Composição de música utilizando LSTM

Alexandre de C. Araújo[1,2] **Nelia Cantanhade Reis**[1,2] **André Monteiro**[1]

[1]Departamento de Informática - Universidade Federal do Maranhão(UFMA)
Av. dos Portugueses, 1966 Bacanga São Luís – MA, 65080805

[2]Núcleo de Ciência Aplicada – Universidade Federal do Maranhão(UFMA)

alexandrearaujo@nca.ufma.br, nelia.reis@nca.ufma.br, andrefelpi@gmail.com

Abstract. *This paper presents a music composition method using a Recurrent Neural Network. The proposed methodology aims to compose musical pieces from a chosen musical style close enough to reality so that a listener can't tell if the music was composed by a human or by a computer. For that end, we use LSTM, a Recurrent Neural Network capable of remembering information through a period of time, something helpful in music composition. The tests applied show good music composition capabilities, as show by the avaliation of the test listeners.*

Resumo. *Esse artigo apresenta um método de criação de música através de uma Rede Neural Recorrente. A metodologia proposta visa criar músicas de um determinado estilo musical próximas o suficiente da realidade tal que um ouvinte não consiga ou tenha dificuldade em identificar se a mesma foi composta por um humano ou não . Para tal objetivo utilizamos LSTM, um tipo de Rede Neural Recorrente capaz de lembrar informação através de um período no tempo, algo útil para composição musical. Os testes realizados mostram um bom desempenho na composição da música, obtendo boa avaliação dos ouvintes.*

1. Introdução

Música é uma mídia que o ser humano consegue descrever e compor com uma quantidade imensa de detalhes. Contudo essa habilidade se prova extremamente elusiva para os computadores atualmente. Neste trabalho usamos uma LSTM, pois esse tipo de rede possui a capacidade de manter informação por longos prazos de tempo, se encaixando muito bem no caso de músicas, uma vez que estas podem ser interpretadas como uma série temporal [Boulanger-Lewandowski et al. 2012].

Usamos a LSTM de maneira similar à [Huang and Wu 2016]. A diferença se dá na maneira como os dados são processados para a entrada na rede. No método que o autor propõe, os dados são representados em um "piano roll"que tem divisões temporais e em que cada divisão é uma lista das notas sendo tocadas naquele momento, enquanto que no método proposto, treinamos a rede nota à nota.

Neste trabalho, o foco será na composição de música clássica. O objetivo final é a criação de modelo que seja capaz de compor música de tal forma que um ouvinte à classifique como clássica

2. Materiais e Métodos

Com esse objetivo em mente podemos assumir que transições entre certas notas e conjunto de notas aparecem mais vezes do que outros, com isso em mente a proposta é a criação de um sistema que:

- Utiliza músicas clássicas como uma base para o aprendizado.
- Seja uma rede neural que tenha uma memória que lembre as transições passadas e saiba quando a informação passada de uma nota não irá ter influência negativa no resultado final.

Com esses objetivos em mente foi utilizada uma rede LSTM(Long short-Term Memory), um tipo de rede que trabalha com memória e é recursiva, baseada no fato de que seres humanos não começam a pensar a cada segundo e sim baseados no entendimento adquirido do mundo, que o pensamento do ser tem persistência.

2.1. Músicas

Foram utilizadas duas bases, *96 músicas clássicas*, que consiste de 96 músicas escolhidas aleatoriamente dentre uma base maior de 5000 músicas e *20 músicas Bach*, que consiste somente de músicas cujo arquivo contém somente um instrumento, piano, de músicas compostas por Bach. As músicas foram obtidas em formato MIDI, um formato universal na indústria da música.

2.2. LSTM

As redes neurais LSTM são um tipo de rede RNN (Recurrent Neural Networks). Tais redes além da entrada normal contêm uma entrada de uma iteração anterior como demonstrado na Figura 1.

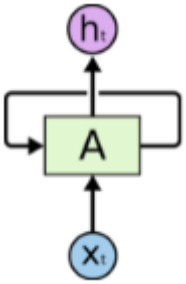

Figura 1. RNN

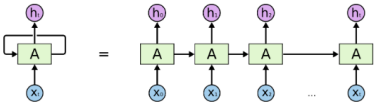

Figura 2. Forma estendida de uma RNN.

A Figura 1 mostra a rede em uma representação gráfica na forma reduzida, porém como o laço pode tornar a compreensão da rede complicada a Figura 2 demostra uma representação estendida da rede.

As RNNs tem como o apelo principal o fato de que elas são capazes de conectar informações antigas com uma informação atual [Karpathy and Fei-Fei 2015] e no caso de informações relativamente recentes elas funcionam muito bem. Porém, existem casos que a informação necessária está muito distante e cria a possibilidade da rede não conseguir conectar a informação ao ponto no qual ela é necessária.

As LSTMs foram projetadas especificamente para evitar o problema com informações de longo prazo, uma vez que essa habilidade se prova útil na maioria dos casos.

O ponto principal para o funcionamento da LSTM é uma célula de estado, a linha horizontal no topo da Figura 3. Essa célula é como uma correia transportadora que deixa a informação passar inalterada.

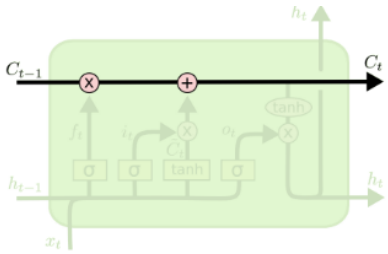

Figura 3. Célula de estado.

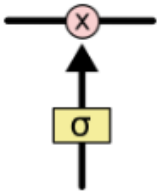

Figura 4. Portão de passagem de informação.

A LSTM tem o poder de retirar ou adicionar informação a essa célula através de "portões", estes são uma maneira de permitir ou não que informações entrem na célula estado.

Esse "portão", como exemplificado na Figura 4 é composto por uma função sigmoide que tem como saída números entre zero e um, onde zero significa que toda a informação é bloqueada e um deixa toda informação passar. Uma LSTM contem três destes "portões".

O primeiro desses "portões" serve para decidir se a informação atual da célula será mantida, também chamado de *Forget Gate Layer*.

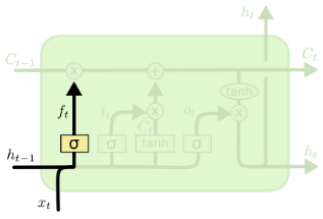

Figura 5. *Forget Gate Layer.*

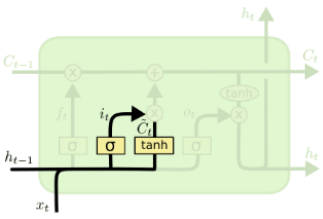

Figura 6. *Input Gate Layer.*

No próximo "portão" será tomada a decisão de qual informação nova irá entrar na célula. O segundo "portão" é chamado de *Input Gate Layer*, ele é acompanhado de uma camada com uma função tanh que cria um vetor dos valores para os novos candidatos.

Há então a junção dos dois "portões" para atualizar o estado antigo da célula, para isso multiplicamos o valor da célula pelo valor do primeiro "portão" e em seguida adicionamos os novos valores resultantes do segundo "portão" a célula.

E finalmente o último "portão", o *output gate*, decidirá a saída da rede. Essa saída será uma versão filtrada da célula onde o valor final é o valor da célula passado através de uma função *tanh*, com valores entre -1 e 1, multiplicado pela saída do "portão".

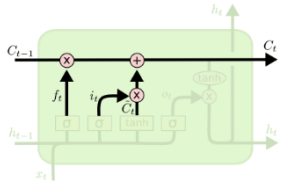

Figura 7. Atualização da célula estado.

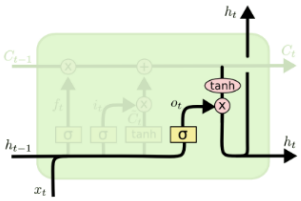

Figura 8. *Output gate*

2.3. Método Proposto

A arquitetura utilizada foram de 3 LSTM. As músicas são vistas como uma grande sequência de notas. Na primeira LSTM temos a entrada nota a nota para a rede. A saída da primeira LSTM é então passada como entrada para a segunda LSTM e o mesmo se repete para a terceira. A saída final é a próxima nota prevista considerando a nota de entrada e o contexto definido pela célula estado.

A geração da música é feita a partir de uma nota inicial aleatoriamente escolhida e então o modelo vai prever as 500 notas que formarão uma música que começa com a nota escolhida.

3. Resultados

Serão apresentados detalhadamente 2 casos de teste. A diferença única entre eles é a base de treino que foi apresentada para a rede. Em ambos os testes a rede treinou por 200 épocas.

O primeiro teste tinha como base 96 músicas aleatoriamente escolhidas de vários autores clássicos. O segundo teste tinha uma base consideravelmente menor por termos de tempo de execução. Foram-se utilizados somente 20 músicas, todas elas de Bach.

Para validação dos resultados foi feito um questionário, que foi aplicado à 20 ouvintes leigos, para avaliar as quatro músicas produzidas pela rede. A primeira música, nomeada *96 músicas clássicas*, era o resultado do treinamento completo do primeiro teste. A segunda música, nomeada *Bach 20 músicas inexperiente*, foi obtida na metade do treinamento do segundo teste e usada para avaliar a diferença entre a qualidade da composição no meio e no fim do treinamento. As duas últimas músicas, nomeadas *Bach experiente primeiro* e *Bach experiente segundo*, foram geradas com o modelo completamente treinado do segundo teste. Estas músicas podem ser acessadas em *https://soundcloud.com/mono-regent/sets/si-musics/s-13Nye*

De cada música foi questionado, utilizando uma escala de 1 a 5, o quanto o ouvinte reconhecia a amostra como música, onde 1 não havia reconhecimento nenhum e 5 o ouvinte reconhecia como música. Em seguida, foi perguntado se o ouvinte classificaria aquela música como clássica ou não.

Analisando as Tabelas 1 e 2 chegamos a certas conclusões. *96 musicas clássicas* obteve a menor nota média pois seu modelo, por ser treinado com músicas de vários compositores, não consegue chegar ao perfil de música clássica, visto que cada compositor tem sua interpretação do que é música.

As três músicas obtidas a partir da base *20 músicas Bach* obtiveram resultados

Nome	Nota 1	Nota 2	Nota 3	Nota 4	Nota 5	Média
96 musicas classicas	**2**	**2**	**6**	**7**	**3**	**3,35**
Bach 20 musicas inexperiente	**1**	**1**	**3**	**6**	**9**	**4,05**
Bach 20 musicas experiente primeiro	**1**	**0**	**3**	**10**	**6**	**4**
Bach 20 musicas experiente segundo	**0**	**1**	**6**	**6**	**7**	**3,95**

Tabela 1. Notas das músicas compostas

Nome	Sim	Não	Talvez
96 musicas classicas	**8**	**5**	**7**
Bach 20 musicas inexperiente	**12**	**2**	**6**
Bach 20 musicas experiente primeiro	**10**	**4**	**6**
Bach 20 musicas experiente segundo	**9**	**4**	**7**

Tabela 2. Classificação das músicas compostas

parecidos. A grande diferença entre elas e a música obtida da primeira base da-se pela configuração dos arquivos das músicas, vários instrumentos na primeira base contra somente 1 instrumento na segunda base.

Na classificação como música clássica, o padrão que vemos nas notas continua, com a música da base *96 músicas clássicas* tendo o pior índice, enquanto as músicas da segunda base tem um resultado muito parecido entre elas.

4. Conclusão

O trabalho obteve sucesso em compor músicas, apesar das mesmas não conseguirem obter um grande consenso quanto a sua classificação. Vale ressaltar que os ouvintes eram todos leigos no campo da música clássica, o que pode levar à uma má classificação das músicas compostas. Como trabalho futuro, desejamos superar o obstáculo do número de instrumentos que um arquivo MIDI pode ter, para que possam ser compostas músicas de vários pretendendo melhorar os resultados obtidos.

Referências

Boulanger-Lewandowski, N., Bengio, Y., and Vincent, P. (2012). Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *arXiv preprint arXiv:1206.6392*.

Huang, A. and Wu, R. (2016). Deep learning for music. *arXiv preprint arXiv:1606.04930*.

Karpathy, A. and Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137.

Sun, Z., Liu, J., Zhang, Z., Chen, J., Huo, Z., Lee, C. H., and Zhang, X. (2016). Grammar argumented lstm neural networks with note-level encoding for music composition. *CoRR*.