. This approach can be applied to solve the problem of insufficient training data in deep learning. The main value is that existing domain data knowledge can be reused without the need for the costly recollection and calibration of large new datasets, and emerging domains can be transferred and applied quickly, demonstrating a time-sensitive advantage.

In deep learning, the model-based transfer learning approach, also called fine-tuning, which is a part of inductive migration, is the most commonly used. According to the literature [59, 60], it is assumed that each parameter w of the task in the deep learning model contains two terms: one is a generic term about the task, and the other is a term specific to the task. In inductive transfer learning, w can be expressed as

```
Input: Character text c_1, c_2, ..., c_m
Output: model evaluation parameters F1, Precision, Recall
Paramerters:
  W_r: the leanable parameter
  N: length of text
emb ⟵ lattic_embedd i ng
for i = 1 ⟶ N do:
  for j = 1 ⟶ N do:
    d_ij^(hh) = head[i] − head[j]
    d_ij^(ht) = head[i] − tail[j]
    d_ij^(th) = tail[i] − head[j]
    d_ij^(tt) = tail[i] − tail[j]
    R_i,j is calculated by equation (6)
    A*_i,j is calculated by equation (8)
  end for
end for
a = Att(A*, V)
b = Add&Norm(emb, a)
c = FFN(b)
d = Add&Norm(b, c)
pred = Linear(d)
return: model evaluation parameters F1, Precision, Recall
```

Algorithm 1: Pseudocode for the optimization model.

$$w_s = w_0 + v_s \text{ and } w_T = w_0 + v_T, \tag{9}$$

where w_s and w_T are the deep learning model parameters for the source and target learning tasks, respectively. w_0 is a generic parameter, and v_s and v_T are specific parameters for the source and target tasks, respectively.

Additionally, the principle of deep migration learning is illustrated in Figure 4 [61]. First, the network is trained in the source domain using a large-scale training dataset. Then, part of the network pretrained for the source domain is transformed into a part of the new network designed for the target domain. Finally, the transferred subnetwork is updated with a fine-tuning strategy to enhance its generalization ability.

5. Results of Ablation and Validation Experiments

5.1. Experimental Environment and Evaluation Criteria

5.1.1. Experimental Environment. The Python programming language was used to code the NER algorithm, which was implemented on a 64-bit Ubuntu 18.04.5 LTS. PyTorch was installed to evaluate the performance of the designed model. All methods were tested on the same computer model with the following main computer configurations: an Intel(R) Core (TM) i7-9700F CPU@ 3.00 GHz, 8.00 GB of RAM, and an NVIDIA RTX2080Ti graphics processing unit (GPU). The FLAT model [52] was used for the ablation experiments, and Lexicon was released by Zhang and Yang [32]. The main parameter settings on the training CLUENER2020 dataset are shown in Table 1.

The parameter settings on the AM_NER dataset were fine-tuned after migration of the best model obtained after training on the CLUENER2020 dataset, and the main parameter settings are shown in Table 2.

5.1.2. Experimental Data. The article used three datasets: CLUENER2020 [62], Weibo-NER [63, 64], and AM_NER (ours). CLUENER2020 is a fine-grained Chinese NER dataset based on the THUCNEWS open-source text classification dataset from Tsinghua University, and the data were selected for fine-grained annotations. The dataset contains 10 entity categories, such as organizations, people, addresses, companies, governments, books, games, movies, jobs, and attractions, with a more balanced distribution among the entity categories. The Weibo-NER data were filtered based on historical data from Sina Weibo from November 2013 to December 2014. The dataset contains four categories: people, organizations, addresses, and geopolitical entities, and each category can be subdivided into specific and generic categories. The AM_NER dataset is the dataset we constructed (see 5.4.1 for details). The characteristics of each dataset are shown in Table 3.

5.1.3. Evaluation Criteria. In this study, we mainly used the precision, recall, and F1 score metrics as the model evaluation parameters. Precision is judged by the prediction results as the proportion of the samples that are correctly predicted as positive cases. The recall is based on the actual sample as the proportion of correctly predicted cases among the actual positive cases. The values of the precision, recall, and F1 score metrics are between 0 and 1.

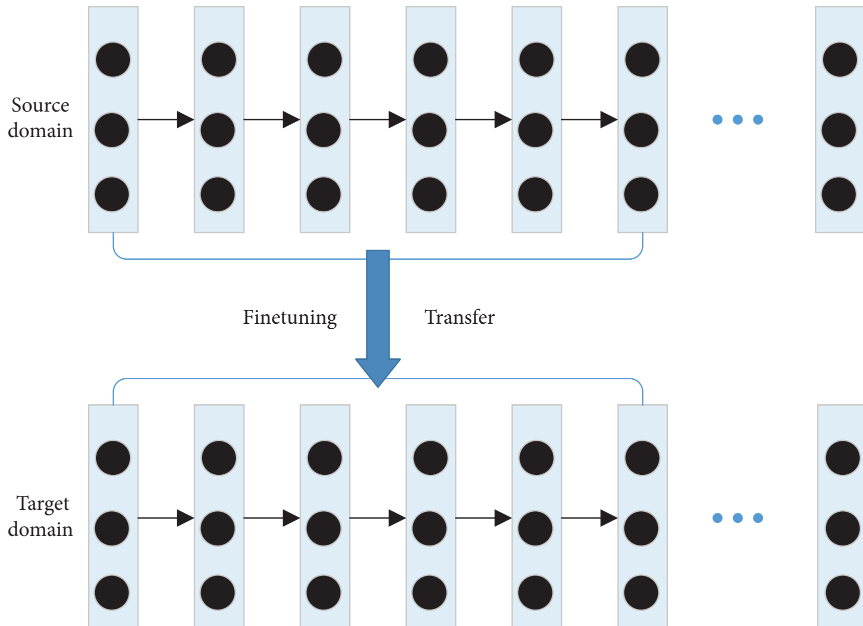

Figure 4: Principle of model-based transfer learning.

Table 1: Main parameter settings on the CLUENER2020 dataset.

Parameters	Values
Batch	10
Epoch	100
Output dropout	0.3
Embed dropout	0.5
Heads	8
Warmup	0.1

Table 2: Main parameter settings on the AM_NER dataset.

Parameters	Values
Batchsize	10
Epoch	100
Warmup	0.2
Learning_rate	$8e-4$
Head_dim	22

Table 3: The characteristics of each dataset.

Dataset	Type	Train	Dev	Test
CLUENER2020	Sentence char	10.748 k	1.343 k	1.343 k
Weibo	Sentence char	1.4 k	0.27 k	0.27 k
AM_NER	Sentence char	1.03 k	0.316 k	0.316 k

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

$$\text{Precision} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \times 100\%, \tag{10}$$

$$F_1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \times 100\%.$$

5.1.4. Experimental Design. Due to the small size of the constructed dataset, we designed the following two main experiments:

(1) Optimization and improvement experiments involving the FLAT model. Based on the FLAT model, the CLUENER2020 dataset was used to derive the best-performing model according to its performance achieved under different parameter settings; then, the resulting optimal model was tested against the baseline model to verify the excellence of the model.

(2) Validation experiments based on transfer learning. The purpose of this experiment was to combine the best model obtained from Experiment 1, migrate this model to the Weibo and AM_NER datasets, and verify whether the model could be well adapted to our UAS text entity recognition task.

5.2. Results of Ablation Experiments Conducted with the FLAT Model. Ablation tests were designed for each parameter according to 3.1, and the results are shown in Figure 5.

As seen from Figure 5, the parameters used by the best-performing version of the model are as follows: (1) Choosing SGD as the optimization function can achieve better results; (2) the model performs better when layer = {1, 2}; the model applies lr = $\{6e-4,\ 8e-4\}$; and the model performs better when head_dim = {22, 20}.

Based on the experimental results in Figure 5, the best-performing model was selected to make the model achieve its relatively optimal performance in terms of the optimizer, *lr*, layer, and head_dim parameters for the combination test, the details of which are shown in Table 4.

The abovementioned model was tested, and the test results are shown in Figure 6.

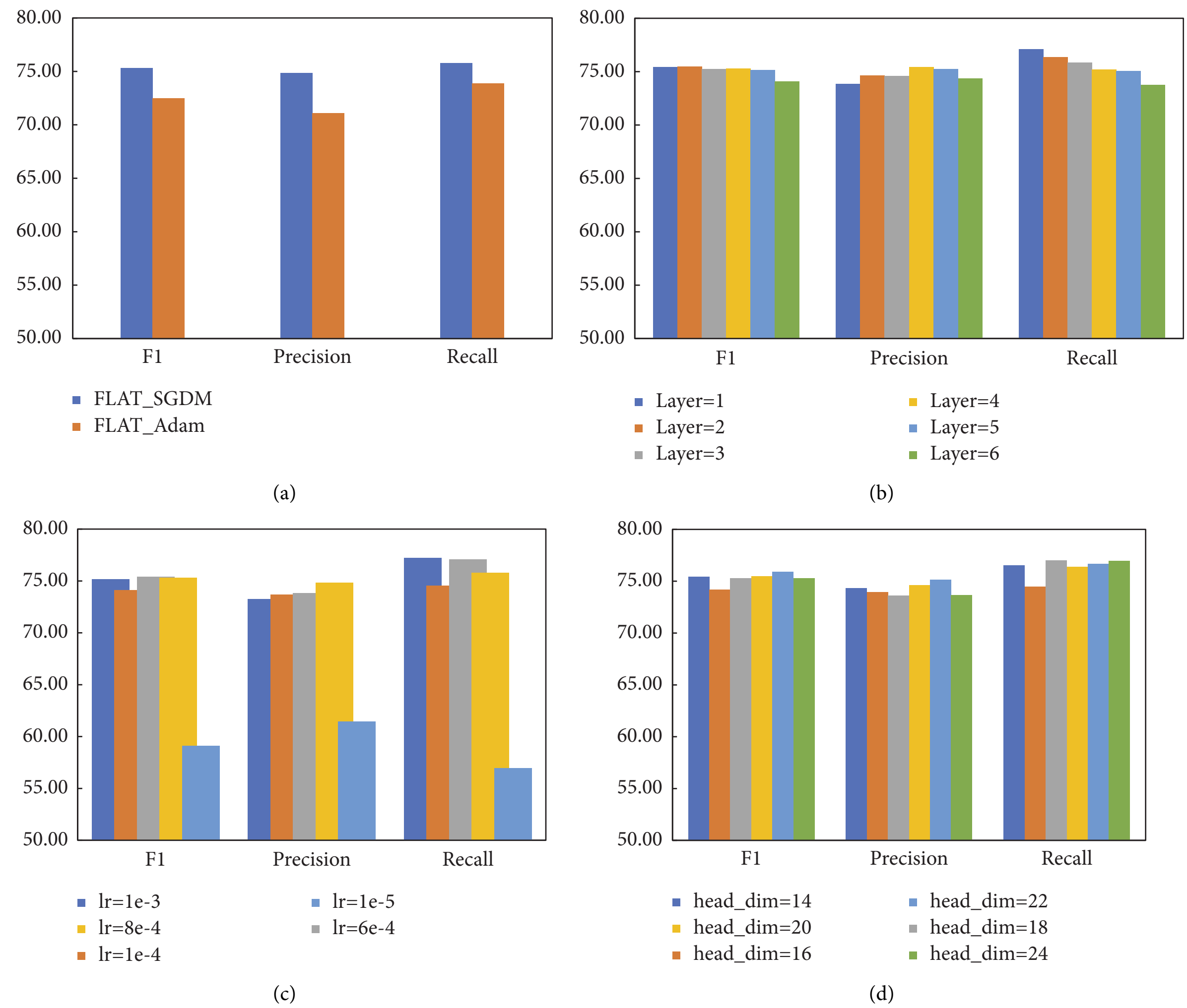

Figure 5: Optimization results obtained on CLUENER2020 (%). (a) Performance of the model with different optimization models; (b) performance of the model with different numbers of layers; (c) performance of the model with different values of *lr*; (d) performance of the model with different head_dim values.

From Figure 6, it can be seen that the F1 score reached 76.26%; the recall value was maximized at 77.36% when the model was set to the fifth group of test parameters; and the precision reached 75.83% when the model was set to the sixth group of test parameters. Furthermore, the precision value was 75.20% in the fifth set of trials. Thus, the validation experiments showed that the model performed best when its parameters were set to the fifth set of parameters: optimizer = SGDM, $\mathrm{lr} = 8e-4$, layer = 2, and head_dim = 22; better performance was achieved when the sixth set of parameters was used: optimizer = SGDM, $\mathrm{lr} = 8e-4$, layer = 2, and head_dim = 20. Therefore, we used the FLAT_5th Experiment version as the base model for the Optimized FLAT model.

Table 4: Statistics regarding the best values for each model parameter.

Experiment number	*lr*	Layer	head_dim	Optimizer
1st experiment	$6e-4$	1	22	SGDM
2nd experiment	$6e-4$	1	20	SGDM
3rd experiment	$8e-4$	1	22	SGDM
4th experiment	$8e-4$	1	20	SGDM
5th experiment	$8e-4$	2	22	SGDM
6th experiment	$8e-4$	2	20	SGDM
7th experiment	$6e-4$	2	22	SGDM
8th experiment	$6e-4$	2	20	SGDM

5.3. Comparison Experiments with the Baseline Models without Transfer Learning

5.3.1. Baseline Models. BiLSTM: This model is a combination of forward LSTM and backward LSTM, which not only inherits most of the characteristics of RNN models and solves the vanishing gradient problem due to the gradient backpropagation process but can also capture semantic dependencies in both directions better than LSTM.

BiLSTM_CRF [9]: This model is also a classic network in NLP that combines both the BiLSTM and CRF approaches to complete the sequence annotation task and achieves good

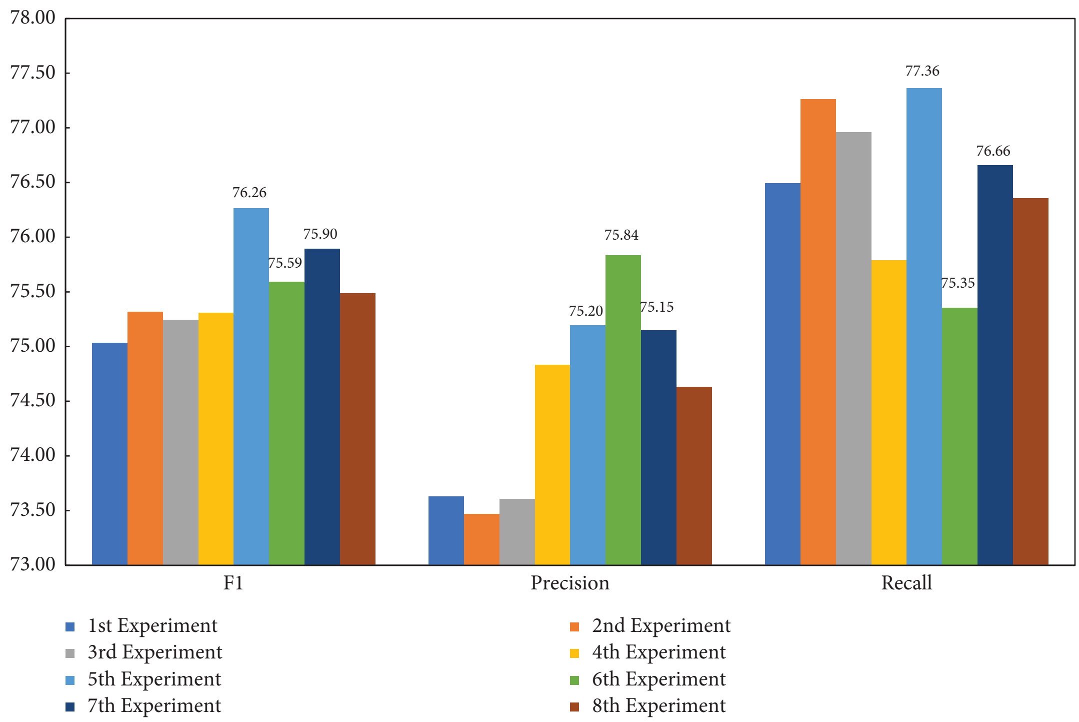

Figure 6: Comprehensive results of the validation experiments.

performance with strong robustness and less dependence on word embedding properties.

Transformer [50, 53]: This model was proposed by Vaswani et al. of the Google team in 2017. The transformer outperforms RNNs and CNNs on machine translation tasks while using only encoder-decoder and attention mechanisms to achieve good results; its biggest advantage is that it can be efficiently parallelized.

Lattice-LSTM [32]: Lattice_LSTM selects the most relevant characters and words by encoding the input character sequences and matching all potential words with the dictionary with the help of gated circular cells. The model effectively avoids the shortcomings of character-based and word-based approaches by utilizing not only word and word sequence information but also without cutoff errors.

Optimized FLAT model (ours): This model can fully model the lattice input directly using a transformer, whose attention mechanism allows characters to interact directly with any potential word, including automatic word matching. Unlike the original FLAT model, we find the optimal model by experimenting with optimization models such as SGD and Adam and different parameters (layers, lr, head_dim); then, the experimentally derived optimal model is compared with the more classic NER method to verify the superiority of the resulting model. Additionally, the parameter of the dropout function is set with reference to the literature [57] as 0.3–0.5, and the multihead attention matrix is changed to a sparse matrix by using dropout in the multihead attention layer to improve the running speed of our model. Other parameter settings are derived from the literature [52].

5.3.2. Experimental Results Compared with Those of the Baseline Model. We used the CLUENER2020 dataset for comparison tests involving BiLSTM, BiLSTM_CRF, Transformer, Lattice_LSTM, and the optimized FLAT model, and the results are shown in Table 5.

As seen from Table 5, the optimized FLAT model achieved the best performance, with F1 score improvements of 11.25%, 5.94%, 4.49%, and 2.88% over the BiLSTM, BiLSTM_CRF, transformer, and Lattice_LSTM models, respectively; precision value improvements of 4.32%, 2.04%, 4.72%, and 1.84%, respectively; and recall value improvements of 14.20%, 7.68%, 15.19%, and 3.97%, respectively, as shown in Figure 7.

As shown in Table 6, compared to the other baseline models, the optimized FLAT model has the highest GPU utilization of 21.61% and the lowest CPU occupancy of 12.2%, with a total time of 4580 s, and the total number of parameters is approximately 9,508,591. Specifically, compared to Lattice_LSTM, the improvement is 91.64%, the GPU usage is 169.12%, and the CPU usage is 48.74% lower. The abovementioned numerical results show that the proposed model can make good use of the parallel computing of GPUs.

5.4. Experimental Validation of TL_FLAT for a UAV Power System Text Dataset

5.4.1. Construction of the AM_NER UAV Power System Dataset. Although some researchers have conducted related studies in the area of manufacturing patents, fewer studies have been conducted on UAV power system texts.

Table 5: Results obtained by different models on CLUENER2020 (%).

Models	F1	Precision	Recall
BiLSTM	68.55	72.08	65.36
BiLSTM_CRF	71.99	73.69	70.38
Transformer	72.99	71.80	74.21
Lattic_LSTM	74.13	73.84	74.41
Optimized FLAT model	76.26	75.20	77.36

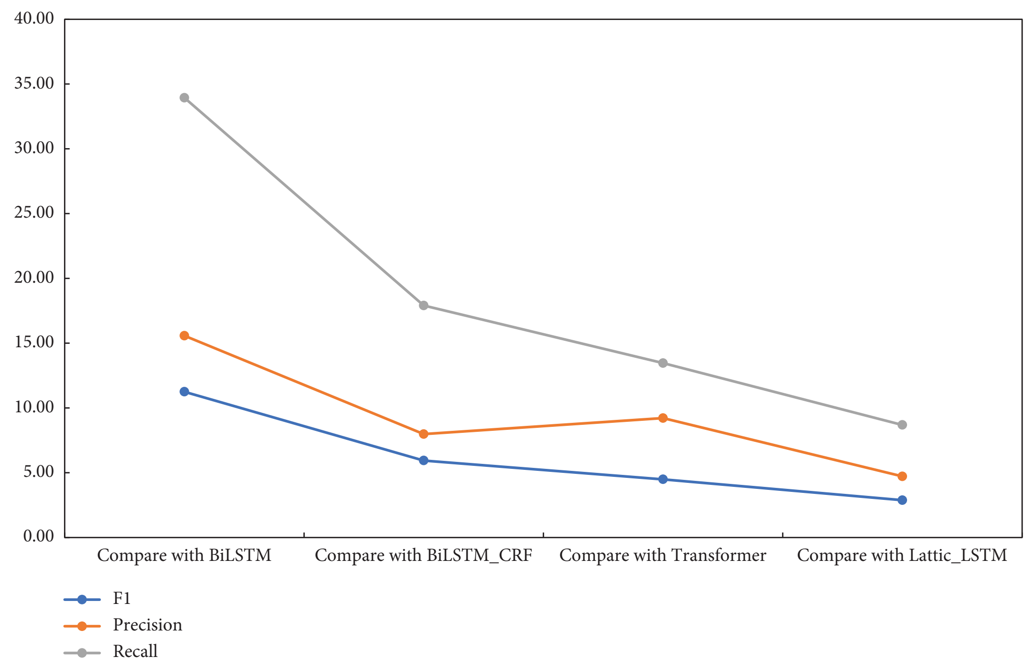

Figure 7: Improvements of the optimized FLAT model values over those of the comparison models (%).

Table 6: Numerical results showing the complexity of the CLUENER2020 dataset.

Models	GPU memory consumption ratio (%)	CPU memory consumption rate (%)	Time consumptions (s)	Number of model parameters
BiLSTM	8.63	17.1	2981	6,368,093
BiLSTM_CRF	9.07	13.4	9539	6,373,493
Transformer	6.80	13.2	2979	9,437,487
Lattice_LSTM	8.03	23.8	54766	11,104,850
Optimized FLAT model	21.61	12.2	4580	9,508,591

Since there is no publicly available dataset, the aviation-manufacturing-NER (AM_NER) dataset used in this paper includes text data crawled by Scrapy from web pages and knowledge networks, mainly product standard documents, product design specifications, patents, technical reports, and so on, as shown in Table 7. Annotation was performed manually on the key features of 7 types of components (parts, components, parameters, materials, functions, structures, and features) using the YEDDA [53] system, a lightweight collaborative text cross-annotation tool. The corpus was in Chinese, and the BIO annotation method was used to obtain high-quality label prediction results and add the corresponding constraints, with each element annotated as "B-Entity," "I-Entity," or "O". "B-X" means that the entity type is X and that the element is located at the beginning of the entity; "I-X" means that the entity type is X and that the element is located in the middle of the entity; "O" means that the entity does not belong to any type.

5.4.2. Experimental Results and Analysis before and after Applying Transfer Learning. According to the experimental results in 5.2–5.3, the proposed TL_FLAT model achieved good NER performance on the CLUENER2020 data. Due to the small size of the dataset, we constructed the trained TL_FLAT model, which was migrated based on the idea of transfer learning after being fine-tuned on the public Weibo dataset and the AM_NER dataset. The before and after results are shown in Table 8.

Table 7: Dataset descriptions of AM_NER.

Dataset	Train	Dev	Test	Classes	Entities
AM_NER	1030	316	316	7	Models, parts, parameters, materials, functions, structures, and features

Table 8: Experimental results of the TL_FLAT model.

	Weibo		AM_NER	
	Before	After	Before	After
F1	60.61	62.18	72.96	76.26
Precision	69.08	70.23	74.90	76.98
Recall	53.98	55.78	71.11	75.56

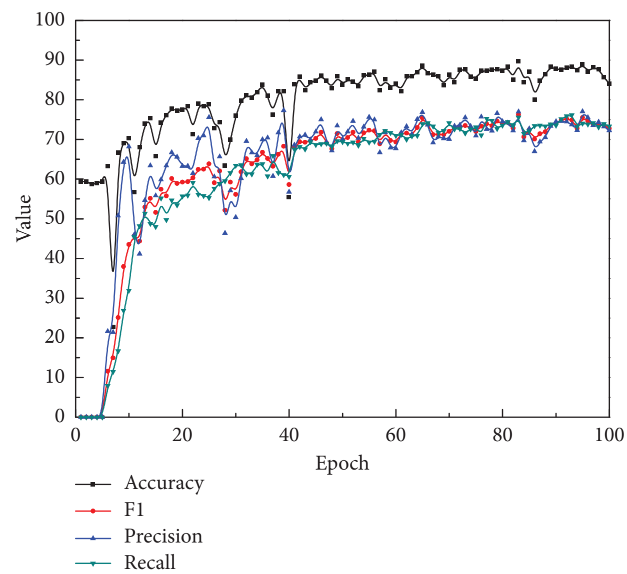

Figure 8: AM_NER results relative to the number of epochs for the TL_FLAT model.

As shown in Table 8, the model performance improved on the Weibo dataset before and after model migration; in particular, the F1 score improved most significantly from 60.61% to 62.18%, which proved the effectiveness of the TL_FLAT model. On the AM_NER dataset, after model migration and fine-tuning, the F1 score improved by 4.53%, the precision improved by 2.78%, and the recall improved by 6.25%. From Figure 8, the accuracy, F1, precision, and recall metrics converge gradually with the increase in the number of epochs. This indicates that the proposed method is applicable to our dataset and can achieve good results.

6. Conclusions

In this paper, the NER model for Chinese text datasets involving UAV power systems was studied using deep learning, and experimental validations showed that the proposed model is applicable. In response to the lack of available text data for UAV power systems, a Chinese assembly text dataset was created and annotated with entities to address this issue for aerospace power systems. The effectiveness of the method proposed in this paper, which can be used to solve the problem of automatic entity extraction from the Chinese text datasets of UAV power systems, lays the foundation for the construction of knowledge graphs for assembling important components. The following are some special conclusions:

(1) It was concluded from 19 sets of ablation tests that the FLAT model performed better when SGD was chosen as the optimization function with layer = {1, 2}, lr = {$6e-4$, $8e-4$}, and head_dim = {22, 20}. Then, the results of the validation tests showed that the goal of improving the performance of the FLAT algorithm could be achieved by using the SGDM optimizer and performing hyperparameter adjustments.

(2) A comparative validation of the optimized model with the main parameters set to optimizer = SGDM, lr = $8e-4$, layer = 2, and head_dim = 22 was performed based on the CLUENER2020 dataset. Compared with the BiLSTM, BiLSTM_CRF, and transformer models, the optimized model performed the best.

(3) A Chinese text dataset for UAV power system assemblies was created and named AM_NER. Based on transfer learning, the TL_FLAT model was further proposed and validated on the dataset: the F1 score reached 76.26%, the precision value reached 76.98%, the recall value reached 75.56%, and the accuracy reached 89.65%. The performance of the model was improved after transfer learning was performed based on the Weibo dataset and AM_NER.

Data Availability

The AM_NER dataset used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Authors' Contributions

Mingming Shen wrote the original draft and reviewed and edited the article. Shaobo Li performed project administration, provided funding acquisition, and wrote, reviewed, and edited the article. Jing Yang wrote, reviewed, and edited the article. Ansi Zhang developed software and wrote, reviewed, and edited the article. Qiuchen He developed software, and reviewed and edited the article. Ruiqiang Pu developed software.

Acknowledgments

The authors wish to express thanks for the support from the National Key R&D Program Project (2020YFB1713300), the National Natural Science Foundation of China (Grant no. 52275480), the Guizhou Provincial Science and Technology Projects (Grant no. GHB [2021]005), Guizhou Province Colleges and Universities Integrated Key Platform Project (Grant no. KY [2020]005), and Guizhou Higher Education Talent Cultivation Base Project (QianjiuheKY word [2020] 009).

References

[1] L. Jianhua, S. Qingchao, C. Hui et al., "The state-of-the-art, connotation and developing trends of the products assembly technology," *Journal of Mechanical Engineering*, vol. 54, pp. 1–28, 2018.

[2] H. C. G. L. Wang, "Research progress and perspective of robotic equipment applied in aviation assembly," *Acta Aeronautica et Astronautica Sinica*, vol. 43, 2022.

[3] H. Li, M. Hagiwara, Q. Li, and H. Ji, "Comparison of the impact of word segmentation on name tagging for Chinese and Japanese," in *Proceedings of the 9th Int. Conferences. Lang. Resour. Eval. Lr*, pp. 2532–2536, Reykjavik, Iceland, May 2014.

[4] P. Cao, Y. Chen, K. Liu, J. Zhao, and S. Liu, "Adversarial transfer learning for Chinese named entity recognition with self-attention mechanism," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 182–192, Brussels, Belgium, October 2018.

[5] F. Wu, J. Liu, C. Wu, Y. Huang, and X. Xie, "Neural Chinese named entity recognition via CNN-LSTM-CRF and joint training with word segmentation," in *Proceedings of the World Wide Web Conference on WWW '19*, pp. 3342–3348, WWW, CA, San Francisco, USA, May 2019.

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

[7] C. Wang, Hu Zhang, X. Wang, and T. Chen, "User intention recognition based on sensitive word rules and character-level RCNN mode," *Computer Appl. Softw.*, vol. 37, pp. 160–165, 2020.

[8] X. Xiang and Z. H. Shi Xiaodong, "User intention recognition based on sensitive word rules and character-level RCNN model," *Comput. Appl. Softw.*, vol. 37, pp. 160–165, 2020.

[9] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF Models for Sequence Tagging," 2015, https://arxiv.org/abs/1508.01991.

[10] T. S. Majtner, S. Yildirim, and J. Y. H. Yayilgan, "Combining deep learning and hand-crafted features for skin lesion classification," in *Proceedings of the 2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA)*, Oulu, Finland, December 2016.

[11] Y. Cui, W. Che, T. Liu, B. Qin, and Z. Yang, "Pre-training with whole word masking for Chinese BERT," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3504–3514, 2021.

[12] Y. Ren, Y. H. Yu Hong, and L. Xiao, "Recognition of quantitative indicator of fishery standard using attention mechanism and the BERT+BiLSTM+CRF model," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 37, pp. 135–141, 2021.

[13] D. Sui, Z. Tian, Y. Chen, K. Liu, and J. Zhao, "A large-scale Chinese multimodal NER dataset with speech clues," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 2807–2818, Bangkok, Thailand, November 2021.

[14] F. Zhang, L. Ma, J. Wang, and J. Cheng, "An MRC and adaptive positive-unlabeled learning framework for incompletely labeled named entity recognition," *International Journal of Intelligent Systems*, vol. 37, no. 11, pp. 9580–9597, 2022.

[15] J. H. Kim and P. C. Woodl, "A rule-based named entity recognition system for speech input," in *Proceedings of the 6 th International Conference on Spoken Language Processing*, pp. 528–531, Beijing China, July 2000.

[16] R. Rq, Q. Vwhpv, R. Rq, O. H. Dqgud, and D. D. O. Qhud, "Named entity recognition over electronic health records through a combined dictionary-based approach," *Procedia Computer Science*, vol. 100, pp. 55–61, 2016.

[17] K. Humphreys, R. Gaizauskas, S. Azzam, and F. Zhang, "Description of the LaSIE- II systemas used for MUC- 7," in *Proceedings of the 7th MessageUnderstanding Conference*, pp. 1–20, Fairfax, Virginia, April 1998.

[18] G. R. Krupka and H. K. Xiao, "IsoQuest inc.: description of the NetOwl $^{\mathrm{TM}}$ extractor system as used for MUC- 7," in *Proceedings of the Proceedings of the 7th Message Understanding Conference*, pp. 1–10, Fairfax, Virginia, June 1998.

[19] W. J. Black, F. Rinaldi, and M. D. Zhao, "FACILE: description of the NE system used for MUC- 7," in *Proceedings of the Proceedings of the 7th Message Understanding Conference*, pp. 1–10, Fairfax, Virginia, November 1998.

[20] C. Aone, L. Halverson, T. Hampton, and T. Chen, "SRA: Description of the IE2 system used for MUC-7," in *Proceedings of the Proceedings of the 7th Message Understanding Conference*, pp. 1–14, Beijing China, June 1998.

[21] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticæ Investigationes. International Journal of Linguistics and Language Resources*, vol. 30, no. 1, pp. 3–26, 2007.

[22] M. S. Y. Collins, "Unsupervised models for named entity classification," in *Proceedings of the Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 1–11, NJ Florham Park, June 1999.

[23] D. Nadeau, P. D. Turney, and M. S. Chen, "Unsupervised named-entity recognition: generating gazetteers and resolving ambiguity," in *Proceedings of the LNCS 4013: Proceedings of the 19th Conference of the Canadian Society for Computational Studies of Intelligence*, pp. 266–277, Québec, Quebec, Canada, April 2006.

[24] S. R. Eddy, "Hidden Markov models," *Current Opinion in Structural Biology*, vol. 6, no. 3, pp. 361–365, 1996.

[25] J. N. Kapur, *Maximum Entropy Models in Science and Engineering Livros*, p. 635, Jagat Narain Kapur, Delhi, 1989.

[26] S. Suthaharan, "Support vector machine," *Nature biotechnology*, vol. 24, 2016.

[27] J. Lafferty, A. Mccallum, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data Abstract," *Penn Engineering*, vol. 19, pp. 282–289, 2001.

[28] V. Yadav and S. Bethard, "A survey on recent advances in named entity recognition from deep learning models," pp. 2145–2158, 2018, https://arxiv.org/abs/1910.11470.

[29] A. Colert, J. Weston, M. Karlen et al., "Natural Language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[30] M. E. Peters, M. Neumann, M. Iyyer et al., "Deep contextualized word representations," in *Proceedings of the NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, pp. 2227–2237, Beijing China, October 2018.

[31] X. Ma and E. Hovy, "End-to-End sequence labeling via Bidirectional LSTM-CNNs-CRF," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 Long Papers*, pp. 1064–1074, Berlin, Germany, August 2016.

[32] Y. Zhang and J. Yang, "Chinese NER using lattice LSTM," in *Proceedings of the ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, pp. 1554–1564, Melbourne, Australia, July 2018.

[33] W. Liu, T. Xu, Q. Xu, J. Song, and Y. Zu, "An encoding strategy based word-character LSTM for Chinese ner," in *Proceedings of the NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, pp. 2379–2389, Minneapolis, Minnesota, June 2019.

[34] T. Gui, R. Ma, Q. Zhang, L. Zhao, Y. G. Jiang, and X. Huang, "CNN-based Chinese NER with lexicon rethinking," in *Proceedings of the IJCAI International Joint Conference on Artificial Intelligence*, pp. 4982–4988, Macao, China, May 2019.

[35] T. Gui, Y. Zou, Q. Zhang et al., "A lexicon-based graph neural network for Chinese ner," in *Proceedings of the EMNLP-IJCNLP 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing*, pp. 1040–1050, Hong Kong, China, November 2019.

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

[37] R. Ding, P. Xie, X. Zhang, W. Lu, L. Li, and L. Si, "A neural multi-digraph model for Chinese NER with gazetteers," in *Proceedings of the ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1462–1467, Florence (Italy), August 2020.

[38] H. Yan, B. Deng, X. Li, and X. T. E. N. E. R. Qiu, "Adapting Transformer Encoder for Named Entity Recognition," 2019, https://arxiv.org/abs/1911.04474.

[39] R. Ma, M. Peng, Q. Zhang, Z. Wei, and X. Huang, "Simplify the Usage of Lexicon in Chinese NER," pp. 5951–5960, 2020, https://arxiv.org/abs/1908.05969.

[40] B. Jin, L. Cruz, and N. Goncalves, "Deep facial diagnosis: deep transfer learning from face recognition to facial diagnosis," *IEEE Access*, vol. 8, pp. 123649–123661, 2020.

[41] Y. Shen, H. Yun, Z. C. Lipton, Y. Kronrod, and A. Anandkumar, "Deep active learning for named entity recognition," in *Proceedings of the 6th International Conference on Learning Representations, ICLR 2018 Conference Track Proceedings*, Vancouver, BC, June 2018.

[42] L. Gligic, A. Kormilitzin, P. Goldberg, and A. Nevado-Holgado, "Named entity recognition in electronic health records using transfer learning bootstrapped neural networks," *Neural Networks*, vol. 121, pp. 132–139, 2020.

[43] V. C. Tran, N. T. Nguyen, H. Fujita, D. T. Hoang, and D. Hwang, "A combination of active learning and self-learning for named entity recognition on twitter using conditional random fields," *Knowledge-Based Systems*, vol. 132, pp. 179–187, 2017.

[44] H. K. Kung, C. M. Hsieh, C. Y. Ho, Y. C. Tsai, H. Y. Chan, and M. H. Tsai, "Data-augmented hybrid named entity recognition for disaster management by transfer learning," *Applied Sciences*, vol. 10, no. 12, p. 4234, 2020.

[45] M. Zhang, G. Geng, and J. Chen, "Semi-supervised bidirectional long short-term memory and conditional random fields model for named-entity recognition using embeddings from language models representations," *Entropy*, vol. 22, no. 2, p. 252, 2020.

[46] L. Grieco, G. Boggia, G. Piro, Y. Jararweh, and C. Campolo, "Ad-hoc, mobile, and wireless networks," in *Proceedings of the 19th International Conference on Ad-Hoc Networks and Wireless, ADHOC-NOW 2020*, Springer Nature, Bari, Italy, October 2020.

[47] G. Iacovelli, A. Coluccia, and L. A. Grieco, "Channel gain lower bound for IRS-assisted UAV-aided communications," *IEEE Communications Letters*, vol. 25, no. 12, pp. 3805–3809, 2021.

[48] G. Iacovelli and L. A. Grieco, "Drone swarm as mobile relaying system: a hybrid optimization approach," *IEEE*

Transactions on Vehicular Technology, vol. 70, no. 11, pp. 12272–12277, 2021.
[49] B. Zhou, J. Bao, Z. Chen, and Y. Liu, "KGAssembly: knowledge graph-driven assembly process generation and evaluation for complex components," *International Journal of Computer Integrated Manufacturing*, vol. 35, no. 10-11, pp. 1151–1171, 2021.
[50] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 103, pp. 5999–6009, 2017.
[51] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," 2021, https://arxiv.org/abs/2106.04554.
[52] X. Li, H. Yan, X. Qiu, and X. Huang, "FLAT: Chinese NER Using Flat-Lattice Transformer," pp. 6836–6842, 2020, https://arxiv.org/abs/2004.11795.
[53] Z. Dai, Z. Yang, Y. Yang et al., "Attentive Language models beyond a fixed-length context," in *Proceedings of the ACL 2019 - 57th Annu. Meet. Assoc. Comput. Linguist. Proc. Conferences*, pp. 2978–2988, Florence, Italy, June 2020.
[54] I. Tolstikhin, O. Bousquet, B. Schölkopf et al., "An overview of gradient descent optimization algorithms," *Lecture Notes in Computer Science*, vol. 12, pp. 1–9, Article ID 11046, 2018.
[55] D. P. Kingma, J. Ba, and L. Adam, "A method for stochastic optimization," in *Proceedings of the 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc*, San Diego, CA, USA, May 2015.
[56] K. I. L. Lee and J. U. N. H. O. Yim, "Hyperparameter Optimization with Neural Network Pruning," pp. 1–17, 2020, https://arxiv.org/abs/2205.08695.
[57] M. Shen, J. Yang, S. Li, A. Zhang, and Q. Bai, "Nonlinear hyperparameter optimization of a neural network in image processing for micromachines," *Micromachines*, vol. 12, p. 1504, 2021.
[58] J. Howard and S. Ruder, "Universal Language model fine-tuning for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 328–339, Melbourne, Australia, June 2018.
[59] S. Panigrahi, A. Nanda, and T. Swarnkar, "A survey on transfer learning," *Smart Innovation, Systems and Technologies*, vol. 194, pp. 781–789, 2021.
[60] T. Evgeniou and M. Pontil, "Regularized multi-task learning," in *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD*, pp. 109–117, Seattle WA USA, August 2004.
[61] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," *Artificial Neural Networks and Machine Learning – ICANN 2018*, vol. 11141, pp. 270–279, 2018.
[62] L. Xu, Q. Dong, C. Yu et al., "CLUENER2020: fine-grained name entity recognition for Chinese," 2020, https://arxiv.org/abs/2001.04351.
[63] N. Peng and M. Dredze, "Named entity recognition for Chinese social media with jointly trained embeddings," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 548–554, Lisbon, Portugal, September 2015.
[64] H. He and X. Sun, "F-score driven max margin neural network for named entity recognition in Chinese social media," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pp. 713–718, Valencia, Spain, April 2017.