CSCI 544 Project Proposal: The Exploration of Personalized Poetry Generation Techniques

Haidan Tang haidanta@usc.edu
Jie Zhang jzhang03@usc.edu
Siyi Tang siyitang@usc.edu
Qianru Hou qianruho@usc.edu
Xiao Chen xchen533@usc.edu

1 Introduction

1.1 Background

From ancient Egyptian hieroglyphs to contemporary news articles, it is indisputable that literature has always been and will continue to be a crucial part of our lives. Thinking of literature, everyone has the desire to feel the cultural connotations, to experience the sensational changes, and to reflect on our lives. We hope to use deep learning strategy and statistical methods to dive deeper into this field.

1.2 Project Overview

Without doubt, the classification of sentiments, rhymes and opinions of various text has attracted the attention of experts is a fundamental task of natural language processing. In our project, we focus on building a natural language processing application about poem literature analysis and content generation. Our data is collected from Kaggle. Our primary aim is to effectively train some machine learning models to conduct poem genre classification task and poem generation applications. On the one hand, given a input poem, the system can classify it into one of the common poem category. On the other hand, the system can automatically generate poem based on users' customized requests. Based on our selected dataset, we will apply statistical analysis to tune and evaluate our model.

Besides, for better demonstration purpose and user experience, we plan to design a front-end website with an interactive panel to demonstrate the overall literature trend and pattern development for this field. We will explore detailed relationships between different entities and attributes.

In our application, we support flexible search and filter functionality over multiple options with built-in text auto-complete systems. Finally, we plan to build a question answering(QA) system to provide users with customized results and help them better explore this literature domain.

1.3 Dataset

Here are some Kaggle dataset candidates that we consider to use.

- kaggle.com/michaelarman/poem-generation-with-transformers/data
- kaggle.com/ultrajack/poetry-analysis-using-ai-machine-learning/data

2 Related Work and Review

Apparently, our project is composed of three parts: poem representation learning, poem classification and poem generation. Before we dive deeper into different implementations. We need to understand the characteristics of a poem.

2.1 Poetic Property

According to Gonçalo Oliveira, poetic text must hold the following three properties [1]:

- **Meaningfulness:** Covey a conceptual message, which is meaningful under some interpretation.
- **Grammaticality:** Obey linguistic conventions prescribed by a given grammar and lexicon.
- **Poeticness:** Exhibit poetic features.

We consider constructing a numerical evaluation system based on these three properties when comes to evaluate our generated poem.

2.2 Text Representation

Word2Vec and **GloVe** can serve as word-level methods to learn the static poem representations [4]. For dynamic representations, **ELMo** utilize bidirectional information for the contextual representation. Another bidirectional transformer-based language model is **BERT**, trained on masked token prediction and more efficient on long-term word dependency problems [5]. **GPT-2**, another transformer-based language model developed by **OpenAI** utilizes unsupervised learning methods on a large quantity of data during training, during which it looks for patterns within a set of data [2]. Tranformer-based language models typically use byte-pair encoding as their tokenization method. Using character-based tokenization might cause slow training speed and other case formatting issue. In the case of byte-pair encoding, the inputs are compressed to the shortest combination of bytes without losing format information [3]. This method can effectively generate tokens for combinations of characters that frequently occur together, which matches our interests for poem generation.

2.3 Poem Classification

Text classification requires that text be represented as vectors of statistical features. Some previous works categorize the poems using Support Vector Machine with a Radius Basic Function (RBF) and linear kernel function [6]. According to Jamal, Mohd and Noah's work in 2012, the classification work involves three processes, which are: Document representation, feature selection and transformation to SVM data format [6]. In their study, they conclude that applying methods to semantically identify poetry according to themes can improve the quality of poetry retrieval or even finding suitable usage of poetry according to certain contextual situations [6]. We find a great connection between this insight and our project goal, so we will extend this idea and implement our own classification algorithm.

2.4 Poem Generation

Poem generation is another hard but popular task many researchers have tried to deal with. Some of the previous works apply RNN-based (**GRU**, **LSTM**, etc.) encoder-decoder framework which can solve the task to a certain extent [7]. According to Chang, Xie and Rastogi's work on Shakespearean Sonnet Generation, models that combine word and character level information, such as a **Gated LSTM** and a **CNN-based LSTM**, significantly outperform the **baseline word-LSTM** and **char-LSTM** models [7].

Besides, instead of handling the classification and generation tasks separately, a group of scholars at Sichuan Univeristy proposed a **BERT**-based model that can classify the poems as well as generating a poem given some keywords [8]. In that study, they focus on ancient Chinese poem classification and evaluation. In terms of evaluation metrics, it involves much of human effort to evaluate both syntactic and semantic aspects [8]. Even though human effort in the evaluation process of a deep learning model development process, we hope to design a semi-automatic or automatic evaluation system to help us fine-tune our model.

When evaluating the classification task, we can simply use accuracy, precision, recall and F-1 scores as metrics. However, for the generation task, we need more complicated standards. As mentioned in class, perplexity can serve the purpose; that is the model with a lower perplexity is better. **BLEU-2** is another choice for this problem applying the N-gram matching rules.

3 Technical Challenges

3.1 Data Acquisition

For data acquisition, we plan to use Kaggle to find relevant structured text-based dataset and select suitable candidate corpus as our training dataset. For instance, a possible candidate corpus may contain poems composed by varied authors of different genres, and these poems are classified based on their content tags, like alone, beauty, Chicago.

3.2 Tag Summarization

The challenge now turns to the question, how should we make use of these mini tags. It is inappropriate to classify poem in over 100 detailed tags, as we may lose the insight to discover the underlying pattern of these poem categories. Therefore, given the fact that the overall goal of tag summarization is to describe the content of each poem as precisely, we plan to concatenate these mini tags into some general tag categories to improve accuracy.

3.3 Text Phrase Normalization

After we collected the data from different sources, we will conduct data processing step to normalize these string attributes to help improve the algorithm performance. We plan to convert all reviews into lower-case and remove URL, HTML, non-alphabetical and extra spaces. We also need to make sure that our data is stored in a consistent format. For abbreviation problem, we extracted the initial letter of each word and combined them as acronyms for the future string comparison during entity linking. For multilingual problem, we used *Unidecode*, a Python library, to format each text piece into English letters. There are different types of poem format, and we expect them differ in terms of length, consistency and grammar (it may not be a complete sentence). Furthermore, we also need to consider whether we should keep or highlight the core words to strengthen its feature parameters.

3.4 Recommendation system

Based on the user searching record, we can build a recommendation system that recommends several poems that we think the user could find interesting. Given the nature of this project, we may not have access to a large amount of users' poem searching records. For the recommendation system, this is usually called a cold start challenge.

With the challenge that we mentioned above: we think a feasible and reasonable solution for our project would be an item-based collaborative filtering recommendation system. This IBCF recommendation system was first introduced by Amazon to deal with recommending items for large numbers of users who have no previous purchase record.

The intuitive working procedures for IBCF are as follows: when a user searches one poem, the item-based CF will calculate the similarities between the searched poem and other poems. Then it will sort the similarity table and return several poems that are most similar to the one searched by the user.

3.5 Search and Filter Functionality

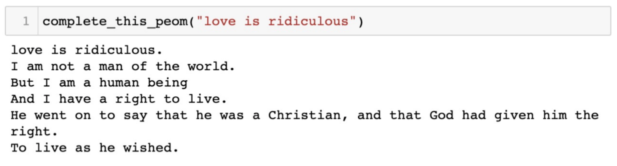
We want to make our application simple but powerful. Our goal was to support flexible search and filter functionalities based on our database, so that users can easily find what they are looking for when they have certain preferences. As for search functionality, we plan to implement search system that allows users to use keywords to search for the matched poems. When a user searches some context from our system, we first tokenize and lemmatize the context, and convert it to some keywords, and then search most matched items from our database. We also plans to support filter functionality. The user can find the poems by filtering with certain attributes. Each poem has its published time, type of poem, and author. The user may browse a list of poems that fit into the corresponding filtering conditions. Finally, to enhance our search and filter functionalities, we also constructed a text auto-complete system based on user input for better user experience.

3.6 Poem Classification

The first part of our project is to classify the category for a given piece of poem. Based on our exploration, poems can be categorized based on philosophical concept, experience, emotion, interpretation and human understanding, etc. Depending on our proposed model and features that we hope to represent, to enable the model to learn effectively, it is necessary to choose an appropriate way to transform each word and represent it using an embedding vector. After feature representation process, the proposed model will extract the information from the embedding layer. Another factor that we need to consider is context information, as we may hope to develop a system capable of retaining contextual information which can interpret useful tokens. Eventually, we believe that this classification task could help us to better understand poetry construction and provide users with more relevant, customized recommendation in our application.

3.7 Poem Generation

An important aim of our project is to study, analyze and perform sentence completion tasks. And we plan to apply sentence completion methods to design functions, including generating a poem randomly and completing a poem with given words. After literature review, we know that there are many methods to perform sentence completion, like **Gated LSTM**, **CNN-based LSTM**, **RNN-based LSTM** and **BERT-based models**, etc. We will compare and then decide which of them performs better considering the scale of the data. In this process, we can encode the poem as a dimensional matrix and then perform our forward pass using the model which computes what it thinks the most likely next character is and uses this prediction to compute the loss. We can adjust parameters during the test process to get the best model with the highest accuracy. Finally we can finish this task by generating the whole poem using given words. The function we want to complete like the following:



```
complete_this_peom("love is ridiculous")
```

```
love is ridiculous.
I am not a man of the world.
But I am a human being
And I have a right to live.
He went on to say that he was a Christian, and that God had given him the
right.
To live as he wished.
```

Eventually, we aim to complete the following functionalities:

- Given a random beginning sentence, we will generate a complete poem based on this starting point.
- Given a random text phrase, we will generate a complete acrostic poem based on this phrase.
- Given a random tag category, within or not within our existing category set, we will generate a complete poem based on this topic.

4 Stretching Goals

In addition to these main goals, we also have some stretching goals that we hope to complete in the future. We plan to add a question answering system to enhance the search functionality. We may also extend our project domain to cover more literature genres, such as lyrics, short story, news report, etc. Besides, we plan to conduct emotion detection and sentiment analysis to offer a broader perspective when appreciating literature. We also hope to provide multilingual supports that cover other languages like Chinese, Spanish and Hindi. Different language may have different grammar rules, so it may require us to further discover different language's grammar, formats and other representative characteristics to build a general poem model which is compatible with different languages.

References

[1] Hugo Gonçalo Oliveira. (2009). Automatic generation of poetry: an overview.

[2] Scott Duda. (2020). Generating an Edgar Allan Poe-Styled Poem Using GPT-2

[3] Max Woolf. (2019). How To Make Custom AI-Generated Text With GPT-2.

[4] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. (2014). GloVe: Global Vectors for Word Representation.

[5] Andreas Pogiatzis. (2019). NLP: Contextualized word embeddings from BERT.

[6] Noraini Jamal, Masnizah Mohd and Shahrul Azman Noah Knowledge Technology Research Group, Faculty of Information Science and Technology. (2012). Poetry Classification Using Support Vector Machines

[7] Max Chang, Stanley Xie, Ruchir Rastogi, Department of Computer Science, Stanford University. Deep Poetry: Word-Level and Character-Level Language Models for Shakespearean Sonnet Generation

[8] Huishuang Tian, Kexin Yang, Dayiheng Liu, Jiancheng Lv, College of Computer Science, Sichuan University. (2021). AnchiBERT: A Pre-Trained Model for Ancient Chinese Language Understanding and Generation