UNIVERSITY OF PISA

AUTOMATIC LYRICS GENERATION

Text Analytics Project

Riccardo **FIGLIOZZI**
609652

A.A. 2020/2021

Contents

Intro

The aim of this project is to generate new song lyrics based on the data from other songs. The dataset used for the task of this project is available on Kaggle at the following link: https://www.kaggle.com/neisse/scrapped-lyrics-from-6-genres.

It is divided in two other datasets *artists-data.csv* and *lyrics-data.csv*, with both containing data about six musical genres :

- Rock
- Hip Hop
- Pop music
- Sertanejo (Basically the Brazilian version of Country Music)
- Funk Carioca (Originated 60s US Funk, a completely different genre in Brazil nowadays)
- Samba (Typical Brazilian music)

The goal of the project is to use the lyrics of the dataset in order to generate new lyrics by using different input seeds and compare the results of a LSTM (Long Short Term Memory) and a GPT2 (Generative Pretrained Transformer 2) model.

Moreover there will be some explanation of the two models used to generate the text in order to explain the differences in terms of construction, efficiency and cost.

Data understanding

The two datasets were treated differently in terms of data understanding and data preparation (which will be discussed in the next section). This is because this phase has been useful in understanding how the dataset is composed and structured and any characteristics to be taken into consideration.

The datasets have different data as said in the intro (artists-data.csv and lyrics-data.csv), one corresponding to the artists and their associated musical genre, and the other containing the songs and their lyrics. To join them it is possible to use the “Link” column and drop all the other ones. Before merging them it is necessary to process each dataframe separately.

In the dataset there are other languages that it is possible to use, since the english is the biggest one the dataset used for the task will contain just english songs.

```
ENGLISH           114723
PORTUGUESE         85085
SPANISH             4812
ITALIAN              626
FRENCH               471
GERMAN               314
KINYARWANDA           88
ICELANDIC             47
SWEDISH               27
FINNISH               24
INDONESIAN            17
ESTONIAN              12
GALICIAN              12
DANISH                 9
HAITIAN_CREOLE         9
```

By looking at the genres it is possible to see that the dataset is unbalanced, so in order to balance it we can delete the three least genres since they could be also considered as outliers.

```
Rock            47534
Pop             25647
Hip Hop         13661
Sertanejo          51
Samba              42
Funk Carioca       15
Name: Genre, dtype: int64
```

This will be something to consider for the data preparation and the training process of the models.

Data preparation

Now that we have done the data understanding, we have understood which are the aspects to consider in order to use the dataset, therefore before using the two models we will have to prepare the dataset to use for the training phase.

For both models we will use a dataset consisting of rock english songs with a popularity level greater than 5. The data will be processed differently based on the needs of the models to be used.

	ALink	SName	SLink	Lyric	Idiom	Artist	Genre	Link
0	/4-non-blondes/	What's Up	/4-non-blondes/whats-up.html	Twenty-five years and my life is still. Trying...	ENGLISH	4 Non Blondes	Rock	/4-non-blondes/
1	/4-non-blondes/	Spaceman	/4-non-blondes/spaceman.html	Starry night bring me down. Till I realize the...	ENGLISH	4 Non Blondes	Rock	/4-non-blondes/
2	/4-non-blondes/	Pleasantly Blue	/4-non-blondes/pleasantly-blue.html	Every time you wake in the mornin'. And you st...	ENGLISH	4 Non Blondes	Rock	/4-non-blondes/
3	/4-non-blondes/	Train	/4-non-blondes/train.html	What ya gonna do child. When your thoughts are...	ENGLISH	4 Non Blondes	Rock	/4-non-blondes/
4	/4-non-blondes/	Calling All The People	/4-non-blondes/calling-all-the-people.html	How can you tell, when your wellness is not we...	ENGLISH	4 Non Blondes	Rock	/4-non-blondes/

LSTM

For the LSTM model we need to insert at the beginning and at the ending of each lyric a specific label in order to let understand the model when a song start and end, in this case we will insert a '"<SOT>" (Start Of Text) and a “<EOT>” (End Of Text).

After doing this, we need to join all the lyrics in order to get one unique text and vectorize it in order to put it as input for the train phase of the model.

This is the LSTM model that we will use for the text generation:

```
model = Sequential()
model.add(Bidirectional(LSTM(256), input_shape=(maxlen, len(chars))))
model.add(Dropout(0.1))
model.add(Dense(len(chars), activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
bidirectional (Bidirectional (None, 512)               870400
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 168)               86184
=================================================================
Total params: 956,584
Trainable params: 956,584
Non-trainable params: 0
_________________________________________________________________
```

Now that the data have been processed we can train the LSTM:

```
[ ]  # train the model, output generated text after each epoch
     model.fit(x, y,
               batch_size=64,
               epochs=10,
               validation_split=0.1)
```

```
Epoch 1/10
704/704 [==============================] - 13s 14ms/step - loss: 3.1927 - accuracy: 0.2051 - val_loss: 2.3567 - val_accuracy: 0.3534
Epoch 2/10
704/704 [==============================] - 9s 13ms/step - loss: 2.2754 - accuracy: 0.3648 - val_loss: 2.1193 - val_accuracy: 0.3830
Epoch 3/10
704/704 [==============================] - 9s 13ms/step - loss: 2.0276 - accuracy: 0.4184 - val_loss: 1.9952 - val_accuracy: 0.4248
Epoch 4/10
704/704 [==============================] - 9s 13ms/step - loss: 1.8847 - accuracy: 0.4541 - val_loss: 1.9249 - val_accuracy: 0.4450
Epoch 5/10
704/704 [==============================] - 9s 13ms/step - loss: 1.7488 - accuracy: 0.4877 - val_loss: 1.8488 - val_accuracy: 0.4636
Epoch 6/10
704/704 [==============================] - 9s 13ms/step - loss: 1.6343 - accuracy: 0.5223 - val_loss: 1.8090 - val_accuracy: 0.4858
Epoch 7/10
704/704 [==============================] - 9s 13ms/step - loss: 1.5282 - accuracy: 0.5520 - val_loss: 1.8070 - val_accuracy: 0.4778
Epoch 8/10
704/704 [==============================] - 9s 12ms/step - loss: 1.4327 - accuracy: 0.5810 - val_loss: 1.7719 - val_accuracy: 0.4966
Epoch 9/10
704/704 [==============================] - 9s 13ms/step - loss: 1.3206 - accuracy: 0.6102 - val_loss: 1.7991 - val_accuracy: 0.4900
Epoch 10/10
704/704 [==============================] - 9s 13ms/step - loss: 1.2342 - accuracy: 0.6385 - val_loss: 1.8191 - val_accuracy: 0.4960
```

GPT2

The GPT2 is different from the LSTM since it is a pretrained model which can already generate decent quality text. However, for a specific context it is better to fine-tune it on our data.

As done for the LSTM, firstly, we need to tokenize the data in order to use the GPT2 by ensuring that every song respects a maximum of 1024 tokens.

To do so, we import the pretrained GPT2 model and the tokenizer. However, since the size of the GPT2 is huge, it is necessary to accumulate the gradients in order to not get a *CUDA Out of Memory error*. This will be done by using the following function:

```
def pack_tensor(new_tensor, packed_tensor, max_seq_len):
    if packed_tensor is None:
        return new_tensor, True, None
    if new_tensor.size()[1] + packed_tensor.size()[1] > max_seq_len:
        return packed_tensor, False, new_tensor
    else:
        packed_tensor = torch.cat([new_tensor, packed_tensor[:, 1:]], dim=1)
        return packed_tensor, True, None
```

The main idea is to sum all the gradients of the different operations and then perform the optimization of the gradient descent. To get fewer calculations, we will divide the total by the number of accumulated steps, in order to get an average loss over the training sample.

Eventually, we can train the model in order to finetune the GPT2 with our data:

```
#Train the model on the specific data we have
model = train(dataset, model, tokenizer)
```

```
0it [00:00, ?it/s]Training epoch 0
0
12000it [18:00, 11.11it/s]
0it [00:00, ?it/s]Training epoch 1
tensor(1.3729, device='cuda:0', grad_fn=<NllLossBackward>)
12000it [18:07, 11.03it/s]
0it [00:00, ?it/s]Training epoch 2
tensor(1.9431, device='cuda:0', grad_fn=<NllLossBackward>)
12000it [18:08, 11.02it/s]
0it [00:00, ?it/s]Training epoch 3
tensor(1.4939, device='cuda:0', grad_fn=<NllLossBackward>)
12000it [18:04, 11.06it/s]
0it [00:00, ?it/s]Training epoch 4
tensor(1.5333, device='cuda:0', grad_fn=<NllLossBackward>)
12000it [18:07, 11.04it/s]
```

Results

This is probably the most interesting section of the project since we are going to generate and analyze the texts generated by the two models, in order to compare them and understand which of the two is more efficient.

Comparing ends of the lyrics

Here are the several texts generated by the LSTM and GPT2 model, in black there is the seed used as input and in blue the part generated by the model by using different values of “temperature”.

LSTM

Temperature	Text
0.2	ut if there's a pill to help me forget. She to down to roll shack You know I know that's got the skind And I way I sake you roll I wan way I gonna baby And I heads of the good to be
0.5	ut if there's a pill to help me forget. Ceall the sard the bous And I take you nothing dorn my spous If you want it up the good on me Hear the por chame to me I got a tound ats me the go down
0.8	ut if there's a pill to help me forget. Carit's givn at some to barcy You know I high a tolight In you wellin' mighta toright AraightarProw be spael
1.0	ut if there's a pill to help me forget. Car tho you do I doil', poy afrum It's roplin' Big Jack, Com you give it'm holling ont Fordy mo choms no drels on the sind Get o toing dobred ne
1.2	ut if there's a pill to help me forget. Den't tound, peen my trown<EOT> <S?T>CasQme hayd Yeah reascane anw thomes And theres rocow, de, boonge niem in time 'Cause I'm oh, busn)etrup

GPT2

Temperature	Text
0.2	ut if there's a pill to help me forget. I'm not sure if I'm going to be able to get through this I'm not sure if I'm going to be able to get through this I'm not sure if I'm going to be able to get through this I'm not sure if I'm going to be able to get through this
0.5	ut if there's a pill to help me forget. I'm not sure what to do I'm not sure what to do I'm not sure what to do I'm not sure what to do
0.8	ut if there's a pill to help me forget. But the next time I wake up in the morning, I'll have to try and figure out what I should do If I can't find a solution, what can I do? Well, here's the way If you're new to Buddhism, you might not be familiar with the Buddhist path
1.0	ut if there's a pill to help me forget. I realize I'm probably not the only one that thinks this way I think that whatever there is to realize in the world is what we have to do to get there I also realize that I will never be free I will never be free
1.2	ut if there's a pill to help me forget. And those around me will always wake up and wonder where my yoga session is all I want you to know that I love you so much I love you so much I want you to see me lead the way

We can see the difference between the various texts generated with the different temperatures. The LSTM generates good results between the values of 0.5 and 1.0, the same seems to be true for the GPT2 which seems to understand the context better and generates sentences of more or less the same length.

Songs similarities

The following tables present the lyrics generated by the models and which are the songs they most resemble.

LSTM

Similarities	Text
- Teacher's Pet (0.0) - Suzi (0.0) - Sunrise (0.0)	You've Got to Hide Your Love Away ehHhi--r rr Ul sy ,Eh(hie eeee m,ss e s(<ejcyme sr ra!uiK(aatsaaeeeeueehhseYttaoeaeii yietWuhirtxwdm
- Monica (0.0) - Star(0.0) - Song For Love(0.0)	The Wait dyihu] ohhIcothiBnhTstiuhthmhlmh<hr!httto l-tcooataaSNhhuhhuldiha icncl<bwbruS Sl!B feit Suhoo,thehotleBntouw-[uh uauihhhc?sculfdioirhmxh ea,eehehhBchmhS<ahi

GPT2

Similarities	Text
- Hitch Hike (0.64) - If I Lose Myself (0.64) - You've Got To Hide Your Love Away (0.61)	You've Got to Hide Your Love Away Despite being in all the wrong places, Angelina Jolie says: "She's so clever and so perfect It's just she's so real And that's one of the greatest things about her She can really make me miss her There's a line in her own mind that I would never know her
- Double Crossing Time (0.22) - Ashley (0.20) - Where Will You Go(0.15)	The Waitresses' Blame Are Pieces There's no denying the murder of Amy Grant, the show's very own celebrity advocate Here's some of the thing that some people, many in this world, don't understand How can we all help one another? Let me show you what it's like to be human Once you're inside the hole The fight is still there There are far too many people out there to learn<EOT>

Here unfortunately the LSTM has generated texts without any sense and then the songs that have been attributed as similar have no value, in a different way we can treat the results generated by GPT2 that manages to get a good similarity with the original texts.

Generating text with different languages

In this last test the input was changed using three languages other than English (Italian, Spanish and French) with a temperature equal to 1.

LSTM

Language	Text
Italian	Siamo fuori di testa, ma diversi da loro She llow I hall you natht in the way [Cow I'm a ready dirth beed ir tise Lot's rond We'll great fertevel on meerany For my homes for there amboting ready Bor my held os ferlot ald heave a but elack For my bad wigh a flaprid why
Spanish	Sí, sabes que ya llevo un rato mirándote ? A rock 't' rall Never might ic ut So down, dy my hey strow Anl oo cid to for off ay the gaig Coull me back in byoude Soven'm in the sky
French	La vie c'est plus marrant hii)ciSteaSChoieiB hhlwAiciiomehheecrorniiwtwiethcoaRi)OOoh-hCaoa woaoioi iiihisihu'h<uh-aiwBigyeehhhhhstlheelbyyhluheltoycitimci<u oOcwoowhblhcoooaohetjni(nochninHichii ieilasuveii uuautaoirycOu[hehhh ih ahnpnuih

GPT2

Language	Text
Italian	Siamo fuori di testa, ma diversi da loro Nuestra se le hora destiano aperra dono no se Tan na riadido de una testa

	Anar mi anarto de la mio No way no no da uno Da voie se litra de una testa My chance, di supereconido no da infamy
Spanish	Sí, sabes que ya llevo un rato mirándote Be that as it may, sabes que ya llevo un rato mirándote es mismo<|endoftext|>Young child on its own Just not quite ready Still Trying to think through the thought process I don't know what it feels like
French	La vie c'est plus marranto!" A silvestri se père m'enchanté C'est moi les cabs de que vous sommes quelques zez pas m'avez, l'enflançon des seilères Mi mécuisine est un voir Don't know if I'm going to kill you now, when I'm out here It's summer, so you'll have to take the jump At least let me<EOT>

In this case, it is interesting to see how the LSTM, by taking an Italian or Spanish text as input, is able to generate texts that make more or less sense, something that does not happen with French. As for GPT2, it seems to make a bit of confusion between Italian and Spanish sometimes mixing them together and also putting in some English; different things happen with a French input.

This is most likely due to the fact that the authors of GPT2 have trained and tested the model for English to French translation.

Word importance for text generation

An interesting experiment could also be done using the ECCO library[1] to understand how the GPT2 model works. Through this library it is in fact possible to understand the level of importance of words in the generation of texts. Personally, I have tried to load the model used for this project, however some problems have been found in the loading phase of the model.

It was therefore decided to use, only for this case, the pretrained model of HuggingFace, just to understand in a general way the operation of the model and the library, hoping to give inspiration to further tests.

[1] https://www.eccox.io/

```
text= "You've Got to Hide Your Love Away"

output = lm.generate(text, generate=20, do_sample=True)
```

```
You 've Got to Hide Your Love Away >> \n
\n
The last few albums of The World According to Rih anna were a few verses in the "
```

```
output.saliency()
```

```
You've Got to Hide Your Love Away >> \n
\n
The last few albums of The World According to Rihanna were a few verses in the "
```

```
output.saliency(style="detailed")
```

```
You 've Got to Hide Your Love Away >> \n
2.27% 2.08% 2.30% 3.26% 3.29% 1.59% 1.87% 2.06% 1.84%
\n
1.87%
The last few albums of The World According to Rih anna were a few verses in the "
1.91% 2.14% 2.87% 8.84% 2.40% 1.59% 1.38% 2.72% 1.58% 6.29% 2.72% 3.14% 2.75% 5.61% 31.629
```

As you can see from the image above, here we used as input a text that was previously used with the fine-tuned model and it is in fact possible to analyze the probability that each word has in the creation of the text.

LSTM and GPT2 model descriptions

In this paragraph the two models used in this project will be described examining the technical characteristics and the possible applications in the real world.

LSTM

RNNs have been shown in these years of research to be the only choice to deal with problems related to sequence classification and to preserve information from previous input data by using it to modify the output at each time step. However, if the length of the sequence is long enough, then the gradients calculated during the backpropagation step, may vanish or explode, due to the cumulative multiplication of large values, causing the model to train relatively in slow fashion.

An LSTM network is a type of RNN architecture that aids in training the model over long sequences and preserving memory from previous input time steps that were fed to the model.
It can solve the problem of gradient vanishing or gradient explosion by introducing additional gates, input and forget gates, which allow better control of the gradient, enabling which information to retain and which to forget, thus monitoring information access to the current cell state.

For text sequences it is often the case that a RNN model can perform better if it not only processes sequences from start to end, but also backwards (Bidirectional RNNs). For example, to predict the next word in a sentence, it is often useful to have the context around the word, not only just the words that come before it.

There have been some successful cases using LSTM models that are worth mentioning:

- In 2018, OpenAI trained an LSTM model to control a human-like robotic hand to manually handle physical objects with unprecedented dexterity.

- In 2019, DeepMind's AlphaStar program used a deep LSTM to be able to compete in the video game Starcraft II.

GPT2

GPT2 is the successor of the first GPT built by OpenAI's in 2018, with the aim to train a model with a bigger dataset and generate better texts.

To train the model the authors scraped the Reddit platform by calling the dataset “WebText”. The size of the dataset used for training GPT2 is 40GB of text data from over 8 million documents, huge compared to Book Corpus dataset used for training GPT model.

Here the structure of the GPT2 model:

- 1.5 billion parameters
- 48 layers
- 50,257 tokens
- batch size of 512 and larger context window of 1024 tokens

The applications of GPT2 are vast, it could be used to help journalists writing news articles, to create bots that can reply to comments on social networks or to generate some poetry [2]and other creative applications.

However, implementing GPT-2 requires very expensive resources; the full version of the model is larger than five gigabytes and consumes high amounts of RAM. In addition, running a single prediction can take up a CPU at 100% utilization for several minutes, and even with GPU processing, a single prediction can take seconds.

In a news article, The Register[3] reports that the GPT-2 model used 256 Google Cloud TPU v3 cores for training, which costs US$256 per hour even if OpenAI didn’t specify the training duration.

In a paper[4] from the University of Massachusetts, the authors found that the training process for large AI models can emit more than 626,000 pounds of carbon dioxide-equivalent to five

[2] https://www.gwern.net/GPT-2
[3] https://www.theregister.com/2019/02/14/open_ai_language_bot/
[4] https://arxiv.org/abs/1906.02243

times the emissions of the average American car's life (including the manufacture of the car itself).

Conclusions

In this project we were able to see the various features of the most commonly used templates for text generation. Despite the limited resources available, the results obtained are acceptable even though the models, especially the LSTM, are sensitive to the type of input they are given. Probably for greater efficiency in the texts generated by the models should be optimized hyperparameters of both the training and the fine tuning of the two models. Undoubtedly both models generated interesting results to understand their operations.

References

- https://en.wikipedia.org/wiki/Long_short-term_memory
- https://www.tensorflow.org/guide/keras/rnn?hl=en
- https://colah.github.io/posts/2015-08-Understanding-LSTMs/
- https://www.bioinf.jku.at/publications/older/2604.pdf
- https://en.wikipedia.org/wiki/Transformer_(machine_learning_model)
- https://medium.com/walmartglobaltech/the-journey-of-open-ai-gpt-models-32d95b7b7fb2
- https://snappishproductions.com/blog/2020/03/01/chapter-9.5-text-generation-with-gpt-2-and-only-pytorch.html.html
- https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf
- https://github.com/mariostrbac/eminem-lyrics-generator/blob/main/notebooks/4b-Lyrics-Generator-GPT2.ipynb
- https://en.wikipedia.org/wiki/GPT-2
- https://openai.com/blog/better-language-models/
- https://www.technologyreview.com/2019/06/06/239031/training-a-single-ai-model-can-emit-as-much-carbon-as-five-cars-in-their-lifetimes/