Generation of poems
Machine Learning for Natural Language Processing 2020

Tom Coral
Affiliation
Tom.CORAL@ensae.fr

Etienne Le Naour
Affiliation
Etienne.LeNaour@ensae.fr

- Point to our notebook (on colab).[1]
- Point to our github[2], which isn't very useful, just some data we call from the colab.

Abstract

The aim of our project is to be able to generate sonnets (poems). Thus, we train a language model on 2685 sonnets. Our approach is based on the article *Deep-speare: A joint neural model of poetic language, meter and rhyme* by (Lau and al., 2018). The model we use is more simplified than the one used by the authors. However, we manage to generate poems with sentences that seem elegant.

1 Problem Framing

Nowadays, supervised learning is capable of recognizing complex data structures, we wonder if it is possible to train a language model capable of generating word sequences in order to get a poem. We would like to train a model on sonnets (kind of poems with 14 lines). We want to be able, after learning, to generate poems and make them suitable. In order to learn a language model, we are going to use dataset built thanks to Project Gutenberg (https://www.gutenberg.org/). A suitable quatrain means that verses follow a good meter and rhythm structure and that English grammar is well respected. According to https://www.writing.upenn.edu/~afilreis/88/meter.html, english poetry is based on basic rhythms of varying stressed and rhyming pattern. We will see if we can handle it.
The questions we ask ourselves are as follows

- Are we able to generate suitable poems ?
- Can the constraint of limited data be overcome with a suitable RNN structure?
- Will the algorithm have trouble with Old English?

2 Experiments Protocol

Let's have a look on our train, valid and test datasets (more descriptive statistics in the notebook) :

data set	Number of sonnet	Number of word
train	2685	367281
valid	335	4020
test	335	4020

As our dataset is limited, we will first pre-train a word2vec (skip-gram) model of size 100 from a large corpus of poems (34M Gutenberg poetry data). After that, we apply this word2vec to our vocabulary from the data set.

Now that we've retrieved this data, we want to train a language model on it. Our language model will be a variant of an LSTM encoder-decoder model with attention. In this model, encoder encodes the preceding context (i.e. all sonnet lines before the current line) and the decoder decodes one word at a time for the current line, while attending to the preceding context. In the notebook we explain in more detail the mathematical working of this deep learning model. In order to get an overview of how this model works, let's take a look at the following figure. In the example below, 'shall I compare thee to a' is the preceding context and the decoder decodes 'thou' then 'art' then etc. The yellow boxes represent words in the form of an embedding matrix. The bi-directional-LSTM with selective encoding refers to a recurrent

[1] https://colab.research.google.com/drive/17KBwi38Nv_xxkeCOwlAPKc34BjuhLvY_

[2] https://github.com/EtienneLeNaour/NLP_Project

neural network that has a particular function (see notebook).

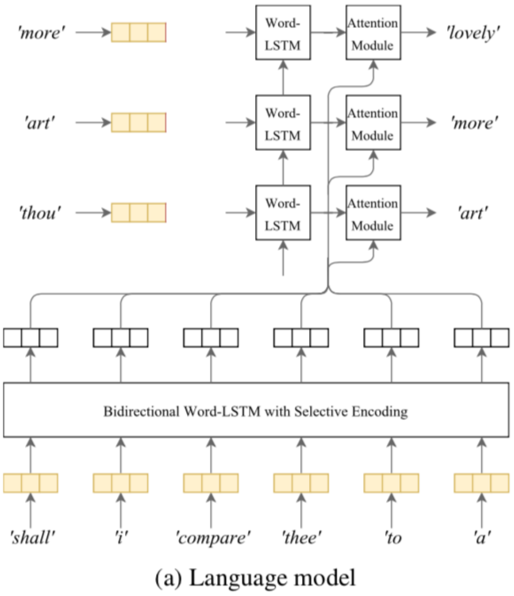

Figure 1: Language model

We train the model by optimizing a standard categorical cross-entropy loss.
Once the model is trained, let's talk about the generation scheme. During generation, we feed the hidden state from the previous time step to the language model's decoder to compute the vocabulary distribution for the current time step. Words are sampled using a temperature between 0.6 and 0.8, and they are resampled if the a set of particular words is generated (see the notebook for these words). Note that our way of generating is much simpler than the one initially proposed by the authors. Indeed, we do not include characters scheme and rhymes scheme in our generator.
Let's take about the implementation. To make this project, we had to read thousands of lines of code from the github of the authors of the article (https://github.com/jhlau/deepspeare). We reused a good part of it and rewrote a large part of it to make it fit our project. A big difficulty was that their code was written in python 2 and we use python 3. For instance, to implement our language model, which is simpler than theirs, we had to isolate the parts of the code that corresponded to what we wanted. After that, we had to rewrite parts of the code to make it work and then call it in the notebook.

3 Results

Here is an example of a poem generated by our model :

some morning soul would leave its heart too fast
reminds me? what is me if all most far
you have no thunder for my waking eyes
should make you more than which we must be true
than if it can remember my romance
that i believe not at its childish fears
for what it is our gods with simple joys
but these have i return; or when we hold
this art can charm from ours! a smaller sphere
seems not his love to heaven; but in an hour
and one of mine! most bright and more serene
as lying with a tranquil, deep in peace
the voice of danger, startled to the steep
from ocean's bosom o'er our country's bay

Even though we did not use the *iambiac pentameter* model like the authors of the article, our sentences seem to be of good length even if they are not always exactly 10 feet long. The rhythm also seems to be respected with variations in the verse. The grammatical nature of classical poetry seems to be respected. Indeed we find the dramatic lexical field ("heaven", "bright" etc.). The grammar is good and most of sentences make sense (in Old English of course). There are some negative points like the fact that there are no rhymes. We do qualitative and quantitative evaluations in the notebook.

4 Discussion/Conclusion

We are satisfied with the poem generator result. Of course it can be greatly improved but for a school project it seems very suitable. The difficulty of generating coherent poems with only a language model, as opposed to a model integrating various constraints of the sonnet form, raises the issue of the generalisation of the work. Indeed, the corpus is one of the largest and most formally coherent currently available, yet it seems still limited and has to be exploited with very specific methods.

References

Jey Hay Lau and al. 2018. Deep-speare: A joint neural model of poetic lan-guage, meter and rhyme. *https://arxiv.org/pdf/1807.03491.pdf*.