CIS530-Project-MS3

Akriti Gupta, Bhumika Singhal, Namita Shukla, Smruti Chourasia

December 2022

1 Introduction

For our baselines, the idea was to simply use character level information and pass it to our defined network. But another major idea behind Haikus is their constrained structure. So now along with character-level information we also wanted to add syllable information.

2 Baseline Extensions

As elaborated before we used FastPunct to clean and structure the data. In addition to this, we now used Phonemizers - which also helps us count syllables. Further, we also extracted the topic of the poem using KeyBert (we hope to use it as feature input).

2.1 CharRNN + Syllable Structure

As discussed in our MS2 results for our strong baseline (Char RNN), even though the output had good poems, they lacked the structural essence of Haikus. Hence, the key idea here is to extend the Character Level RNN network (our strong baseline) by feeding in the syllable structure information into it. The main idea behind the implementation is to encode the syllable structure information. For each line of the poem we encode the syllable information by passing the number of syllables in that line through a dense layer to form the hidden dimension representation. We add this into the hidden state and the cell state of the LSTM network. We then perform the character-level generation of the line using the updated states.

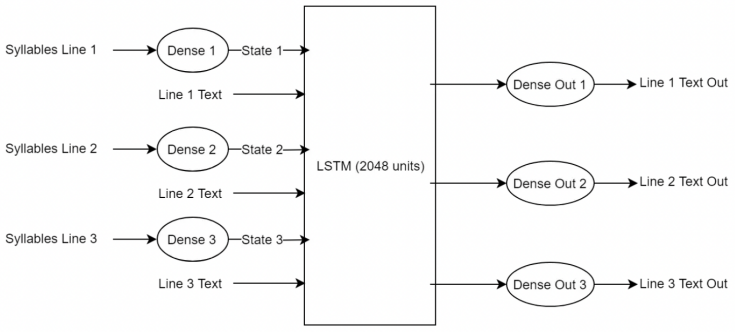

Figure 1: CharRNN Architecture + Syllable Information

Extending our explanation of CharRNN, we one hot encode the text line-wise in each poem. In a loop of three, we pass the tensors into our modified LSTMs. We first add the dense representation of the syllable count according to the line number and we get the output for each line.

```
CharSyllableRNN(
  (lstm): LSTM(67, 1024, batch_first=True, dropout=0.4)
  (feature): Sequential(
    (0): Linear(in_features=1, out_features=1024, bias=True)
    (1): ReLU()
  )
  (dropout): Dropout(p=0.4, inplace=False)
  (fc): Sequential(
    (0): Linear(in_features=1024, out_features=67, bias=True)
  )
)
```

Figure 2: Char Syllable RNN Architecture

The preliminary results are about an average GRUEN score of 0.348 with a median of 0.31. This has improved only a little from the CharRNN model. However, the main motto was to improve the structure of the poem, we can observe that the average number of syllables across line are -
Line 1: 4.96, Line 2: 7.56, and Line 3: 6.2.
In addition, the median of the distribution line wise is - Line 1: 5, Line 2: 7 and Line 3: 6 - showing the generated data learned the structure of the poems in a better way.

3 Next Steps

3.1 Fine tuning GPT-J

We are trying to fine tune GPT-J to be able to learn haiku poem structure. We also plan on sending in some extra features like topic of the poem (extracted during data cleaning) to help the model be coherent and structured.
Since retraining all 6 billion parameters becomes compute intensive and infeasible, we adopt the Low-Rank Adaptation, or LoRA algorithm which freezes the pretrained model weights and injects trainable rank decomposition matrices into each layer of the Transformer architecture, greatly reducing the number of trainable parameters for downstream tasks.