Universidade Federal do Rio Grande do Norte - CCET/DIMAp
Aprendizado Profundo – DIM0446 – 2024.2

Canto para música

Warlike Richard da
Silva Soares

Natal, 29/01/2025.

Resumo: este documento tem como objetivo descrever um modelo de regressão que utiliza redes recorrentes para transformar um áudio de cantarolado em uma música tocada em piano. Para isso foram gravados os cantarolados a partir da base de dados Maestro v3 de arquivos MIDI.

1. Introdução

Neste trabalho eu implementei uma IA que transforma áudios .wav de cantarolados em um espectrograma, que visa corresponder a uma musica que foi tocada em um piano. Para isso, o sistema transforma o áudio do cantarolado em um vetor com os embeddings extraídos do áudio, usando o modelo wav2vec. Esses embeddings passam então pelo modelo que utiliza RNN para ser treinado, gerando então um espectrograma. Sendo assim, o sistema criado utiliza uma outra ferramenta para atingir o objetivo, já que o modelo puro precisa do áudio representado em um vetor, e não do áudio em si.

2. Fundamentação teórica

RNN, ou Rede Neural Recorrente, é um modelo de aprendizado profundo que leva em consideração a recorrência dos dados, ou seja, como as informações são dispostas ao longo do tempo, já que elas dependem das informações anteriores.
Podemos citar como exemplo o Chat GPT, já que para gerar uma resposta ele considera as palavras já geradas para gerar as próximas.
Para que o modelo pudesse capturar essas sequências, os áudios de entrada foram transformados em vetores utilizando o modelo wav2vec2, enquanto que os áudios de saída foram transformados em espectrogramas, para que os dados pudessem ser extraídos de forma efetiva representados de forma sequencial.

3. O modelo implementado

O modelo utilizado foi o LSTM, que é uma RRN importada do keras. Foram utilizadas 4 camadas, sendo uma delas de entrada, duas camadas LSTM, uma camada dense e uma camada de reshape.

4. Resultados

A base de dados utilizada foi a Maestro V3, que pode ser encontrada na página web abaixo:
https://magenta.tensorflow.org/datasets/maestro

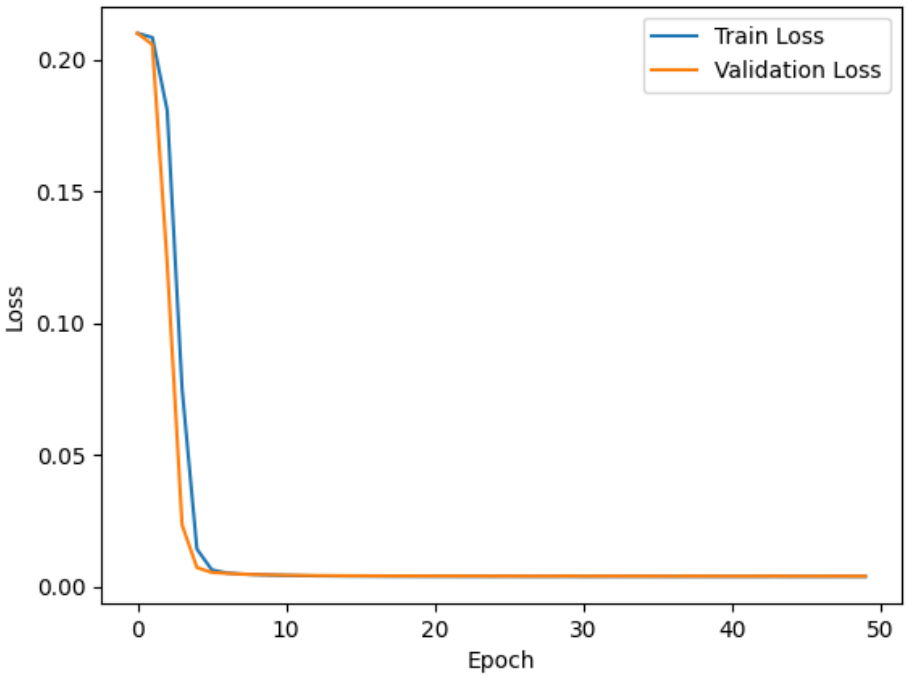
Foram selecionados aleatorialmente 200 arquivos MIDI, que foram recortados para que tivessem duração entre 3 e 14 segundos, sendo a maioria com duração próxima a 10 segundos. Inicialmente eu queria selecionar mais arquivos e com durações maiores, mas houveram complicações na hora de gravar os áudios cantarolando cada música, principalmente devido à complexidade de cada uma delas. Eu utilizei a aplicação Audacity para gravar os áudios de cantarolado enquanto escutava as músicas de piano, ajustando o resultado para que a sua duração correspondesse à dos arquivos de música. Após muitas tentativas e um processo demorado para criar esses novos dados de entrada, eu decidi reduzir a quantidade de dados, que inicialmente seria 300. A duração originalmente também era entre 7 e 14 segundos,

o que foi alterado posteriormente para que eu conseguisse gerar no mínimo 200 dados.

Para os dados de entrada (áudios .wav) foi usado o modelo wav2vec2, que extraiu os embeddings de cada áudio e os salvou em arquivos do tipo .npy. Durante o treinamento, foi necessário adicionar um padding aos embeddings, para que todos tivessem um tamanho de 699 (tamanho do maior embedding).
Para os dados de saída (arquivos MIDI) primeiro foi necessário convertê-los para .wav utilizando o programa fluidsynth, para então converter os resultados para espectrogramas, utilizando a biblioteca librosa. Os espectrogramas foram convertidos para escala de cinza e normalizados, dividindo a imagem com o numpy por 255.

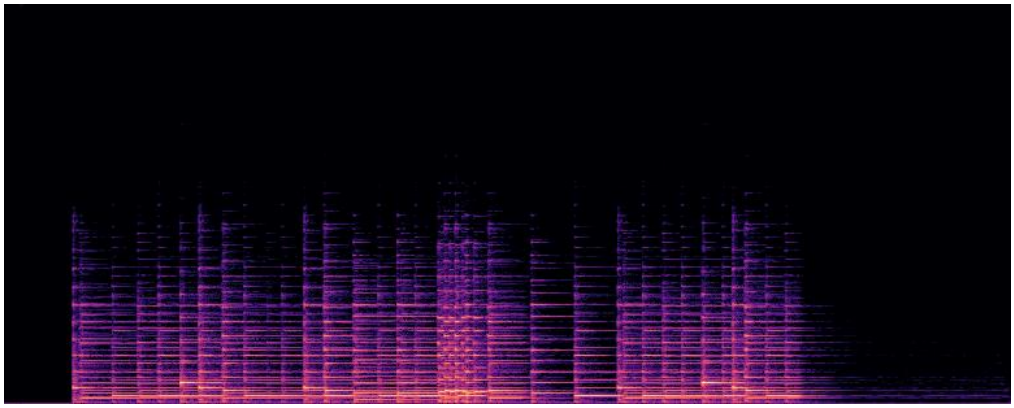
A base de dados foi dividida em 20% de validação e 80% de treino. Como o problema é de recorrência, não vi uma maneira clara de separar os dados de teste, então apenas criei novos áudios (cantarolados) para observar o resultado gerado.

Abaixo é possível ver o training loss e validation loss do modelo:

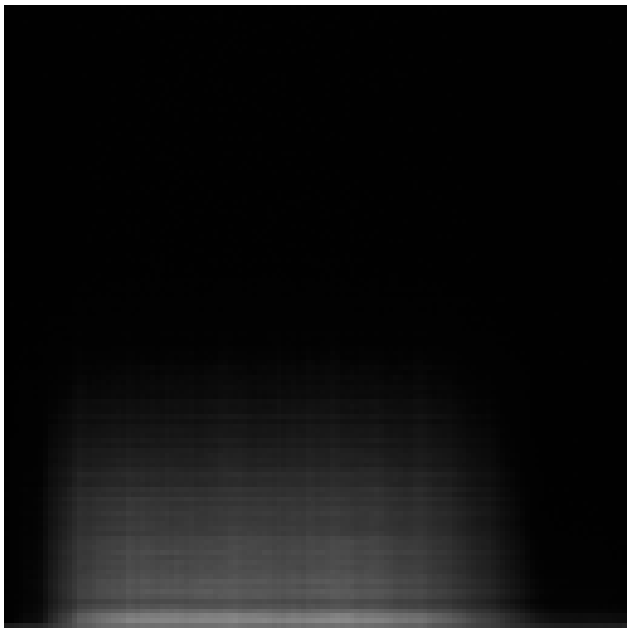


Como o problema abordado foi de regressão, não foi possível determinar a acurácia do modelo, nem criar uma matriz confusão. Foi observado, no entanto, que os espectrogramas gerados pareciam estar consistentes com os áudios, mas não foi viável determinar com precisão a qualidade do modelo pois houveram complicações ao tentar transformar esses espectrogramas em áudio.

Tentei utilizar o primeiro espectrograma da base de dados para comparar com um espectrograma gerado pelo modelo utilizando a mesma entrada de áudio, e pude observar a semelhança, o que não pode ser determinada com grande precisão devido à limitação de visualização desses dados em forma de áudio. Abaixo estão duas imagens comparando esses espectrogramas:



Espectrograma original, gerado diretamente de um dos áudios da base de dados.



Espectrograma gerado utilizando o modelo treinado.

O modelo foi treinado utilizando um processador Ryzen 5 5700x3D, localmente. Os códigos foram feitos utilizando o VS code como IDE, e os passos foram separados em diferentes arquivos.

Como utilizar o modelo: para executar e testar o programa, você pode executar o arquivo runModel.py, que em seguida irá solicitar um diretório/path para um áudio .wav. Após passado esse diretório, o modelo irá criar um espectrograma criado para representar uma música.

5. Conclusão

Foi gerado um modelo de regressão que consegue transformar os cantarolados em um espectrograma que visa representar uma música, mas a qualidade do resultado foi inconclusiva e necessitaria de mais tempo de análise. O objetivo se mostrou difícil de ser alcançado, e pude perceber que, no geral, problemas de regressão costumam ser mais complicados que problemas de classificação. O resultado alcançado parece ter sido um bom “pontapé” para algo que futuramente poderia se tornar funcional, mas que ainda precisa de trabalho para ser aplicado.

Referências bibliográficas

https://aws.amazon.com/pt/what-is/recurrent-neural-network/