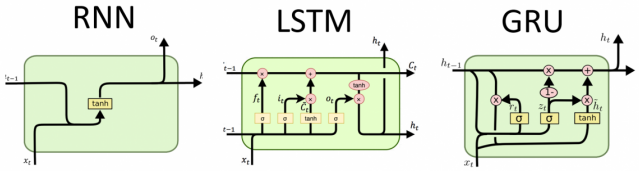
CelticRNN

Uma rede neural recorrente para a geração de música Celta



Claudemir Casa[1], Lucas Aleixo[2], Marcelo Santos[2], Eduardo J. Spinosa[3]

***Abstract*— This paper present CelticRNN, a recurrent neural network (RNN) for automatic generation of celtic music. We propose a network capable to generate music with time, instrument, chord and melody variation. In the course of the work we have shown that it is possible to get good results with a small database and with just a few lines of code.Our approach uses Long Short Term Memory (LSTM) cells to analyze compositions in MIDI format, and we use a total of 3 LSTM layers for satisfactory results.**

I. INTRODUÇÃO

É possível observar um grande avança na utilização de técnicas de Inteligência Artificial (IA) para resolver problemas do mundo real. Tais técnicas estão presente em aplicações de análise de imagens médicas [1], processamento de texto [2], sintetização de voz [3], análise de transações bancarias [4] e até mesmo em pilotos automáticos nos carros autônomos [5]. Dentre as técnicas de IA mais utilizadas, aquelas que ganharam maior destaque foram o aprendizado de máquina (Machine Learning) [6] e o aprendizado profundo (Deep Learning) [7] devido a sua grande aplicabilidade e aos ótimos resultados obtidos. O aprendizado de máquina pode ser definido como um método de análise de dados para a construção de modelos analíticos, e tem como objetivo fazer com que sistemas consigam aprender com dados a identificar padrões, e posteriormente tomar decisões sem ou com a mínima interação humana.

Ainda dentro do escopo do aprendizado de máquina, existe uma categoria de redes neurais artificiais capaz de analisar séries temporais, e são chamadas de Redes Neurais Recorrentes (RNN) [8]. Esse tipo de rede neural é capaz de aprender relações temporais. Além da entrada padrão, a rede recebe também como entrada aquilo que percebeu (processou) anteriormente no tempo, por isso são comumente referenciadas como *"redes que lembram"*. O potencial desse tipo de rede é tão impressionante que elas são capazes de analisar dados do mercado financeiro, gerar textos e criar música. Para a implementação deste trabalho, utilizamos um tipo especial de arquitetura recorrente com camadas de LSTM's, que são camadas com particularidade que não existem nas redes recorrentes comuns. Um exemplo, é que esse tipo de arquitetura possui portões como os de entrada, saída e de esquecimento responsáveis por controlar o fluxo da informação.

*IMAGO Research Group

[1]Universidade Federal do Paraná - UFPR, Departamento de informática, <claudemir.casa@ufpr.br> http://claudemir.casa

[2] Universidade Federal do Paraná - UFPR

[3]Universidade Federal do Paraná - UFPR, Departamento de informática, <spinosa@inf.ufpr.br> http://www.inf.ufpr.br/spinosa/

II. GERAÇÃO DE MÚSICA COM LSTM

A. *LSTM's*

As redes LSTM (Long Short-Term Memory) são um tipo especial de rede recorrente que são utilizadas em séries temporais, como por exemplo, arquivos de áudio. Elas não processam apenas um ponto dos dados como imagens, mas sequências inteiras de dados, como vídeos e voz. Essas redes são compostas por células com portões de entrada, saída e esquecimento e são capazes de lembrar valores durante períodos arbitrários de tempo. Os portões são responsáveis por regular o fluxo de entrada e saída da célula, assim como apresentado na figura 1.

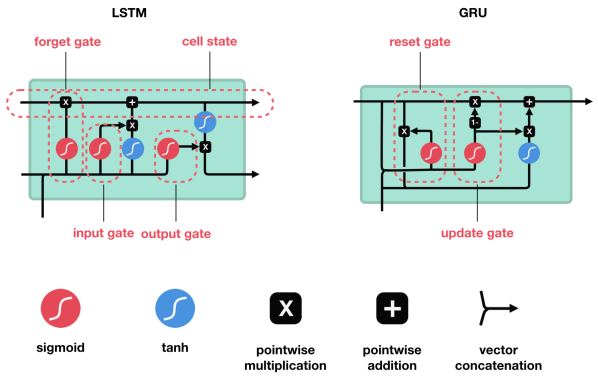


Fig. 1. Visão interna de uma célula LSTM à esquerda

Redes LSTM são utilizadas para classificar, processar e realizar predições com base em séries temporais e podem ser utilizadas em inúmeras aplicações reais, sendo utilizadas para a geração de musicas no formato MIDI[1] no escopo deste trabalho.

[1]É uma linguagem padronizada para escrever musica e reproduzi-la em diferentes sintetizadores.

B. Representação musical

As redes recorrentes são ótimas para a geração de textos, e isso é perfeito, pois a música naturalmente é composta por representações textuais [9]. Em um arquivo MIDI a música, assim como em uma partitura, é representada por uma série de símbolos textuais. Por exemplo, quando precisamos representar uma nota assim como o **Dó**, utilizamos a notação europeia que define a letra **C** para representar essa nota. Também é possível dizer em qual oitava a nota deve ser tocada, se ela possui um acidente musical ou até mesmo a duração da nota. Na verdade a notação europeia não possui tantos símbolos representativos, porém o formato MIDI suporta tanto as representações da partitura quanto da notação europeia. A figura 2 apresenta uma representação de um trecho musical expresso em uma partitura.

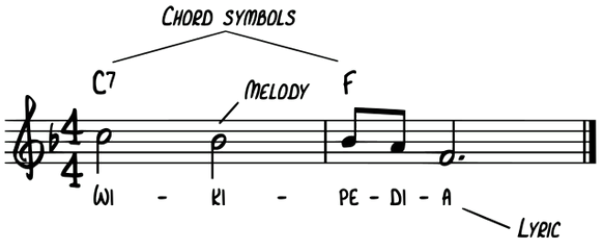


Fig. 2. Trecho de música representada em uma partitura.

Nossa maneira de representar trechos musicais para que a rede neural consiga interpretar, é codificar as informações mais relevantes dentro da música em pequenos trechos textuais. Nós realizamos um pré-processamento no dataset de arquivos MIDI e separamos os trechos de notas e acordes anexados com informações relevantes, como o instrumento utilizado para a execução, o tempo e o offset[2][3]. De forma geral, a etapa de pré-processamento pode ser expressa da seguinte forma:

1) Realiza a leitura de todos os arquivos MIDI;
2) Percorre a coleção de notas e acordes de cada música separadas por instrumento;
3) Salva um arquivo binário contendo sequências de string no formato: <instrumento> <nota ou acorde> <duração> <offset>;

Para armazenar o instrumento decidimos utilizar um número universal da representação MIDI, isso é necessário para reduzir o tamanho do vetor final de classes e consecutivamente reduzir o consumo de memória utilizada no treino. Codificamos também notas simples e conjuntos de notas para formar acordes, que são interpretadas posteriormente como um vetor de notas e analisadas da seguinte forma: se o vetor é unitário, então é interpretado como uma nota, caso contrário é interpretado como um acorde.

Nós treinamos nossa rede com um número reduzido de exemplos e iterações. Utilizamos um dataset com 115 músicas celtas no formato MIDI em um total de 200 iterações. Para obter melhores resultados recomendamos que se utilize maiores números, tanto para amostragem quanto para as iterações. Alteramos a estrutura da rede aumentando o número de camadas e modificando a quantidade de filtros, porém não obtivemos melhores resultados. Então decidimos manter a rede com três camadas LSTM, cada camada com um filtro de 512. Utilizamos também a linguagem Python para a codificação do algoritmo devido aos diversos recursos disponíveis. Também utilizamos a biblioteca music21 [10] do MIT para a codificação/decodificação das representações textuais.

[2]É a distância entre notas e acordes dentro da música.

[3]A saída é um vetor de trechos de strings assim como: ['0 D7 1.0 4.7', '50 G- 3.0 5.0', '2 C# G2 A7 2.7 4.3'].

III. CONCLUSÃO

Apresentamos no decorrer deste trabalho uma forma de gerar música automaticamente utilizando recursos de redes neurais recorrentes e representações no formato MIDI. Ao final de nossa implementação e experimentos, concluímos que essa abordagem pode ser utilizada sem problemas para o escopo do problema retornando bons resultados.

APÊNDICE

O código-fonte necessário para executar a rede neural e realizar experimentos está disponível em https://github.com/claudemircasa/celticrnn. O arquivo **lstm.py** deve ser utilizado para o treino da rede neural e o arquivo **predict.py** deve ser utilizado para realizar as predições.

RECONHECIMENTO

Gostaríamos de referenciar o trabalho de Sigurður Skúli Sigurgeirsson que foi crucial para a nossa proposta. Ele disponibilizou o código fonte para um gerador de música clássica que foi alterado para atender nossas necessidades. O repositório oficial do projeto original pode ser encontrado em https://github.com/Skuldur.

REFERENCES

[1] LITJENS, Geert et al. A survey on deep learning in medical image analysis. Medical image analysis, v. 42, p. 60-88, 2017.
[2] YIN, Wenpeng et al. Comparative study of CNN and RNN for natural language processing. arXiv preprint arXiv:1702.01923, 2017.
[3] KANEKO, Takuhiro et al. Sequence-to-Sequence Voice Conversion with Similarity Metric Learned Using Generative Adversarial Networks. In: INTERSPEECH. 2017. p. 1283-1287.
[4] FU, Kang et al. Credit card fraud detection using convolutional neural networks. In: International Conference on Neural Information Processing. Springer, Cham, 2016. p. 483-490.
[5] BOJARSKI, Mariusz et al. End to end learning for self-driving cars. arXiv preprint arXiv:1604.07316, 2016.
[6] BALTRUŠAITIS, Tadas; AHUJA, Chaitanya; MORENCY, Louis-Philippe. Multimodal machine learning: A survey and taxonomy. IEEE Transactions on Pattern Analysis and Machine Intelligence, v. 41, n. 2, p. 423-443, 2018.
[7] POUYANFAR, Samira et al. A survey on deep learning: Algorithms, techniques, and applications. ACM Computing Surveys (CSUR), v. 51, n. 5, p. 92, 2019.
[8] LIPTON, Zachary C.; BERKOWITZ, John; ELKAN, Charles. A critical review of recurrent neural networks for sequence learning. arXiv preprint arXiv:1506.00019, 2015.
[9] SCRUTON, Roger. Understanding music: Philosophy and interpretation. Bloomsbury Publishing, 2016.
[10] CUTHBERT, Michael Scott; ARIZA, Christopher. music21: A toolkit for computer-aided musicology and symbolic music data. 2010.