Write like a classical poet

Shuimu Zeng
201677429

Statement of ethical compliance

In this project, all the data are derived from the works of great poets which are out of British copyright. The outcome of the project will be evaluated by proper specialist from the English department. Considering above, the data category falls in the category A and the human participant category falls in the category 2. In conclusion, the final category will be A2.

Project description

The ultimate goal of this project is to imitate the work of classical poets. The approach will be realized by using the text analysis techniques namely stylometric analysis. An algorithm will be build to receive works of a poet and make analysis. After the analysis, texts will be generated based on the language model that we obtain.The generated texts will be in the similar style as the poet. For evaluation of results, human specialists will comment on them. Other scientific techniques can also be applied.

Aims&Requirements

There are aims for the project:

1. **Understand the n-grams language model**
2. **Successfully imitate a poet**

Requirements

1. Produce a system building on the n-gram language model that generates verse in the style of a classical poet.

2. To identify the deficiencies in the output and discuss/develop which methods that can improve this.

Key literature&background reading

The main method that we apply in the project according to the requirement is the n-grams model. The n-grams model appears to be a so-called probability machine. "Probabilities are essential in any task in which we have to identify words in noisy, ambiguous input, like speech recognition" (Jurafsky & Martin, 2018). As the model can recognize written patterns, it can also generate texts based on its analysis of certain patterns. The early design of the n-grams method looks for predicting the next word using the probability of the previous sequence. However, the probability of the entire previous sequence is not easily calculated. "One thing we can do is decompose this probability using the chain rule of probability" (Jurafsky & Martin, 2018). As we can see, the application of n-grams model with a really large n seems to be not a proper way. So, we turn to approximate the previous sequence probability with a more manageable approach called the Markov chain. "Markov models are the class of probabilistic models that assume we can predict the probability of some future unit without looking too far into the past" (Jurafsky & Martin, 2018). We can generalize the bigram (which looks one word into the past) to the trigram (which looks two words into the past) and thus to the n-gram (which looks n−1 words into the past) (Jurafsky & Martin, 2018).

After the text-generation process, we need evaluation of the results. We have both human and scientific evaluation. "We may ask English expertise to compare the result texts with texts of the original author and tell if they are similar enough. For scientific evaluation, the best way to evaluate the performance of a language model is to embed it in an application and measure how much the application improves. Such end-to-end evaluation is called extrinsic evaluation" (Jurafsky & Martin, 2018). We may use perplexity instead of raw probabilities because perplexity is a more interpretable and user-friendly metric. "It helps us understand how well a language model predicts text by considering the average uncertainty or surprise of the model's predictions. Lower perplexity values indicate better language modeling, making it a valuable metric for model evaluation and comparison" (Jurafsky & Martin, 2018). To visualize the language model, we may also apply the sampling techniques. "Sampling from a distribution means to choose random points according to their likelihood. Thus sampling from a language model—which represents a distribution over sentences—means to generate some sentences, choosing each sentence according to its likelihood as defined by the model" (Jurafsky & Martin, 2018). Because the project aims to imitate the existing style of poets, we don’t consider how we deal with unknown words (words that don’t exist in the input text). Using the words in the input datasets can work according to our estimation. Having an overview of the background, we can start from what we know and add new things according to our observation during the process.

Development and implementation summary

For the efficiency of introducing new methods. I choose the google colab a the main development environment. Google Colab runs entirely in the cloud, which means I don't need to install any software or configure development tools on my local machine. The regular access to the environment is through a web browser.The project will be implemented on python version 3 because Google Colab is primarily designed for Python development. It supports popular Python libraries and frameworks, including TensorFlow, PyTorch, NumPy, pandas, scikit-learn, and NLPTK which is the main package that I will use for this project.

Creating poetry or text that imitates a poet's work using n-grams is an interesting natural language processing (NLP) project. N-grams are essentially contiguous sequences of n items from a given sample of text or speech. To use n-grams to realize the project, I follow these steps:

Data Collection:

Obtain a substantial corpus of the poet's work. This could be their poems, essays, or any written material you want to mimic.

Preprocessing: Clean the text data by removing any unnecessary characters, converting text to lowercase.

N-gram Generation: Generate n-grams from the preprocessed text. An n-gram could be a unigram (single word), bigram (two words), trigram (three words), or higher order n-grams.

Frequency Analysis: Calculate the frequency of each n-gram in the poet's work. This helps you understand which n-grams are more common and, therefore, more likely to be used in the imitation.

Language Model: Create a language model or a probability distribution based on the n-grams. This model should allow me to predict the next word or phrase based on the previous n-grams.

Text Generation: Use the language model to generate new text. Start with a seed phrase or word and generate the next word or phrase based on the probabilities learned from the poet's work. Continue generating text until I reach the desired length.

Iterative Improvement: I might need to experiment with different n-gram sizes, language models, and techniques like smoothing to improve the quality of the generated text. Some techniques like Markov Chains are commonly used for text generation.

Evaluation: Evaluate the generated text against the poet's work. Metrics like perplexity and human evaluation can help assess how closely the generated text resembles the poet's style.

Refinement: Fine-tune the model and generation process based on the evaluation results. This may involve adjusting the n-gram size, modifying the language model, or expanding datasets.

User Interface (Optional): We need to create an interactive tool, we can develop a user interface where users can input a seed phrase and generate text that imitates the poet's style.

Deployment: The deployment is not so much considered in this project because the algorithms is mainly for scientific usage.

Data sources

For the project of imitating a poet's work based on n-grams analysis, I will primarily be using publicly available text data from the Gutenberg Project, which offers a wide range of literary works, including poems and written texts from various poets. The Gutenberg Project provides its content in the public domain, making it a valuable resource for text analysis and natural language processing.

To access the data, I will retrieve text files from the Gutenberg Project's website, which allows free access to their collection. I will ensure that all data used in this project is obtained legally and with permission, as the texts provided by the Gutenberg Project are in the public domain.

Regarding confidentiality and anonymity, the project's focus is on text analysis and generation, so it will not involve handling personal information or sensitive data. The analysis and generation process will be based solely on the textual content available in the public domain, and no personal information or confidential data will be collected, stored, or processed.

As the project does not involve personal data, there are no privacy or confidentiality concerns related to this specific use case. The primary goal is to leverage publicly available text data to create an imitation of a poet's work using n-grams analysis.

Testing&Evaluation

Human Evaluation:

Qualitative Assessment: A group of human evaluators, preferably individuals familiar with the poet's work, will read and assess the generated texts. They will provide subjective feedback on the quality, coherence, and style of the generated poems.

Scientific Evaluation (N-grams Analysis):

Perplexity Score: Perplexity is a common metric used to evaluate the performance of N-grams language models. Lower perplexity values indicate better model performance. By computing the perplexity of the N-grams model, we can assess how well it predicts the next word in a sequence.

Cross-Validation: The N-grams model can be subjected to cross-validation to

assess its generalization capabilities. By splitting the data into training and testing sets, we can evaluate how well the model performs on unseen data.

BLEU Score: BLEU (Bilingual Evaluation Understudy) is a metric used to evaluate the quality of machine-generated text. It measures the overlap of n-grams (phrases of varying lengths) between the generated text and reference text. A higher BLEU score indicates better text generation quality.

Comparison to Existing Work:

We can compare the output of our N-grams model to existing automated poetry generation methods and assess whether our approach outperforms or matches them in terms of quality and style similarity.

Human Feedback from Beta Testers:

If beta testers are involved, they will provide feedback on the overall user experience, interface usability, and quality of the generated poems.

Objective Criteria:

We may define objective criteria for text generation success, such as generating poems with the correct structure, rhyme scheme, or adherence to known literary rules.

Project Ethics & Human Participants

Since the N-grams text imitation project primarily involves human participants for the evaluation and comments on the generated results, we take them into account. However, they are not expected to use the algorithm

of the project. All they will do is to read the generated texts and compare them to the original author that the algorithm is imitating. The participants in the evaluation process will be kept anonymous unless they allow us to show there identity for certain purpose such as showing the identity of an English literature expertise for a prove of the accuracy of the evaluation.

BCS Project Criteria

Practical and Analytical Skills: The N-grams text imitation project leverages the practical and analytical skills acquired during the degree program. It involves natural language processing, machine learning, and algorithm implementation. The project will demonstrate the application of these skills to develop and improve text imitation models.

Innovation and Creativity: The project exhibits innovation and creativity in the field of text generation. By applying N-grams and Markov models to imitate the work of poets and generate text, it explores creative approaches to language modeling and text generation.

Synthesis of Information and Evaluation: The project involves the synthesis of linguistic information, ideas, and best practices in text generation and language modeling. It will provide a quality solution by developing effective N-grams and Markov models. The project will also critically evaluate the generated text to ensure quality and authenticity.

Meeting Real Needs: The project addresses the need for advanced text generation techniques. It has practical applications in creative writing, content generation, and language modeling, meeting the real needs of content creators, writers, and researchers.

Self-Management: The project is a significant piece of work that requires self-management. It involves multiple phases, including data collection, model development, testing, and evaluation. Project members will need to manage their time effectively to meet project milestones.

Critical Self-Evaluation: The project includes a critical self-evaluation process. Team members will assess the effectiveness of the text generation models and their ability to imitate poets' work accurately. This self-evaluation will lead to model improvements and project refinements.

In summary, the N-grams text imitation project not only aligns with the six BCS outcomes but also embodies the practical application of IT skills, creativity in language modeling, and critical evaluation of text generation models. It meets real needs by advancing text generation techniques and requires self-management to deliver a quality solution.

UI/UX Mockup

Since the project is primarily for scientific purposes and not intended for end-users, the user interface (UI) and user experience (UX) are minimal and functional. Users will not directly interact with the project; instead, they will receive the results of the N-grams analysis and generated texts. But basic descriptions will be given.

User Interface (UI):

The UI for this project will be minimal and command-line-based, given that the primary audience is researchers and individuals interested in the scientific aspects of automated poetry generation. The main components of the UI will include:
Command Line Interface: Users will interact with the project through a command-line interface (CLI). This CLI will allow users to input commands to initiate N-grams analysis and text generation processes.

User Experience (UX):

The UX for this project is straightforward and not interactive. Users won't engage with a graphical interface; instead, they will input specific commands to run the N-grams analysis and text generation. The interaction is akin to executing commands in a terminal or writing scripts. The focus is on providing access to N-grams analysis results and generated text with efficiency.

Project Plan

Task	Start Date	End Date	Duration
Background Reading	2023-09-31	2023-10-25	25 days
Detailed proposal	2023-10-25	2023-11-06	14 days
Discovering methods	2023-11-07	2023-11-21	14 days
Development	2023-11-22	2023-12-31	40 days
Testing	2023-12-31	2024-03-31	91 days
Evaluation	2023-12-31	2024-03-31	91 days
Refinement	2023-12-31	2024-03-31	91 days
Project video	2024-03-31	2023-04-19	19 days
Project Dissertation	2024-04-20	2023-05-10	20 days

Risk&Contingency Plans

Risks	Contingencies	Likelihood	Impact
Changes in Data Availability	Regularly update the datasets and adapt the model	Medium	Low

Algorithmic	**Regularly assess and mitigate biases in the model**	**Medium**	**Medium**
Technical issues	**Debugging, continuous monitoring, and version control**	**Medium**	**Low**
Development environment failure	**Store the copy of project properly**	**Low**	**High**

References

Jurafsky, D., & Martin, J. H. (2018). "Speech and Language Processing" (3rd ed.).