IJLAI Transactions on Science and Engineering (ISSN: 3006-3418) 2024, vol. 2, no. 3, pp: 1–8

English-Chinese Neural Machine Translation Based on Self-organizing Mapping Neural Network and Deep Feature Matching

Shu Ma[1]

College of International Business, Shenyang Normal University
No. 253 Huanghe North Street, Shenyang 110034, China
mashuetd@163.com
Corresponding author: Shu Ma

Abstract. The traditional Chinese-English translation model tends to translate some source words repeatedly, while mistakenly ignoring some words. Therefore, we propose a novel English-Chinese neural machine translation based on self-organizing mapping neural network and deep feature matching. In this model, word vector, two-way LSTM, 2D neural network and other deep learning models are used to extract the semantic matching features of question-answer pairs. Self-organizing mapping (SOM) is used to classify and identify the sentence feature. The attention mechanism-based neural machine translation model is taken as the baseline system. The experimental results show that this framework significantly improves the adequacy of English-Chinese machine translation and achieves better results than the traditional attention mechanism-based English-Chinese machine translation model.

Keywords: Chinese-English translation model, Self-organizing mapping neural network, Deep feature matching, Deep learning.

1. Introduction

Since Cho et al. [10] proposed neural machine translation based on encoder-decoder architecture in 2014, this framework had achieved rapid development in the field of machine translation, and had achieved significant improvement in translation quality compared with statistical machine translation. The encoder and decoder are both Recurrent neural networks (RNN). The encoder RNN encodes the input source language sentence into a vector representation with fixed dimension, and the decoder RNN decodes the representation into the target language sentence. The conditional probability of the input sentence is maximized through joint training of the encoder and decoder [2,3]. On this basis, Zhang et al. [4] proposed a method to add attention mechanism to the end-to-end machine translation model, which effectively improved the quality of neural machine translation. Liang et al. [5] studied two simple and effective attention mechanisms, one focusing on all source words all the time and the other focusing on only a subset of source words. The validity of global attention and local attention in English-German translation is proved respectively. In order to alleviate the problems of gradient disappearance and Long distance dependence in machine translation, the researchers proposed a machine translation system based on the sentence-level Long Short Time Memory (LSTM) model [6]. Although the traditional end-to-end neural machine translation framework has achieved remarkable results, there is still a major disadvantage, which is that the framework tends to translate some source words repeatedly, while mistakenly ignoring some words. This leads to the serious phenomenon of over-translation and missing translation. This is because the traditional encoder-decoder architecture does not have a mechanism to ensure that the information at the source end is fully translated to the target end.

In response to the above problems, Sung et al. [7] proposed a neural machine translation coverage model in 2019, which effectively alleviated the phenomenon of overtranslation and missing translation in neural machine translation with attention mechanism. The principle of this method was to combine the coverage vector and the attention vector in the neural machine translation model based on the attention mechanism. That is, in the process of decoding, the decoder reduced the ”attention” of those words that had been translated from the source language, which reduced the possibility of it being translated again. The immediate solution was to combine the coverage vector with attention, so that the coverage vector could regulate attention and thus play the role of ”correcting” attention. However, this method can only be implemented on neural machine translation models with attention mechanism, and is not applicable to all end-to-end neural machine translation models. Therefore, Qing et al. [8] proposed a method to add a reconfigurator on the basis of the traditional encoder-decoder framework. The aim was to ensure that the information from the source side was fully converted to the target side by adding a mechanism to the traditional encoder-decoder framework, which was applicable to all encoder decoder frameworks. The experimental results showed that the adequacy of translation was closely related to the reconstruction apparatus.

In this paper, an encoder-decoder reconstruction framework for E-C neural machine translation is proposed, and the distributed representation of word vectors is used to convert all data sets used in the experiment into vector form. The traditional encoder-decoder framework with attention mechanism is used as the baseline system. The experimental results show that this method can effectively alleviate the phenomenon of overtranslation and missing translation in E-C neural machine translation.

2. Text Preprocessing

In the process of text preprocessing, the suspended word is first removed and all letters are changed to lower case, and then each sentence is analyzed to obtain the syntax tree of all sentences. We use the NLTK[3] toolkit for text segmentation and the Stanford PCFG grammar parser for syntactic parsing.

2.1. Traditional Natural Language Features

For the answer selection task, we first use some traditional natural language processing models for lexical analysis and syntactic analysis, and extract the features about word and sentence structure [9-14].

A. Tf-idf cosine similarity

Tf-idf (term frequence-inverse document frequency) [15] is a commonly used word weighting technique, which can evaluate the importance of words in a corpus and is widely used in information retrieval systems. Tf-idf weighting technology is mainly considered from two aspects: term frequency (tf) and inverse document frequency (idf).

Word frequency is defined as the number of times a word w appears in document D, as shown in equation (1).

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}}. \tag{1}$$

Where $tf_{i,j}$ represents the word frequency value of the word w_i in document d_j. The numerator $n_{i,j}$ is the number of times the word w_i appears in document d_j. The denominator represents the total number of occurrences of all words in document d_j, and is primarily used for normalization to prevent word frequency values from being biased toward longer documents.

Whereas word frequency is used to measure the importance of words in a single document, inverse document frequency is used to measure the prevalence of words in multiple documents. If a word appears in too many documents, it is more common and less important, as shown in equation (2).

$$idf_i = \log \frac{|D|}{|d_j : w_i \in d_j| + 1}. \tag{2}$$

Where the numerator $|D|$ is the number of all documents in the document library, and the denominator $|d_j : w_i \in d_j|$ is the number of all documents containing the word w_i. The denominator is added by 1 to prevent the divisor from being 0.

It can be seen that the importance of a word in a document is directly proportional to the word frequency and the inverse document frequency respectively. Therefore, the $Tf - idf$ weight of the word w_i in document d_j can be defined as:

$$tfidf_{i,j} = tf_{i,j} \times idf_i. \tag{3}$$

We treat each text as a document d_j, and all text collections as document collections D. The Tf-idf model is used to calculate the Tf-idf weight for each word in the text. Therefore, each text pair (q, a) can be represented by a One-Hot weight vector (v_q, v_a). By calculating the cosine $cos(v_q, v_a)$ between the two vectors, the similarity between the text pairs about the coverage of important words can be obtained, which reflects the similarity between the texts to a certain extent.

B. Longest common subsequence

If a sequence S is a subsequence of two sequences (q, a) and is the longest of all sequences that meet the conditions, S is called the longest common subsequence (LCS) of q and a [16,17]. The longest common subsequence is often used to measure the similarity between two strings. In this paper, the longest common subsequence is applied to measure the lexical similarity between two text sequences, that is, a sentence is regarded as a sequence whose elements are all the words in each sentence. The longest common subsequence can be solved using dynamic programming (DP) methods.

In order to avoid bias of LCS values towards longer texts, the model in this paper also uses the maximum length of the two texts after calculating the LCS of the two texts. The final LCS similarity is shown in equation (4).

$$LCS_{score}(q,a) = \frac{LCS(q,a)}{max(|q| \cdot |a|)}. \tag{4}$$

2.2. Deep Matching Feature

Traditional natural language processing models mainly model the similarity relationship between texts from the perspective of word and sentence structure, and cannot fully consider the semantic connection between two sentences. Therefore, in addition to the above traditional models, we also use word vector technology and depth matching model to model the semantic connections between text pairs and extract the semantic features between sentences [18,19].

A. Word vector cosine similarity

Word emdedding can map each word as a vector in a continuous space, and the semantic similarity between words can be represented by the cosine similarity between word vectors. In this paper, Word2Vec [20] developed by Google was trained on the Qatar Living dataset to obtain the word vector of each word, and the vector dimension was set to 200.

After each word vector is obtained, the next step is to use the word vector to calculate the similarity between sentences. Here, this study first adopts a simple Bag of Word model to get the vector representation of each sentence, that is, to calculate the average value of each word vector in the sentence, as shown in equation (5).

$$s = \frac{1}{|s|}\sum_{i-1}^{|s|} s_i. \tag{5}$$

Where s is the vector representation of the sentence. s_i is the vector representation of each word in the sentence. After the similarity of each sentence is calculated, the cosine similarity can be used to measure the semantic similarity between two sentences. For each text pair (q,a), the calculation method of the cosine similarity of the word vector is shown in equation (6).

$$emb_{score}(q,a) = \frac{q_{emb} \cdot a_{emb}}{|q_{emb}| \times |a_{emb}|}. \tag{6}$$

This method is simple and effective, with high computational efficiency, but it can not deal with challenges such as the structure of phrase matching and the hierarchy of text matching. Therefore, we also use two-way LSTM model and 2D neural network model to model text matching.

B. Encoder-decoder architecture

Since the encoder-decoder architecture was proposed, it has been widely concerned by researchers and has become the basic model of neural machine translation [21]. Its structure is shown in Figure 1.

This article uses a bidirectional LSTM as an encoder and an unidirectional LSTM as a decoder. In this model, the encoder reads the input sentence $x = (x_1, x_2, \cdots, x_n)$ and encodes it as a vector with fixed dimension. First, the forward LSTM encodes the input language as the forward hidden layer state $\overrightarrow{h}_i$, and then the reverse LSTM encodes the input language as the backward hidden layer state $\overleftarrow{h}_i$. Then the overall hidden layer state h_i is obtained by connecting $\overrightarrow{h}_i$ and $\overleftarrow{h}_i$. Finally, the hidden state of the input sequence is transformed into context vector c by nonlinear transformation q.

Among them, the calculation of the forward hidden layer state $\overrightarrow{h}_i$ is shown in equation (7), and the backward hidden layer state $\overleftarrow{h}_i$ and context vector c are obtained in the same way:

$$\overrightarrow{h}_i = f(x_i, h_{i-1}). \tag{7}$$

$$c = q(h_1, h_2, \cdots, h_n). \tag{8}$$

Where f and q are nonlinear functions.

Given the context vector c and the first $i-1$ already generated word $y_1, y_2, \cdots, y_{i-1}$, the decoder is trained to predict the next word y_i. The probability formula of generating word y_i is expressed as:

$$p(y) = \prod_{i=1}^{n} p(y_i, y_2, \cdots, y_{i-1}, c). \tag{9}$$

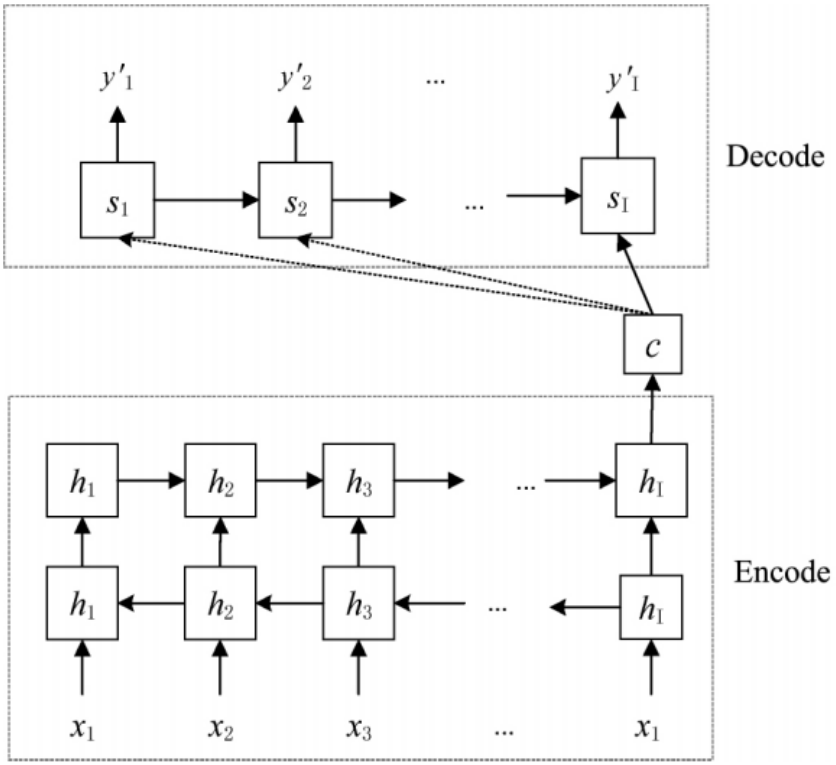

Fig. 1. One kernel

$$p(y_i, y_2, \cdots, y_{i-1}, c) = g(y_{i-1}, S_i, c). \tag{10}$$

Where s_i represents the hidden layer state at the moment of decoder i, expressed as:

$$s_i = f(s_{i-1}, y_{i-1}, c). \tag{11}$$

3. SOM Feature Selection

SOM is an unsupervised, self-learning neural network developed by Finnish scholar KOHONEN T to simulate the autonomous learning ability of human brain neurons. It can find similar parts in the huge sample space and map them to the low-dimensional space, form clusters and remember this logical relationship, and the trained network can automatically classify the input samples.

SOM has the feature of mapping high-dimensional data to low-dimensional space and keeping the original data topology unchanged. The network consists of input layer and output layer. Unlike other neural networks, SOM does not contain hidden layers, its structure is simpler, the algorithm is less complex and effective. The input layer and the output layer are directly connected with the value vectors, and each neuron in the output layer is connected with all the input vectors to ensure that the input vectors can be mapped well on the output layer.

The SOM network structure is shown in Figure 2. In Figure 2, the lower layer is the input layer and the upper layer is the output layer. After the weight connection, the input vectors of the same type can be automatically clustered in the output layer. In Figure 2, different colors of neurons in the output layer represent different data categories.

SOM neural network belongs to competitive learning neural network. After the training starts, the input vector will calculate the Euclide distance between all the neurons in the output layer connected with it, and the distance is taken as the basis for the victory of the neuron. The neuron in the output layer with the smallest distance from the input vector is the winning neuron, and it has the right to update the weight with all neurons in the neighborhood to further reduce the distance between it and the input vector. Different input vectors may correspond to different winning neurons in the output layer, so neurons at different locations in the output layer may be activated.

The algorithm steps are as follows.

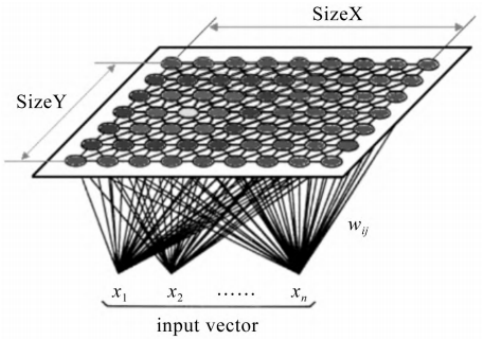

Fig. 2. One kernel

1. Step 1. The number of neurons in output layer, initial weight vector, learning rate, neighborhood size and other parameters are set to initialize the network.
2. Step 2. Let the input vector $x(t)$ in the data sample set x be n-dimensional, and t be the number of iterations. Each iteration randomly extracts $x(t)$ from x.
3. Step 3. Calculate the weight of each neuron in the output layer and the Euclidean distance of the input vector, and find the winning neuron corresponding to the input vector $x(t)$.

$$c = argmin_i||x(t) - m_i(t)||. \tag{12}$$

   Where c is the winning neuron. $m_i(t)$ is the weight vector.
4. Step 4. The weights of the winning neuron c and other nodes in its neighborhood are updated synchronously.

$$m_i(t+1) = m_i(t) + h_{ci}(t)[x(t) - m_i(t)]. \tag{13}$$

   Where $h_{ci}(t)$ is a neighborhood function, and different nodes in the neighborhood update their weights differently according to their distance.
5. Step 5. Update network parameters.

$$h_{ci}(t) = \alpha(t)exp(-sqdist(c,i)/2\sigma^2(t)). \tag{14}$$

   Where $\alpha(t)$ is the learning rate function. $\sigma(t)$ is a neighborhood size function. Both are monotonically decreasing functions of time t (that is, the number of iterations). The value of $\alpha(t)$ is generally between (0,1). $sqdist(c,i)$ is the square of the geometric distance between the winning neuron c and the node i in the neighborhood.
   With the continuous iteration, the neighborhood function will be updated constantly to make the network converge gradually.
6. Step 6. Check whether the results meet the established requirements, if not, continue to iterate; If the requirements are met, the cycle ends and the training is complete.

4. U-Matrix Visualization

The unified distance matrix [22] (U-Matrix) can display SOM classification results. It takes distance as a metric and displays results in a two-dimensional network structure. Each grid point in the U-Matrix represents the distance between that neuron and the surrounding neuron.

According to the nature of clustering, clustering is to use the similarity of data to aggregate and separate chaotic original data sets, so as to divide certain categories. U-Matrix is the method of distance measurement to visualize SOM classification results. U-Matrix can store the distance between each neuron in the SOM and its neighboring neurons, and the size of the value indicates the distance.

In order to represent this distance measurement more vividly, the author introduced RGB colors in UMatrix, and assigned different data values to different colors. The dark color means the distance is far and the value is large, the light color means the distance is close and the value is small, and the size of the distance can be judged according to the depth of the color. Data with high similarity (that is, small distance) belongs to the same class of data, while data with small similarity (that is, large distance) belongs to different class of data.

The U-Matrix actually only serves to store the relative distance of neurons in the SOM, and does not change the position of neurons. Therefore, this approach not only visualizes SOM, but also preserves the topology of the original data.

5. Experiment and Analysis

In model training, all English-Chinese parallel data sets (represented by M) are used, and the encoder-decoder parameter $P(y|x;\theta)$ and the reconstructor parameter $R(x|s;\gamma)$ are trained simultaneously, whose objective function is expressed as:

$$J(\theta,\gamma) = argmax_{(\theta,\gamma 0} \sum_{m=1}^{M} \log P(y^m|x^m;\theta) + \lambda \log R(x^m|s^m;\gamma). \tag{15}$$

The objective function consists of two parts, the likelihood function is used to measure the fluency of the translation, and the reconfigurator is used to measure the adequacy of the translation. λ is the hyperparameter of balancing likelihood and reconstruction, and λ is chosen as 1 in this paper. BLEU value is used to evaluate the translation quality of this paper.

The experimental data consisted of 67273 sentences of English-Chinese parallel corpus. Firstly, the English-Chinese parallel corpus was preprocessed and an English-Chinese alignment dictionary with a size of 30,000 was generated respectively. Then, the corpus was divided by using the automatic method, and the partitioning results were shown in Table 1.

Table 1. Dataset

Data	Number
Training set	60000
Validation set	4000
Test set	3300
Total	67300

The training of English-Chinese neural machine translation model based on encoder-decoder reconstruction framework requires pre-training of encoder-decoder framework based on attention mechanism. In this paper, a neural machine translation model, NMT-Chainer, was used to pre-train 67300 pre-processed English-Chinese parallel corpus through this framework. The BLEU value of the model obtained in the first 30 rounds of testing was shown in Table 2.

Table 2. BLEU value of the pre-trained model

epoch	value
3	2.73
6	6.69
9	10.48
12	14.21
15	16.72
18	18.89
21	20.26
24	21.17
27	22.64
30	23.18

An attention-mechanism-based neural machine translation model (Baseline-NMT) is used as a Baseline system to train an encoder-decoder Reconstructor without pre-training. In addition, the model with the highest BLEU value is selected from the attention-mechanism-based neural machine translation model as the initial model to train the encoder-decoder reconstructor (Jointly). The BLEU values of the three models were tracked respectively, and the comparison results of the BLEU values of the three models were obtained, as shown in Table 3.

As can be seen from Table 3, if the model is not pre-trained before training the encoder-decoder reconstruction framework, the model translation effect after adding the reconfigurator will be lower than the baseline; Through pre-training and then training the encoder-decoder reconstruction framework, the resulting model translation effect increases by 0.89% BLEU over the baseline. The training time of the model with reconfigurator is obviously extended.

Table 3. BLEU value of the comparison experiment

Model	BLEU
Baseline-NMT	23.18
+Reconstructor	20.50
+Reconstructor(Jointly)	24.07

The BLEU values of the baseline model, namely the attention-mechanism-based translation model and the (Reconstructor(Jointly)) model, were tracked, and the changes of the BLEU values of the two models with the increase of the training cycle were obtained, as shown in Table 4.

Table 4. BLEU values with training epoch

Model	3	6	9	12	15	18
Pre-trained model	2.6	6.6	10	14	17	19
Reconstruction model	2.9	7.1	10	15	18	19

6. Conclusion

The end-to-end neural machine translation model is quite mature, but it has the problem of over-translation and missing translation. Therefore, this paper proposes an English-Chinese neural machine translation method based on the encoder-decoder reconstruction framework. In order to alleviate the problem of dimensionality disaster, this paper first uses Word2vec technology to process the English-Chinese parallel corpus. Then, the end-to-end E-C neural machine translation model is pre-trained, and finally the English-Chinese neural machine translation model based on the encoder-decoder reconstruction framework is trained, which effectively alleviates the phenomenon of over-translation and missing translation in the process of E-C machine translation. However, due to the relative scarcity of English-Chinese parallel corpora, the quality of the translation model obtained is not particularly ideal. Therefore, obtaining high-quality English-Chinese parallel corpora with wide coverage has become one of the focuses of future research to improve the effectiveness of machine translation.

7. Conflict of Interest

The authors declare that there are no conflict of interests, we do not have any possible conflicts of interest.

Acknowledgments. None.

References

1. Cho K, Van Merri nboer B, Bahdanau D, et al. On the properties of neural machine translation: Encoder-decoder approaches[J]. arxiv preprint arxiv:1409.1259, 2014.
2. S. Yin, H. Li, Y. Sun, M. Ibrar, and L. Teng. Data Visualization Analysis Based on Explainable Artificial Intelligence: A Survey[J]. IJLAI Transactionss on Science and Engineering, vol. 2, no. 2, pp. 13-20, 2024.
3. Yin S, Li H, Laghari A A, et al. An Anomaly Detection Model Based On Deep Auto-Encoder and Capsule Graph Convolution via Sparrow Search Algorithm in 6G Internet-of-Everything[J]. IEEE Internet of Things Journal, 2024. doi: 10.1109/JIOT.2024.3353337.
4. Zhang B, Xiong D, Su J. Neural machine translation with deep attention[J]. IEEE transactions on pattern analysis and machine intelligence, 2018, 42(1): 154-163.
5. Liang J, Du M. Two-Way Neural Network Chinese-English Machine Translation Model Fused with Attention Mechanism[J]. Scientific Programming, 2022, 2022(1): 1270700.
6. Xiong H, He Z, Hu X, et al. Multi-channel encoder for neural machine translation[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2018, 32(1).
7. Sung T W, Liu J Y, Lee H, et al. Towards end-to-end speech-to-text translation with two-pass decoding[C]//ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019: 7175-7179.
8. Qing-dao-er-ji R, Su Y L, Liu W W. Research on the LSTM Mongolian and Chinese machine translation based on morpheme encoding[J]. Neural Computing and Applications, 2020, 32: 41-49.

9. Xie S, Xia Y, Wu L, et al. End-to-end entity-aware neural machine translation[J]. Machine Learning, 2022, 111(3): 1181-1203.
10. Yin S, Li H, Teng L, et al. Attribute-based multiparty searchable encryption model for privacy protection of text data[J]. Multimedia Tools and Applications, 2023: 1-22.
11. Xiao Q, Chang X, Zhang X, et al. Multi-information spatialCtemporal LSTM fusion continuous sign language neural machine translation[J]. IEEE Access, 2020, 8: 216718-216728.
12. Di Gangi M A, Negri M, Cattoni R, et al. Enhancing transformer for end-to-end speech-to-text translation[C]//Proceedings of Machine Translation Summit XVII: Research Track. 2019: 21-31.
13. Zhou J, Cao Y, Wang X, et al. Deep recurrent models with fast-forward connections for neural machine translation[J]. Transactions of the Association for Computational Linguistics, 2016, 4: 371-383.
14. Jiang Y, Yin S. Heterogenous-view occluded expression data recognition based on cycle-consistent adversarial network and K-SVD dictionary learning under intelligent cooperative robot environment[J]. Computer Science and Information Systems, 2023, 20(4): 1869-1883.
15. Christian H, Agus M P, Suhartono D. Single document automatic text summarization using term frequency-inverse document frequency (TF-IDF)[J]. ComTech: Computer, Mathematics and Engineering Applications, 2016, 7(4): 285-294.
16. Adamson D, Kosche M, Ko T, et al. Longest common subsequence with gap constraints[C]//International Conference on Combinatorics on Words. Cham: Springer Nature Switzerland, 2023: 60-76.
17. Bringmann K, Cohen-Addad V, Das D. A Linear-Time n 0.4-Approximation for Longest Common Subsequence[J]. ACM Transactions on Algorithms, 2023, 19(1): 1-24.
18. L. Teng, et al. FLPK-BiSeNet: Federated Learning Based on Priori Knowledge and Bilateral Segmentation Network for Image Edge Extraction[J]. IEEE Transactions on Network and Service Management, vol. 20, no. 2, pp. 1529-1542, June 2023, doi: 10.1109/TNSM.2023.3273991.
19. Yin S, Wang L, Shafiq M, et al. G2Grad-CAMRL: an object detection and interpretation model based on gradient-weighted class activation mapping and reinforcement learning in remote sensing images[J]. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 16, 3583 - 3598, 2023.
20. Yonglan L, Wenjia H. English-Chinese Machine Translation Model Based on Bidirectional Neural Network with Attention Mechanism[J]. Journal of Sensors, 2022.
21. Hegde A, Gashaw I, Hl S. Mucs-machine translation for dravidian languages using stacked long short term memory[C]//Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages. 2021: 340-345.
22. Al-Jarrah M, Alsusa E, Masouros C. A unified performance framework for integrated sensing-communications based on KL-divergence[J]. IEEE Transactions on Wireless Communications, 2023, 22(12): 9390-9411.

Biography

Su Ma is with the College of International Business, Shenyang Normal University. Research direction is English-Chinese translation, English linguistics and AI.