Modelagem de Letras de Músicas - Top 10 Nacionais

Pré-Processamento de Palavras

Benicio Ramos Magalhaes

Motivação:

Realizar uma primeira tentativa de criação de um treino de letras de músicas com inteligência artificial. Neste contexto, selecionamos 10 letras de música da banda Charlie Brown Jr. obtidas do site letras.com.br.

In [1]:
```
#Criando letras de musicas do Charlie Brown Jr.

#Bibliotecas
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.layers import Embedding, LSTM, Dense, Dropout, Bidirectional
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras import regularizers
import tensorflow.keras.utils as ku
import numpy as np
import glob
import nltk
```

In [5]:
```
#lendo arquivos com letras de musicas
files = glob.glob("musicas/charliebrownjr/*")
print('Musicas:')
print('----------------------------------------------------------------------------------------------------')
print(files)
print('----------------------------------------------------------------------------------------------------')

data = ''
for fname in files:
    data = data + '\n' + f.read()
```

```
Musicas:
----------------------------------------------------------------------------------------------------
['musicas/charliebrownjr\\cbjr-ceu-azul.txt', 'musicas/charliebrownjr\\cbjr-como-tudo-deve-ser.txt', 'musicas/charliebrownjr\\cbjr-dias-luta-dias-gloria.txt', 'musicas/charliebrownjr\\cbjr-lugar-ao-sol.txt', 'musicas/charliebrownjr\\cbjr-meu-novo-mundo.txt', 'musicas/charliebrownjr\\cbjr-pontos-indestrutiveis.txt', 'musicas/charliebrownjr\\cbjr-proibidoproibir-mtm.txt', 'musicas/charliebrownjr\\cbjr-sendo-o-tempo.txt', 'musicas/charliebrownjr\\cbjr-so-os-loucos-sabem.txt', 'musicas/charliebrownjr\\cbjr-vozio-de-lola.txt']
----------------------------------------------------------------------------------------------------
```

In [8]:
```
#dividindo todas as letras de músicas e separando em palavras
tokenizer = Tokenizer()

corpus = data.lower().split("\n")


tokenizer.fit_on_texts(corpus)
total_words = len(tokenizer.word_index) + 1


input_sequences = []
for l in corpus:
    token_list = tokenizer.texts_to_sequences([l])[0]
    for i in range(1, len(token_list)):
        n_gram_seq = token_list[:i+1]
input_sequences.append(n_gram_seq)



max_seq_len = max([len(x) for x in input_sequences])
input_sequences = np.array(pad_sequences(input_sequences, maxlen=max_seq_len, padding='pre'))

predictors, label = input_sequences[:,:-1],input_sequences[:,-1]

# one hot
label = ku.to_categorical(label, num_classes=total_words)
```

In [10]:
```
model= Sequential([
    Embedding(total_words, 100, input_length=max_seq_len-1),
    Bidirectional(LSTM(150, return_sequences = True)),
    Dropout(0.2),
    LSTM(100),
    Dense(total_words/2, activation='relu', kernel_regularizer=regularizers.l2(0.01)),
    Dense(total_words, activation='softmax')
])

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
print(model.summary())
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 21, 100)           61100

bidirectional (Bidirectional (None, 21, 300)           301200

dropout (Dropout)            (None, 21, 300)           0

lstm_1 (LSTM)                (None, 100)               160400

dense (Dense)                (None, 305)               30805

dense_1 (Dense)              (None, 611)               186966
=================================================================
Total params: 740,471
Trainable params: 740,471
Non-trainable params: 0
_________________________________________________________________

None
```

In [11]:
```
num_epochs= 100
history = model.fit(predictors, label, epochs=num_epochs, verbose=1)
```

```
Epoch 1/100
84/84 [==============================] - 7s 86ms/step - loss: 6.6096 - accuracy: 0.0350
Epoch 2/100
84/84 [==============================] - 7s 81ms/step - loss: 5.7117 - accuracy: 0.0376
Epoch 3/100
84/84 [==============================] - 7s 81ms/step - loss: 5.4351 - accuracy: 0.0342
Epoch 4/100
84/84 [==============================] - 7s 81ms/step - loss: 5.2900 - accuracy: 0.0414
Epoch 5/100
84/84 [==============================] - 7s 85ms/step - loss: 5.1519 - accuracy: 0.0500
Epoch 6/100
84/84 [==============================] - 7s 84ms/step - loss: 5.0392 - accuracy: 0.0504
Epoch 7/100
84/84 [==============================] - 7s 86ms/step - loss: 4.9691 - accuracy: 0.0515
Epoch 8/100
84/84 [==============================] - 7s 88ms/step - loss: 4.8744 - accuracy: 0.0635
Epoch 9/100
84/84 [==============================] - 7s 89ms/step - loss: 4.7849 - accuracy: 0.0654
Epoch 10/100
84/84 [==============================] - 8s 89ms/step - loss: 4.7131 - accuracy: 0.0673
Epoch 11/100
84/84 [==============================] - 8s 89ms/step - loss: 4.6417 - accuracy: 0.0620 3s - los
Epoch 12/100
84/84 [==============================] - 8s 90ms/step - loss: 4.5656 - accuracy: 0.0823
Epoch 13/100
84/84 [==============================] - 8s 93ms/step - loss: 4.4989 - accuracy: 0.0812
Epoch 14/100
84/84 [==============================] - 8s 96ms/step - loss: 4.4344 - accuracy: 0.0895
Epoch 15/100
84/84 [==============================] - 8s 96ms/step - loss: 4.3659 - accuracy: 0.0977
Epoch 16/100
84/84 [==============================] - 8s 90ms/step - loss: 4.3203 - accuracy: 0.1128
Epoch 17/100
84/84 [==============================] - 8s 94ms/step - loss: 4.2567 - accuracy: 0.1169
Epoch 18/100
84/84 [==============================] - 8s 93ms/step - loss: 4.1811 - accuracy: 0.1256
Epoch 19/100
84/84 [==============================] - 8s 92ms/step - loss: 4.1254 - accuracy: 0.1331
Epoch 20/100
84/84 [==============================] - 7s 88ms/step - loss: 4.0730 - accuracy: 0.1376
Epoch 21/100
84/84 [==============================] - 7s 89ms/step - loss: 3.9962 - accuracy: 0.1481
Epoch 22/100
84/84 [==============================] - 7s 88ms/step - loss: 3.9589 - accuracy: 0.1523
Epoch 23/100
84/84 [==============================] - 7s 88ms/step - loss: 3.8895 - accuracy: 0.1688
Epoch 24/100
84/84 [==============================] - 8s 90ms/step - loss: 3.8491 - accuracy: 0.1662
Epoch 25/100
84/84 [==============================] - 7s 89ms/step - loss: 3.7570 - accuracy: 0.1932
Epoch 26/100
84/84 [==============================] - 7s 88ms/step - loss: 3.7007 - accuracy: 0.2011
Epoch 27/100
84/84 [==============================] - 7s 88ms/step - loss: 3.6352 - accuracy: 0.2135
Epoch 28/100
84/84 [==============================] - 7s 87ms/step - loss: 3.5690 - accuracy: 0.2323
Epoch 29/100
84/84 [==============================] - 7s 87ms/step - loss: 3.5243 - accuracy: 0.2402
Epoch 30/100
84/84 [==============================] - 7s 87ms/step - loss: 3.4655 - accuracy: 0.2545
Epoch 31/100
84/84 [==============================] - 7s 87ms/step - loss: 3.3925 - accuracy: 0.2699
Epoch 32/100
84/84 [==============================] - 8s 90ms/step - loss: 3.3256 - accuracy: 0.2883 1s
Epoch 33/100
84/84 [==============================] - 7s 87ms/step - loss: 3.2733 - accuracy: 0.3053
Epoch 34/100
84/84 [==============================] - 8s 90ms/step - loss: 3.2300 - accuracy: 0.3090
Epoch 35/100
84/84 [==============================] - 8s 101ms/step - loss: 3.1869 - accuracy: 0.3199
Epoch 36/100
84/84 [==============================] - 8s 94ms/step - loss: 3.1164 - accuracy: 0.3342
Epoch 37/100
84/84 [==============================] - 7s 86ms/step - loss: 3.0722 - accuracy: 0.3470
Epoch 38/100
84/84 [==============================] - 7s 83ms/step - loss: 3.0221 - accuracy: 0.3534
Epoch 39/100
84/84 [==============================] - 8s 90ms/step - loss: 2.9564 - accuracy: 0.3805
Epoch 40/100
84/84 [==============================] - 8s 93ms/step - loss: 2.9159 - accuracy: 0.3789
Epoch 41/100
84/84 [==============================] - 8s 91ms/step - loss: 2.9192 - accuracy: 0.3808
Epoch 42/100
84/84 [==============================] - 8s 93ms/step - loss: 2.8234 - accuracy: 0.3959
Epoch 43/100
84/84 [==============================] - 7s 87ms/step - loss: 2.7562 - accuracy: 0.4090
Epoch 44/100
84/84 [==============================] - 7s 87ms/step - loss: 2.6987 - accuracy: 0.4282
Epoch 45/100
84/84 [==============================] - 7s 88ms/step - loss: 2.6883 - accuracy: 0.4323
Epoch 46/100
84/84 [==============================] - 7s 86ms/step - loss: 2.6391 - accuracy: 0.4395
Epoch 47/100
84/84 [==============================] - 7s 89ms/step - loss: 2.5779 - accuracy: 0.4590
Epoch 48/100
84/84 [==============================] - 7s 88ms/step - loss: 2.5373 - accuracy: 0.4628
Epoch 49/100
84/84 [==============================] - 8s 98ms/step - loss: 2.4698 - accuracy: 0.4789
Epoch 50/100
84/84 [==============================] - 7s 87ms/step - loss: 2.4378 - accuracy: 0.4707
Epoch 51/100
84/84 [==============================] - 7s 89ms/step - loss: 2.3906 - accuracy: 0.4906
Epoch 52/100
84/84 [==============================] - 7s 88ms/step - loss: 2.3907 - accuracy: 0.4974
Epoch 53/100
84/84 [==============================] - 7s 89ms/step - loss: 2.4106 - accuracy: 0.4865
Epoch 54/100
84/84 [==============================] - 7s 85ms/step - loss: 2.3019 - accuracy: 0.5147
Epoch 55/100
84/84 [==============================] - 7s 88ms/step - loss: 2.2394 - accuracy: 0.5383
Epoch 56/100
84/84 [==============================] - 8s 90ms/step - loss: 2.1860 - accuracy: 0.5447
Epoch 57/100
84/84 [==============================] - 8s 91ms/step - loss: 2.1270 - accuracy: 0.5624
Epoch 58/100
84/84 [==============================] - 8s 92ms/step - loss: 2.1022 - accuracy: 0.5688 1s - loss: 2.1021 - accuracy: - ETA: 0s - loss: 2.1
Epoch 59/100
84/84 [==============================] - 7s 88ms/step - loss: 2.0432 - accuracy: 0.5808
Epoch 60/100
84/84 [==============================] - 7s 84ms/step - loss: 2.0201 - accuracy: 0.5921
Epoch 61/100
84/84 [==============================] - 8s 92ms/step - loss: 1.9831 - accuracy: 0.6004
Epoch 62/100
84/84 [==============================] - 7s 86ms/step - loss: 1.9787 - accuracy: 0.5981
Epoch 63/100
84/84 [==============================] - 8s 90ms/step - loss: 1.9182 - accuracy: 0.6177
Epoch 64/100
84/84 [==============================] - 7s 84ms/step - loss: 1.8882 - accuracy: 0.6267
Epoch 65/100
84/84 [==============================] - 7s 86ms/step - loss: 1.8523 - accuracy: 0.6323
Epoch 66/100
84/84 [==============================] - 7s 86ms/step - loss: 1.8481 - accuracy: 0.6331
Epoch 67/100
84/84 [==============================] - 7s 88ms/step - loss: 1.7853 - accuracy: 0.6534
Epoch 68/100
84/84 [==============================] - 8s 93ms/step - loss: 1.7573 - accuracy: 0.6504
Epoch 69/100
84/84 [==============================] - 7s 88ms/step - loss: 1.7363 - accuracy: 0.6571
Epoch 70/100
84/84 [==============================] - 7s 87ms/step - loss: 1.7023 - accuracy: 0.6733
Epoch 71/100
84/84 [==============================] - 7s 87ms/step - loss: 1.6727 - accuracy: 0.6808
Epoch 72/100
84/84 [==============================] - 7s 82ms/step - loss: 1.6534 - accuracy: 0.6801
Epoch 73/100
84/84 [==============================] - 7s 88ms/step - loss: 1.6206 - accuracy: 0.6910
Epoch 74/100
84/84 [==============================] - 7s 77ms/step - loss: 1.5815 - accuracy: 0.7049
Epoch 75/100
84/84 [==============================] - 7s 77ms/step - loss: 1.5828 - accuracy: 0.7011 0s - loss: 1.5743
Epoch 76/100
84/84 [==============================] - 6s 77ms/step - loss: 1.6170 - accuracy: 0.6932
Epoch 77/100
84/84 [==============================] - 6s 77ms/step - loss: 1.5897 - accuracy: 0.6977
Epoch 78/100
84/84 [==============================] - 6s 77ms/step - loss: 1.5358 - accuracy: 0.7041
Epoch 79/100
84/84 [==============================] - 7s 80ms/step - loss: 1.4794 - accuracy: 0.7289
Epoch 80/100
84/84 [==============================] - 7s 80ms/step - loss: 1.4540 - accuracy: 0.7380
Epoch 81/100
84/84 [==============================] - 6s 75ms/step - loss: 1.4602 - accuracy: 0.7286
Epoch 82/100
84/84 [==============================] - 6s 76ms/step - loss: 1.4084 - accuracy: 0.7447
Epoch 83/100
84/84 [==============================] - 6s 74ms/step - loss: 1.3898 - accuracy: 0.7504
Epoch 84/100
84/84 [==============================] - 6s 75ms/step - loss: 1.4180 - accuracy: 0.7455
Epoch 85/100
84/84 [==============================] - 6s 75ms/step - loss: 1.3470 - accuracy: 0.7658
Epoch 86/100
84/84 [==============================] - 6s 75ms/step - loss: 1.3312 - accuracy: 0.7703
Epoch 87/100
84/84 [==============================] - 6s 75ms/step - loss: 1.3127 - accuracy: 0.7741
Epoch 88/100
84/84 [==============================] - 6s 75ms/step - loss: 1.3000 - accuracy: 0.7733
Epoch 89/100
84/84 [==============================] - 6s 75ms/step - loss: 1.2774 - accuracy: 0.7774
Epoch 90/100
84/84 [==============================] - 6s 74ms/step - loss: 1.2429 - accuracy: 0.7883
Epoch 91/100
84/84 [==============================] - 6s 75ms/step - loss: 1.3489 - accuracy: 0.7632
Epoch 92/100
84/84 [==============================] - 6s 75ms/step - loss: 1.2437 - accuracy: 0.7880
Epoch 93/100
84/84 [==============================] - 6s 75ms/step - loss: 1.2400 - accuracy: 0.7801
Epoch 94/100
84/84 [==============================] - 6s 75ms/step - loss: 1.2003 - accuracy: 0.8056
Epoch 95/100
84/84 [==============================] - 6s 75ms/step - loss: 1.1686 - accuracy: 0.8056
Epoch 96/100
84/84 [==============================] - 6s 74ms/step - loss: 1.1752 - accuracy: 0.7981
Epoch 97/100
84/84 [==============================] - 6s 75ms/step - loss: 1.1298 - accuracy: 0.8132
Epoch 98/100
84/84 [==============================] - 6s 75ms/step - loss: 1.1199 - accuracy: 0.8229
Epoch 99/100
84/84 [==============================] - 6s 75ms/step - loss: 1.1040 - accuracy: 0.8150
Epoch 100/100
84/84 [==============================] - 6s 75ms/step - loss: 1.0968 - accuracy: 0.8211
```

In [37]:
```
start_text = "eu te amo"
num_words = 100  #numero de palavras

for _ in range(num_words):
    token_list = tokenizer.texts_to_sequences([start_text])[0]
    token_list = pad_sequences([token_list], maxlen=max_seq_len-1, padding='pre')
    predicted = model.predict_classes(token_list, verbose=0)
    out_word = ""
    for word, index in tokenizer.word_index.items():
        if index == predicted:
            out_word = word
            break
    start_text += " " + out_word
```

In [36]:
```
print('Resultado:')
print('----------------------------------------------------------------------------------------------------')
print(start_text)
print('----------------------------------------------------------------------------------------------------')
```

```
Resultado:
----------------------------------------------------------------------------------------------------
eu te amo
----------------------------------------------------------------------------------------------------
```

```
facil me deixa aqui a toa mal uso como voce mas em troca de uma foto com sorriso inocente nem pouco q
ue olhando por chover nossos nossa loucura um brown jr tudo que olhando sozinho fazer nem som vc
me feliz guerra mais a vida tudo um brown brown brown brown brown fe a gente rala mais a gente creze
chover feito por de nossos nossa nossos nossos brown brown brown brown brown brown brown brown brown bro
wn brown brown brown brown brown brown brown
----------------------------------------------------------------------------------------------------
```

In [ ]:
```

```