CS 7180 Special Topics in AI: Deep Learning
Northeastern University, Spring 2019

Assignment 2 (10 Points)

Due 11:59 p.m. Sunday, Feb 24, 2019

Submission Instructions

Please submit your assignment as a .zip archive with filename `LastnameFirstname.zip` (replacing `Lastname` with your last name and `Firstname` with your first name), containing a PDF of your assignment writeup **in the main directory** with filename `LastnameFirstname_1.pdf` using the provided LaTeX template. (`http://roseyu.com/CS7180/latex_template.tex`). Submit your code as Jupyter notebook .ipynb files or .py files, and **include any images generated by your code along with your answers in the solution .pdf file.** We ask that you use Python 3.6 for your code, and that you comment your code such that the TAs can follow along and run it without any issues. Failure to do so will result in a **1 point deduction**.

1 Poem Generation

William Shakespeare is perhaps the most famous poet and playwright of all time. Shakespeare is known for works such as Hamlet and his 154 sonnets, of which the most famous begins:

Shall I compare thee to a summer's day?
Thou art more lovely and more temperate:

Shakespeare's poems are nice for generative modeling because they follow a specific format, known as the Shakespearean (or English) sonnet.[1] Each sonnet is 14 lines, spread into 3 quatrains (section with 4 lines) followed by a couplet (section with 2 lines). The third quatrain is known as the *volta*[2] and has a change in tone or content. Shakespearean sonnets have a particular rhyme scheme, which is *abab cdcd efef gg*.
Shakespearean sonnets also follow a specific meter called *iambic pentameter*[3]. All lines are exactly 10 syllables long, and have a pattern of unstressed stress. For example, the famous Sonnet 22 begins:

Stress	x	\	x	\	x	\	x	\	x	\
Syllable	Shall	I	com -	pare	thee	to	a	sum-	mer's	day?

Here, each x represents an unstressed syllable and every \represents a stressed syllable. Try saying it out loud!
The goal for this assignment is to generate poems that Shakespeare may have written by training a recurrent neural network on his 154 sonnets. His sonnets are available in the data file `shakespeare.txt`, provided in the data folder. Read the sequence modeling chapter (Chapter 10) of the Deep Learning book for more details of RNNs.

[1] `https://en.wikipedia.org/wiki/Sonnet#English_.28Shakespearean.29_sonnet`
[2] `https://en.wikipedia.org/wiki/Volta_%28literature%29`
[3] `https://en.wikipedia.org/wiki/Iambic_pentameter`

2 Recurrent Neural Networks

2.1 Pre-processing (2 points)

The first step is to pre-process the dataset before you train on it. How you pre-process is completely up to you. Here are a couple of questions to help you decide how to do pre-processing: How will you tokenize the data set? What will consist of a singular sequence, a poem, a stanza, or a line? Do you keep some words tokenized as bigrams? Do you split hyphenated words? How will you handle punctuation? It may be helpful to get syllable counts and syllable stress information from CMU's Pronouncing Dictionary available on NLTK. You might also find the file `Syllable_dictionary.txt`, provided in the data folder, to be helpful; please see the associated file called "syllable_dict_explanation" for an explanation.

2.2 Model Training (5 points)

Try doing poem generation using a recurrent neural network (RNN). Please follow these guidelines in this section:

- Train a **character-based LSTM** model. A single layer of 100-200 LSTM units should be sufficient. You should also have a standard fully-connected output layer with a softmax nonlinearity.
- Train your model to minimize categorical cross-entropy. Make sure that you train for a sufficient number of epochs so that your loss converges. You don't necessarily need to keep track of overfitting/keep a validation set.
- Your training data should consist of sequences of fixed length (40 characters is a good number for this task) drawn from the sonnet corpus. The densest way to do this is to take all possible subsequences of 40 consecutive characters from the dataset. To speed up training, using *semi-redundant* sequences (i.e. picking only sequences starting every n-th character) works just as well.
- To generate poems, draw softmax samples from your trained model. Try to lay around with the *temperature* parameter and generate different outputs, which controls the variance of your sampled text.

2.3 Improving recurrent models (3 points)

There are many ways you can improve the basic LSTM model. For example, you can try using word-embeddings or other morphological representations instead of characters. You can also try to include the attention mechanism as well as more complicated recurrent models. Evaluate your improvements over the baseline model both qualitatively and quantitatively (e.g. using the perplexity metric). You may use any publicly-available code and papers as long as you reference them appropriately.

3 Report

Pre-Processing Your report should contain a section dedicated to pre-processsing. Explain your choices, as well as why you chose these choices initially. What was your final

pre-processing? How did you tokenize your words, and split up the data into separate sequences? What changed as you continued on your project? What did you try that didn't work? Also write about any analysis you did on the dataset to help you make these decisions.

RNN Explain in detail what model you implemented? What parameters did you tune? Comment on the poems that your model produced. Does the LSTM successfully learn sentence structure and/or sonnet structure? How about the runtime/amount of training data needed? Include generated poems using temperatures of 1.5, 0.75, and 0.25 with the following initial 40-character seed: "shall i compare thee to a summer's day?\n", and comment on their differences.

Improving RNNs Explain in detail your improvement over the basic LSTM model. What module(s) did you change? What are the improvements both qualitatively and quantitatively? Feel free to be include visualizations and interpretations, but remember that accurately representing data is still your primary objective. Your figures, tables, and diagrams should contribute to a discussion about your model.

4 Additional Resources

- TED talk: Can a computer write poetry?
- Botpoet
- Natural Language Processing Toolbox
- Markov Contraints for Generating Lyrics with Style
- Unsupervised Rhyme Scheme Identification Hip Hop Lyrics Using Hidden Markov Models