Research on Intelligent Writing Poetry Model Based on Neural Network

Hanyu Liu

Software College, Jiangxi Normal University, Nanchang 330022, China

ABSTRACT. *Poetry is a special and important cultural heritage in human history. The arrival of the era of big data and the development of deep learning technology have promoted the innovation of algorithm design ideas in the field of artificial intelligence. Deep learning has also developed greatly in poetry generation technology. Even to a certain extent, poetry produced may be difficult for ordinary people to distinguish. However, at present, poetry generation technology still focus on poetry avoid of thoughts and feelings and human spirituality. The work of machine poetry began in the 1970s. The traditional method of poetry generation relies heavily on the professional knowledge of the field of poetry. It requires experts to design a large number of artificial rules to constrain the rhythm and quality of the generated poetry. With the development of deep learning technology, the study of poetry generation has entered a new stage. Many methods of poetry generation based on deep learning have been proposed, such as the method based on RNN language model, the framework based on encoder-decoder, and the method of poetry generation based on GAN. This paper needs to modify this reality from human writing. Based on the iterative polishing mechanism proposed by Rui Yan et al., it adds a more perfect poetry evaluation system and proposes intelligent writing based on deep learning that can be automatically modified for multiple times. The poetry model, after trial and manual evaluation, proves that the method has certain effects.*

KEYWORDS: *deep learning, poetry generation, NLP*

1. Introduction

Poetry is the oldest and most literary style of literature. Since the emergency of "Book of Songs" in China, the poems of the past two thousand years have been colorful. Using intelligent machine to write poetry has always been a challenging task in the field of artificial intelligence. Generating poetry through machine began in the 1970s. The computationalization of Chinese classical poetry began in the mid-1990s. So far, some preliminary results have been achieved in terms of corpus establishment, lexical semantic analysis, creative style analysis, and joint language response.

The earliest poetry generation model, Word Salada, is based on a simple calculation program, which uses a method of connecting randomly generated vocabulary. The result is only the vocabulary of the vocabulary failing to meet the semantic grammar requirements. This method shows little consideration for the content, form and meaning of poetry, and its resuls cannot be called poetry in a strict sense. The traditional method relies heavily on the professional knowledge of the field of poetry, and requires experts to design a large number of artificial rules to constrain the rhythm and quality of the generated poetry. At the same time, the ability to migrate is also poor, and it is difficult to apply directly to other styles and languages. It is difficult to apply directly to other styles and languages.

With the development of deep learning technology, the study of poetry generation has entered a new stage. Based on the RNN language model [1], the overall content of poetry is sent to the RNN language model for training as a training corpus. After the training is completed, some initial content is given, and then the next word can be sampled according to the probability distribution of the language model output, and the process is repeated to produce a complete poem. Xingxing Zhang et al. proposed the RNNPG model [2]. First, the first sentence is generated by the keyword given by the user, and then the second sentence is generated by the first sentence, and the third sentence is generated by one or two sentences. The process goes on until the poetry is completed. Yu, Lantao et al. proposed a sequence generation framework called SeqGAN [3], which used the anti-generation network in the image to generate text. The generated network is an RNN that directly generates the entire poem. The discriminant network is a CNN. It is used to judge whether the poem is written by humans or machine-generated, and the gradient is transmitted back to the generation network by means of reinforcement learning.

From traditional methods to deep learning, poetry generation technology has developed greatly, and even to a certain extent, it has been possible to produce poetry that is difficult for ordinary people to distinguish between true and false. However, at present, the poetry generation technology is still only the probability distribution of knowledge, that is, the collocation law between poems and verses. I did not learn that poetry contains thoughts and feelings. Therefore, although the poems produced seem to be modeled, they still make people feel that they are only in the table, lacking a trace of the spirituality of humanity.

This paper needs to modify this reality from human writing, based on the iterative polishing mechanism proposed by R.Yan et al. [4], and adds a more perfect poetry evaluation system, proposing that it can be automatically used based on deep learning. The modified intelligent writing poetry model has been tested and manually evaluated to prove that the method has certain effects. Figure 1 is a schematic diagram of the model. The main contributions of this paper are as follows:

1) Different from the previous generation methods in neural networks, this paper imitates the process of human writing poetry, and proposes one or more iterations of the generated poems to improve the wording and make it more poetic.

2) Improve the evaluation system of poetry and realize the evaluation of poetry - modify the iteration of writing poems. In addition to the iteration of the process of

writing, after the poetry is finally generated, it will be modified one or more times according to the evaluation of the poetry system.

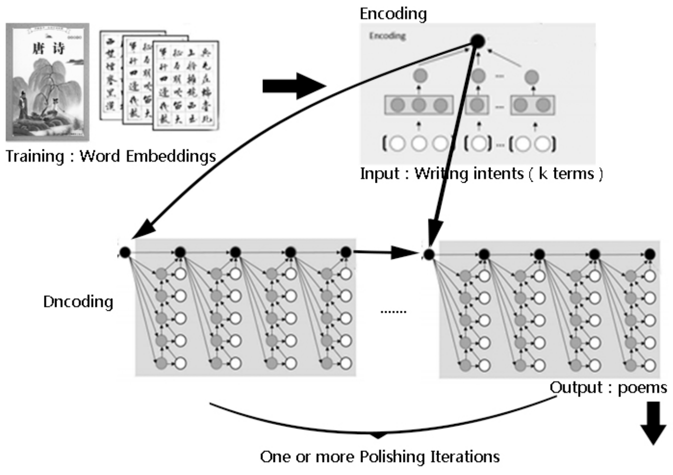
Encoding
Training : Word Embeddings
Input : Writing intents (k terms)
Dncoding
....
Output : poems
One or more Polishing Iterations

Figure 1 Automatically modify the schematic diagram of the intelligent writing poetry model

2. Methodology

Traditional models often treat words as discrete markers, so the intrinsic relationship between similar words will be lost. Word embedding is a standard method of text processing based on neural networks. A word is mapped to a low-dimensional real-value vector. This process is called vectorization and it captures some of the underlying meaning. Given enough data, usage, and context, word embedding can make highly accurate guesses about the meaning of a particular word. Embedding can be equivalently treated as a word first represented as a single heat vector and multiplied by a lookup table [5]. In the model, all words are vectorized first using their embedding.

2.1 Intentional representation

User intent is specified as a keyword term, and each term includes one or more characters. First, a multi-layer convolutional neural network or a recurrent neural network is used to capture the meaning of keyword terms; then, the collection layer can integrate representations into different terms as a way of semantic composition [6].

we let a term k have |k| characters, $c_1, \cdots, c_{|k|}$. A convolutional neural network

(CNN, Figure 2.a) applies a fifixed-size window to extract local (neighboring) patterns of successive characters. Suppose the window is of size t, the detected features at a certain position $x_i, \cdots, x_{i+t-1}$ is given by

$$y_i = f(W[x_i;\cdots;x_{i+t-1}]+b)$$

Where x is the vector representation of the character (ie, embedded). W and b are the parameters of the convolution. A semicolon indicates a column vector connection. f(·) is a nonlinear activation function, and we use the Tanh function in our experiments. If the character does not have enough subsequent characters to fill the slot in the convolution window, zero is padded at the end of the term. In this way, we obtain a set of detected features y1,...,yn. Then, the largest pooling layer aggregates information of different characters into a fixed-size vector, namely:

$$y[j] = \max\{y_1[i],\cdots,y_n[i]\}$$

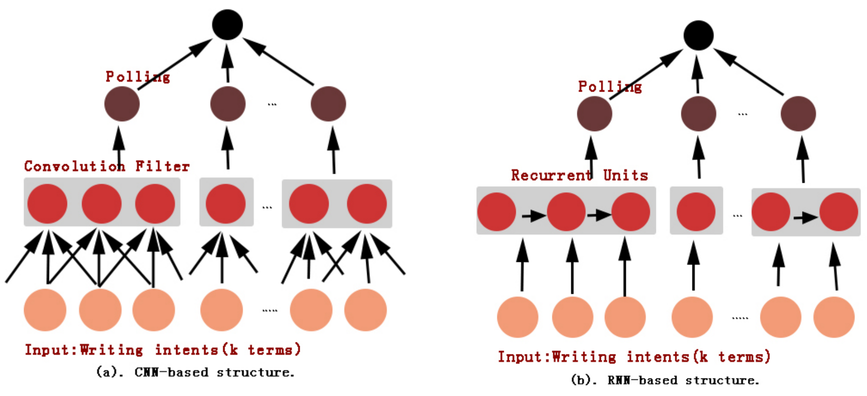
Polling
Convolution Filter
Input:Writing intents(k terms)
(a). CNN-based structure.
Polling
Recurrent Units
Input:Writing intents(k terms)
(b). RNN-based structure.

Figure 2: User intention representation structures

Alternatively, we can iteratively obtain the character sequence $x_i, \cdots, x_{i+t-1}$ using a recurrent neural network (RNN, Figure 2.b) information. For each character, the RNN assigns a hidden state h_i, which depends on the embedding x_i of the current character and the previous state h_{i-1}. Since the term usually contains 2-3 characters, which is actually not a very long sequence, it is sufficient for us to use a vanilla RNN with basic interaction, ie.,

$$h_i = f(w_h h_{i-1} + w_x x_i + b)$$

We also use the largest pool of all hidden states in CNN; this is usually more efficient than using the last hidden state as a vector representation of the sequence.

RNN can handle sentence boundaries smoothly, rather than CNNs that require zero padding; therefore, we can expect RNN to be more robust in this case. We will verify the performance of the two structures in the experimental part. For unification, we use h_{in} to record the encoding of the write intent given by the CNN or RNN.

2.2 Sequential generation

After representing the user's intent as a fixed-size vector, we provide it for a layered natural language generator for poetry synthesis.

The global level RNN (the black circle on the right side of Figure 1) captures the global information representation and results in the generation of a line of the poem. The starting hidden vector of the global RNN chain is given by the user intent and then changed as the RNN generates a new row. Let $h_i^{(hight)}$ be its hidden vector, we have:

$$h_i^{(global)} = f(w_h h^{(local)} + w_h h_{i-1}^{(global)})$$

Here $h^{(local)}$ is the last hidden state of the low-level RNN (gray circles in Figure 1) in the lower hierarchy, which serves as the sentence generator. Each hidden state controls the generation of one character. Its input is the word embedding of the previous character, augmented with the global information vector (black circle in Figure 1),namely $h^{(global)}$, as a static attention mechanism. The output is a softmax classifier predicting the probability of a certain character at the current step. Formally, we give the formula for computing the hidden layer as follows.

$$h_i^{(local)} = f(w_x w_{i-1} + w_g h^{(global)} + w_h h_{i-1}^{(local)})$$

2.3 Iterative polishing

In the process of writing, human beings usually need to modify the touches repeatedly. Inspired by this move, this paper proposes a multi-automatic modification model based on the iterative polishing mechanism proposed by R. Yan et al. In particular, the model should know the generated poem itself after a generation. Therefore, we consider the hidden state of the global-level RNN (corresponding to the last row) as the "subject" of the overall semantic representation of the poem, which functions like the user's intention. At the same time, an original written intent representation is provided to further mix the information during each iteration. This will not deviate from the original writing intention.

$$\mathrm{h}_0^{(global)} = \begin{cases} f\,(\mathrm{W}_i h_{in}) & (1stiteration) \\ f\,(\mathrm{W}_i h_{in} + \mathrm{W}_h h^{(global)}) & (Po;ishings) \end{cases}$$

Where h^{global} is the last global information representation in the RNN chain during the last iteration. Mix the generated poetry performance with the original writing intent as the initial global state for each polishing iteration. The stopping criteria are as follows.

After each iteration, we have the main point representation of the entire generated poem h^{global}. The algorithm iteration is stopped when the cosine similarity between two h^{global} from two successive iterations exceeds a threshold.

After 5 iterations, we will submit the generated poems to the relevant Chinese and English literature for evaluation, and at the same time give revisions, and then re-train the 5th generation according to the revised opinions.

3. Experiments

3.1 Data set.

The ancient Chinese poems collected by the author are the Tang poems included in the whole Tang poems. Through the library's "Full Tang Poetry", OCR (OpticalCharacter Recognition) technology is used to scan the paper version of the whole Tang poem into the computer to form an electronic version of the data set for use. The entire Tang poetry data set includes 49,403 ancient poems, and the data format is “Ancient Poetry: Ancient Poetry Content”.

3.2 Data processing.

For ancient poetry data, since the data set includes many residual sentences, it is necessary to use an algorithm to remove the endgame. Optical Character Recognition (OCR) is the use of optical technology and computer technology to scan and recognize the text in the paper, and convert it into a computer acceptable. The understandable format, due to the OCR technology, has a certain recognition garbled rate. Therefore, data preprocessing is needed. The purpose of preprocessing is to improve the accuracy and readability of ancient poetry generation. Therefore, the first step of preprocessing needs to denoise the 50,000 ancient poetry data in the data set. Special characters such as "_", "(", ")", "", "", "[", "]" and old poems that are too long and too short.

Extract the pre-processed ancient poems, count the number of texts that have appeared, and sort according to the number of occurrences, select the extracted words, and remove the ten words with the least number of occurrences in order to reduce the occurrence of uncommon words. Finally, the readability of the ancient poems is generated. Finally, these words are mapped to digital IDs to form a dictionary, so that the ancient poems are mapped into a string. In fact, the main purpose of data preprocessing is to establish a numerical representation of characters, because Chinese characters cannot be directly imported into the neural network for training, so you need to use different numbers to represent different characters.

3.3 Training.

The goal of the training is to predict the cross entropy error of the character distribution and the actual character distribution in our corpus. An L2 regularization item is also added to the target. The training model is propagated back through time, and the length is the time step. Minimize the target by random gradient descent. During training, the cross-entropy error of the output propagates back to the write intent through all hidden layers.

In this paper, we use 128-dimensional word embedded through vectorization and randomly initialize and learn during training. We use the ReLU function as an activation function in the neural network. Since the data set is Chinese, we split the standard Chinese into characters. We set the width of the convolutional decimal to 3. In order to improve the network, we use random gradient descent and mixed small batches for optimization. The gradient is provided by standard backpropagation. The initial learning rate is set to 0.8 and the multiplication learning rate is applied. We use the validation set to stop early.

4. Conclusion

4.1 Evaluation

The effects of computer-generated poems are often not well evaluated. We evaluate the experimental results from two different evaluation indicators:

(1) Perplexity (PPL): PPL is an indicator used to measure the quality of a language model in the field of natural language processing (NLP). It mainly estimates the probability of a sentence appearing according to each word, and uses the length of the sentence as normalize. The formula is:

$$\begin{aligned} pp(S) &= p(w_1 w_2 \cdots w_N)^{-\frac{1}{N}} \\ &= \sqrt[N]{\frac{1}{p(w_1 w_2 \cdots w_N)}} \\ &= \sqrt[N]{\prod_{i=1}^{N} \frac{1}{p(w_i \mid w_1 w_2 \cdots w_N)}} \end{aligned}$$

S stands for the sentence, N is the length of the sentence, and p(wi) is the probability of the ith word. The first word is p(w1|w0), and w0 is the start of the sentence, which is a placeholder.

The smaller the PPL, the larger the p(wi), and the higher the probability that the sentence we expect will appear. Perplexity can be thought of as the average branch factor, which indicates how many choices can be made when predicting the next word. The PPL drops to 90. It can be understood as the case that the next word has a reasonable choice of 90 when the model generates a sentence. The fewer the number of optional words, the more accurate the model is. That is, the smaller the PPL, the better the model. In this sense, the perplexity of poetry is getting lower and lower, the purer performance is better, and the generated poetry may be better.

(2) Human evaluation. The evaluator is asked to comment on the automatically written poems. Human evaluation requires clear criteria. We use four indicators: “fluency”, “liquidity”, “liquidity” and “liquidity” to evaluate. The Faculty of Arts and the students were invited to assist in the assessment. The evaluator only needs to assign 0-1 points according to the four criteria ("No", "Yes"). After that, the total score of poetry is calculated by summarizing four separate scores, ranging from 0 to 4 in the 5-point scale. The evaluation process is conducted as a blind review.

4.2 Conclusion

Poetry creation is a difficult task in natural language generation. This paper needs to modify this reality by starting from human writing. Based on the iterative polishing mechanism proposed by Rui Yan et al., this paper offers a more perfect poetry evaluation system and presents intelligent writing based on the system. In the future, the trial will include more poetic features into the process of production and continually refine the Model.

References

[1] T. Mikolov, M. Karafifiat, L. Burget, J. Cernock y, and S. Khudanpur. Recurrent neural network based language model. In INTERSPEECH, pages 1045–1048, 2010.
[2] Zhang, X., and Lapata, M. 2014. Chinese poetry generation with recurrent neural networks. In EMNLP, 670–680.
[3] L. Yu, W. Zhang, J. Wang, and Y. Yu. Seqgan: Sequence generative adversarial nets with policy gradient. arXiv preprint arXiv:1609.05473, 2016. 2
[4] R. Yan. i, poet: Automatic poetry composition through recurrent neural networks with iterative polishing schema. In IJCAI, 2016.
[5] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Effificient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.
[6] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. In NIPS, pages 2042–2050, 2014.