Chinese Traditional Poetry Generating System Based on Deep Learning

Chenlei Bao[1], Lican Huang[1,2]

***Abstract*—Chinese traditional poetry is an important intangible cultural heritage of China and an artistic carrier of thought, culture, spirit and emotion. However, due to the strict rules of ancient poetry, it is very difficult to write poetry by machine. This paper proposes an automatic generation method of Chinese traditional poetry based on deep learning technology, which extracts keywords from each poem and matches them with the previous text to make the poem conform to the theme, and when a user inputs a paragraph of text, the machine obtains the theme and generates poem sentence by sentence. Using the classic word2vec model as the preprocessing model, the Chinese characters which are not understood by the computer are transformed into matrix for processing. Bi-directional Long Short-Term Memory is used as the neural network model to generate Chinese characters one by one and make the meaning of Chinese characters as accurate as possible. At the same time, TF-IDF and TextRank are used to extract keywords. Using the attention mechanism based encoding-decoding model, we can solve practical problems by transforming the model, and strengthen the important information of long-distance information, so as to grasp the key points without losing important information. In the aspect of emotion judgment, Long Short-Term Memory network is used. The final result shows that it can get good poetry outputs according to the user input text.**

***Index Terms*— Deep learning; natural language generation; ancient poems; keyword extraction; emotional analysis**

I. INTRODUCTION

Chinese traditional poetry stands as a cornerstone of China's intangible cultural heritage, encapsulating profound life philosophies and historical narratives from ancient times to the present. Serving as a powerful artistic medium, it conveys intricate layers of thought, culture, spirit, and emotion. However, due to the strict rules and constraints of ancient poetry, machine-generated compositions remain a formidable challenge.

In recent years, great breakthroughs have been made in the related technologies of deep learning, which has been widely and successfully applied to various aspects such as speech recognition, image recognition, natural language processing and so on. The automatic generation of poetry has attracted more attention[1,4]. The machine should not only learn the structural characteristics, rules of poetry, but also learn the various meanings of Chinese characters and the author's emotion contained in the characters. This paper mainly studies the application of deep learning technology to the automatic generation of Chinese traditional poetry.

The automatic generation of poetry in Western countries predates that in China. However, early machines could only use simple programs to combine or mechanically copy words, lacking the deep understanding of ancient poetry and songs. The main research into mechanical poetry generation began in the 1970s. The following section introduces several traditional methods of poetry generation by machines.

The "Word Salad" method is the earliest approach to poetry generation. This method treats words as raw material and mixes them randomly. However, because this approach does not consider the meanings of the words or the requirements of poetry, the resulting poetic effect is often poor.

The "Template and Pattern" method for poetry generation is akin to filling in the blanks in sentences. By predefining template formats and identifying elements such as subjects and predicates, this method uses the original text as a poetry template and fills in the blanks with words that fit the specified format, thus generating a new poem. While this approach helps avoid the issue of words not aligning with the nature of poetry, it can be quite rigid due to its fill-in-the-blank nature. Subsequently, a pattern-based generation method emerged. This approach involves generating poetry within a predefined framework, effectively creating poems by adhering to the characteristics of each position's word and the tonal rhyme and oblique tones of poetry. This method can to some extent meet the requirements of grammar and rhythm.

Changle Zhou, et.al[2] proposed a method for generating poetry based on genetic algorithms. This approach treats the poetry generation process as a search process, beginning with a random line and then using a predefined evaluation method to assess each iteration. Through iterative processes, the algorithm aims to produce the desired poem. While this method may be effective in generating individual lines of poetry, it may not yield satisfactory results for the entire poem.

Rui Yan, et.al [20] drew inspiration from automatic summarization technology and applied it to poetry generation. Through the words entered by the user, the system searches and selects similar words in the corpus, sorts and selects the keywords according to the conditions of semantics, tonal rhyme, etc. Finally, the algorithm modifies and replaces these keywords through optimization of restrictive conditions to generate the poetry.

Jing He, et.al [3] approach poetry generation as a machine translation problem. They use the previous sentence as the input for predicting the next sentence, by adding constraints to guide the prediction process. and continue this iterative process until a complete poem is generated.

Lican Huang is the corresponding author (e-mail: huang_lican@yahoo.co.uk). [1] was affiliated with Zhejiang Sci-Tech University, [2] is affiliated with Hangzhou Domain Zones Technology Co., Ltd.

Manuscript received December 1, 2023, Revised March 25, 2024, Accepted March 30, 2024.

In this paper, we use deep learning method to generate chinese traditional poetry. The draft preprint version can be found in arXiv[23].

II. RELATED WORK

2.1 Word vector model

Text is a form of unstructured data that cannot be directly computed or utilized. The purpose of text representation is to convert unstructured information into a structured format, enabling us to effectively perform calculations and analyses on this textual information.

Word Embedding can be understood as a mapping process where words are calculated, transformed, and projected into a numerical space that can be interpreted by computers. This process allows computers to understand the meaning and relationships between words based on their numerical representations.

Word2vec[5,6,7] is a method of Word Embedding that transforms text into vectors through a function f(x) -> y. This method utilizes a weight matrix that is continuously improved during the training process to transform words into one-dimensional matrices, known as word vectors.

Word2vec has two main methods, CBOW (Continuous Bag-of-Words Model) and Skip-Gram (Continuous Skip-gram Model)[8]. Among them, CBOW predicts the current value through the context, and Skip-Gram: predicts the context through the current value.

2.2 Keyword extracting

Keyword extraction methods for text can be broadly classified into three categories: supervised, semi-supervised, and unsupervised.

1. Supervised keyword extraction: The dataset has predefined keywords marked in advance, and the extraction process involves identifying similarity to these predetermined keywords.
2. Semi-supervised keyword extraction: This method involves providing a portion of labeled data for training. After training, the model can identify keywords in new data, which are then confirmed and added to the dataset. This process iterates to improve the keyword extraction over time.
3. Unsupervised keyword extraction: This method directly extracts keywords from the given text without relying on any preexisting labels or training data.

In this article, two unsupervised keyword extraction methods, TF-IDF[11]and TextRank[10], are primarily utilized.

2.2.1 TF-IDF

The main idea behind TF-IDF (Term Frequency-Inverse Document Frequency) is to measure the importance of a word in a document relative to a collection of documents. TF (Term Frequency) calculates how often a word appears in a document relative to the total number of words in that document. IDF (Inverse Document Frequency) measures the degree of discrimination of a word. It reflects how unique or rare a word is across a collection of documents. Words that are common across many documents will have a lower IDF score, while words that are rare and unique to specific documents will have a higher IDF score. The TF-IDF value for a word is calculated by multiplying the TF value by the IDF value. Words with a high TF-IDF value are considered important and are likely to be keywords that distinguish the document from others in the collection.

The Term Frequency (TF) of word w in the article is calculated as:

$$TF_w = \frac{a}{b} \quad (1)$$

This formula gives the frequency of word w (a) in the article , normalized by the total number of words (b) in the article.

The Inverse Document Frequency (IDF) of word w is calculated as:

$$IDF_w = \log\left(\frac{N}{N_w}\right) \quad (2)$$

Where:

N is the total number of documents in the collection.

N_w is the number of documents in the collection that contain word w.

The TF-IDF value of a word w can be obtained by multiplying its Term Frequency (TF) by its Inverse Document Frequency (IDF).

$$TF-IDF_w = TF_w \times IDF_w \quad (3)$$

This formula combines the TF and IDF scores to determine the importance of word w in the article relative to the entire collection of documents.

2.2.2 TextRank

The TextRank algorithm is a graph-based ranking algorithm used for text processing, particularly for keyword extraction and text summarization. The text is represented as a directed weighted graph G=(V,E), where V is the set of vertices (nodes) representing units of text (e.g., words or phrases), and E is the set of edges representing relationships between these units. Each edge in E is a pair of vertices, indicating a connection between them. The weight of an edge between vertices Vi and Vj is denoted as Wji.

The score of a vertex Vi is calculated using the following formula:

$$Score(V_i) = (1-d) + d \times \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{V_k \in Out(V_j)} w_{jk}} \times Score(V_j) \quad (4)$$

Where:

d is a damping factor, typically set between 0.85 and 0.95, used to balance between the importance of the graph structure and the importance of the individual vertex.

Wji is the weight of the edge from vertex Vj to vertex Vi.

In(Vi) is the set of vertices pointing to Vi.

Out(Vj) is the set of vertices Vj points to.

Score(Vj) is the score of vertex Vj.

The algorithm iteratively calculates the score for each vertex until convergence is reached. The vertices with the

highest scores are considered the most important and are selected as keywords or key phrases.

The basic outline of how TextRank works for keyword extraction is as following:

1. Text Preprocessing: The text is divided into units, such as words or phrases, and any necessary preprocessing (like removing stopwords or stemming) is performed.
2. Graph Construction: A graph is constructed, where each node represents a unit (word or phrase) and edges represent relationships between units. The relationships can be based on co-occurrence within a window of words or other metrics.
3. Graph-based Ranking: Each node is assigned an initial score, and these scores are iteratively updated based on the scores of neighboring nodes. The update rule typically involves taking a weighted sum of the scores of neighboring nodes.
4. Convergence: The iteration continues until the scores converge, indicating that the ranking has stabilized.
5. Keyword Extraction: The units with the highest scores are selected as keywords, as they represent the most important or central concepts in the text.

By using this graph-based approach and leveraging the relationships between words or phrases, TextRank can effectively identify keywords and key phrases in a text.

2.3 Network Model BiLSTM

Recurrent neural network(RNN)[12,13,14,15,16,17,18] is a type of neural network that processes sequential data by maintaining a memory of previous inputs. This memory allows RNNs to exhibit temporal dynamics and make predictions based on the current input and past inputs. RNNs are particularly well-suited for tasks involving sequential data, such as time series analysis, natural language processing, and speech recognition.

One key characteristic of RNNs is their ability to maintain a hidden state that captures information from previous time steps. This hidden state is updated at each time step based on the current input and the previous hidden state. This recurrent nature allows RNNs to capture long-range dependencies in sequential data.

In many scenarios, understanding context requires considering relationships between words or sentences that are not immediately adjacent. Traditional RNNs struggle with this because they can forget information from earlier time steps as new information is processed, especially over long sequences.

Long Short-Term Memory (LSTM) networks, a type of RNN, were specifically designed to address this issue. LSTMs have a more complex architecture compared to traditional RNNs, including a memory cell and three gates: input gate, forget gate, and output gate. By selectively updating the memory cell and gates, LSTMs can learn to capture long-range dependencies in sequential data.

While the LSTM model can handle sentence structures better, it still faces a limitation—it cannot effectively process information from both the front and the back of a sequence. Additionally, when dealing with more nuanced content, it is important to consider the interaction of emotion words and degree words.

The Bidirectional LSTM (BiLSTM) addresses this limitation by combining both forward and reverse LSTM architectures. This allows the model to process the input sequence in both directions, capturing information from both past and future contexts. The final hidden state at each time step is obtained by concatenating the hidden states from the forward and backward LSTMs, resulting in a more comprehensive representation of the input sequence.

2.4 Encoder-decoder model

The encoder-decoder model is a type of architecture used in sequence-to-sequence learning tasks, such as machine translation and text summarization. It consists of two main components: an encoder and a decoder.

Encoder: The encoder processes the input sequence and produces a fixed-length representation (often called the context vector) that captures the input sequence's meaning or context. The encoder can be a recurrent neural network (RNN), such as an LSTM or GRU, or a convolutional neural network (CNN).

Decoder: The decoder takes the context vector produced by the encoder and generates an output sequence based on it. Like the encoder, the decoder can be an RNN or any other type of neural network. During training, the decoder is fed the correct output sequence, while during inference (or testing), it generates the output sequence one token at a time, using its own predictions as input for the next time step.

2.5 Emotion analysis

Emotion analysis[19] is mainly divided into the following processes:

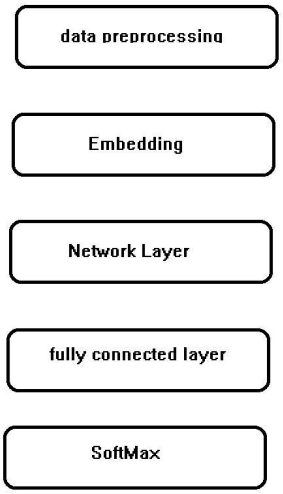


Fig.1 Emotion analysis process

Data preprocessing involves several steps, including removing garbled characters, eliminating stop words, and segmenting words. Since the length of each sentence varies, and many ancient poems consist of either five or seven characters, the longest sentence in the current batch can be chosen as the benchmark. And and fill the length with 0 to the length of the current bach_size to facilitate the follow-up of the data deal with.

The Embedding layer is used to convert one-dimensional word vectors into higher-dimensional vectors, thereby capturing more semantic information. In this article, a dimensionality of 300 is chosen for the high-dimensional vectors.

The network layer is initialized first, and then the input vector is superimposed with the initial hidden state, the weight is updated through the network, and the hidden state h is obtained. The previously extracted features are synthesized through a fully connected layer, the information after the integration result is output through a certain linear operation, and the processed data is used for classification.

The Softmax output layer classifies n categories. The input of the fully connected layer is passed through the Softmax function to obtain n probability values, the largest of which is the result of the classification.

III. CHINESE TRADITIONAL POETRY AUTHORING SYSTEM IMPLEMENTATION

3.1 Poetry data source and data preprocessing

The poetry data used in this project is sourced from the GitHub project Chinese Poetry[22], which is known for being the most comprehensive database of Chinese Poetry Classics. It includes hundreds of thousands of ancient poems and songs from the Tang and Song dynasties, provided in JSON format.

Since ancient Chinese poetry is mainly categorized into five-character and seven-character poems, the poetry data is first separated into these two categories.

Read the five character and seven character data respectively, extract the content of each poem, obtain the keyword for each poem, and store them in a new file. The format should include the keyword followed by the current poem and the previously processed text. For example, the poem "床前明月光，疑是地上霜(Moonlight in front of bed, suspected to be frost on the ground)" is stored as "月光|床前明月光(Moonlight | moonlight in front of bed)" and "霜|床前明月光，疑是地上霜(frost | moonlight in front of bed, suspected to be frost on the ground)".

Set four special character marks, namely, space, unknown, start and end, to facilitate the determination of the beginning and end of poetry, special characters and other unqualified contents during prediction and training. Traverse the data and build a sequence index .

Set the maximum sequence length in each batch as the benchmark for the current batch, and pad the remaining sentences shorter than this length with spaces until they reach the maximum length.

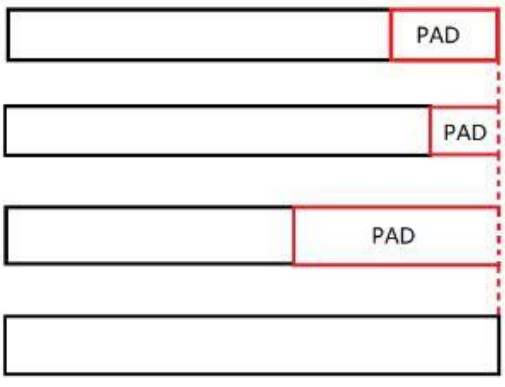


Fig.2 Completing sequence with PAD

3.2 Chinese word vector

Chinese Wikipedia is used as the training set, and Skip-Gram method (word2vec) is used for training. The method based on context co-occurrence and PCA dimension reduction is used for dependency vectors. The number dimension of the trained word vector is 20029 and the dimension is 300.

In poetry training, the entire pre-trained file is read. If a word is not found in the trained word vector file, the word is randomly assigned a value appropriate for the dimension and added to the word vector. In subsequent training, the value of the word is adjusted to align it as closely as possible with the position of the word's meaning in space.

3.3 Construction Network Model

Read the pre trained data, build id2word and word2id sequences, and build embedding.

For the encoder side, use the built embedding to obtain the keyword and the previous content data in the current batch. The bidirectional long-term and short-term memory network (BiLSTM) is used as the network layer to create neurons with two keywords and previous content for forward and backward respectively. Then build the network of keywords and the previous content respectively. In the neural networks of the keyword and the previous content, the output hL and hR obtained in the forward and backward directions are spliced and combined into the final h, and the dimension is specified as 1. Finally, the key words and the output of the previous content are spliced to form a whole.

For the decoder side, the same as the encoder side, use the built embedding and get the contents in the batch. The functions used for training at the decoder are TrainingHelper and GreedyEmbeddingHelper respectively. TrainingHelper does not take the output of t-1 stage as the input of t-stage, but directly inputs the real value to RNN, so that each input is the real value of text. For example, "床前明月光(bright moon in front of bed)", the input of TrainingHelper is ['床', '前', '明', '月', '光' ('bed', 'front', 'bright', 'moon', 'light')]. GreedyEmbeddingHelper is generally used for decoder decoding in the prediction stage. It does not use the real value of the text, but embeds the value obtained from each prediction as the next input value. The difference between GreedyEmbeddingHelper and TrainingHelper is that it embeds the output under t-1 and then inputs it to RNN.

3.4 Model Training

The training process involves iteratively feeding batches of data to the model, calculating the loss, computing gradients, and updating the model parameters until the model converges to a satisfactory solution.

Sequence Loss Function: Softmax Evaluation: Apply the softmax function to the output logits to obtain probabilities for each word in the vocabulary. Cross-Entropy Selection: Calculate the cross-entropy loss between the predicted probabilities and the actual labels (one-hot encoded). Calculate Average: Average the cross-entropy losses across

all time steps and sequences in the batch to get the final loss.
Adam Optimizer: Use the Adam optimizer to minimize the loss function. Adam is an adaptive learning rate optimization algorithm that computes individual adaptive learning rates for different parameters.
Calculate Gradients: Use automatic differentiation to calculate the gradients of the loss function with respect to each trainable parameter in the model. This is typically done using backpropagation through time (BPTT) for RNNs.
Update Variables: Use the calculated gradients to update the corresponding variables (weights and biases) in the model. This is usually done using the update rule for the optimizer, which adjusts the parameters in the direction that reduces the loss.

3.5 Prediction Implementation

The poetry generation method works as follows: the user inputs a paragraph of text, from which key words are extracted to form a poetry outline. Each verse to be generated is matched with a keyword to ensure coherence. If the extracted keywords are insufficient to generate the required number of poem lines, synonyms are used to expand the keywords.

During prediction, the extracted keywords from user input are sequentially fed into the computational diagram. For the first keyword, the preceding text is empty. Subsequently, the generated poem line is added to the previous list in order, and both the poem lines and keywords are input into the computational diagram. For instance, given the keywords "月光 (Moonlight)" and "霜 (frost)", the first line "床前明月光 (Moonlight in front of bed)" is obtained by inputting "月光 (Moonlight)" and an empty preamble. Then, the second line "疑是地上霜 (suspected to be frost on the ground)" is obtained by inputting the keyword "霜 (frost)" and the preceding text "床前明月光 (Moonlight in front of bed)".

3.6 Evaluation of Model

This paper uses the perplexity degree to judge the quality of the model. The basic idea of perplexity degree is: the model which assigns a higher probability value to the sentences in the test set is better. After the model is trained, When the test set is full of normal sentences, the higher the probability of the trained model on the test set, the better of the model. The formula is as follows:

$$PP(w) = \sqrt[n]{P(w_1 w_2 \dots w_n)} \qquad (5)$$

According to the formula, the better the model, the higher the sentence probability, and the lower the perplexity.

3.7 Emotional data source and its pre-preprocessing

The emotional data, collected from a dataset by Minlie Huang et al. [21], comprises over 40,000 sentences categorized into six emotions: others, likes, sadness, disgust, anger, and happiness, labeled from 0 to 5. This dataset originates from the NLPCC Emotion Classification Challenge and was manually annotated after filtering microblog data. Due to the dataset's web-sourced nature, it requires preprocessing and data cleaning.

The sentences are converted into digital sequences and the sentence lengths are standardized. Selecting an optimal length is crucial; if the benchmark length is too long, it may introduce many zeros, while if it is too short, it might truncate the sentence's meaning. Therefore, it is essential to actively select and test a suitable length based on the data.

The dataset is split into training and test sets in a 1:4 ratio.

IV. Experimental Results and Analysis

4.1 Experimental environment

The experimental environment includes Operating System, Compile Language, Deep Learning Framework, Compile Tool, etc. as Table 1 shown.

Table 1 Experimental environment table

Environment	Version and parameters
Operating system	Windows10
Compile language	Python 3.7.0
Deep Learning Framework	TensorFlow-gpu 1.14.0
Compile Tool	PyCharm 2020.3.5
GPU	NVIDIA Tesla V100 16G

4.2 Experimental results

When a user inputs text, the system generates a poem based on the input content. Here are some examples of the generated poems.

In Figure 3, the input is "黄昏，一缕轻烟从烟囱里飘出来，抬头望着天空，只见晚霞已经染红了大半片天空，旁边的青山郁郁葱葱，其上有飞鸟掠过。(At dusk, a wisp of light smoke drifted out of the chimney. Looking up at the sky, one could see the evening glow had already dyed half the sky red. The nearby green mountains were lush, with birds flying over them.)" . The generated poem is "晚霞孤鹜带晴晖，万里风烟入翠微。一片青山依旧好，不知何处是春归。(The evening glow, like a solitary duck, brings a clear light. Wind and smoke from afar blend into the emerald green. The green mountains are still beautiful, but it's unclear where spring has gone.)" . The emotion category is "喜好 (liking)".

In Figure 4, the input is "树木长得葱葱笼笼，一棵棵大树，似一顶顶绿绒大伞遮住了阳光，绿叶一簇堆在另一簇上面,不留一点儿缝隙。(The trees grow lush and dense, each large tree resembling a green velvet umbrella covering the sunlight. Clusters of green leaves pile on top of each other, leaving no gaps.)" . The generated poem is "万木森森一径开,绿阴深处独徘徊。一枕清凉梦觉时，草径春风绿满怀。(A path opens through a dense forest, where I wander alone in the deep shade. Awaking from a cool dream, I find myself embraced by the spring breeze on a grassy path.)". The emotion category is "高兴(Gladness)".

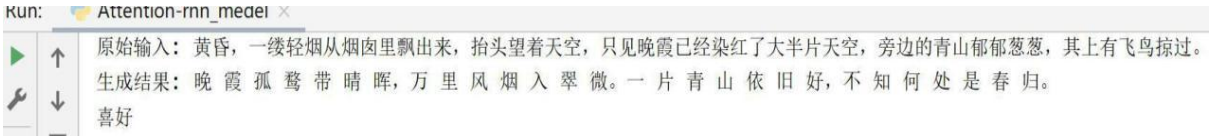


Fig.3 poem a generated

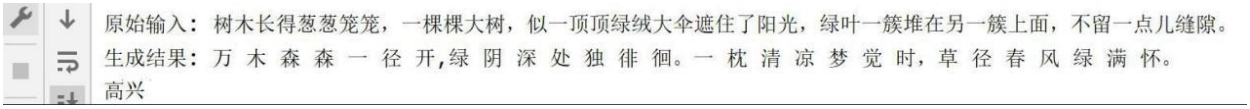


Fig.4 poem b generated

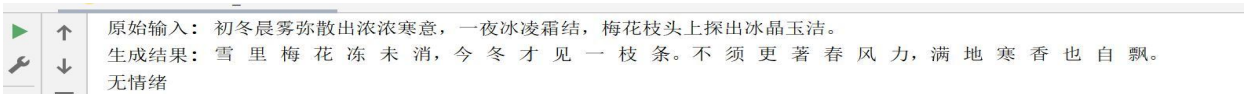


Fig.5 peom c generated.



Fig.6 peom d generated.

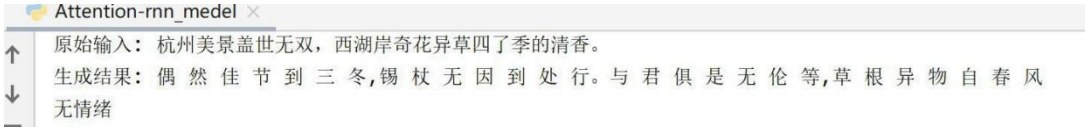


Fig.7 poem e generated.

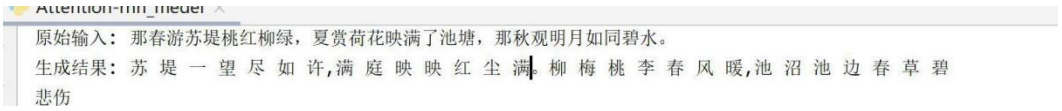


Fig.8 peom f generated.

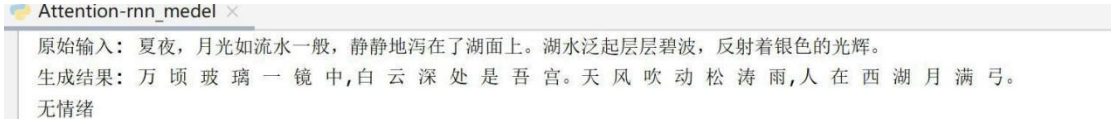


Fig.9 peom g generated.

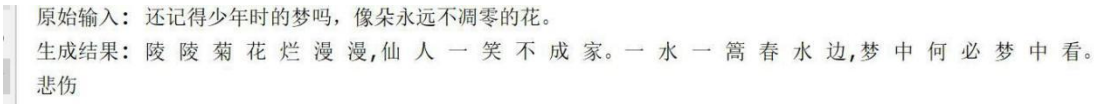


Fig.10 peom h generated.

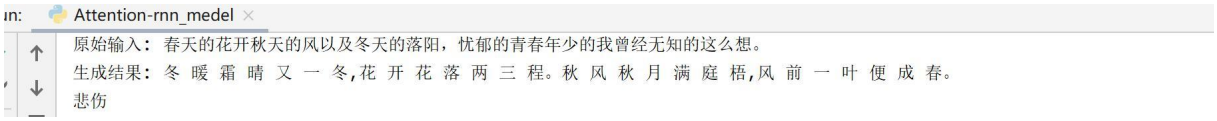


Fig.11 poem i generated.

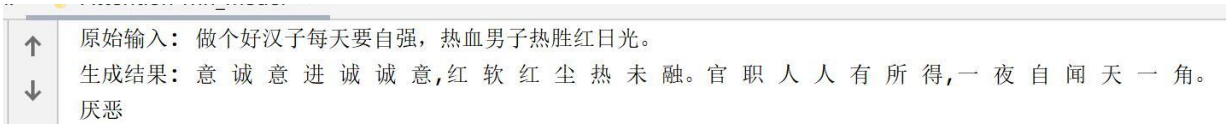


Fig.12 peom j generated.

In Figure 5, the input is “初冬晨雾弥散出浓浓寒意，一夜冰凌霜结，梅花枝头上探出冰晶玉洁。(In early winter, the morning mist spreads a thick chill, Ice and frost have formed overnight, the plum blossom branches glisten.)” . The generated poem is “雪里梅花冻未消,今冬才见一枝条。不须更著春风力，满地寒香也自飘。(Amidst the snow, the plum blossom remains unfazed, This winter, it's the first branch to be raised. No need for the spring breeze to lend its might, The fragrance of cold drifts naturally, pure and light.)”. No emotion category .

In Figure 6, the input is “燕山雪花大如席，片片吹落轩辕台。(Snowflakes on Yanshan Mountain as large as mats, Drifting down piece by piece on Xuanyuan Terrace.)” . The generated poem is “天心似玉月如霜,万里风埃冻色凉。一夜西山生白练，满城和雾作寒光。(The sky's heart resembles jade, the moon akin to frost, For miles, frozen dust brings a chilly hue, not lost. Overnight, the western mountains are clad in white, The city shrouded in mist, a cold light in sight.)”. The emotion category is "厌恶(Aversion)”.

In Figure 7, the input is “杭州美景盖世无双，西湖岸奇花异草四了季的清香。(Hangzhou's beautiful scenery is unparalleled, On the banks of West Lake, unique flowers and exotic grasses exude the fragrance of all four seasons.)” . The generated poem is “偶然佳节到三冬,锡杖无因到处行,与君俱是无伦等,草板异物白春风。(By chance, the festival arrives in the third winter, I wander with my walking stick without reason. We are Unmatched kind, Even the grass and wood are touched by the white spring breeze.)”. No emotion category .

In Figure 8, the input is “那春游苏堤桃红柳绿，夏赏荷花映满了池塘，那秋观明月如同碧水。(In spring, the Su Causeway is adorned with peach blossoms and green willows, In summer, lotus flowers bloom, filling the pond with their reflections. In autumn, the bright moon shines like jade in the clear water.)” . The generated poem is “苏堤一望尽如许,满庭映映红尘洲柳梅桃李春风暖,池沼池边春草碧。(As far as the eye can see on the Su Causeway, The courtyard is reflected with red dust isle. Willows, plum, apricot, and cherry bloom, warmed by the spring breeze, The pond is green with spring grasses by the edge.)”. The emotion category is "悲伤(Sadness)”.

In Figure 9, the input is “夏夜，月光如流水一般，静静地泻在了湖面上，湖水泛起层层碧波，反射着银色的光辉。(On a summer night, moonlight flows like water, Quietly pouring onto the lake's surface, The lake ripples with layers of green waves, Reflecting silver brilliance.)” . The generated poem is “万顷玻璃一镜中,白云深处是吾宫,天风吹动松涛雨,人在西湖月满号。(In a mirror, vast as glass, all is reflected, My palace lies deep in the white clouds, The wind stirs the pines, rain falls gently, I am in West Lake, where the moon shines bright.)”. No emotion category .

In Figure 10, the input is “还记得少年时的梦吗，像朵永远不凋零的花。(Do you still remember the dreams of your youth, Like a flower that never fades?)”. The generated poem is “陵陵菊花烂漫漫,仙人一笑不成家,一水一篙春水边,梦中何必梦中看。(Chrysanthemums bloom abundantly, The immortal's smile never settles, By the spring water with a pole in hand, Why dream within a dream?)”. The emotion category is "悲伤(Sadness)”.

In Figure 11, the input is “春天的花开秋天的风以及冬天的落阳，忧郁的青春年少的我曾经无知的这么想。(The flowers blooming in spring, with the autumn wind and the setting sun of winter, all these used to be thought by my melancholic, youthful, and ignorant self.)” . The generated poem is “冬暖霜晴又一冬,花开花落两三程。秋风秋月满庭栝,风前一叶便成春。(Winter warmth, frosty and clear, another winter; flowers bloom and fall, two or three rounds. Autumn wind, autumn moon, fill the courtyard with locust trees; before the wind, a leaf shows spring.)”. The emotion category is "悲伤(Sadness)”.

In Figure 12, the input is “做个好汉子每天要自强，热血男子热胜红日光。(Be a good man, strive for self-improvement every day; passionate men are hotter than the sun.)” . The generated poem is “意诚意进诚诚意,红软红尘热未融。官职人人有所得,一夜自闻天一角。(Intentions progress with sincerity, earnest intentions; the red soft dust has not yet melted in the heat. Every official has something to gain, and me hearing the corners of heaven in a single night sonely.)”. The emotion category is "厌恶(Aversion)”.

The generated poems are generally good, but there are instances where the themes slightly differ. The emotion catogories sometimes are unsuitable to the themes.

V. CONCLUSION AND FUTURE WORK

This paper mainly studies the application of deep learning technology to the automatic generation of Chinese traditional poetry. The key words of each poem are compared with the existing previous text for training, so that the trained model can be generated sentence by sentence while keeping the theme unchanged. At the same time, the poetry emotion judgment function is added, which can make the poetry emotion meet the needs of users as much as possible. Because there is no poetry emotion data set that has been emotion classified, we use other article emotion classification data sets, and the effect is not satisfactory in the actual test. In addition, there are many skills in poetry, such as metaphor, comparison, etc. it is difficult to use machine generation alone, which still needs more in-depth research.

REFERENCES

[1] Z Y. Liu. Research on automatic generation algorithm of Chinese poetry based on deep learning [D]. Hunan Normal University, 2018. (in Chinese)

[2] C L. Zhou, W. You , X J. Ding. A genetic algorithm for automatically generating Song Ci and its machine implementation [J]. Acta Sinica Sinica, 2010,21 (03): 427-437. (in Chinese)

[3] J. He, M. Zhou, and L. Jiang. Generating chinese classical poems with statistical machine translation models. In Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

[4] J S. SU, C L. Zhou, Y H. Li. Establishment of a corpus of CI segmentation in the whole Song Dynasty based on statistical word extraction and metrical [J]. Acta Sinica , 2007 (02): 52-57.(in Chinese)

[5] L. Zhou. Working principle and application of word2vec [J]. Sci tech information development and economy, 2015,25 (02): 145-148. (in Chinese)

[6] D. Huang and J. Wang. An approach on Chinese microblog entity linking combining baidu encyclopaedia and word2vec[J]. Procedia Computer Science, 2017, 111 : 37-45.
[7] F. Xiang, et al. Automatic Construction of a Chinese Sentiment Lexicon in the Field of Education Based on Word2vec[J]. Data Science and Industrial Internet, 2019, 2(1).
[8] L. Zhang, Y L. Zhu. Vocabulary information acquisition in natural language: an improved skip gram model [J]. Journal of Beijing Institute of Electronic Science and technology, 2020,28 (02): 19-26. (in Chinese)
[9] N. Alami , M. Meknassi and N. En-nahnahi. Enhancing unsupervised neural networks based text summarization with word embedding and ensemble learning[J]. Expert Systems With Applications, 2019, 123.
[10] C X. Meng, Y. Zhang, N N. Li. Research on improved keyword extraction method based on textrank [J]. Computer and digital engineering, 2020,48 (12): 3022-3026. (in Chinese)
[11] Y. Luo, S L. Zhao, X C. Li, et al. Text keyword extraction method based on word frequency statistics [J]. Computer applications, 2016,36 (03): 718-725. (in Chinese)
[12] J. Zhang, D. Qu, Z. Li. Recurrent neural network language model based on word vector features [J]. Pattern recognition and artificial intelligence, 2015,28 (04): 299-305. (in Chinese)
[13] Y J. Huang, J. Zhu, J C. Yang. Application and comparative analysis of common neural network models in automatic generation of ancient poetry [J]. Journal of Dongguan Institute of technology, 2020,27 (05): 55-60. (in Chinese)
[14] H S. Dong. Generation of ancient poetry based on long short term memory network [J]. Modern computer (Professional Edition), 2018 (32): 18-21 + 26. (in Chinese)
[15] Y. Wen, et al. Chinese poetry and couplet automatic generation based on self-attention and multi-task neural network model[C]. Kitakyushu, Japan, 2020.
[16] Z L. Zhu, Y Rao, Y Wu, et al. Research progress of attention mechanism in deep learning [J]. Chinese Journal of information, 2019,33 (06): 1-11. (in Chinese)
[17] Y. Nie, et al. Attention-based encoder-decoder model for answer selection in question answering[J]. Frontiers of Information Technology & Electronic Engineering, 2017, 18(4) : 535-544.
[18] A S. Varghese , et al. Bidirectional LSTM joint model for intent classification and named entity recognition in natural language understanding[J]. International Journal of Hybrid Intelligent Systems, 2019, 16(1) : 13-23.
[19] X H. Wu, L. Chen, T T. Wei, et al. Sentiment analysis of Chinese short texts based on self attention and Bi LSTM [J]. Acta Sinica Sinica Sinica, 2019,33 (06): 100-107. (in Chinese)
[20] Y.Rui, et.al, Poet: Automatic Chinese Poetry Composition through a Generative Summarization Framework under Constrained Optimization, Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence i, 2014
[21] M. Huang, et.al, https://www.biendata.xyz/ccf_tcci2018/datasets/emotion/
[22] Github project: https://github.com/chinese-poetry/chinese-poetry
[23] C.Bao, L.Huang. Chinese Traditional Poetry Generating System Based on Deep Learning. https://arxiv.org/pdf/2110.12335.pdf