**Ancient Chinese Poem Generator**

Shengyu Chen 1568011188 [shengyuc@usc.edu](mailto:shengyuc@usc.edu)

Shuai Zhou 4264203725 [zhou990@usc.edu](mailto:zhou990@usc.edu)

Yizhao He 4256071445 [yizhaohe@usc.edu](mailto:yizhaohe@usc.edu)

Zichen Yang 8236922562 [zichenya@usc.edu](mailto:zichenya@usc.edu)

# **ABSTRACT**

Ancient Chinese poetry generation is a challenging and interesting task in natural language processing. In this paper, we will train a multi-layer Recurrent Neural Network(RNN) with Long Short-Term Memory (LSTM) model to generate ancient Chinese poem.

# **INTRODUCTION**

Ancient Chinese poem, as one of the most famous cultural heritage, has been applied widely to express author’s feelings, such as homesickness, friendship, etc. The golden age of ancient Chinese poetry was in Tang dynasty, which was the most prosperous dynasty in Chinese history.

There are some fixed rules for ancient Chinese poetry written during Tang dynasty. Basically, there are either four or eight sentences in a poem and five or seven Chinese characters in each sentence. In this paper, we will focus on the most popular poetry style, which consists of four sentences with five characters in each sentence. Of course, in order to sound comfortable when reading the poem, there are restrictions on rhyme, like Ping (the level tone) or Ze (the downward tone). Based on these requirements, we collected 16,000 poems written during Tang dynasty as our dataset.

With the development and rising popularity of the deep learning, poetry generation have been a hot topic. There are a lot of valuable resources and methods based on previous research[1-4]. These methods usually generate the first line by selecting one line from the dataset of poems according to the user’s writing intents, and the other three lines are generated based on the first line and the previous lines. Here, we use the similar way to generate our poem. However, the creativity in our project is that we integrate acrostic poem, which means besides the first sentence, we will give first character of the other three sentences to generate the remaining part of the poem. Because acrostic poem is a special genre, every first character in each sentence can be formed in sequence to cleverly present another idea, which makes this poem seems fantastic and highlights the author’s capability.

The rest of this paper is organized as follows. Section 2 we will talk about preprocessing data. And section 3 specifically describe the generation of ancient Chinese poetry. Then section 4 discusses evaluation. At last, section 5 concludes the paper.

# **METHOD**

* 1. Materials

The source corpus of ancient Chinese poetry is collected from Jackie Gao's GitHub repository. The whole source data consists approximately fifty-five thousand Tang poems of ten thousand famous poets in total. For each poem, the basic information of author, paragraph and tone is well structured in the corpus.

* 1. Procedure

Basically, to train our model, we first preprocess and clean the training data. Then a multi-layer LSTM is trained to generate Tang style Chinese poems. We apply tone pattern constraints on the generating procedure to make poems more human-written alike.

* + 1. *Data preprocessing and cleaning*

We store each poem as a dictionary (python data type) in which three keys represent paragraph, tone and title respectively. For tone representation, ‘p’ stands for the level tone (平) and ‘z’ stands for the downward tone (仄). Since we only need five-character poems for training, we filter out all the poems whose format is not classified as ancient Chinese five-character poems. For simplicity, we convert traditional Chinese to simplified Chinese, remove all the commas and periods and replace them with ‘$’ to represent end of sentence(<*EOS*>). To obtain high-quality data, we need to remove incorrect records and unnecessary symbols in the data set. In order to save as much efforts as possible and at the same time obtain larger training corpus, we come up with a filtering strategy. The strategy is to remove any poem whose contents are considered as incompletion or format is not correct and modify the poem which contains extra unrecognized characters. Before applying the strategy, we obtain around 28,000 poems from the source corpus. After applying, only 16,000 effective poems with correct tone and complete paragraph are selected for training.

* + 1. *Ancient Chinese poem generator*

Thanks to LSTM’s ability of modelling long-term dependencies[5], we use a rule-based multi-layer LSTM to generate the following sentences of the poem. The structure of this generator is shown in Fig.1. This neural network is basically a sequence-to-character model, which models probability of a character given the preceding *n-*character sequence:

We use one-hot encoding to represent each Chinese character so that the model can be a generative model which outputs the probabilities of all the candidates. This makes it easier to apply tone pattern constraints on the poem, which will be discussed later.

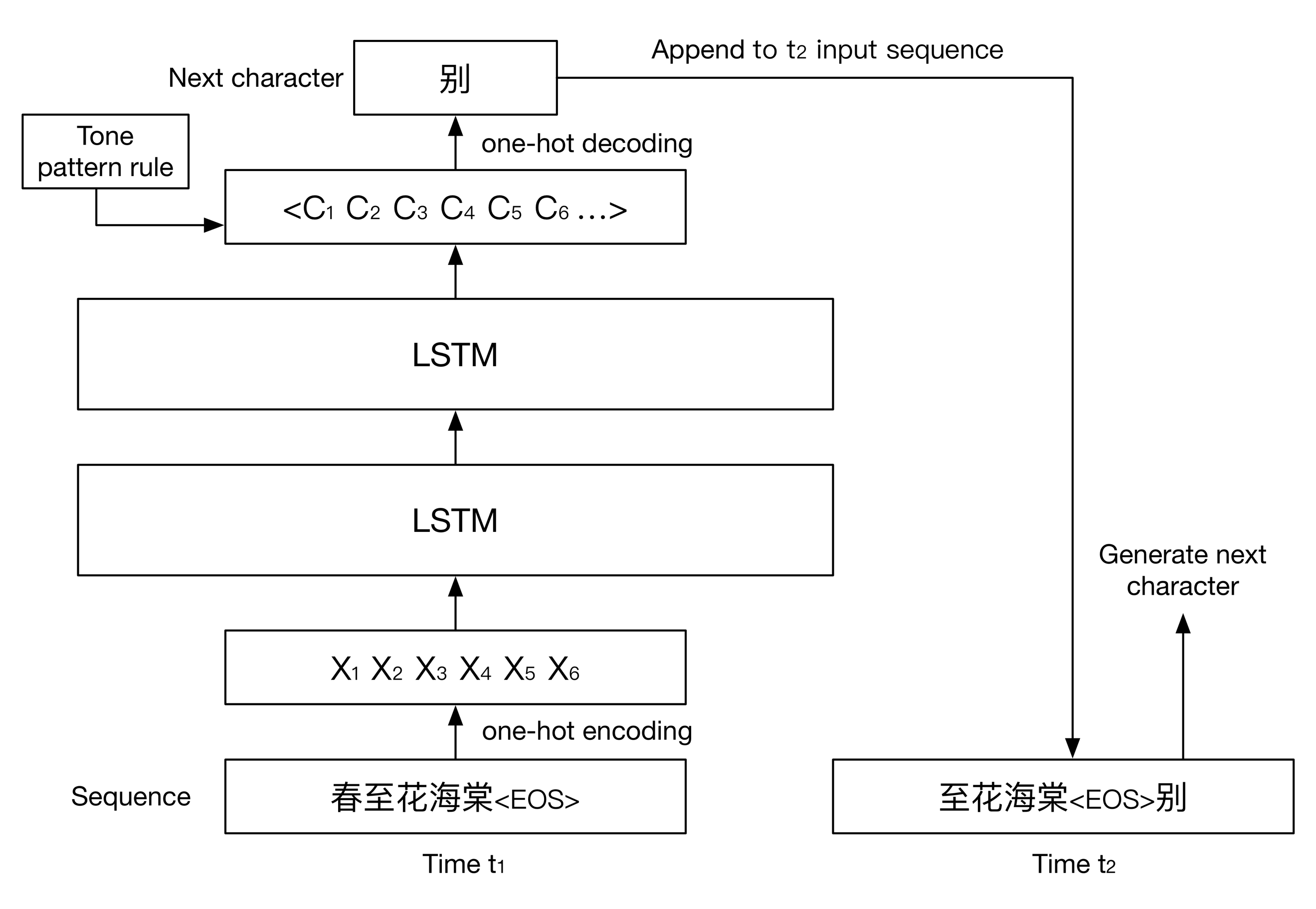


Fig. 1 Structure of following sentences generator with two layers LSTM

(Input sequence means “Spring comes, begonia blossoms”. Output character means “Depart”)

*Training*

All processed data containing about 16,000 5-character ancient Chinese poems written in Tang dynasty is used on training the model. We tried different number of layers and memory units to get the best result. Other configurations and hyper parameters are fixed: we use *softmax* as the activation function; loss is calculated through cross entropy; *Adam*[6] is used to optimize the learning rate; dropout is 0.2; sequence length is 6 (for better estimating each 5-character sentence since we also encode *<EOS>* as a character); batch size is 128.

*Generating*

To generate the following sentences given the first sentence which is also a 6-character sequence ended with *<EOS>*, we simply input the first sentence and use the most likely character as the next character and recursively generate the whole poem:

To make the poem more human-written alike, there are some rules we need to consider. First, we want the model learn to generate poem in rhyme, so we encode *<EOS>* to represent the end of sentence. However, *<EOS>* should never be output in the middle of each sentence, and must be output at the end. The other thing is we want the poems to be in correct tone patterns. There are four different tone pattern templates for 5-character ancient Chinese poem. Under this constraint, the generator will choose the most likely character that also follow the tone pattern rule.

* + 1. *Poem generator user interface*

In order to provide a way for normal users to access our poem generator and visualize poem generation results, we craft an elegant user interface.  Procedures of our poem generator are as follows: Firstly, gather user input, including initial characters for each sentence (optional) and the first sentence.  When the user clicks the submit button, a GET request will be sent to our server. A Python script will be invoked to load the model and also vocabulary dictionary to generate the poem.  After receiving the JSON object that wrapping the result, our UI will display the poem sentence by sentence, word by word.



Fig. 2 User interface

* 1. Evaluation

*2.3.1 Evaluation Metrics*

Since evaluation of automatically generated poetry is extremely difficult, we prefer to evaluate them manually. We decide to invite ten Chinese people who have a comprehensive and authoritative knowledge of Chinese poetry. The experts will rate the output poems by our model using 1-10 scale with the higher score the closer to human written poems based on three evaluation standards in following general grading guidelines.

General grading guidelines:

(a) Fluent expression: ideas succinctly stated; proper mechanics

(b) Meaningfulness: have meaning, function, or purpose

(c) Coherence: structural, phonological, and semantic requirements are satisfied

*2.3.2 Experiment*

We implemented several poetry generation methods as **baselines** and employed the same pre-processing method for all the methods:

1.model1

2.model2

We performed two experiments:

In the first experiment, the first step is to find ten five-character poems from Tang dynasty which are not in our training corpus. Then generates ten five-character poems on each model given the first sentence and first character of each sentence from the unseen ten poems.

In the second experiment, we generated the poems only using the first sentence from the unseen ten poems.

All generated poems are rated by ten human evaluators and the final score will be the average. To eliminate bias, it's necessary to remove outliers that will significantly affect the final score.

1. **RESULTS**
2. **DISCUSSION**

**REFERENCES**

[1] Xingxing Zhang and Mirella Lapata. 2014. Chinese poetry generation with recurrent neural networks. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 670–680, Doha, Qatar, October. Association for Computational Linguistics.

[2] Xiaoyuan Yi, Ruoyu Li, and Maosong Sun. 2016. Generating chinese classical poems with rnn encoder-decoder. CoRR, abs/1604.01537.

[3] Qixin Wang, Tianyi Luo, Dong Wang, and Chao Xing. 2016. Chinese song iambics generation with neural attention-based model. CoRR, abs/1604.06274.

[4] Zhe Wang, Wei He, Hua Wu, Haiyang Wu, Wei Li, Haifeng Wang, Enhong Chen. Chinese Poetry Generation with Planning based Neural Network. arXiv:1610.09889.

[5] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.

[6] Kingma, D. P., & Ba, J. L. (2015). Adam: a Method for Stochastic Optimization. International Conference on Learning Representations, 1–13.

**DIVISION OF LABOR**

**WORD COUNT**