Desenvolvimento de Composições Musicais Procedurais Para Jogos Eletrônicos Utilizando Aprendizado Profundo

Marcus Daniel de Almeida
Dept. de Ciência da Computação
IF Sudeste MG
Rio Pomba, MG, Brasil
maarcusalmeida@gmail.com

Lucas Grassano Lattari
Dept. de Ciência da Computação
IF Sudeste MG
Rio Pomba, MG, Brasil
lucas.lattari@ifsudestemg.edu

Matheus de Freitas Oliveira Baffa
Dept. de Computação e Matemática
Universidade de São Paulo
Ribeirão Preto, SP, Brasil
mbaffa@usp.br

Frederico Miranda Coelho
Dept. de Ciência da Computação
IF Sudeste MG
Rio Pomba, MG, Brasil
frederico.coelho@ifsudestemg.edu

***Abstract*—No contexto dos jogos eletrônicos, a música está relacionada à uma boa experiência de jogo, por proporcionar imersão e melhor aproveitamento dos recursos de *game design*, como elementos dramatúrgicos e de sons do ambiente. Atualmente, o estudo e desenvolvimento de novas técnicas de geração de áudio procedural para jogos tangem a utilização de metodologias inteligentes, como as Redes Neurais Artificiais, para a produção de conteúdo audiovisual. As Redes Neurais Recorrentes (RNN), mais especificamente as de Memória de Longo Prazo (LSTM-RNN), apresentam resultados promissores para problemas de geração de áudio para jogos. Na atualidade existem projetos que visam a implementação da LSTM para a sintetização automática de músicas, por exemplo, o Google Magenta. Desta forma, neste trabalho foram utilizados modelos de Redes Neurais Profundas, como a Performance RNN da Magenta, em conjunto com o GANSynth, para a geração procedural de trilhas sonoras em jogos eletrônicos. A partir de um conjunto de músicas de *games* existentes, como *Final Fantasy*, foram criadas composições artificiais. Essas foram aplicadas a um jogo protótipo desenvolvido para a avaliação do resultado por parte de entrevistados. As avaliações foram realizadas seguindo o conceito de "teste de Turing musical", no qual 36 participantes avaliaram o quão agradável e imersiva eram as trilhas, bem como em tentar adivinhar se eram gerados por computadores ou por humanos. Os resultados obtidos neste experimento demonstraram uma ótima capacidade em gerar composições artificiais com a mesma percepção de qualidade se comparada a das composições humanas.**

***Index Terms*—geração procedural, jogos eletrônicos, músicas para jogos, aprendizado de máquina, inteligência artificial**

I. Introdução

A música está relacionada com experiências que envolvem o prazer como assistir um filme, uma apresentação de teatro, um concerto musical e muitas outras. Uma tendência artística que tem se intensificado nos últimos anos são as trilhas sonoras para jogos digitais. Nestas aplicações audiovisuais, as músicas utilizadas atuam como recursos para melhorar a experiência dos usuários e proporcionar a sensação de imersão no jogo [9], [18].

Um dos efeitos das músicas, de maneira geral, é o estímulo das células cerebrais, podendo melhorar o humor de quem as experiencia [1]. Quatro características que influenciam o humor de alguém observadas ao escutar uma música são (i) o volume, (ii) o timbre, (iii) o ritmo e (iv) as dissonâncias. O volume está relacionado com a intensidade e excitação do som. O timbre está relacionado com o "quão agradável" a música soa aos ouvintes. O ritmo, por sua vez, é particionado em três características: força, regularidade e tempo, podendo com suas variações gerar sensações diferentes. Por fim, a dissonância é a justaposição de duas ou mais notas formando uma discordância harmônica [15], [22].

A música pode contribuir para o realismo e o entendimento da narrativa do jogo ao proporcionar a sensação de espaço, de tempo, de caracterização e de atmosfera. Esta possui a capacidade de evocar emoções e, consequentemente, auxiliar ou prejudicar a imersão do jogador no universo experimentado junto ao jogo. A música de fundo cativa e desperta seu interesse. Seus efeitos podem levar quem estiver jogando a desconsiderar os ruídos externos e focar de maneira mais eficiente em sua partida [22]. O áudio, como elemento de *game design*, é um dos principais fatores que contribuem para uma melhor experiência de jogo e imersão no ambiente virtual [9].

O jogo *Grand Theft Auto: San Andreas* por Rockstar Games em 2004 é lembrado pela utilização da música como elemento imersivo. Venceu diversos prêmios, incluindo o de Melhor Trilha Sonora e Melhor Performance Humana do Video Game Awards de 2004 [3]. Durante a partida, ao dirigir um carro, pode-se ouvir diversas músicas no rádio combinados com os sons presentes na cidade, estabelecendo um cenário próximo da realidade. Além disso, é possível modificar as músicas ouvidas alternando as estações de rádio. Em contraponto

existem jogos como *The Elders Scrolls IV: Oblivion* por Bethesda Game Studios em 2006 que apesar de apreciado pelo público e pela crítica, há certos momentos em que o áudio pode causar desconforto. Isso ocorre, por exemplo, quando alguns gritos, sons e músicas são ouvidos com um volume muito elevado [3].

No jogo *Super Mario World* por Nintendo em 1990 quando o avatar utiliza o dinossauro *Yoshi* como montaria, ele obtém novas habilidades e é necessário enfatizar isso ao jogador. Para assim representar, é adicionada a música uma camada de percussões do instrumento Bongô. Mantendo a consistência, quando o Mario não está mais utilizando o *Yoshi*, a camada de som é retirada da música. A estratégia para se criar esse efeito poderia ter sido feita realizando uma mudança completa da música, mas isto afetaria o suave fluir da jogatina, sendo assim, o uso de camadas musicais foi uma melhor opção. Apesar do jogo *Super Mario World* ter sido criado há mais de 30 anos, o uso da sobreposição de camadas instrumentais não é uma técnica que se tornou antiquada. Ela ainda é muito utilizada, por exemplo nos jogos *Journey* desenvolvida por *thatgamecompany*, em 2012 e *Titan Souls* por Acid Nerve em 2015 [21].

O jogo *Journey* é um título consagrado em seu meio, sendo indicado ou mesmo vencendo premiações diversas sob várias categorias, como o BAFTA (Melhor Trilha Original) e o Grammy, quando foi a primeira trilha musical de videogame a concorrer ao prêmio de Melhor Composição para Mídia Visual [10], [17]. No jogo, a construção de camadas pode ser percebida, por exemplo, quando o avatar aciona o primeiro dos dispositivos necessários para a construção de uma grande ponte que o levará para o próximo destino. Nessa ocasião, a música surge com um orquestral adicionando um contrabaixo. Após a ativação do segundo mecanismo, um violoncelo solo é adicionado à orquestra deixando a música mais bem definida, anteriormente difusa [21]. Sendo assim, este jogo usa da estratégia de empregar uma música base e alternar os instrumentos como recurso para que de forma coerente com a situação do avatar, camadas instrumentais sejam adicionadas a música para desta maneira extrair dinamicamente as emoções certas nos momentos devidos.

As músicas produzidas por algoritmos são estudadas, no mínimo, desde o século XVIII, com parte das melodias sendo pré-compostas por Mozart e Haydn, além de algumas terem sido desenvolvidas com procedimentos aleatórios no século XX por Cage e Xenakis [16]. Em sequência, Hiller et al. [14] foram os primeiros a utilizarem algoritmos computacionais para compor músicas, por meio de Cadeias de Markov e Gramáticas Generativas.

Para abordagens mais modernas envolvendo composições procedurais, tem-se o uso de sistemas especialistas baseados em regras. Em 1996, Cope [5] desenvolveu experimentos em Inteligência Musical utilizados para compor músicas com diferentes estilos musicais históricos. Recentemente, com a popularização do Aprendizado Profundo (*Deep Learning*), aumentou-se o interesse na aplicação de Redes Neurais Profundas para diferentes áreas criativas, como a composição de músicas.

Com o crescente poder computacional, juntamente com a grande quantidade de dados existentes, tornou-se viável desenvolver modelos de Aprendizado de Máquina baseados em Redes Neurais, com bastante robustez. Por meio de bibliotecas como *TensorFlow, Keras* e *PyTorch*, a implementação de algoritmos para *Machine Learning* é facilitada, tornando acessível o uso destas tecnologias não só por grandes corporações, mas também para o pequeno empreendedor, com um determinado conhecimento tecnológico.

Dentre as atuais tecnologias computacionais capazes de compor músicas de maneira autônoma, destaca-se o projeto Google Magenta[1]. Este projeto é desenvolvido pela equipe do Google Brain e está disponível com código-fonte aberto para artistas e desenvolvedores que desejam trabalhar, principalmente, em projetos artísticos que envolvem músicas.

Usufruindo do poder de processamento computacional atual, da grande quantidade de dados existente que encontram-se disponíveis na internet, bem como dos modelos de Aprendizado de Máquina como as Redes Neurais Profundas, surgem alguns questionamentos. Dentre eles: Seria possível uma Rede Neural ser artisticamente tão criativa quanto um humano? Se essas redes podem se tornar especialistas naquilo para o qual foram treinadas? Elas podem criar alguma coisa que possa ser confundida com a criação de uma pessoa?

Por isso, o presente trabalho tem por objetivo desenvolver uma metodologia baseada em Redes Neurais Profundas, que seja capaz de criar trilhas sonoras para jogos digitais a partir de modelos de Aprendizado de Máquina e músicas inseridas a priori em uma base de dados. Para avaliar o método, o trabalho foi colocado em teste para que se pudesse averiguar se as musicas criadas proporcionavam a imersão deseja no jogador.

II. Trabalhos Relacionados

Como um trabalho sobre geração de música procedural, o *MidiNet* proposto por Yang et al. [24] é inspirado no *WaveNet*, desenvolvido por Oord et al. [19]. Este tipo de aplicação emprega CNNs para gerar ondas realistas de áudios musicais. Desta forma, este trabalho traz modificações na arquitetura típica dessa Rede Neural para aprender as distribuições de melodias. Ele conta com um acréscimo de componentes, encontrado em Redes Generativas Adversariais (GAN), como a Rede Geradora e também a Rede Discriminadora, tendo assim uma arquitetura DC-GAN.

No *MidiNet* [24], como entrada para o gerador foram utilizados ruídos que são informados por duas camadas totalmente conectadas com 1024 neurônios e 512 neurônios para serem remodelados em uma matriz. O gerador ainda conta com mais quatro camadas de convolução transposta. O modelo ainda agrega um condicionador, que pode ser visto como o reverso do gerador, treinado simultaneamente com ele e influenciando na geração de novos áudios.

No discriminador, foram adotadas duas estratégias de decaimento para contornar o problema da dissipação do gradiente,

[1] https://magenta.tensorflow.org/

erro cujos neurônios nas camadas anteriores aprendem muito mais lentamente que os neurônios nas camadas posteriores [2]. A primeira estratégia realiza atualizações no gerador, no condicionador e no discriminador para cada iteração. A segunda estratégia foi a modelagem do discriminador utilizando duas camadas de convolução seguidas por uma camada totalmente conectada.

O modelo implementado com o Tensorflow é capaz de gerar melodias do começo, seguindo uma sequência de acordes ou uma melodia base a ele passado. Em Yang et al. [24] é desenvolvido um método cujo resultado da avaliação com usuários mostra que os áudios gerados pelo *MidiNet* possuem um desempenho que pode ser comparado com o *MelodyRNN* da Google.

Para um trabalho mais recente utilizando as DCGANs, o *WaveGAN* desenvolvido por Donahue et al. [6] é um modelo capaz de sintetizar segmentos de um segundo de áudio aprendido a partir de exemplos reais. Quando treinado, o *WaveGAN* pode aprender a gerar palavras, sintetizar áudio de instrumentos, como bateria e piano e também vocalizações de pássaros.

A arquitetura da *WaveGAN* é baseada na DCGAN proposta por Radford et al. [20]. Por esse motivo utiliza a operação de convolução transposta, mas conta com alterações como os filtros unidimensionais maiores que 5x5 e os filtros de *upsample* (que transformam vetores / matrizes menores em outros maiores) de tamanho 4 ao invés de 2 em cada camada. Outra modificação acontece no discriminador, com filtros de tamanho 25 em uma dimensão e aumentando nos passes (*strides*) de 2 para 4. O modelo de Donahue et al. [6] ainda conta com algumas outras mudanças as normalização do lote (*batch*) do gerador e do discriminador, removendo-as, além de empregar a estratégia de treinamento do modelo WGAN-GP [11]. A rede foi treinada com (i) amostras de discursos com um grande vocabulário (2,4 horas), (ii) com efeitos sonoros de bateria (0,7 horas), (iii) com vocalizações de pássaros (12,2 horas) e (iv) com sons de piano (0,3 horas).

Para avaliar as gerações foram usadas várias métricas quantitativas além de uma avaliação com juízes humanos. Como resultado, descobriu-se que o *WaveGAN* captura modos semânticos, como bumbo e caixa para efeitos de bateria. Para a vocalização de pássaros, o modelo gera sons diversos. Em relação ao piano, as composições foram coerentes com os dados de treinamento, com uma variedade de assinaturas principais e padrões rítmicos. Para o conjunto de dados de fala com vocabulário elevado, o *WaveGAN* teve resultados interessantes, comparáveis a modelos autoregressivos incondicionais [6], [19].

O trabalho de Eck e Schmidhuber [7] é um dos primeiros na utilização de LSTM para lidar com os problemas de dependências de longo prazo na generalização de músicas. Nele é possível ter uma ideia do poder da LSTM na criação de melodias. Os autores mostram que o modelo por eles implementado é capaz de aprender com sucesso a estrutura de músicas do gênero *blues* e reproduzi-las do zero ou dar continuidade a uma composição a partir de sequências de acordes a ele passado. Também é possível observar que a LSTM pode aprender a estrutura global da música melhor do que uma RNN convencional. A base de dados para o treinamento da rede foi desenvolvida a partir de um único conjunto de dados de acordes de *blues*, juntamente com melodias criadas pelo primeiro autor. Essas composições foram concatenadas com seus acordes associados para serem servirem de amostra para a base de dados final.

Foram feitos dois experimentos a fim de verificar a qualidade das composições geradas pelo modelo. No primeiro, a Rede Neural é treinada para reproduzir a progressão de acordes do aprendizado mesmo com a ausência da melodia. Para isso, a rede foi treinada usando entropia cruzada como uma função objetivo para prever a probabilidade de uma determinada nota ser ou não aplicável em seguida, dada uma entrada anterior. Como resultado verificou-se que a LSTM lidou facilmente com a tarefa sendo capaz de prever com êxito toda uma sequência de acordes.

No segundo experimento, a rede é desafiada a reproduzir tanto a progressão dos acordes quanto a melodia. O treinamento acontece de maneira similar ao primeiro passo. A rede é interrompida quando a estrutura dos acordes é memorizada e o erro de entropia cruzada é relativamente baixo. Após o aprendizado ser interrompido, a rede é iniciada com uma nota ou uma série de notas musicais para então prever a nota seguinte que por sua vez, atua como entrada para próxima previsão.

Não foi feito nenhum método estatístico para avaliar a qualidade das músicas geradas pelo algoritmo mas os autores discutem sobre alguns resultados percebidos. Segundo Eck e Schmidhuber [7], o modelo produz músicas que mixam trechos aprendidos com passagens que são condizentes com o estilo de música escolhido. Apesar da simplicidade do modelo e da base dados, esse trabalho foi de muita importância para campo da modelagem de linguagem musical com Rede Neurais.

Mauthes [16] também implementou Redes de Memória de Longo Prazo em seu modelo de geração de músicas procedurais chamado de VGM-RNN. Não foi um dos primeiros trabalhos da área a empregarem LSTMs, mas inovou ao aplicá-la para geração de músicas para jogos eletrônicos. O autor desenvolve sua base de dados com trilhas sonoras dos jogos do console NES. O algoritmo foi desenvolvido utilizando, entre outras bibliotecas o Python, o Pretty-Midi, o Tensorflow e o Keras.

O VGM-RNN contém uma única camada LSTM com 256 unidades ocultas. A camada de saída final da rede usa a função de ativação *softmax* para criar uma distribuição categórica de probabilidades de notas. Para o treinamento, foi escolhido uma taxa de aprendizado (ou *learning rate*) de 0,01, o tamanho lote (ou *batch size*) de 50 e uma duração de sequência (ou *sequence length*) de 64, o que corresponde a 4 compassos em uma composição 4/4. Também foi fixado um *pitch* MIDI entre as notas MIDI 36 e 84, o que corresponde a uma faixa de quatro oitavas de C2 a C6 [16].

O autor descobriu que o modelo por ele implementado gerou resultados coerentes após treinado entre 20 a 50 épocas.

Por ter uma base de dados pequena as gerações refletem a tonalidade das músicas, o que não é um problema. Isso quer dizer que a rede gerou notas corretas para as chaves, soando agradavelmente. No entanto, alguns problemas foram encontrados, como notas excessivamente repetidas.

Como forma de avaliar a qualidade do áudio gerado, foi feita uma pesquisa administrada *on-line* usando o Google Forms. Neste formulário são mostrados cinco clipes de dez segundos de música, como uma espécie de "teste musical de Turing", onde os participantes avaliam a qualidade das composições ao mesmo tempo em que tentam distinguir se essas foram compostas por um humano ou uma máquina. A pesquisa revelou uma média de 5,56 para cada clipe de áudio. Onde as composições humanas obtiveram 5,42 e as pela LSTM 5,67 além de obter o clipe mais bem avaliado pelos entrevistados com 6,33 [16].

III. Metodologia para Criação de Base de Dados Musical

Geralmente, os modelos generativos usados para modelar a música adquirem o aprendizado a partir de uma distribuição de dados musicais, de forma que o modelo define a probabilidade de uma determinada nota ou acorde ocorrer após uma sequência anterior de notas. Idealmente, os dados do conjunto de treinamento devem ocupar pouco espaço de armazenamento em memória, pois demandam uso intenso de processamento computacional. O formato simbólico de música MIDI (*Musical Instrument Digital Interface*), bastante empregado em jogos eletrônicos, foi escolhido por ser ideal para treinar esse tipo de modelo [16]. Milhares de arquivos MIDI podem ser adquiridos de forma gratuita na Web, como músicas clássicas, jazz e também trilhas sonoras de jogos, as quais interessam o projeto aqui desenvolvido.

Após selecionar as trilhas que farão parte da base de dados (Seção III-A), foi realizado um processo de categorização das mesmas (Seção III-B). Os arquivos foram separados em 12 conjuntos com temáticas distintas referentes as cenas de um jogo. Combinações desses grupos formam as coleções de dados para o treinamento e validação do modelo de Aprendizado de Máquina baseado em Rede Neural Artificial Profunda.

A. Seleção de Trilhas Sonoras para a Base de Dados

O primeiro passo foi adquirir o conjunto de dados, que consiste em arquivos MIDI provenientes do Web Site thefinalfantasy.com[2]. No momento da redação deste artigo, estão disponíveis o total de 227 trilhas sonoras da franquia de jogos *Final Fantasy (Final Fantasy 1* até *Final Fantasy XV*), além de uma obra derivada (*Final Fantasy Tactics*) organizada em conjuntos do primeiro ao último lançamento. A grande popularidade desses jogos e de suas trilhas sonoras que se tornaram clássicos, sendo até disponibilizadas em plataformas de *streaming* musical como Spotify e Apple Music, certamente influenciaram a seleção deste conjunto de dados. No entanto, esse não é o único fator que justifica essa escolha: o fato de ser uma franquia com muitos jogos torna possível o desenvolvimento de uma base de dados relativamente grande e homogênea, ou seja, com músicas sonoramente parecidas entre si.

Entretanto, esse conjunto de dados não foi suficiente para esse trabalho. Nas últimas gerações de músicas para os experimentos realizados, foi necessário adicionar trilhas sonoras de outros jogos para complementar os dados de treinamento, a fim de se gerar músicas com maior variabilidade sonora. Para esse acréscimo, foram utilizadas as trilhas sonoras dos jogos advindos dos consoles NES disponibilizada por nmauthes[3] na plataforma GitHuB. Esse conjunto de arquivos MIDI contém 4.194 arquivos adquiridas no Web Site VGMusic.com.

Apenas um subconjunto dessa outra base de dados escolhida foi utilizada neste projeto, um total de 94 músicas. Um processo de categorização foi realizado na primeira, separando por tema suas trilhas sonoras. Algumas músicas da segunda foram usadas para complementar esses conjuntos. O processo de categorização é explicado mais detalhadamente na Seção III-B.

B. Categorização das Trilhas Sonoras

Uma alternativa para auxiliar o treinamento da Rede Neural, a fim de se gerar músicas destinadas aos experimentos, foi a categorização das músicas da base de dados. O processo foi feito a partir da separação dos arquivos MIDI em pastas. Cada conjunto contém trilhas sonoras com um tema associado, são esses: (i) 22 músicas com melodia motivadora; (ii) 7 músicas de batalha; (iii) 18 músicas de batalha com "chefe de fase"; (iv) 37 músicas calmas; (v) 7 músicas de castelos; (vi) 21 músicas de suspense; (vii) 33 músicas excitantes; (viii) 8 músicas melancólicas; (ix) 22 músicas para cidades; (x) 21 músicas tristes; (xi) 9 músicas após vitória em batalha e (xii) 22 músicas usadas em mapas (florestas, bosques, etc).

Os títulos das músicas subsidiaram a categorização. Muitas estão nomeadas sugestivamente, como: FF1town, FF2battl, FF5worl3, etc. Para um determinado subconjunto foi necessário ouví-las e compará-las para encaixá-las empiricamente em um grupo. O intuito desse processo é ter a possibilidade de criar trilhas sonoras para cenários ou momentos específicos de um jogo. Por exemplo, para um cenário de uma floresta verde e alegre, pode-se combinar os conjuntos: músicas de mundo, calmas e motivadoras. Já para uma floresta morta e sombria, só é preciso substituir as melodias calmas e animadoras por tristes e de suspense. Isso torna possível um treinamento mais preciso e coeso, o que implica em resultados diversos e positivos nas gerações.

C. Preparação da Base de Dados

A representação de um arquivo MIDI é convertida em uma sequência de 413 eventos diferentes: 128 eventos *note-on* (Ativação de nota), 128 eventos *note-off* (desativação de nota), 125 eventos eventos de mudança de tempo (*time-shift events*) e 32 eventos de velocidade. Os arquivos MIDI advindos do

[2]https://www.thefinalfantasy.com/site/midi-collection.html

[3]https://github.com/nmauthes/vgm-rnn

conjuntos de dados original usado para o treinamento são convertidos em *Note Sequences* e em seguida em *Sequence Examples* [4].

Para o primeiro passo, é empregado um algoritmo de criação de *NoteSequences*, que é um formato de dados de processamento rápido, eficiente e mais fácil de se trabalhar do que arquivos MIDI. Obtém-se como entrada de dados a coleção de músicas MIDI e, gerando como saída, o arquivo notesequences.tfrecord contendo as sequências de notas.

Para o segundo passo, realiza-se a aplicação de um algoritmo de criação de *SequenceExamples*. Inicialmente, extrai-se as informações advindas do TFRecord criado no passo anterior e gera-se o arquivo training_performances.tfrecord, contendo um vetor de codificação com 413 dimensões de entrada e saída. Cada *SequenceExample* possui uma sequência de entradas e de rótulos que representa um desempenho da base de dados [4]. Nesse passo também é solicitada uma escolha de pacote de configuração. Neste trabalho foi utilizado o performance_with_dynamics, modelo que inclui mudanças de velocidade quantizadas.

IV. Sintetização de Músicas usando Deep Learning

Para a execução do projeto foram selecionadas arquiteturas de redes neurais baseadas em Aprendizado Profundo (*Deep Learning*) para geração de músicas. A partir de experimentações com diferentes arquiteturas disponíveis em código aberto em plataformas como o GitHub e na literatura acadêmica, os módulos da plataforma Magenta[4] (Google) obtiveram os melhores resultados sonoros. Dentre eles, para a criação das trilhas musicais foi escolhido o Performance RNN. No entanto, as criações do Performance RNN são músicas apresentadas com a sonoridade de piano. Para alcançar um resultado semelhante ao encontrados nos jogos eletrônicos, foi necessário adicionar outros instrumentos para as composições. Para a sintetização dos áudios foi utilizado o GANSynth, também disponível no Magenta.

O Performance RNN (Fig. 1) é definido pelos autores Simon e Oore [23] como uma rede neural recorrente baseada em LSTM, projetada para modelar músicas polifônicas com tempo e dinâmica expressivos. O performance RNN é capaz de gerar tempo e dinâmica expressivos a partir de um fluxo de eventos MIDI. O MIDI cronometra precisamente eventos contendo nota sim e nota não, além de especificar seu tom e velocidade. Assim, esses eventos são disponibilizados para um sintetizador padrão, a fim de se gerar o som esperado em um piano. A representação do arquivo MIDI pode ser observada na imagem central da Fig. 1. Resumidamente, o modelo não cria o áudio diretamente, apenas determina quais serão as notas tocadas, quando serão tocadas e qual será a intensidade de sua batida [23].

O GANSynth é uma ferramenta capaz de gerar áudio de alta fidelidade com as Redes Generativas Adversárias (GANs), em que é possível sintetizar áudios interpolando o timbre ao longo da peça. O GANSynth utiliza a base de dados NSynth de notas de instrumentos musicais e é capaz de controlar independentemente o tom e timbre, além de ter facilidade em trabalhar com recursos globais como a altura (*pitch*). Com uma arquitetura Progressive GAN juntamente com o aumento da resolução de frequência, há uma melhora no desempenho do método ao trabalhar com espaços harmônicos [8].

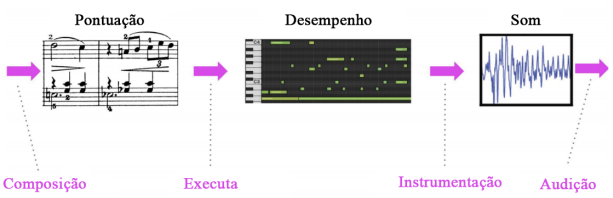

Fig. 1. Performance RNN - Processo de Geração de Música. Fonte: Jiyanbo et al. [4], traduzido.

Tanto o Performance RNN[5] quanto o GANSynth[6] estão disponíveis em código aberto e podem ser encontrados no site do Magenta. Com o Performance RNN é possível gerar composições a partir de um modelo pré-treinado via Web em uma demo disponibilizada na plataforma Google Colab. O GANSynth também possui uma demo no Colab[7] em que usuários podem sintetizar áudios e escolher quais instrumentos preferem usar. Com a instalação do Magenta pode-se treinar os modelos com conjunto de dados diferentes.

Nas próximas seções serão descritos os passos realizados em ambas as ferramentas, deixando mais claro os seus funcionamentos.

A. Treinamento das Redes Neurais Profundas

Após convertida a coleção de arquivos MIDI em *SequenceExamples* (Seção III), estes são inseridos no algoritmo durante o treinamento e a avaliação. O modelo utiliza o *Keras*, uma interface de alto nível para bibliotecas de aprendizado profundo, com um *back-end* em *TensorFlow* para implementar a Rede Neural Recorrente. O *TensorFlow* inclui o *TensorBoard*, uma *framework* de Aprendizado de Máquina que permite o acompanhamento da estatística da Rede Neural em si, com métricas e visualização de grafos, histogramas de pesos, entre outros.

Para a utilização do algoritmo de aprendizado é necessário indicar o pacote de configuração (deve ser o mesmo utilizado na geração do *SequenceExamples*), o diretório em que os pontos de verificação e os dados do *TensorBoard* para execução serão armazenados e o caminho do arquivo TFRecord de *SequenceExamples* que alimenta com dados o modelo.

Opcionalmente pode-se especificar o número de passos de treinamento (*num_training_steps*). Caso não seja especificado, o ciclo de treinamento será executado até que seja encerrado

[4]https://github.com/magenta/magenta

[5]https://magenta.tensorflow.org/performance-rnn
[6]https://magenta.tensorflow.org/gansynth
[7]https://magenta.tensorflow.org/demos/colab/

manualmente. Além disso, tem-se os hiperparâmetros (*hparmams*), usados para especificar outros elementos além dos padrões.

Neste trabalho, foram utilizados os seguintes parâmetros: três camadas ocultas LSTMs o qual cada camada contém 512 neurônios, tamanho de lote (*batch_size*) de 64, comprimento da atenção (*attn_length*) de 0 e taxa de aprendizagem (*learning_rate*) de 0,001.

O treinamento para cada combinação do conjunto de dados categorizado para cenas específicas de um *jogo* demandou entre 20 à 24 horas para obter cerca de 0,99 de acurácia e 0,01 de perda. Após a conclusão do treinamento, as configurações finais de peso adquiridas pelo modelo são serializadas e salvas no disco para serem usadas durante o processo de geração com o intuito prever novas sequências de notas.

B. Sintetização de Trilhas Sonoras

O processo de geração é mais direto e requer menos preparação do que o treinamento. Deve ser especificado o diretório usado para o trabalho de treinamento, referente ao local em que se encontra os arquivos de ponto de verificação; os hiperparâmetros, que devem ser os mesmos utilizados durante o treinamento; o diretório de saída em que os arquivos MIDI gerados serão salvos; o número de melodias que serão criadas e o número de passos, que representam o tempo de duração de cada melodia.

Para gerar novos dados MIDI, pode-se preparar a rede com uma sequência de notas (*primer_melody*), uma nota inicial (*primer_melody* com apenas uma nota no vetor), ou arquivo MIDI (*primer* MIDI). Durante a previsão, a rede usará essa informação juntamente com o arquivo de ponto de verificação mais recente para prever uma nova sequência. É possível optar por nenhum "*primer*", assim uma nota aleatória do intervalo de notas do modelo será escolhida como a primeira nota e as restantes serão geradas pelo modelo.

Para as gerações deste projeto, não foram utilizados nenhum *primer*. Optou-se por 10 arquivos MIDI de saída (*num_outputs*) para cada geração e 3000 como número de etapas (*num_steps*), resultando em 30 segundos para cada uma das 10 composições (uma etapa é equivalente a 10 milissegundos).

C. Definição de Instrumentos para as Músicas

Para sintetizar as músicas por meio do Performance RNN, foi utilizado o modelo GANSynth, também do Google Magenta. A versão demo do algoritmo disponibilizada no Colab usado por este trabalho pode ser encontrado no site do mesmo. Essa versão oferece duas alternativas para instrumentalizar as músicas. A primeira consiste em uma interpolação de instrumentos de maneira aleatória. A segunda permite utilizar até 16 instrumentos, escolher quais utilizar e o tempo relativo para cada um deles. Por ser necessário que as composições se adéquem as cenas específicas de jogos digitais, a segunda opção foi escolhida por permitir uma instrumentalização que adicionasse, as trilhas sonoras, o humor requerido.

O processo é relativamente simples: são necessários alguns passos para realizar a sintetização através do GANSynth. O primeiro passo consiste em configurar o ambiente, fazer o *download* do modelo, instanciar parâmetros, definir funções auxiliares, instalar pacotes e bibliotecas *Python*, dentre outros processos. Em seguida é necessário selecionar um arquivo no formato MIDI. É disponibilizado um áudio para teste, além da opção "submeter um arquivo do computador". Depois encontram-se as duas opções de instrumentalização. A interpolação aleatória é simples e direta, apenas executando a célula de código e o áudio é gerado e disponibilizado para *download*.

Para a outra possibilidade, é preciso definir a quantidade de instrumentos a serem disponibilizados e assim, escolher quais deles usar. A ordem dos instrumentos durante a sintetização e o tempo são especificados a partir do preenchimento de vetores. No vetor de instrumentos é possível definir a quantidade de instrumentos. No vetor de tempo dos instrumentos é possível definir um tempo relativo em um intervalo que deve iniciar em 0 e terminar em 1,0. Após esses passos basta executar a célula de código para gerar a interpolação e fazer o *download* do áudio.

V. Experimentos e Resultados

Como forma de fazer a avaliação qualitativa das músicas produzidas, realizou-se uma validação inspirada no trabalho de Mauthes [16], que administrou virtualmente uma pesquisa usando o Google Forms. No início, os participantes são questionados sobre sua experiência musical. Em seguida, são apresentados cinco clipes de áudio com cerca de trinta segundos de duração. Cada clipe é uma música original gerada pela IA ou retirada do conjunto de dados de treinamento.

O participante é informado de que pelo menos um dos áudios foi composto por um computador e pelo menos um é de uma trilha sonora real de jogos eletrônicos, composta por um ser humano. Depois de ouvir cada clipe, é perguntado se eles acham que esse foi composto por um ser humano ou um computador além de avaliar sua experiência subjetiva ouvindo a música, por meio da Escala de Likert (de 1 a 10). Dessa forma, a pesquisa atua como uma forma de "teste de Turing musical", no qual os participantes avaliam a qualidade da música ao mesmo tempo em que tentam determinar se a mesma foi composta por um ser humano ou um computador.

A seguir, o formulário avalia as composições musicais como um recurso de imersão em jogos. Após a etapa anterior, uma pergunta coringa foi adicionada. Nela o participante tem de escolher um entre dois *emojis*, indo para seções diferentes do formulário, dependendo de sua escolha. Isso é feito para sortear se os mesmos farão parte de um grupo controle ou grupo teste. O gruo controle, de forma geral, é ter um parâmetro do uso de músicas criadas em relação ao que já existe. Contribuindo assim, na verificação da disparidade significativa entre os grupos, o que mostra que há espaço para se usar modelos inteligentes.

Desta maneira foi possível que uma parte dos avaliadores fizessem o *download* e jogassem versões com trilhas sonoras

diferentes do mesmo jogo. Uma parte jogou o jogo do grupo teste, com as trilhas sonoras produzidas mediante Aprendizado de Máquina e em outra o jogo do grupo controle, com trilhas sonoras provenientes da base de dados.

Aos que jogaram o jogo do grupo teste, esses são informados que devem fazer *download* e jogar até o final e anotar no formulário a sequência de números que são exibidos ao final da partida. Essa informação foi utilizada na primeira pergunta desta seção, em que os participantes marcaram em uma caixa de seleção as sequência de números que representam as trilhas sonoras sorteadas durante sua partida. Em seguida, é mostrada uma questão pedindo a avaliação de 1 a 10 às trilhas sonoras tocadas como recurso de imersão no jogo. Ao término, é perguntado se eles consideram que as trilhas sonoras selecionadas em suas respectivas partida foram criadas por um humano ou computador.

Para aqueles que foram direcionados para o jogo do grupo controle, as perguntas são as mesmas, com exceção da seleção de músicas tocadas durante a partida. Uma vez que não se faz necessário por estas serem fixas e não serem sorteadas. Para fins de replicabilidade, todas as músicas, os jogos e os formulários criados encontram-se disponíveis em repositório[8] do GitHub.

A. *Modelagem do Jogo Avaliativo*

A fim de se testar a qualidade das músicas geradas e se estas podem possibilitar a imersão dos usuários, um pequeno jogo RPG foi desenvolvido para a aplicação dos áudios. O protótipo foi produzido por meio do *RPG Maker VX Ace* [9], da Enterbrain, um *software* para criação de jogos RPG.

O jogo foi nomeado como ”O Destino de Vhotória” e conta a história do personagem Erick. No início da partida, Erick acorda no meio de um bosque, com dor de cabeça e sem lembranças do seu passado. Ao seu lado encontra-se presente um ser espiritual, chamada Lauren, que explica que ele sofreu queda em batalha, bateu com a cabeça e ela o resgatou. Ela o teletransportou para aquele bosque e cuidou de seus ferimentos. Lauren diz a Erick que ele deve procurar por Elen, que está a sua espera. Ao encontrá-la, Elen explica que o Grande Demônio está acordando e que os moradores de Vhotória precisam das habilidades de Erick. Assim, ela dá a Erick toda força e vitalidade que possui, o que leva ao fim de sua existência. Mas antes de desaparecer, Elen diz que Patrick está a espera do herói e que ele pode fornecer equipamentos para o preparar de sua grande batalha. Erick então encontra-se com Patrick, recebendo mais informações sobre o inimigo, como o local de seu esconderijo. Patrick entrega a Erick uma armadura, arma e poções de cura, além de abençoá-lo com o poder de abrir portais. O personagem então segue sua caminhada e encontra o portal.

Uma vez dentro do esconderijo do grande demônio, Erick encontra seu inimigo e trava uma batalha que decidirá o destino de Vhotória. Caso o herói seja derrotado, ele morre e o jogo acaba. Caso Erick saia vitorioso, o Grande Demônio morre e Lauren aparece e o teletransporta de volta para o bosque, agradece-o pelo seu esforço e o jogo encerra com uma mensagem dizendo que o Reino de Vhotória tornou-se feliz e em plena harmonia.

Para o processo de sonorização do jogo, foi necessário introduzir trilhas sonoras para três cenários diferentes (bosque, masmorra e batalha (Fig. 2), além de música de vitória, derrota e de tela de título. Foram desenvolvidas três músicas diferentes para serem sorteadas durante a partida para cada cenário, com exceção da tela de título e de derrota (*Game Over*). Estas não receberam mais de uma opção de trilha sonora pelo *RPG Maker VX Ace* não fornecer um recurso para a programação necessária para esse sorteio. Todos as áudios foram editados no *software* Audacity para a retirada dos espaços em silêncio em seus inícios e fins, além de ser adicionado *fade-in* (aparecer gradual) e *fade-out* (desaparecer gradual) a fim suavizar as transições.

Fig. 2. Cenas do jogo desenvolvido. Fonte: autor.

1) Primeira Cena: Na primeira cena, o personagem acorda confuso e desmemoriado em um bosque. As músicas geradas para essa cena foram construídas a partir do treinamento do modelo Performance RNN com a combinação dos grupos: músicas tranquilas, de mundo e também de castelos e cidades. Das 10 gerações resultantes do treinamento, foram escolhidas empiricamente as três músicas que soavam agradáveis e que são distintas sonoramente entre si.

No jogo, Erick acorda confuso e ao decorrer da cena recebe informações de outros personagens, assim descobrindo um pouco mais sobre quem ele é e qual sua missão. Os personagens secundários que auxiliam o personagem principal atuaram como eventos. A cada conversa a trilha sonora sofre uma alteração para enfatizar essa obtenção de conhecimento. Para isso, para cada arquivo MIDI gerado no passo anterior, quatro versões dos áudios foram desenvolvidas no GANSynth. A começar por uma trilha sonora com instrumentos que soam como uma espécie de coral, juntamente com outros de timbres confusos. Segue-se com outra versão, em que são subtraídos alguns desses instrumentos por outros que soam mais bem definidos aos ouvidos. Até chegar na última música que será tocada quando Erick for até o esconderijo e essa está mais clara, com instrumentos com timbres mais fortes e algumas vezes até mesmo metalizados.

2) Segunda Cena: Para a cena da masmorra, nenhum evento especial é adicionado. O personagem apenas caminha pelo esconderijo até encontrar seu inimigo. Assim, não foi necessário adicionar versões diferentes de instrumentos de um mesmo áudio. Para treinamento do Performance RNN

[8]https://github.com/marcusalmda/Desenvolvimento-de-Composicoes-Musicais-Procedurais-Para-Jogos-Eletronicos-Utilizando-DL

[9]https://www.rpgmakerweb.com/products/programs/rpg-maker-vx-ace

para esse cenário, a combinação dos grupos categorizados para formar a base de dados foram: músicas excitantes, melancólicas, motivadoras e de suspense. Das dez gerações, as três músicas diferentes entre si que soam mais parecidas com algo criado por humanos foram escolhidas para a sintetização no GANSynth. Para essa cena, foram escolhidos instrumentos com timbres marcantes e metalizados ou instrumentos com timbres mais sombrios, por exemplo algo similar ao som de um órgão.

3) Terceira Cena: Para a batalha final foram utilizadas três trilhas sonoras sorteadas, com a combinação dos grupos categorizados: músicas de batalha, de ”chefão”, exitantes e motivadoras, além de um acréscimo de 22 áudios com a base de dados de músicas dos consoles NES. Esse acréscimo foi feito através de uma pesquisa e seleção no diretório com músicas a partir de chaves, como batalha e “chefão” (*boss*). Os arquivos MIDI gerados foram escolhidos da mesma maneira que nas cenas anteriores. O processo de sintetização no GANSynth foi realizado a partir de uma escolha de instrumentos de timbres mais vibrantes. Com o intuito de uma imersão maior dos jogadores durante a partida, algumas transições entre a mesma trilha sonora com um acréscimo em sua temperatura foi feita gradualmente ao longo da batalha, para levar algum estímulo ou excitação ao usuário. O recurso de aumentar ou diminuir a temperatura do áudio é disponibilizado pela própria plataforma utilizada para a criação do jogo.

4) Cenas Finais e Tela de Título: Para as cenas finais, tem-se a vitória e derrota. Caso o personagem seja vitorioso na batalha, toca-se uma entre as três músicas pré-compostas. O processo de desenvolvimento é parecido com os das cenas anteriores. O modelo performance RNN é treinado com uma combinação dos grupos categorizadas da base de trilhas dos jogos *Final Fantasy*. Os grupos selecionados foram: músicas motivadoras e de vitória. Foi necessário um acréscimo com a segunda base de dados, que também foi feito através de seleção e pesquisa de chave com músicas com títulos relacionados a vitória, como “*win*” e “*victory*”, um total de 35 áudios. As três composições que soam com algo feito por humanos que mais se diferem entre si passaram pela sintetização no GANSynth. Essa instrumentalização foi feita com instrumentos de timbres mais claros e alegres.

Pode acontecer do personagem sair derrotado da batalha, mas como no caso da tela de *“game over”* só pode ser escolhido um áudio, apenas uma música foi selecionada dentre as dez que foram geradas. A base de dados escolhida para essa cena foi desenvolvida pelos grupos categorizados: músicas tristes e de melancólicas, com acréscimo de 37 áudios da segunda base de dados com músicas relacionadas a *“game over”* em seu nome. A sintetização dessa trilha no GANSynth foi feita, assim como as outras, com instrumentos com timbres que condizem com a mensagem que se deseja passar no momento que será tocado, o que diz respeito a instrumentos que “soam tristes” de certa maneira.

Para a tela de título nenhum treinamento ou geração específico foi feito. Selecionou-se uma trilha sonora que não foi escolhida para outras cenas mas que ainda assim apresentava bom resultado sonoro. Essa foi instrumentalizada de forma parecida com as trilhas sonoras para a cena da masmorra.

B. Resultados e Discussão

No total, foram realizadas 36 entrevistas. Os resultados indicam que os participantes da pesquisa não usufruíam de muita experiência musical, com 69,4% dos entrevistados indicando ”Gosto de ouvir música no meu tempo livre, mas não toco instrumento”, e apenas 22,2% escolhendo ”Tenho alguma experiência tocando um instrumento, mas toco apenas ocasionalmente ”, 5,6% “Eu estudo/estudei teoria e composição musical” e 2,8% “Eu me considero um músico e toco o tempo todo”.

A qualidade média dos áudios, de acordo com as classificações dos usuários foi de 7,36, mediana de 8, com 6 e 9 no primeiro e terceiro quartil respectivamente, indicando uma aceitação bastante positiva. Não parece haver uma diferença estatística significativa entre a classificação média dos clipes criados por humanos 7,67, mediana 8 com 7 e 9 no primeiro e terceiro quartil respectivamente e os gerados pelo Performance RNN e GANSynth com média 7,16, mediana 8 com 6 e 9 no primeiro e terceiro quartil respectivamente. Curiosamente, o áudio com a segunda classificação mais alta, com uma pontuação de 7,72, mediana 8, com 7 e 9 no primeiro e terceiro quartil respectivamente foi gerado por computador e apontado como humano por 63,9% dos entrevistados, a maior porcentagem entre os clipes.

Os resultados indicam que os entrevistados tiveram dificuldade em determinar precisamente se um clipe foi composto por um ser humano ou por um computador. De acordo com a pesquisa, eles conseguiram distinguir apenas 51,6% das vezes. O gráfico abaixo contém a porcentagem de erros e acertos dos participantes (Fig. 3).

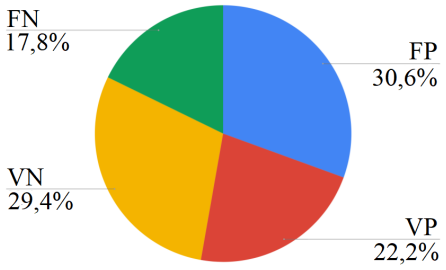

Fig. 3. Gráfico mostrando o resultado do ”teste de Turing musical”. Verdadeiro positivo (VP) indica que o participante classificou a música como autor humano e, de fato, era produzida por humano. Em Verdadeiro negativo (VN), o participante classificou a máquina como autora e, notadamente era a máquina. Falso negativo (FN) revela que o participante classificou a máquina como autora, mas na verdade era produzida por um humano. Finalmente, no Falso positivo (FP), o participante classificou a música como autor humano, mas na verdade era a maquina. Fonte: autor.

Comparado ao trabalho de Mauthes [16], o resultado percebido neste projeto teve um resultado interessante. O autor obteve, a partir de um total de seis entrevistados, uma qualificação média de 5,56 nos áudios, sem apresentar distinção significativa entre a classificação média com relação

aos criados por humanos (5,42) e gerados pelo modelo usado, o VGM-RNN (5,67). O áudio mais bem avaliado gerado pelo modelo proposto pelo autor teve uma média de 6,33. Ainda de acordo com sua pesquisa, os participantes foram capazes de distinguir proveniência das músicas (se foi criado por um humano ou máquina) em apenas 56,6% das vezes.

É possível observar uma melhoria no resultado apresentado nesse projeto se comparado ao vgm-rnn ([16]). Notou-se um valor superior em: qualidade média dos áudios no geral, qualidade média de áudios gerado por computador, além de uma menor porcentagem de percepção entre os reais autores das músicas, se por humanos ou computador. No entanto, Mauthes teve o áudio mais bem avaliado, uma trilha sonora criada por seu modelo, a qual obteve 100% dos entrevistados identificando-a como criada por um humano. Neste projeto, um dos áudios criado por computador ficou em segundo lugar, com 58,35% atribuindo-o a um humano, mas é preciso levar em consideração a maior quantidade de participantes nesta pesquisa.

Quanto a avaliação das trilhas sonoras geradas como um recurso de imersão em jogos, cerca de 61,1% estiveram no grupo controle (jogaram o jogo com as trilhas sonoras originais dos autores) e 38,9% no grupo teste (jogaram o jogo com as trilhas sonoras geradas por RNAs). Com o grupo controle foi possível observar uma satisfação média de 7,59, mediana de 7 com 6 e 9 no primeiro e terceiro quartil respectivamente em relação a imersão segundo a Escala de Likert. Os participantes acertaram 47,8% das vezes e 52,2% presumiram que, em sua jogatina, as músicas tocadas foram geradas por uma IA. Quanto ao grupo teste, a nota para a satisfação média de imersão com auxílio da música foi de 7,78, mediana de 8 com 7,25 e 9 no primeiro e terceiro quartil respectivamente. Sobre a diferenciação das trilhas sonoras, 57,1% dos entrevistados adivinharam corretamente, enquanto 42,9% acreditaram ter jogado um jogo com trilhas sonoras composta por humano. O gráfico, Fig. 4, contém a porcentagem de erros e acertos dos participantes.

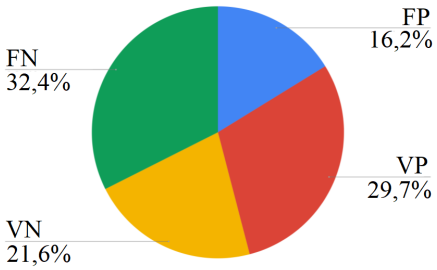

Fig. 4. Gráfico mostrando o resultado do teste dos jogos. As definições de Verdadeiro Positivo (VP), Verdadeiro Negativo (VN), Falso Positivo (FP) e Falso Negativo (FN) são semelhantes à descrita na Fig. 3. Fonte: autor.

VI. Conclusão

Conclui-se que a arquitetura LSTM do Performance RNN consegue modelar recursos musicais, incluindo a melodia, a harmonia, o ritmo e apresenta tempo e dinâmica expressivos. Quanto ao GANSynth, este trouxe uma sintetização a qual auxiliou no aperfeiçoamento do resultado final das trilhas sonoras, introduzindo distintos instrumentos musicais. Além disso, mostra-se que a música gerada pelos modelos nem sempre foram diferenciadas em relação às músicas produzidas por seres humanos, segundo os entrevistados. Embora ainda permita-se muitas melhorias, esses resultados trazem implicações importantes com relação à produção sonora de jogos eletrônicos. Ainda, pode-se dizer que o poder das RNNs, mais precisamente a de Memória de Longo Prazo, são arquiteturas promissoras para a realização de tarefas de aprendizagem de sequência, incluindo a geração automática de música.

O Performance RNN é modelo mais recente do Magenta disponibilizado em código aberto, mas já existe um novo modelo, o Music Transformer [12], disponibilizado em versão demo que apresenta resultados potencialmente superiores. Essa é uma Rede Neural baseada na coerência a longo prazo. Como a bateria também é importante para a música de videogame, um modelo de linguagem musical dedicado exclusivamente para tambores, como em Hutchings et al. [13] pode ser uma ótima opção para trabalhar em paralelo a melodia. Um possível trabalho futuro é empregar tais tecnologias à metodologia proposta.

Agradecimentos

Os autores gostariam de agradecer ao IF Sudeste MG, pelo financiamento da pesquisa, espaço e oportunidade e ao Programa de Educação Tutorial (PET) pela orientação e auxílio.

References

[1] José Carlos Areias. A música, a saúde e o bem estar. *Nascer e crescer*, 25(1):7–10, 2016.

[2] Eduardo Bezerra. Introdução à aprendizagem profunda. *Artigo–31º Simpósio Brasileiro de Banco de Dados–SBBD2016–Salvador*, 2016.

[3] Eric Stefan Boury and Pollyana Notargiacomo Mustaro. Um estudo sobre áudio como elemento imersivo em jogos eletrônicos. *XII Simpósio Brasileiro de Jogos e Entretenimento Digital (SBGames 2013), Anais... ISSN*, pages 2179–2259, 2017.

[4] Jiyanbo Cao, Jinan Fiaidhi, and Maolin Qi. A review of automatic music generation based on performance rnn. 2020.

[5] David Cope. *Experiments in musical intelligence*, volume 1. AR editions, 1996.

[6] Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis. *arXiv preprint arXiv:1802.04208*, 2018.

[7] Douglas Eck and Juergen Schmidhuber. A first look at music composition using lstm recurrent neural networks. *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale*, 103:48, 2002.

[8] Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts. Gansynth: Adversarial neural audio synthesis. 2019.

[9] Tracy Fullerton. *Game design workshop: a playcentric approach to creating innovative games*. CRC press, 2014.

[10] Grammy. First-time grammy nominee: Austin wintory, 2020. Disponível em: https://www.grammy.com/grammys/news/first-time-grammy-nominee-austin-wintory. Acesso em: 08/08/2020.

[11] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

[12] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, and Douglas Eck. Music transformer: Generating music with long-term structure. *arXiv preprint arXiv:1809.04281*, 2018.

[13] Patrick Hutchings. Talking drums: Generating drum grooves with neural networks. *arXiv preprint arXiv:1706.09558*, 2017.

[14] Lejaren A Hiller Jr and Leonard M Isaacson. Musical composition with a high speed digital computer. In *Audio Engineering Society Convention 9*. Audio Engineering Society, 1957.

[15] Dan Liu, Lie Lu, and Hong-Jiang Zhang. Automatic mood detection from acoustic music data. 2003.

[16] Nicolas Mauthes. Vgm-rnn: Recurrent neural networks for video game music generation. 2018.

[17] British Academy of Film and Television Arts BAFTA. Games in 2013, 2020. Disponível em: http://awards.bafta.org/award/2013/games. Acesso em: 08/08/2020.

[18] Raul Paiva de Oliveira et al. Música computacional e jogos digitais: influência da composição por algoritmo na sensação de imersão do jogador. 2017.

[19] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[20] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[21] Luiz Fernando Valente Roveran et al. Música e adaptabilidade no videogame= procedimentos composicionais de música dinâmica para a trilha musical de jogos digitais= music and adaptability in video games: dynamic music compositional procedures for video game scores. 2017.

[22] Marco Scirea, Peter Eklund, Julian Togelius, and Sebastian Risi. Can you feel it? evaluation of affective expression in music generated by metacompose. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 211–218, 2017.

[23] Ian Simon and Sageev Oore. Performance rnn: Generating music with expressive timing and dynamics. https://magenta.tensorflow.org/performance-rnn, 2017.

[24] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. Midinet: A convolutional generative adversarial network for symbolic-domain music generation. *arXiv preprint arXiv:1703.10847*, 2017.