COGS 181 Final Project Report

Samuel Sunarjo
A12340230

Abstract

Recently, RNNs based on characters, also known as char-RNN, have shown significant results in many applications, such as literature, Wikipedia, Algebraic Geometry, Linux source code. The core strength of RNN compared to general neural networks or the Convolutional Neural Networks (CNN) is that RNN operates over sequences of inputs and seemingly avoid the constraint of computational steps and fixed-sized vector as input or output. One of the main reasons behind the magical performance of RNN is the information passing from one time step to the next time step in the sequential signal. RNN generally requires more computations and larger dataset to train. Long Short-Term Memory (LSTM), a specific model of recurrent neural network (RNN), was designed to perform better in terms of modeling and accuracy for temporal and long-range dependencies. LSTM addresses some critical issues in regular RNN; exploding gradient is controlled by gradient clipping, and vanishing gradient is controlled with additive interactions. In this project, the LSTM char-RNN will be applied to a dataset of song lyrics, which is a form of writing that is somewhat similar to literature and poems. The performance of the LSTM char-RNN will be tested and evaluated based on different set of datasets, hyperparameters, and network structure.

Introduction

Char-RNN has been studied and experimented and recently illustrated to be capable of producing meaningful and impressive results for generating texts with different specific structures [1]. It is capable of learning the important structures in different text applications. The problem can be classified as a many-to-many type of neural network. Music is a huge part in most people’s lives nowadays, however producing a song is certainly time consuming and not an easy task. Being able to produce more than one hundred songs during a singer’s or writer’s lifetime is quite a career accomplishment. Also famous singers who have retired or passed away would no longer produce any new music for their fans. But what if we can build a network that can generate song or lyrics that contains resemblance with the original work of a particular singer or genre? This could certainly speed up the song production process, provide new or inspirational ideas for singers, and produce new songs of famous singers from the past. Multiple Char-RNN will be trained on this dataset with different selections, network structure, and hyperparameters to compare results.

Method

Typically, a char-RNN would require at least one million characters as training set to be able to produce meaningful results. It is quite reckless to just combine song lyrics from different artists and genres to increase the size of the training set, which could introduce significant amount of variance and disturbance into the style and format of the song lyrics. This project will focus on a Kaggle dataset containing 57650 entries, each includes information such as the artist, song name, and unmodified lyrics of the song [2]. This dataset is processed and manipulated with the Python module Pandas. The singer in this dataset who produced the highest number of songs is Donna Summer with 191 songs, followed by Gordon Lightfoot with 189 songs. These two singers have many differences, so are their songs.

In this project, the char-RNN will base on the implementation and source code from the Keras framework [3], which is a high level wrapper on top of Google Tensorflow. The general architecture would begin with a LSTM layer, followed by a dense layer and activation function. In each iteration of training, the char-RNN looks at a block of the characters from current to a certain length in the past and predict the next character as the output based on the distributions over the set of possible characters. This predicted character will then be compared to the true next character for computing the loss to update the weights in the LSTM model. During each training step, the model is trained on a batch sampled randomly from the lyrics and generate lyrics starting from a randomly selected sentence. These lyrics will be generated according to several values of temperature, or diversity, which specifies the degree of variation in the model outputs. This allows various model observations to be made during each training iteration.

Experiment

First Vanilla Char-RNN. This network is trained with all the song lyrics from Donna Summer only, which contains a total of 236908 characters. The length of each block of sequence is 40 characters with step size of 3. The LSTM hidden layer has 128 units, followed by the dense layer with size of the character set and softmax activation. The model is optimized through RMSprop on categorical cross entropy. The batch size is 128, with 60 training iterations. The diversity values are 0.2, 0.5, 1.0, and 1.2. After only 1 training iteration, the model is already capable of producing words that are understandable:

```
Starting sentence: "again \nWe'll stand there hand in hand  "

again
We'll stand there hand in hand

I want to the way a wanna the want
You to the way a wanna do want to to the way

When you
```

```
Oh you

So love you
```

Each training iteration takes about 45 seconds. The loss decreased from 2.0118 at the first training iteration down to 0.7257 at the last iteration. Based on this trained char-RNN model, some observations about the diversity values become apparent. With a small diversity value, for example 0.2, the model generates output that has much more repetition in characters and words. The generation process can also easily trap in a state of generating the same set of characters:

```
Starting sentence: "ou're standing unaware \nI'm filled with"

ou're standing unaware
I'm filled with me
And I want you to stay go

Ooh baby, back in love
I'm a wanderer

I want to make you
It's not the way that I need
I want to me baby
I love to love you baby
I love to love you baby
I love to love you baby
I love to love you baby
I love to love you baby
```

The outputs generated with higher diversity value, for example 1.2, result in more spelling errors and more variance in words and structure:

```
Starting sentence: "ou're standing unaware \nI'm filled with"

ou're standing unaware
I'm filled with but he's ten it

Our love
Like the only get what passs we could night of I know to go   never had a botting high
With hambe sonm that't you

Me feeling flaw hold
And don't kep umben 'cause the joyeas again
The onegen set it one heart of his sing
We're all my dreas and out,
Water for ery lofe mesued me
Lothiny will be love you'll promided go
i lone live and baby I fever you
```

This property of char-RNN seems to be very powerful and useful. A trained network can produce a wide range of outputs based on a parameter that is adjustable after training. A simple LSTM char-RNN with relatively small training set is already capable of producing decent outputs that are pretty understable to humans. Note that the network also knows how to deal with new lines, which are represented as '\n' in the texts.

Vanilla Char-RNN with Different Artists. The second char-RNN is trained on song lyrics from another artist Gordon Lightfoot, which contains a total of 230377 characters. The structure and parameters of this network are identical to the previous network for Donna Summer. Each training iteration takes about 55 seconds. The loss decreased from 2.0612 at the first training iteration down to 0.8494 at the last iteration. The trained network follows the same pattern for the different diversity values, as mentioned in the previous network.

The third char-RNN is trained on song lyrics from both artists Donna Summer and Gordon Lightfoot. This overlapping dataset consists of half of the song lyrics from Donna and half of the song lyrics from Gordon, in order to keep a similar size of training set. Each training iteration takes about 40 seconds. The loss decreased from 2.0655 at the first training iteration down to 0.8259 at the last iteration.

To test the similarity between these three trained networks, ten different starting sentences are fed into the networks and one of the results with diversity value of 0.5 is shown below. We can see that Donna’s char-RNN produces words with short length, simple words, and mostly associated to the first-person perspective. Whereas Gordon’s char-RNN produces words with relatively longer length, more complex words, and mostly associated to the third-person perspective. The mixture char-RNN seems to reflect some of the characteristics described from both the previous two char-RNN:

```
Starting sentence: "Loneliness is always looking for a frien"

Donna Summer char-RNN:
Loneliness is always looking for a friend, you and me, we can

There no the day we was a please
And I don't want to get hurt
I'm not know it's all of life
This is hard to be good you
If you the time
I don't want to get on a singt
...


Gordon Lightfoot char-RNN:
Loneliness is always looking for a frient
Baby the waters in the evenin' is tonight
A fistred in the dawn the flame
The road the promises in my mind
As a saute, where is the dawn
If you not soil conner
To the teld you when I can see

…

Donna and Gordon char-RNN:
```

```
Loneliness is always looking for a friene

She got all I ever think about
I got your love you out there
If you're looke for a while of the sky

Well, have understand who mistake a wart
I'm a promise of me
…
```

Different Parameters for char-RNN. The first char-RNN with Donna Summer dataset is further experimented with different settings. The hidden units in the RNN is increased from 128 to 512, while retain other parameters. This trained network actually produces surprising result shown below with diversity value of 0.5. This network simply fails to learn any meaningful structure from the training set and cannot even spell a correct English word. This behavior could be caused by many different reasons. After multiple round of testing, I was not able to find the correct values of other parameters that could fix this network of 512 hidden units:

```
Starting sentence: "Loneliness is always looking for a frien"

Loneliness is always looking for a frientceeo'aneonofsoo


oraoo'ehkge
egI
n naeaeassneoee

fe
…
```

Another char-RNN is trained based on the first char-RNN with Donna Summer except with two layers of LSTM with 128 units. The output of this trained network with diversity value of 0.5 is shown below. Again, this network did not produce any meaningful result either:

```
Starting sentence: "old on, yeah \nCan't we just sit down an"

old on, yeah
Can't we just sit down an aa see te cod the yh wha ush hhaw
ho
haa na the
hd ha

t  u pah
…
```

The main reason of the failure of these two networks is very likely that there are simply too many parameters in the LSTM char-RNN model for this tiny dataset and little training iterations and epochs. A training set of 200k characters and training duration of 40 minutes is insufficient for training these deeper complex networks to produce meaningful results.

Conclusion and Future Work

A small and quick LSTM char-RNN seems to be capable of producing meaningful results from training on song lyrics. We trained the network with song lyrics from different artists and examined the influences the training dataset have over the network. Network with larger and deeper structure parameters were attempted for training but failed to produce meaningful result due to the limited amount of dataset and training time. LSTM char-RNN certainly requires a large amount of training data and time to become meaningful and effective. Another limitation of text generating networks is that there is no good metric standardized to evaluate the performance and quality of the texts generated by the network. One possible implementation in the future is to create another network to evaluate the text quality or perform classification on the text generated by the network. Training in the future should incorporate a larger dataset of song lyrics and longer training time and iterations to further study the effectiveness of LSTM char-RNN in the domain of song and lyrics.

Reference

[1] Karpathy, Andrej. “The Unreasonable Effectiveness of Recurrent Neural Networks.” The Unreasonable Effectiveness of Recurrent Neural Networks, 21 May 2015, karpathy.github.io/2015/05/21/rnn-effectiveness/.

[2] Kuznetsov, Sergey. 55000+ Song Lyrics. 5 Jan. 2017, www.kaggle.com/mousehead/songlyrics.

[3] keras-team. “Keras Source Code.” GitHub, 15 Dec. 2017, github.com/keras-team/keras.