Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Redes Neurais para Improviso Musical Interativo

Thales G. Grilo

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientador
Prof. Dr. Luís Paulo Faina Garcia

Brasília
2021

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Redes Neurais para Improviso Musical Interativo

Thales G. Grilo

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Prof. Dr. Luís Paulo Faina Garcia (Orientador)
CIC/UnB

Dr. Marcelo Antonio Marotta
Universidade de Brasília

Dr. Vinicius Ruela Pereira Borges
Universidade de Brasília

Prof. Dr. Marcelo Grandi Mandelli e Prof. Dr. Wilson Henrique Veneziano
Coordenador do Bacharelado em Ciência da Computação

Brasília, 15 de outubro de 2021

Ficha catalográfica elaborada automaticamente,
com os dados fornecidos pelo(a) autor(a)

Gr Grilo, Thales
Redes Neurais para Improviso Musical Interativo / Thales Grilo; orientador Luis Paulo Faina. -- Brasília, 2021.
52 p.

Monografia (Graduação - Ciência da Computação) -- Universidade de Brasília, 2021.

1. Redes Neurais. 2. Inteligência Artificial. 3. Aprendizado de Máquina. 4. Música. I. Faina, Luis Paulo, orient. II. Título.

Dedicatória

Dedico esse trabalho ao Sensei Nelson Takayanagi. Descanse em paz.

Agradecimentos

Sou grato em primeiro lugar pela oportunidade de poder estudar em uma instituição pública de qualidade. Por esse privilégio, agradeço a Darcy Ribeiro e a todos aqueles que valorizam e defendem o seu legado. Nenhuma conquista é irreversível, e assim é preciso estar sempre atento e forte.

Agradeço pela sorte de ter estudado com professores que admiro, e que transformam minha forma de ver o mundo. Alexandre Zaghetto, Yuri Dumaresq, Célius Magalhães, Flávio Moura, Aleteia Araujo, Carla Castanho, Sérgio Nogueira, e finalmente ao professor Luis Paulo Faina Garcia por orientar esse trabalho ao longo de tantos meses, sempre paciente e atencioso. Muito obrigado.

Agradeço ao MediaLab/UnB, onde tive o primeiro contato com o mundo da arte tecnologia, e que me proporcionou amigos e parceiros para projetos que eu nunca teria concebido. Agradeço à professora Suzete Venturelli pela iniciativa e a todos os demais colaboradores. Joênio Costa, Phil Thomson, Jackson Marinho, Arthur Cabral, Lorena Ferreira, Leandro Muñoz, Alexandre Rangel, Eufrásio Prates, vocês são parte disso. Obrigado por estarem comigo nessa jornada.

Agradeço aos que de formas diversas me ajudaram a encontrar o caminho dessa pesquisa: Mathias Gatti, Maria Fernanda do Carmo, Ian Nery, Daniel Pantoja e os professores Thiago Barros e Alexandre Romariz.

Agradeço à minha família pelo suporte, e também aos bons amigos que tenho. As conversas, conselhos, referências e risadas foram ainda mais preciosas nessa realidade tensa que está sendo a pandemia. Agradeço especialmente à minha mãe, que um dia perguntou se era difícil fazer uma máquina tocar Jazz. A resposta é sim.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES), por meio do Acesso ao Portal de Periódicos.

Resumo

O improviso musical, às vezes definido como Composição Instantânea, é parte quase imprescindível à maioria dos processos de criação musical. Apesar de novos modelos de Aprendizado de Máquina permitem a síntese de música com estrutura elaborada, o improviso musical humano-máquina permanece um desafio longevo. Os sistemas desenvolvidos para criação musical automatizada possuem limitações que restringem seu uso a casos específicos, e nenhum permite o uso de modelos diferentes para geração de improviso em cenários dinâmicos. Por não haver uma quantificação padrão de qualidade musical, é difícil avaliar resultados de modelos de improviso. Além disso, ainda não existe um consenso com relação a métricas de avaliação, avaliações objetivas são escassas e há poucas comparações entre modelos distintos. Estudando os sistemas de geração musical encontrados na literatura e suas limitações, esse trabalho propõe o conceito de um Sistema de Geração Dinâmica Musical, classe de sistemas que permitem a geração automatizada de improviso. Esse trabalho define as características primárias de sistemas dessa classe e propõe uma nova arquitetura com menos limitações.

A abordagem proposta baseia-se na adaptação de algoritmos de Modelagem de Sequência para geração dinâmica, permitindo incorporar à geração de improviso os avanços recentes em composição musical baseada em Aprendizado de Máquina. Para validar essa adaptação, o trabalho emprega o estado da arte em avaliação de geração musical, estendendo a metodologia avaliativa proposta para incluir parâmetros novos, específicos à geração dinâmica. Tomando como base Modelos de Sequência voltados à geração estática de música, compara-se os resultados obtidos entre suas versões originais e adaptadas através de duas avaliações complementares: uma quantitativa, com base em dois indicadores de distância entre métricas objetivas (*Overlap* e *KL-Divergence*) e uma qualitativa, composta de um Teste de Turing e de um estudo de preferência. Os autores validam assim a adaptação proposta, mostrando que a qualidade musical de modelos adaptados se preserva com relação a um conjunto de métricas objetivas e subjetivas. Isso constitui uma prova de conceito importante que pode servir de base a trabalhos futuros.

Palavras-chave: Redes Neurais, Inteligência Artificial, Aprendizado de Máquina, Música

Abstract

Musical Improvisation, or *Instantaneous Composition* as it is often defined, is historically an intrisic component to most of music creation. Even though new Machine Learning models allow for the synthesis of music with elaborate structure, human-machine musical improvisation remains a long-withstanding challenge. Systems designed for automated music creation show limitations that restrict their use to specific use-cases, and none was found which allowed for the use of different models for the same task. Moreover, since there is no standard quantification of musical quality, it's difficult to evaluate works in this area, and evaluations found in literature are precarious when compared to other areas of Artificial Intelligence. Objective model evaluations are scarce and the comparison of different models tends to be very limited. Investigating the music generation systems found in literature and their limitations, this work proposes the concept of a Dynamic Music Generation System (DMGS), a class of systems that allow for the automated generation of musical improvisation. It also defines the primary qualities of systems of that kind and proposes a new implementation that aims to overcome the limitations of its alternatives.

The approach is based on the adaptation of Sequence Models for dynamic generation, allowing recent advancements in Machine Learning-based musical composition to be incorporated into improvisation generation. To validate this adaptation, the authors use the state of the art in objective evaluation of musical generation, extending the proposed method to include new parameters which are specific to dynamic generation. This method is then used to validate the proposed adaptation, showing that the musical quality of the adapted models is preserved in regards to a set of objective metrics. This constitutes an important proof of concept that can be extended further on future works in the area of automatic music generation.

Keywords: Neural Networks, Artificial Intelligence, Machine Learning, Music

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AM Aprendizado de Máquina.

ANS Aprendizado Não-Supervisionado.

AR Aprendizado por Reforço.

AS Aprendizado Supervisionado.

BPM Batidas Por Minuto.

CA Criatividade Artificial.

CF Esquecimento Catastrófico.

FDP Função Densidade de Probabilidade.

GAN Rede Generativa Adversarial.

GD Geração Dinâmica.

GE Geração Estática.

IA Inteligência Artificial.

LSTM *Long Short-Term Memory*.

MIR Processamento de Informação Musical.

MS Modelagem de Sequência.

NADE *Neural Autoregressive Distribution Estimation*.

NLP Processamento de Linguagem Natural.

NNs Redes Neurais.

NoteSeq *Note Sequence.*

OSC *Open Sound Control.*

RNN Redes Neurais Recorrentes.

Seq2Seq *Sequence-to-Sequence Modeling.*

SGDM Sistema de Geração Dinâmica Musical.

SME Subject Matter Expert.

VC Visão Computacional.

Capítulo 1

Introdução

São raras as técnicas ou formas de criação musical não originadas ou essencialmente influenciadas pela prática do improviso (Ferand, 1962) [1]. Para o teórico Gorow, a prática é talvez a epítome da criação artística [2]. Apesar de haver divergência quanto a uma definição precisa, há consenso que o improviso musical possui fundamentalmente um caráter extemporâneo, sendo frequentemente definido como *composição instantânea* [3, 2].

Referência em jazz contemporâneo e improvisação livre, Bailey, D. (1993) [3] afirma não ser possível a existência de um método que descreva por completo a criação musical. O autor mostra como diferentes culturas e autores criam ou empregam seus próprios conjuntos de regras para improvisação, além da importância que teve o improviso conjunto no processo criativo de coletivos diversos, como *New Phonic Arts Group* [1] e *Spontaneous Music Ensemble* [2]. O que se mostra inevitável para essa prática é a associação temporal entre criação e experiência do som. De fato, Bailey afirma que o improviso enquanto prática possui como propósito final celebrar o próprio momento em que se manifesta.

O processo de criação musical é associado à criatividade, que pode se definir no estudo de linguística como a capacidade humana de gerar sequências inéditas para uma dada linguagem [4, 5]. Em paralelo, o interesse no uso de computadores para esse tipo de tarefa tem início na década de 1950 [6, 7] e técnicas recentes têm gerado amostras sintéticas cada vez mais semelhantes a peças criadas por autores reais [8, 9], tensionando os limiares do que entendemos como criatividade.

A análise e síntese automática de arte, em suas diferentes linguagens, define um campo de estudo na área de Inteligência Artificial (IA), chamado de Criatividade Artificial (CA) [10]. Pelo recorte escolhido, esse estudo se encontra na intersecção entre CA e a área

[1] https://rateyourmusic.com/artist/new_phonic_art

[2] https://rateyourmusic.com/artist/spontaneous-music-ensemble

focada na análise e processamento computacional de informação musical, chamada Processamento de Informação Musical (MIR) [11].

É comum empregar-se IA como ferramenta auxiliar em processos criativos de diferentes formas, como fonte por exemplo de fragmentos musicais [12] ou de continuações de um fragmento original [8]. Essa delegação de criatividade pode ser vantajosa na produtividade do músico pelo simples volume e variedade de ideias resultantes.

Hoje o estado-da-arte emprega técnicas de Aprendizado de Máquina (AM), campo em IA que estuda aprendizado indutivo, onde algoritmos (também chamados de modelos) são sinteticamente construídos para uma determinada tarefa a partir de dados [13]. A construção de modelos com AM acontece, de forma resumida, nas seguintes etapas: (*i*) Determina-se as métricas relevantes e um valor de sucesso; (*ii*) Estabelece-se na forma de um sistema um fluxo de aprendizado, que inclui estimar os valores das métricas de interesse; (*iii*) Instrumenta-se o sistema para determinar problemas de desempenho e possíveis causas, e; (*iv*) Faz-se mudanças incrementais sobre a configuração do sistema conforme seu efeito sobre as métricas-alvo [13].

Tradicionalmente, algoritmos de AM pressupõem um tamanho fixo para suas entradas e saídas, o que limita sua aplicabilidade para domínios como geração musical, cujas amostras variam naturalmente em comprimento [14]. Em contraste, uma abordagem chamada Modelagem de Sequência (MS) consiste na modelagem de dados como sequências de valores cujo comprimento pode ser variável. Por isso, é também chamada *Sequence-to-Sequence Modeling* (Seq2Seq). Por oferecer essa flexibilidade com relação aos tamanhos de suas amostras, MS torna-se particularmente popular para tarefas de geração musical [15, 16].

O maior corpo de estudo na aplicação de AM para aplicações musicais é parte de uma iniciativa colaborativa da *Google AI* chamada *Magenta Research* [12]. Sua contribuição para o campo inclui MSs como *PerformanceRNN* (2017) [9] e, mais recentemente, *Music Transformer* (2019) [8]. Ambos são exemplos de arquiteturas de *AM* chamadas NNs, treinadas para a geração de música clássica para piano incluindo nuances de tempo e intensidade inerentes ao exercício da música por intérpretes humanos. A principal característica que os difere é o mecanismo que empregam para Modelagem de Sequência: *PerformanceRNN* implementa uma Redes Neurais Recorrentes (RNN), enquanto *Music Transformer* utiliza um mecanismo mais recente, chamado *Relative Self-Attention*. Os dois modelos são discutidos em detalhe no capítulo seguinte, e através deles é possível gerar amostras de 16 e 50 segundos de duração, respectivamente, sem que se possa identificá-las como artificiais de forma consistente.

Em particular, os autores do *Magenta Research* disponibilizam seus modelos na forma de uma biblioteca [12], porém, como descrito no próximo capítulo, diferentes modelos

e ferramentas tendem a ser desenvolvidas isoladamente, sobre diferentes infra-estruturas (*TensorFlow* [8] e *PyTorch* [17], por exemplo) e representações musicais. Por conta disso, interfaces e integrações com outros sistemas acabam sendo reimplementadas em função de cada aplicação proposta [18, 12, 17].

Esse não é o único obstáculo para o emprego de IA em processos criativos. É difícil, por exemplo, propor uma solução computacional para algo que transcende método, segundo Bailey, D. (1993) [3]. A ausência de controle ou determinismo sobre a música gerada, seja na forma de limitações impostas, de parâmetros sonoros ou intervenção direta, é algo que inibe maior adoção dessa tecnologia nessa área [19, 20].

Há também dificuldades na avaliação rigorosa de trabalhos nessa área. A natureza musical do resultado dificulta categorização e análise objetiva: o caráter altamente subjetivo e cultural da música dificulta a quantificação de emoção e significado, ou a definição de um conjunto universal de regras. Assim, é difícil descrever música semanticamente [21], e por consequência, determinar e argumentar sobre a qualidade objetiva de resultados musicais.

Além disso, o uso de diferentes bases de dados e a forma como se apresentam os resultados de estudos na área são fatores que normalmente impedem a comparação cruzada entre modelos [10, 22]. O uso da mesma base de treinamento na comparação entre modelos distintos elimina os obstáculos avaliativos relativos a emoção, significado e sintaxe - essa concessão é comum por promover maior conclusividade nos resultados de cada estudo [23, 8, 17, 24]. Como consequência, os resultados encontrados tendem a ser dependentes da base de dados escolhida, e não são imediatamente relacionáveis.

Por fim, a prática do improviso exige que a criação e a reprodução da música sejam simultâneas. Isso será referido ao longo do trabalho como Geração Dinâmica (GD), e exige que um sistema gere informação musical continuamente conforme ela é executada. Os modelos mencionados empregam Geração Estática (GE), gerando música em um momento e permitindo a experiência do som apenas mais tarde. Isso portanto os inviabiliza como ferramentas de improviso contínuo. Existem modelos na literatura desenvolvidos especificamente para esse fim [18, 17], mas são menos numerosos e implementam individualmente suas próprias soluções de reprodução, sincronização e interface.

Falta na literatura pesquisada um sistema que permita o uso de diferentes modelos para GD de música, ou uma forma de adaptação de GE para GD nesse domínio. Essa ausência restringe as opções disponíveis para improviso automatizado, e impede a exploração computacional de um componente importante do processo criativo musical. Esse trabalho investiga um paradigma alternativo ao empregado tradicionalmente por sistemas de GD: propõe-se um *framework* para a adaptação de MS pré-treinados para GD sem restrições quanto à infra-estrutura empregada. A solução, implementada na forma

de um sistema de software, será avaliada de maneira sistemática sob critérios objetivos e subjetivos, através de uma metodologia que estende o estado-da-arte para contemplar parâmetros de GD estabelecidos pelo *framework* adaptativo.

1.1 Objetivos

O objetivo desse estudo é propor e validar um *framework* que possibilita o uso de *MS*s estáticos para a GD de música. Para isso, uma técnica de adaptação de MSs de GE para GD é proposta com base em um ciclo de geração iterativa. A técnica é então implementada na forma de um sistema e utilizada para adaptar quatro MSs encontrados na literatura: *PerformanceRNN*, *PolyphonyRNN*, *MelodyRNN* e *PianorollRNN-NADE*. Propõe-se uma classe de sistemas de GD de música, chamada Sistema de Geração Dinâmica Musical (SGDM), e mostra-se como o sistema construído cumpre lacunas funcionais que impedem o uso de sistemas semelhantes como ferramentas de geração de improviso.

Por fim, para determinar a qualidade da adaptação, é necessário medir seu impacto sobre a qualidade da saída dos modelos. Os autores portanto propõem uma metodologia que compara a qualidade da saída dos modelos pré-adaptação (GE) e pós-adaptação (GD). Essa comparação é composta de duas avaliações complementares: objetiva e subjetiva.

O estado-da-arte em análise objetiva de informação musical propõe um conjunto de 12 métricas descritivas [21] e uma técnica que as emprega para aferir a semelhança entre amostras musicais. Isso é feito através do cálculo de dois indicadores objetivos de distância entre métricas: *Overlap* e *KL-Divergence*. A avaliação objetiva desse trabalho avalia a variação entre os valores de distância pré e pós-adaptação. São introduzidos também dois novos parâmetros: os comprimentos de entrada e saída empregados no ciclo de geração iterativa. Esses parâmetros são utilizados para otimizar os valores de distância de cada MS, e para controlar seu tempo de geração, a fim de garantir sua viabilidade para GD.

Para avaliação musical subjetiva, as técnicas mais comuns são testes de audição [19], incluindo um Teste de Turing voltado à detecção de amostras artificiais e uma comparação de preferência para quantificação da qualidade subjetiva de conjuntos de amostras. Para essa etapa da avaliação, foi construído um questionário com base nas amostras estudadas, tanto reais quanto geradas artificialmente, para determinar: (i) O quão mais identificáveis são as amostras dinamicamente geradas, e (ii) Se soam significativamente piores que as amostras de GE.

1.2 Hipótese

É possível adaptar MSs GE para GD sem comprometer a qualidade sonora de sua saída. Essa adaptação soluciona a lacuna apontada na literatura, promovendo aos modelos de geração de composição o aspecto temporal que permite usá-los como ferramentas de improviso. Isso é possível sem impor restrições aos modelos quanto às

Esse estudo investiga a hipótese que MSs podem ser adaptados de GE para GD sem comprometer a qualidade sonora de sua saída. Acredita-se que isso é possível sem a necessidade de impor restrições sobre os MS de interesse quanto à própria arquitetura ou infra-estrutura.

Para investigar essa hipótese, o experimento construído parte de um conjunto de MSs que variam com relação à infra-estrutura e base de treinamento. Para cada modelo, compara-se a semelhança que as amostras geradas sob cada modalidade de geração (GE e GD) apresentam com relação a amostras reais, objetivamente e subjetivamente. Caso não se observe diferença estatisticamente significativa entre as duas modalidades, conclui-se que a qualidade musical dos modelos originais (GE) e adaptados (GD) é equivalente, e não é portanto afetada pela adaptação. Isso valida o *framework* adaptativo proposto, e portanto a hipótese principal do estudo.

1.3 Organização do documento

Esse trabalho divide-se em seis capítulos. Neste primeiro, apresentamos os problemas a serem abordados, objetivos e hipótese central do estudo. No próximo capítulo exploramos a literatura, estabelecendo conceitos importantes e definindo o escopo do estudo. O Capítulo 3 apresenta a proposta de adaptação e o sistema que a implementa. O Capítulo 4 explica a construção do experimento e as etapas avaliativas: quantitativa e qualitativa. No Capítulo 5, discute-se os resultados obtidos com os experimentos do capítulo anterior. Por fim, o Capítulo 6 discute as contribuições do artigo para o campo e apresenta limitações atuais e direções futuras para o trabalho.

Capítulo 2

Revisão da Literatura

Processamento de Informação Musical (MIR) é a área da Ciência da Computação que estuda a análise e processamento de informação musical [11]. Para que seja possível modelar computacionalmente tarefas relacionadas à criação musical, é necessário entender quais aspectos desse domínio podem ser explorados, bem como as abordagens computacionais que foram estudadas no passado. Esse capítulo estabelece de ambos os domínios os conceitos relevantes ao estudo, incluindo fundamentos teóricos da música, de AM e uma revisão das técnicas de geração e de avaliação encontradas na literatura.

A seção 2.1 elucida os conceitos básicos de teoria musical necessários para o restante do estudo. Na segunda seção, explora-se brevemente o histórico da composição musical baseada em algoritmos e discute os protocolos mais comuns no contexto de criação musical com sistemas computacionais. A seção 2.3 traz um resumo da evolução do Aprendizado de Máquina (AM) até as arquiteturas mais usadas hoje para a tarefa em questão. A quarta seção explora o conceito de Modelagem de Sequência (MS) e os benefícios que motivam seu uso para a geração musical. Por fim, a seção 2.5 discute a estratégia mais robusta de avaliação de Modelagem de Sequência (MS) para música atualmente.

2.1 Conceitos Musicais

Visto que música não possui uma única definição, ou uma definição absoluta, existem múltiplas formas de estudar e representá-la (Dannenberg, 1998) [25]. Para efeito desse estudo, será empregada a teoria musical padrão historicamente aceita no mundo ocidental. Khulusi, et al (2020) [26] resume algumas noções musicais importantes a qualquer abordagem de visualização ou representação musical, e inclui:

- Altura (Pitch): uma quantificação da frequência associada a um som, tradicionalmente notado no eixo vertical. O intervalo entre cada par de frequências relacionados

pelo fator 2 chama-se *Oitava*. O sistema mais comum na música ocidental divide cada oitava em 12 posições logaritmicamente equidistantes - esse sistema chama-se *Temperamento Igual* [2]. Nos referiremos a cada um dos 12 valores possíveis de altura dentro de uma oitava como uma Classe de Altura, ou Classe de *Pitch*.

- Intensidade: Energia associada a um som, proporcional à amplitude de sua onda sonora. Sons com alta intensidade propagam-se mais amplamente através do espaço e são mais facilmente audíveis com relação a sons de baixa intensidade, que propagam-se menos e tendem a ser ofuscados por sons mais intensos. Na representação padrão, graus de intensidade são quantificados em ordem crescente na seguinte escala: $\{ppp, pp, p, mp, m, mf, f, ff, fff\}$.
- Dinâmica: Nível de variação de intensidade entre notas de um mesmo instrumento ou conjunto ao longo de um segmento musical, ocasionalmente referido na literatura também como *Expressividade* [27]. Estruturas complexas de Dinâmica são historicamente associadas à criação musical humana e difíceis de replicar automaticamente de forma verossímil.
- Nota: Simbolização de um som musical formado pela união da informação relativa à sua Altura (Pitch), Duração e Intensidade. Notas formam a base da composição para a teoria ocidental, sendo possível então representar composições musicais como distribuições de notas ao longo do tempo.

Além disso, existem também unidades temporais importantes, úteis e necessárias para organizar a informação musical. Hewitt (2008) [28] define as seguintes:

- Batida: Unidade básica usada para medir o comprimento de notas.
- Compasso (ou Ciclo Métrico): Unidades de tempo compostas de batidas agrupadas em um comprimento consistente. Compassos são uma unidade conveniente para organizar tanto informação musical como também métricas e dados de análise [21].
- Fórmula de Compasso (Time Signature): Fração que determina o número e subdivisão de batidas dentro de um compasso. Compõe-se por dois números: o numerador indica a quantidade de elementos por ciclo métrico, enquanto o denominador indica o comprimento (em batidas) de cada elemento.
- Batidas Por Minuto (BPM): Unidade que quantifica a velocidade de execução de uma obra ou segmento musical, indicando literalmente o número esperado de Batidas para cada intervalo de mesma duração de tempo.

Esses conceitos são exemplificados na figura 2.1, que ilustra uma visualização da informação musical referente a um fragmento musical. Essa visualização baseia-se no protocolo

MIDI, descrito na seção 2.2. Na figura, observa-se uma grade onde notas são dispostas. O eixo vertical do plano indica a altura de uma nota, quantizada novamente em 12 valores possíveis por Oitava. O eixo horizontal representa o tempo, quantizado em Batidas e Compassos. Notas são denotadas como células dentro da grade: sua altura equivale à sua posição vertical, e sua duração, à largura da própria célula.

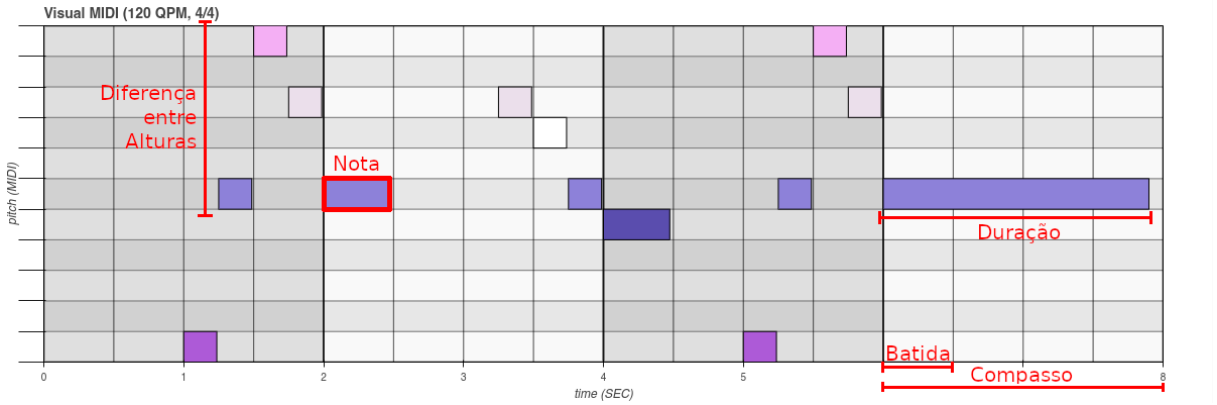


Figura 2.1: Fragmento Transcrito de "Bum Bum Tan Tan" (MC Fioti, 2017)

Outro conceito importante diz respeito à simultaneidade do som - o conceito de *Textura* musical, definido pelo teórico McKay [29] como uma categoria ou forma elementar de unidade musical, podendo caracterizar-se como, entre outros: *1. Monofonia*, havendo apenas uma nota a cada dado momento, ou; *2. Polifonia*, onde é possível que diferentes sons de mesma natureza aconteçam e interajam simultaneamente. Observa-se haver na figura 2.1 no máximo uma nota a cada dado instante, tratando-se assim de uma composição monofônica - trechos polifônicos, em contraste, apresentariam múltiplas notas simultâneas.

Por fim, é importante fazer a distinção entre duas formas importantes de criação - tarefas para as quais existem modelos de geração treinados. *Composição* refere-se à geração de sequências de notas sem que haja necessariamente alguma restrição ou condição. Já *Geração de Acompanhamento* refere-se a uma técnica específica de criação que parte de um conjunto monofônico de notas e gera como saída um conjunto polifônico dentro do mesmo intervalo de tempo: *BachDoodle* [30] é um exemplo de sistema de GE de Acompanhamento, gerando notas sobre uma composição curta do usuário, e BachDuet [31] executa Geração Dinâmica de Acompanhamento, compondo uma peça polifônica sobre uma entrada monofônica conforme o usuário a executa.

2.2 Representação Computacional de Música

A década de 1950 foi marcada por um grande interesse, por parte de autores da época, em formas experimentais de composição, como Composição Aleatória - técnica definida pela delegação parcial ou completa do resultado musical a operações de chance [32]. Isso se observa por exemplo na obra *Music of Change* [32], do renomado autor John Cage, construída com base em resultados do I-Ching, oráculo textual ancestral chinês utilizado para divinação através de números aleatórios.

Nessa mesma década, surge o estudo do uso de computadores para fins musicais, principalmente envolvendo sistemas de regras e técnicas de modelagem probabilística, como Cadeias de Markov [33]. Compositores como Iannis Xenakis passaram a empregar computadores na automação parcial do processo criativo a partir de 1960. Desde então, diversas abordagens distintas foram exploradas, como autômatos celulares, sistemas evolutivos e AM [10].

A partir de 1970, crescem os estudos em CA na exploração de outras vertentes de criação como poemas, haikus, textos e imagens [34], consolidando o campo em um espectro mais amplo. Para o estudo de CA, uma máquina é considerada mais criativa quão menos conhecimento prévio possuir por intervenção humana. Assim, técnicas de AM tornam-se muito atrativas para a exploração desse campo, por minimizarem o viés humano sobre a criatividade de um sistema [34, 13].

Em 1985, é apresentado um protocolo padrão para codificação e comunicação de informação musical a nível de software e hardware, chamado MIDI [35], que desde então foi base para inúmeros trabalhos na área de MIR [19, 8, 17, 36, 9] e tornou-se uma importante integração entre sistemas musicais, "dominando a música popular" [37]. Entre outras informações e eventos musicais, o protocolo representa Notas Musicais como eventos compostos por três elementos básicos: Altura (*Pitch*) e Intensidade (*Velocity*), cada um quantificado na forma de um número inteiro no intervalo [0:127], e Comprimento, quantificado sob uma unidade escalar própria, chamada *tick*, e calculada com base na distância entre os dois eventos que compõem uma nota: *Note On* e *Note Off*, representando seu início e fim, respectivamente.

Além disso, o protocolo também armazena informações temporais, como o quão rápido um arquivo deve ser reproduzido, propriedade chamada *Tempo* [35]. Dessa forma, MIDI é capaz de representar música - um amplo subconjunto dela -, como uma sequência de eventos, o que possibilita o uso de técnicas de MS para sua análise. A Figura 2.2 ilustra um fragmento musical representado através do protocolo MIDI. Cada nota é representada como um retângulo em tonalidades distintas de roxo, usadas para reforçar a distinção entre alturas. A altura de uma nota é representada no eixo vertical. Seu início e fim são representados no eixo horizontal de tempo.

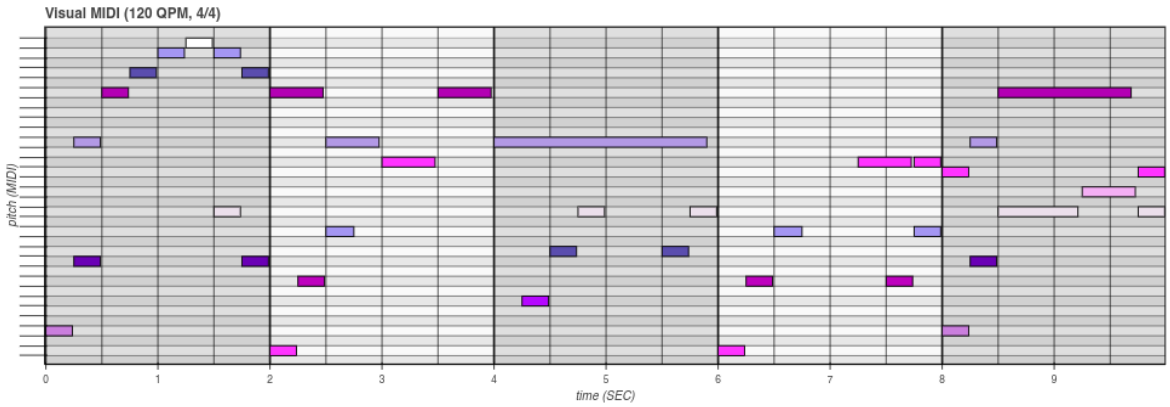


Figura 2.2: Representação MIDI de fragmento de Dire Dire Docks (Koji Kondo, 1996)

Além de um protocolo de informação, MIDI é também um formato de arquivo, possuindo três tipos básicos: Arquivos MIDI Tipo-1 contém apenas uma sequência, que inclui o cabeçalho da composição e todos os demais eventos. Arquivos Tipo-2 possuem múltiplas sequências síncronas, reservando a primeira faixa para eventos de tempo, compartilhados por todas as faixas. Arquivos Tipo-3 são menos comuns, e contém sequências paralelas temporalmente independentes.

Em 1997, surge outro protocolo, chamado *Open Sound Control* (OSC). Com o objetivo de ser mais flexível, extensível, reativo e semântico que o MIDI [38]. OSC é implementado sobre o protocolo de rede UDP, possibilitando integrações não previstas pelo protocolo MIDI e expandindo o potencial criativo de aplicações musicais e multi-mídia [39]. A Figura 2.3 exemplifica a comunicação OSC para um sistema hipotético, onde um instrumento (à esquerda) possui parâmetros tímbricos Volume, Frequência e Formato, que podem então receber novos valores via mensagens como à direita da figura.

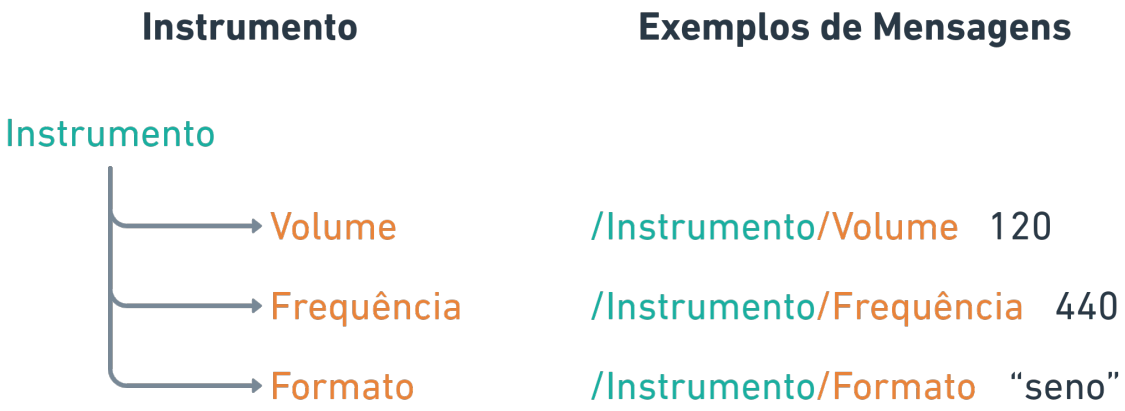


Figura 2.3: Exemplo de Codificação OSC num Sistema Hipotético

Desde então, tem sido comum o uso de sistemas capazes de traduzir entre ambos

[40] em ferramentas que exploram a criação computacional de padrões musicais [41], permitindo que se tire proveito de um ecossistema criativo já maduro e ao mesmo tempo abrindo espaço para a construção de controles adequados para cada aplicação. OSC passa então a ser adotado em sistemas que exigem alta interatividade e extensibilidade [42, 40].

2.3 Aprendizado de Máquina e Redes Neurais

Aprendizado de Máquina é uma área de pesquisa que estuda os algoritmos baseados em aprendizado dedutivo [13]. Esses algoritmos são capazes de aprender - ou seja, construir conhecimento sobre uma determinada tarefa - com base em amostras do mundo real. Para Murphy (2012) [43], AM é um conjunto de técnicas computacionais para construção de sistemas autônomos baseados na detecção de padrões em dados, com fins voltados a tomada de decisão ou predição de dados futuros.

Redes Neurais são técnicas de modelagem estatística fundamentais para AM, cujo objetivo é aproximar uma determinada função $f*$ [13]. Para isso, NNs buscam inspiração na neurociência, organizando-se internamente como grafos cujos nós são unidades lógicas análogas a neurônios. Em uma Rede Neural, nós são organizados em camadas consecutivamente interconectadas, ou seja valores de ativação de uma camada são alimentados aos nós da camada seguinte. O papel de um nó é calcular um valor de saída para as entradas recebidas com base em uma função de ativação, cuja fórmula tipicamente inclui um coeficiente escalar chamado peso. Os pesos associados aos nós de NNs armazenam seu conhecimento, adquirido através de amostras de dados, e o processo de ajuste dos desses pesos é o que se chama *treinamento* (ou *aprendizado*) de uma Rede Neural.

Há algumas modalidades distintas de aprendizado em AM, sendo as mais comuns: Aprendizado Supervisionado (AS), em que ambas a entrada e a saída esperada são conhecidas; Aprendizado Não-Supervisionado (ANS), em que a saída não é definida; e Aprendizado por Reforço (AR), em que a saída do modelo é avaliada a cada iteração, e seu comportamento é então ajustado [44].

Historicamente, houve maior interesse na adoção de Aprendizado Supervisionado para NNs [45], em parte por conta das dificuldades associadas ao ANS [13]. Em 2016, Metz et al. [45] propõem um classe de Redes Neurais chamadas DCGAN - baseando-se em uma arquitetura já conhecida, chamada Rede Generativa Adversarial (GAN) -, mostrando ser possível o aprendizado não-supervisionado de conceitos semânticos elaborados em imagens. Desde então, GANs e suas variantes foram empregadas em diversas aplicações, incluindo na síntese de música [46].

Música é uma linguagem estruturada, significando que possui sub-estruturas identificáveis e recorrentes ao longo de uma mesma amostra [4]. Um dos maiores desafios,

portanto, na modelagem de música através de AM, é exatamente a geração de música com estrutura. Buscando aprimorar essa capacidade, expande-se entre 1980 e 2000 a exploração de modelagem musical usando arquiteturas de Redes Neurais (NNs) [4].

O treinamento de NNs para MS apresenta um desafio particular, constatado inicialmente ao fim da década de 1980, chamado Esquecimento Catastrófico (CF). O termo se refere à perda de conhecimento que NNs de sequência sofrem ao aprender com novos dados, efetivamente desfazendo completamente seu processo de treinamento [47]. Uma consequência disso para a geração de música é que a saída de MSs tende a se tornar repetitiva ou incoerente a partir de um determinado comprimento - recentemente abordagens para mitigação desse efeito têm sido estudadas, como RL-Tuner [48].

Em paralelo à geração de música, NNs passam a demonstrar bons resultados para tarefas de outras áreas de IA, notavelmente nos campos de Visão Computacional (VC) e Processamento de Linguagem Natural (NLP) [4]. A abordagem mais frequente em NLP é a modelagem Seq2Seq, normalmente conjugada ao uso de arquiteturas que permitem retenção de informação ao longo do processamento de sequências, como Redes Neurais Recorrentes (RNN)s e LSTMs [49].

Redes Neurais Recorrentes são redes cujos grafos possuem ciclos [13], técnica usada principalmente para a detecção de padrões em sequências. Hochreiter, Sepp e Schmidhuber (1990) [50] propõem a arquitetura LSTM que consiste em um arranjo recorrente específico de neurônios junto a um algoritmo de aprendizado específico, buscando melhorar a retenção de aprendizado durante o processamento de sequências longas. Para o mesmo fim, Vaswani et al (2017) [51] propõe uma arquitetura de NNs chamada *Transformer*, mais simples que as disponíveis até então, sem o uso de recorrência e exigindo menor tempo de treinamento e de resposta para tarefas de NLP.

Observando-se as semelhanças entre os domínios musical e linguístico - amostras de comprimento variável, caráter estrutural dos dados e vasta disponibilidade de dados não-rotulados [49] - tem sido comum então o uso de arquiteturas originalmente usadas em Processamento de Linguagem Natural (NLP) para tarefas de modelagem musical [4]. De fato, modelos como *PerformanceRNN* [9] foram desenvolvidos para geração de música para piano, sendo capazes de gerar amostras coerentes de até 16 segundos em média, e esse resultado é então superado em 2018 com o uso de modelos *Transformers*, gerando amostras acima de 50 segundos mantendo coerência e estrutura interna [8].

2.4 Modelos de Sequência Musical

A Tabela 2.1 reúne MSs voltados a geração musical encontrados hoje na literatura, incluindo as arquiteturas que cada modelo emprega, as bases de dados sobre as quais foram

treinados, as tarefas para as quais foram concebidos, a disponibilidade de modelos pré-treinados (abreviado na tabela como PT) e suas modalidades de geração (estática ou dinâmica).

Tabela 2.1: Modelos de Geração

Name	**Arquitetura**	**Base**	**Geração**	**PT**	**Estilo**
RL Duet	RNN-RL	Bach Chorales	Dinâmica	Sim	Contraponto
BachDuet	RNN-LSTM	Bach Chorales	Dinâmica	Não	Contraponto
Continuator	Markov	Autoral	Estática	Não	Jazz
JIG	Markov	Autoral	Estática	Não	Jazz
Music SketchNet	RNN-Attention	Irish Scott. Melodies	Estática	Sim	Irish Folk
REMI	Self-Attention	Autoral	Estática	Sim	Pop
Bassline	RNN-LSTM	Autoral	Estática	Não	Eletrônico
BebopNet	RNN-LSTM	Autoral	Estática	Sim	Jazz
MelodyRNN	RNN-LSTM	Autoral	Estática	Sim	Pop
ImprovRNN	RNN-LSTM	Autoral	Estática	Sim	Pop
PerformanceRNN	RNN-LSTM	Piano-e-Comp.	Estática	Sim	Piano Solo
Pianoroll RNN-NADE	RNN-LSTM NADE	Bach Chorales	Estática	Sim	Contraponto
PolyphonyRNN	RNN	Bach Chorales	Estática	Sim	Contraponto
Music Transformer	Self-Attention	Piano-e-Comp.	Estática	Sim	Piano Solo

A textura musical de cada modelo (monofônica ou polifônica), sua expressividade, sua arquitetura e sua base de treinamento são fatores que tendem a restringir o uso de sistemas baseados sobre eles a contextos específicos. Para permitir a adaptação de MSs arbitrários para geração musical dinâmica, é importante que a técnica proposta não possua as mesmas restrições. Assim, é necessário validar a técnica proposta sobre um conjunto mínimo de modelos dentro do qual esses quatro aspectos variem. Além disso, prioriza-se os modelos treinados sobre bases de dados conhecidas em MIR. *Bach Chorales*, *Piano-e-Competition* e *LAKH* são exemplos de bases polifônicas populares, usadas no treinamento de diversos modelos [8, 17, 9, 52, 53] .

Visando otimizar a aplicabilidade da técnica, o escopo do estudo será limitado aos modelos que: (*i*) Possuam código aberto e versões pré-treinadas prontamente disponíveis para análise; (*ii*) Gerem informação musical estruturada, e não simplesmente aleatória; (*iii*) Baseiem-se em AM, conforme motivado pelo estudo de CA, e; (*iv*) Empreguem codificações compatíveis com o protocolo MIDI [35]. *RL-Duet* [17], *BachDuet* [31], *Continuator*

[54], *JIG* [55] e Bassline [56] não possuem modelos abertos pré-treinados, e portanto não serão considerados para efeito dessa avaliação.

Os modelos criados pelo *Magenta Research* variam com relação aos quatro aspectos citados, e além disso possuem duas características que simplificam sua análise. Primeiro, há versões pré-treinadas para esses modelos [1], e segundo, apesar de possuírem entradas e saídas distintas, os modelos compartilham internamente uma mesma representação de informação musical concebida pelos autores, chamada *Note Sequence* (NoteSeq) [2]. *Note Sequence* é um procolo de representação de sequências de notas musicais a nível de software análoga ao MIDI que possibilita a transferência de informação musical entre sistemas com baixo custo de memória e tempo.

Dessa forma, os modelos *MelodyRNN*, *PerformanceRNN*, *PolyphonyRNN* e *PianorollRNN-NADE* já constituem um espaço amostral suficientemente variado, e servirão portanto como base desse estudo.

2.4.1 MelodyRNN

MelodyRNN [57] é um modelo voltado para geração de melodias. No contexto do trabalho, entende-se como melodia uma sequência monofônica principal, que prevalece sobre as demais sequências paralelas, sejam monofônicas ou polifônicas. O modelo emprega uma arquitetura LSTM, e é treinado sobre uma base de dados própria, construída a partir de melodias e linhas vocais de músicas populares.

Os autores fornecem quatro variantes para esse modelo: *BasicRNN*, *MonoRNN*, *LookbackRNN* e *AttentionRNN*. *BasicRNN* implementa uma LSTM simples e serve como base para geração de melodias. Sua entrada e saída são ambas codificadas como vetores *one-hot*, onde cada posição indica uma altura de nota dentro no intervalo $[48 : 84]$. Para simbolizar informação MIDI, o modelo emprega como possíveis rótulos para a entrada recebida os eventos *note-on*, indicando que uma nota deve ser executada, *note-off*, indicando que deve ser interrompida e *no event* que indica extensão da duração de quaisquer eventos anteriores, incluindo silêncio.

A primeira adaptação, *MonoRNN*, simplesmente amplia o intervalo de alturas possíveis para a entrada e saída par o intervalo MIDI $[0 : 127]$. *LookbackRNN* baseia-se no *BasicRNN*, alterando sua entrada para incluir informação relativa aos dois últimos compassos e à posição atual da nota. Isso é feito através de um vetor de cinco posições incorporado à entrada do modelo, que codifica binariamente a batida atual dentro de um compasso de até 2^5 batidas de comprimento. Em adição a isso, o modelo adiciona dois novos rótulos ao conjunto inicial: *repeat-1-bar-ago* e *repeat-2-bars-ago*, que indicam a

[1] https://github.com/magenta/magenta/tree/main/magenta/models
[2] https://github.com/magenta/note-seq

repetição do rótulo aplicado há um ou dois compassos atrás. Isso permite que a saída do modelo retenha um grau maior de estrutura comparado às versões anteriores. A última adaptação, *AttentionRNN*, emprega *Self-Attention* com o mesmo propósito, relacionando os últimos n passos da própria saída atual à entrada recebida. Isso fornece ao modelo acesso à informação anterior da própria saída sem a necessidade de armazená-la no estado de sua *RNN*.

Os autores não descrevem os parâmetros de treinamento das versões pré-treinadas do modelo. Também não há uma descrição de suas bases de treinamento, informando-se apenas que milhares de arquivos MIDI representando melodias monofônicas foram usados no processo. Não se encontra também qualquer avaliação da qualidade sonora do modelo, apenas constata-se empiricamente sua capacidade de gerar melodias estruturadas. Yang, Chou & Yang (2017) [58] comparam as três variantes através de um teste de escuta, e concluem que *AttentionRNN* e *LookbackRNN* possuem resultados semelhantes com relação à preferência do ouvinte, ambos superando a variante *BasicRNN* [59]. Esse estudo usará a variante *AttentionRNN*, pelo uso de *Self-Attention* que distingue sua arquitetura das demais.

2.4.2 PerformanceRNN

PerformanceRNN [9] é um modelo concebido para gerar música semelhante a um pianista profissional tocando - especificamente, busca gerar composições polifônicas expressivas. O modelo é construído sobre uma *LSTM* e treinado sobre a base de dados *Piano-e-Competition*. Diferente dos outros algoritmos, o *PerformanceRNN* modela pequenas nuances em intensidade e em velocidade, com o objetivo de aproximar melhor um instrumentista real.

A arquitetura do modelo emprega três camadas LSTM escondidas, com 512 nós por camada. Sua entrada e saída são ambas codificadas como vetores *one-hot* de 413 posições. Para modelar nuances relativas a tempo e intensidade, emprega-se uma quantização de tempo com taxa amostral de $125Hz$, de forma que a 120BPM, intervalos de $8ms$ são modelados. Para informação de intensidade sonora, emprega-se uma escala discreta de 32 posições, suficientemente precisa para modelar intensidades entre *ppp* e fff.

Os autores fornecem duas versões pré-treinadas do modelo. A versão *performance* modela notas incluindo nuances temporais, mas não modela informação relativa à dinâmica das amostras, já *performance_with_dynamics* realiza também uma modelagem de dinâmica que a difere dos demais modelos estudados, motivando seu uso. O treinamento de ambas é feito com amostras de $30s$ de duração da base *Piano-e-Competition*, composta de amostras de recitais de piano clássico por pianistas profissionais, através de um processo estocástico decaimento de gradiente, com passos de 64 amostras e taxa de

aprendizado de 0.001. São empregadas duas aumentações de dados para cada amostra: (i) Desloca-se globalmente as alturas de suas notas num intervalo $[-4, +4]$ com relação às alturas originais, resultando em 8 amostras novas, ou , $[-6, +6]$, gerando 11 novas amostras (ii) Multiplica-se a escala de tempo uniformemente por $\pm 2.5\%$ e $\pm 5\%$, gerando 4 novas amostras, ou até $\pm 10\%$.

A avaliação de resultados é informal e baseada em testes de audição, não constituindo um resultado comparável aos demais modelos. Vaswani et al (2017) [51] mostra que o modelo é capaz de gerar amostras verossímeis (difíceis de distinguir de amostras reais) com duração em torno de 16 segundos.

2.4.3 PolyphonyRNN

PolyphonyRNN [52] busca modelar composições polifônicas para piano, ignorando portanto as nuances presentes na execução humana. O modelo é uma RNN-LSTM que utiliza uma linguagem própria de representação musical e possui três camadas ocultas, com 256 nós por camada por padrão. Os autores providenciam uma versão pré-treinada do modelo (*polyphony_rnn*), treinada com a base *Bach Chorales* sob um processo de decaimento de gradiente com 64 amostras por etapa, taxa de *dropout* de 0.5 e taxa de aprendizado 0.001.

Informações sobre o modelo são relativamente escassas: os autores providenciam instruções para a extração de métricas básicas (perda e acurácia) [52], porém não se encontra na literatura explorada uma avaliação dos resultados musicais do modelo, ou mesmo comparações com outros algoritmos.

2.4.4 PianorollRNN-NADE

PianorollRNN-NADE [53] é um modelo construído para a modelagem de composições polifônicas para piano. Baseia-se na arquitetura RNN-NADE [60], uma RNN híbrida que incorpora uma técnica chamada *Neural Autoregressive Distribution Estimation* (NADE). Assim como *PolyphonyRNN*, *PianorollRNN-NADE* quantifica o tempo de sua entrada e saída em batidas, e foi também treinado sobre a base *Bach Chorales*. São disponibilizadas duas versões pré-treinadas para o algoritmo: *rnn-nade* usa três camadas de RNN com 128 nós por camada, enquanto *rnn-nade* usa apenas duas, adotando em contrapartida um mecanismo de atenção, com 32 posições de comprimento, ambas empregando 128 unidades ocultas de NADE em sua arquitetura.

Semelhantemente ao modelo *PolyphonyRNN*, não há muita informação sobre seu treinamento além de terem sido as treinadas versões mencionadas sobre a base *Bach Chorales* com taxa de aprendizado de 0.001, taxa de *dropout* de 0.5 e 64 e 48 amostras por etapa, respectivamente. Apesar dos autores do *Magenta Research* não oferecerem uma avaliação

da saída do modelo, Boulanger-Lewandowski, et. al (2012) [60] mostram que *RNN-NADE* apresenta maior acurácia e menor perda que RNNs convencionais na modelagem de música polifônica.

2.5 Avaliação de Modelos de Sequência

Em Processamento de Linguagem Natural, há duas métricas padrão adotadas para a avaliação objetiva de modelos: chamam-se ROUGE [61] e BLEU [62], ambos indicadores escalares obtidos da comparação entre a saída de um modelo e um conjunto de saídas esperadas (ou, de referência). Para a saída gerada, o cálculo-base do BLEU considera, a porcentagem média de palavras e de sequências consecutivas de 2, 3 e 4 palavras presentes em alguma referência. Inversamente, ROUGE calcula a mesma média sobre uma saída de referência em comparação com a saída gerada

Mesmo para o domínio proposto, essas métricas ainda apresentam problemas fundamentais, por focarem no resultado léxico da saída e ignorar o aspecto semântico [49]. Avaliar resultados musicais é uma tarefa ainda mais desafiadora, não havendo sequer uma métrica padrão. As métricas tradicionais usadas no treinamento de modelos de AM (acurácia e erro) dizem muito pouco sobre a musicalidade de uma amostra ou conjunto. Por isso, é comum empregar-se uma combinação entre uma análise quantitativa, medindo aspectos objetivos de um conjunto amostral sonoro, e uma análise qualitativa, tipicamente na forma de um teste subjetivo de audição [10].

Yang et. al (2018) [21] propõe uma metodologia avaliativa para resultados musicais de modelos de sequência, com base em um conjunto de 12 métricas objetivas, usada em trabalhos como *BachDuet* para a avaliação do próprio modelo. Os autores fornecem a biblioteca *mgeval* [21], que implementa a extração das métricas propostas, que são:

- *Contagem de Alturas (PC):* Número de diferentes valores de *Pitch* de notas contidas em uma amostra. É um valor escalar no intervalo $[0 : 127]$.
- *Histograma de Classes de Pitch (PCH):* Histograma de dimensão 12 que representa o conteúdo relativo a *pitch* contido em uma amostra, correspondendo às 12 classes de *Pitch* na escala cromática (quantização padrão do espectro de frequência), sem distinção entre oitavas. Os valores do histograma são escalares no intervalo $[0 : \infty]$
- *Matriz de Transição entre Pitches (PCTM):* Histograma bidimensional (12×12) que quantifica transições entre pares de Classes de Pitch no espaço de uma amostra
- *Alcance de Pitch (PR):* Diferença entre as alturas máxima e mínima de um intervalo, indicando a dinâmica melódica do trecho. É um valor escalar.

- *Intervalo Médio entre Pitches (PI):* Distância absoluta média entre o pitch de cada par de notas consecutivas, em semi-tons. É um valor escalar.
- *Contagem de Notas (NC):* Número de notas contidas na amostra. Ao contrário do PC, é uma métrica rítmica, não contendo informação de *pitch*. É um valor inteiro no intervalo $[0 : \infty]$.
- *Intervalo Médio entre Onsets (IOI):* Intervalo médio, em segundos, entre os inícios de cada par de notas consecutivas, representado como um valor decimal no intervalo $[0 : \infty]$.
- *Histograma de Comprimentos de Notas (NLH):* Histograma de dimensão variável (12 ou 24) representando a distribuição de diferentes comprimentos de notas dentro de uma amostra. Os valores do histograma são escalares positivos no intervalo $[0 : \infty]$
- *Matriz de Transição entre Comprimentos de Notas (NLTM):* Semelhante à matriz anterior, um histograma bi-dimensional de dimensão 12×12 ou 24×24 que corresponde à transição entre diferentes comprimentos de notas no espaço de uma amostra. Os valores da matriz são inteiros positivos no intervalo $[0 : \infty]$

Para três dessas métricas (*PC*, *PCH* e *NC*), além do valor total sobre cada amostra, extrai-se também a média de cada valor ao longo dos diferentes compassos de uma peça:

- *Pitch Count per Bar (PC/bar):* Número de diferentes valores de *pitch* dentro de um mesmo, no intervalo $[0 : 127]$
- *Pitch Class Histogram per Bar (PCH/bar):* Histograma de Classes de Pitch por Compasso
- *Note Count per Bar (NC/bar):* Número total de notas dentro de um mesmo compasso

As métricas propostas não possuem uma única aplicação, podendo ser usadas tanto para comparação entre bases de dados quanto entre modelos de geração musical. Para o segundo caso, os autores propõem os seguintes passos: Suponha dois conjuntos de amostras musicais A e B. Primeiro, extrai-se o valor de cada métrica m para cada amostra em A e em B. Em seguida, para cada $C \in \{A, B\}$ e para cada m, compara-se o valor de m_C entre cada amostra de C através de uma validação cruzada exaustiva [63], gerando um histograma de distâncias H_{m_C}. Por fim, estima-se a Função Densidade de Probabilidade (FDP) de cada H_{m_C}, e calcula-se a distância entre H_{m_A} e H_{m_B} através de dois indicadores: *Overlap* e *KL-Divergence*.

Para cada métrica, quão maior seu *Overlap* e menor sua *KL-Divergence*, então mais próximo é o comportamento entre dois conjuntos. Os dois indicadores são valores decimais positivos: *Overlap* representa a integral da sobreposição entre duas FDPs e é portanto normalizado, enquanto *KL-Divergence* não possui limite superior. Dito isso, nos exemplos encontrados no trabalho, dados dois conjuntos A e B semelhantes, os valores de *Overlap* tendem a se encontrar no intervalo $[0 : 1]$, e os de *KL-Divergence*, no intervalo $[0 : 1.2]$. Por fim, é importante notar que os autores não propõem hierarquia entre as métrica, que podem apenas ser consideradas complementares.

2.6 Sistemas de Geração de Música

Na literatura é possível observar sistemas baseados em AM voltados à GD de música - aos quais nos referiremos com o termo Sistema de Geração Dinâmica Musical (SGDM) -, para tarefas como geração de trilha sonora dinâmica para jogos [42] e improviso interativo [64, 17]. Há exemplos na literatura de SGDMs desenvolvidos sobre modelos pré-existentes: AI-Duet e PSC2 [64] usam o MelodyRNN, enquanto BachDoodle [30] emprega o modelo COCONET [65] .

Uma solução arquitetural frequente nesses sistemas assemelha-se ao mecanismo *produtor-consumidor* [66]: adota-se um *buffer* para armazenar a informação musical [67, 64] e duas *threads* paralelas (leitura e escrita) ou mais, para as outras tarefas do sistema. Além disso, para que SGDMs sejam minimamente usáveis, são necessários componentes como interface e comunicação ou integração, que exigem também seus respectivos esforços de implementação.

Observa-se um retrabalho decorrente do desenvolvimento individual de cada sistema: cada nova ferramenta exige uma nova implementação dos mesmos componentes funcionais. Além disso, cada sistema mencionado baseia-se em um modelo específico [64, 17, 67] e restringe-se portanto a um caso de uso específico. A despeito de sua qualidade, para usar cada modelo como ferramenta de improviso é necessário implementar um SGDM próprio, o que é indesejável tendo em vista a frequência com que novos modelos para a tarefa de composição são desenvolvidos [36, 9, 17, 8, 24] . É possível elencar as seguintes características como aspectos relevantes para um SGDM ideal:

1. Geração Dinâmica: um sistema de geração estática não permite que a música seja ouvida conforme gerada, e portanto não cumpre com o requisito fundamental do improviso. Assim, sistemas que não implementem essa modalidade de geração não serão considerados SGDMs.

2. Interatividade: formas de controle do processo de geração oferecidas pelo sistema. Aplicações musicais exigem um nível elevado de responsividade, controle e interação [19] para que vejam uso prático.

3. Desacoplamento: para minimizar o custo de construção que futuras iterações de um SGDM com base em sucessores ou alternativas a seu próprio modelo, é importante que ambos não estejam fortemente acoplados.

4. Integração: conjunto de protocolos de transmissão de informação implementados pelo sistema, que possibilita seu uso em combinação com outros *softwares* de produção musical.

A tabela 2.2 resume os sistemas encontrados com relação às quatro características propostas. Nela, observa-se que nenhum dos sitemas encontrados permite simultaneamente GD e uso de diferentes modelos. Também não há na literatura uma forma de adaptar GE para GD de música. Dessa forma, o sistema proposta nesse trabalho é o único que cumpre os quatro requisitos elicitados.

Tabela 2.2: Sistemas de Geração Musical

Nome	Dinâmico	Interativo	Acoplado	Integração
AMS	**Sim**	**Sim**	Sim	OSC
AI Duet	**Sim**	**Sim**	Sim	MIDI
COCOCO[68]	Não	Não	**Não**	
BachDoodle[30]	Não	**Sim**	Sim	
BachDuet[31]	**Sim**	**Sim**	Sim	MIDI
NONOTO[69]	Não	Sim	Sim	
PSC2[64]	**Sim**	Não	Sim	MIDI
Sistema Proposto	**Sim**	**Sim**	**Sim**	MIDI/OSC

Capítulo 3

Proposta

A solução proposta envolve dois aspectos, descritos nesse capítulo: primeiro, uma metodologia para a adaptação de qualquer MS genérico de GE para GD é discutida na seção 3.1. Segundo, apresenta-se na seção 3.3 um sistema que implementa essa metodologia, que pode ser usado portanto para adaptar futuros MSs para GD de música. Para aferir a qualidade da solução proposta, propõe-se uma metodologia avaliativa que expande a avaliação proposta por Yang, et al. (2018) [21] para comparar resultados obtidos da GD resultante contra a GE original.

3.1 Adaptação de Modelos para Geração Dinâmica

Seja M um modelo de GE musical que empregue as representações R_E para sua entrada e R_S para sua saída. A adaptação de M para GD é possível através de um algoritmo de geração iterativa parametrizada. Esse algoritmo é construído em três etapas, a primeira das quais é a criação de um ciclo de realimentação:

1. Geração: Fornece-se a M uma entrada em R_E, gerando uma saída em R_S.
2. Tradução: traduz-se a saída de R_S para R_E.
3. Realimentação: A saída traduzida é concatenada à sequência de entrada original, gerando uma nova sequência de entrada em R_E.

A segunda etapa da adaptação é o controle dos parâmetros de geração. Os comprimentos das sequências tanto de entrada quanto da saída de um modelo impactam seu tempo de geração. Para não comprometer o caráter de GD do sistema, é necessário garantir que a geração ocorra dentro de um intervalo razoável de tempo. Isso torna necessário parametrizar ambos os comprimentos, referidos então como Comprimento de Janela de

Entrada e Comprimento de Janela de Saída. A combinação C_s^e de comprimentos de janelas de entrada e e de saída s é um fator determinante para o tempo de geração e para a qualidade da saída de M em GD, e será chamado de sua **configuração**.

A etapa final da adaptação é a implementação de um mecanismo de leitura e escrita simultânea da informação musical gerada, para permitir a execução da saída de M paralelamente à sua geração. A arquitetura produtor-consumidor é uma solução simples e eficaz para esse problema [66].

Concluídos os três passos, o algoritmo resultante executa GD de música usando M, originalmente construído para GE. Como resultado, o algoritmo M passa a cumprir com um requisito de tempo não contemplado previsto em sua versão original. O procedimento não impõe sobre M nenhuma restrição além da capacidade de gerar uma saída traduzível para a própria entrada, que vale para todos os modelos estudados.

3.2 Viabilidade Temporal de Modelos de Sequência

Complementarmente à viabilidade musical, é preciso determinar a viabilidade temporal de cada modelo. Visto que o tempo de geração de saída cresce em função da configuração de cada modelo, é necessário um limite de tempo que permita selecionar as configurações viáveis para uso em tempo real.

Para realizar uma comparação justa, é necessário determinar a configuração que forneça os melhores valores de *Overlap* métrico para cada modelo, dentro dos limites de tempo. Assim, o comprimento total de saída será fixo à duração de 16 compassos - aproximadamente 32 segundos sob 120 BPM - para o comprimento total de uma composição.

Para que uma configuração seja considerada viável, seu tempo de geração não pode ultrapassar a duração total da saída gerada - do contrário, será causado um *underflow* no *buffer*, ou seja, a execução da composição será interrompida por falta de dados até que a geração conclua, comprometendo seu caráter de GD. É importante observar que o tempo de saída gerado varia em função do andamento da peça, representado pelo seu BPM.

Assim, o BPM médio de cada conjunto de teste deve ser calculado e usado para filtrar as configurações de cada modelo. Se uma configuração não gera uma saída musicalmente coerente e maior do que o próprio tempo de geração dentro do BPM médio de seu próprio estilo, isso significa que não é possível usá-la em tempo real para mais da metade de seus casos de uso esperados (valores de BPM acima da média), e então conclui-se que não é apta para uso em tempo real para aquele estilo.

3.3 Arquitetura do Sistema Proposto

Propomos um SGDM chamado *Ornette*[1], que permite através da adaptação proposta o uso de diferentes MSs para GD de música. Sua arquitetura implementa o mecanismo descrito de geração iterativa: a saída gerada pelo modelo servirá de entrada para seu próximo passo de geração, até que se atinja o comprimento desejado de uma composição. Assim, é possível gerar composições de comprimentos variáveis de forma parametrizável, partindo ou não de uma sequência inicial. Para reproduzir os resultados musicais, o sistema emprega uma solução produtor-consumidor, conforme implementada por outros SGDMs.

A execução do sistema começa com a inicialização um servidor que pode receber comandos, indicando quando começar e quando finalizar a geração de improviso. Dado o comando de início, o programa inicia o ciclo de geração iterativa, alimentando ao modelo os dados armazenados em memória e construindo progressivamente o improviso. Enquanto acontece a geração iterativa, a informação MIDI gerada é consumida e sinais OSC são disparados para sistemas externos, que podem capturar essa informação para a geração de som. Assim, é possível ouvir a composição gerada simultaneamente à sua criação. É possível carregar um arquivo MIDI para servir como base ao processo de geração ou simplesmente solicitar ao modelo que gere informação nova. Ao fim do processo, a critério do usuário, um arquivo MIDI pode ser criado, contendo o improviso gerado.

A Figura 3.1 ilustra a alto nível a arquitetura interna do sistema quando inicializado com um modelo específico, mostrando o fluxo de informação descrito. À direita da figura, encontra-se o sistema: a hierarquia entre seus componentes é descrita pela relação entre as caixas ilustradas. As setas representam o fluxo de informação (musical e de controle) tanto entre os componentes quanto com sistemas externos. O ciclo principal do programa (o processo de geração iterativa) se encontra entre os dois componentes internos do servidor em evidência: a informação MIDI (em memória) e o MS selecionado. As saídas de interesse - Som e Arquivos MIDI - estão representadas como caixas azuis na parte inferior à esquerda, e as caixas amarelas à esquerda da figura representam quaisquer fontes e destinos externos de informação. A cor verde representa partes do sistema que podem ser estendidas ou manipuladas pelo usuário, especificamente o MS e o mecanismo de tradução baseado em filtros, descrito em 3.3.2.

3.3.1 Interatividade

O controle do processo generativo exige que o sistema implemente uma série de rotinas, para as quais utiliza-se a abstração de *comandos*. Os comandos do sistema implemen-

[1] https://github.com/ghalestrilo/ornette

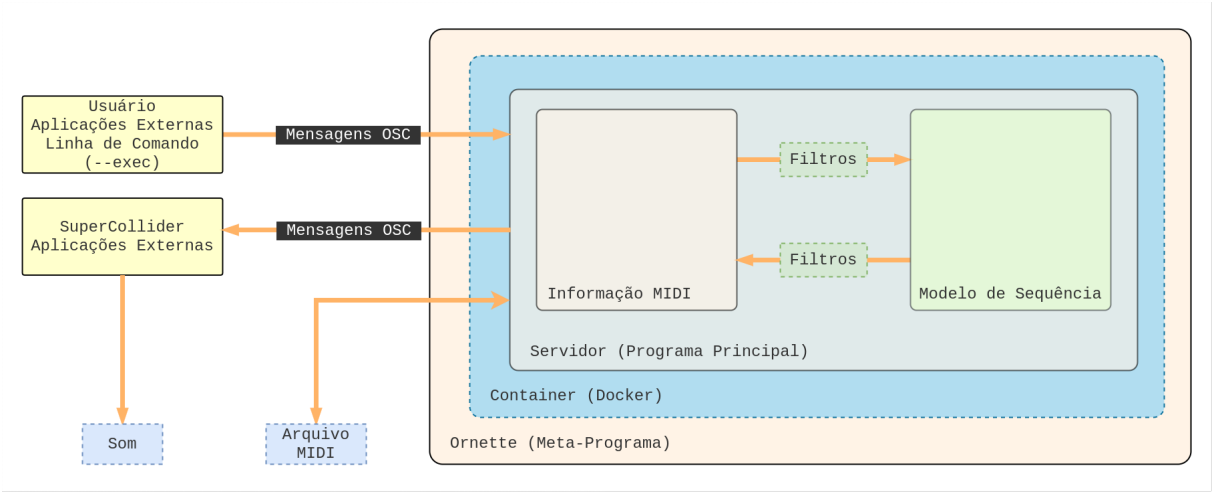


Figura 3.1: Arquitetura do Sistema Proposto

tam operações básicas de controle sobre o modelo inicializado, incluindo rotinas para carregar/salvar um arquivo MIDI, descartar notas antes ou depois de um determinado intervalo, gerar uma quantidade determinada de compassos ou definir constantes internas ao programa, como o comprimento de entrada e saída. Há duas formas imperativas de executar-se os comandos descritos:

- *Linha de Comando:* Com o uso da flag `--exec`, é possível alimentar ao comando inicial do sistema uma sequência de instruções - isso é usado nos *scripts* avaliativos para simplificar o processo de geração de amostras.
- *Comunicação OSC:* A integração principal do sistema ocorre através de comunicação OSC, dessa forma, mensagens OSC, quando recebidas pela porta UDP reservada pelo programa, dão início à execução do comando solicitado.

3.3.2 Desacoplamento

A estrutura mais comum nos sistemas estudados parte de um modelo específico, construindo sobre ele mecanismos para cumprir quaisquer requisitos de interação e usabilidade. Com o objetivo de permitir o uso de modelos distintos, explora uma abordagem alternativa: cada modelo é encapsulado sob uma mesma interface e o sistema então gerencia a geração e realimentação do modelo escolhido. A figura 3.2 ilustra essa diferença: à esquerda da figura, exemplifica-se um SGDM tradicional, em que um MS é parte integrante do sistema, e à direita, o princípio organizacional adotado no desenvolvimento do *Ornette*, onde cada MS é compartimentado, permitindo a integração de novas redes conforme necessário.

As diferentes infra-estruturas de AM usadas para desenvolver MSs não são necessariamente compatíveis, e é ocasionalmente inviável prover suporte simultâneo a elas a nível de configuração de software. Assim, para garantir que seja possível usar modelos construídos sobre diferentes infra-estruturas intercambiavelmente é necessário simular ambientes distintos. A ferramenta *Docker* [70] permite a virtualização de ambientes a nível de sistema operacional usando uma abstração chamada *container*. Um *container* é uma unidade de virtualização cuja configuração pode ser definida através de um documento de configuração, chamado Imagem. Assim, para dar suporte a diferentes infra-estruturas, *Ornette* emprega imagens *Docker* para virtualizar o ambiente em que o MS será executado.

O aspecto multi-modelo da solução emprega o conceito de um *Módulo*. Para o sistema, um *Módulo* é a unidade que encapsula cada modelo generativo, e compõe-se de dois elementos básicos:

1. Classe *Ornette*: Apesar das infra-estruturas distintas, a implementação dos modelos estudados compartilha da mesma linguagem de programação: *Python*. Portanto, adota-se um arquivo *Python* com uma definição de classe como ponto de entrada para a execução de cada modelo, permitindo assim o uso do código já existente, em acordo com a proposta desse trabalho.
2. Imagem Docker: esse documento define as características de software do *container* dentro do qual a classe será executada.

Essa estrutura é suficiente para permitir ao sistema o uso de diferentes modelos - a Figura 3.2 ilustra-a de forma simplificada. Dentro de um módulo, o autor tem total autonomia para transformar o *buffer* recebido conforme desejado, podendo inclusive convertê-lo entre formatos/estruturas de dados distintos, contanto que a informação retornada pelo módulo seja MIDI-compatível.

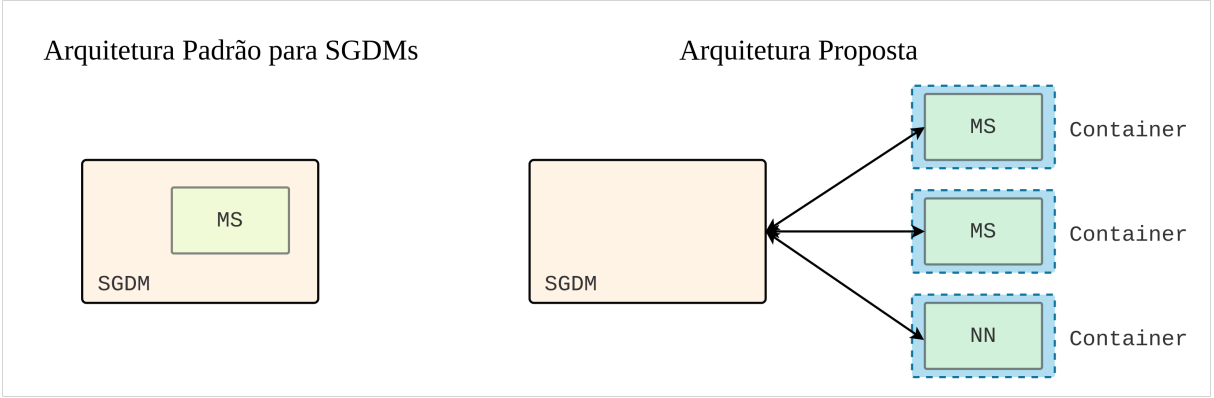


Figura 3.2: Arquitetura Proposta para SGDMs

É possível que mais de um modelo represente informação musical usando estruturas semelhantes, como é o caso das quatro RNNs selecionadas para esse estudo (*MelodyRNN*, *PerformanceRNN*, *PolyphonyRNN* e *PianorollRNN-NADE*), que internamente armazenam informação musical em NoteSeqs. Assim, é possível simplificar a etapa de tradução do ciclo de geração iterativa, extraindo-se lógicas de conversão entre representações como componentes funcionais reusáveis. O sistema oferece uma abstração para essas unidades de conversão, chamando-as de filtros, que podem ser declarativamente empregados por cada módulo. Na Figura 3.1, observa-se o posicionamento dos filtros, intermediando o fluxo circular de dados entre a Informação MIDI armazenada no servidor e o MS selecionado.

Para adaptação dos quatro modelos estudados, tendo em vista que empregam uma mesma representação interna, foram criados dois filtros para implementar a etapa de tradução entre a representação usada por cada modelo e a informação MIDI armazenada no sistema. São eles: (*i*) *midotrack2noteseq*, que converte faixas MIDI para NoteSeqs, e; (*ii*) *noteseq2midotrack*, que converte NoteSeqs para faixas MIDI. Usando esses dois filtros, o sistema consegue então realimentar a cada MS sua própria entrada.

Capítulo 4

Metodologia

Esse capítulo descreve a metodologia avaliativa usada para a avaliação da técnica proposta. O processo compõe-se de três fases principais: (*i*) a fase de geração das amostras; (*ii*) o estudo de viabilidade de uso em tempo real; e (*iii*) as avaliações objetiva e subjetiva.

A Figura 4.1 resume através de um fluxograma o processo completo de avaliação. Na figura, as setas indicam o fluxo de dados, os blocos amarelos indicam transformações sobre esses dados e os blocos roxos indicam operações auxiliares que não modificam os dados sobre os quais operam. As figuras azuis indicam conjuntos de amostras musicais, sendo pontilhados os conjuntos fabricados experimentalmente. Por fim, as caixas vermelhas ao fim do processo indicam o método estatístico usado para comparar as duas formas de geração através dos dados gerados, constatando se houve ou não diferença significativa decorrente da adaptação proposta.

Partindo das bases de testes usadas no treinamento dos modelos selecionados, conduz-se GE e GD de amostras de cada modelo. Em seguida, um breve pré-processamento prepara as amostras para a extração de métricas, resultando em bases sintéticas usadas durante a terceira fase. Isso é feito também com as bases de testes, para servirem como base de comparação para as amostras geradas.

Uma vez geradas as amostras, é feito um estudo de viabilidade de uso em tempo real das configurações de GD usadas. Conduz-se uma análise preliminar das bases de teste para determinar os limites de tempo adequados para a GD, usando-os então para selecionar apenas as configurações viáveis.

Por fim, as bases sintéticas são usadas para a fase de avaliação objetiva e subjetiva. Para a etapa objetiva, extraem-se métricas das amostras de cada base, que serão usadas para comparar a distância que cada base de geração possui com relação à base de teste. Para GD, selecionam-se os parâmetros de configuração que minimizam essa distância. Essa distância é comparada com a distância obtida por GE através de um teste estatístico.

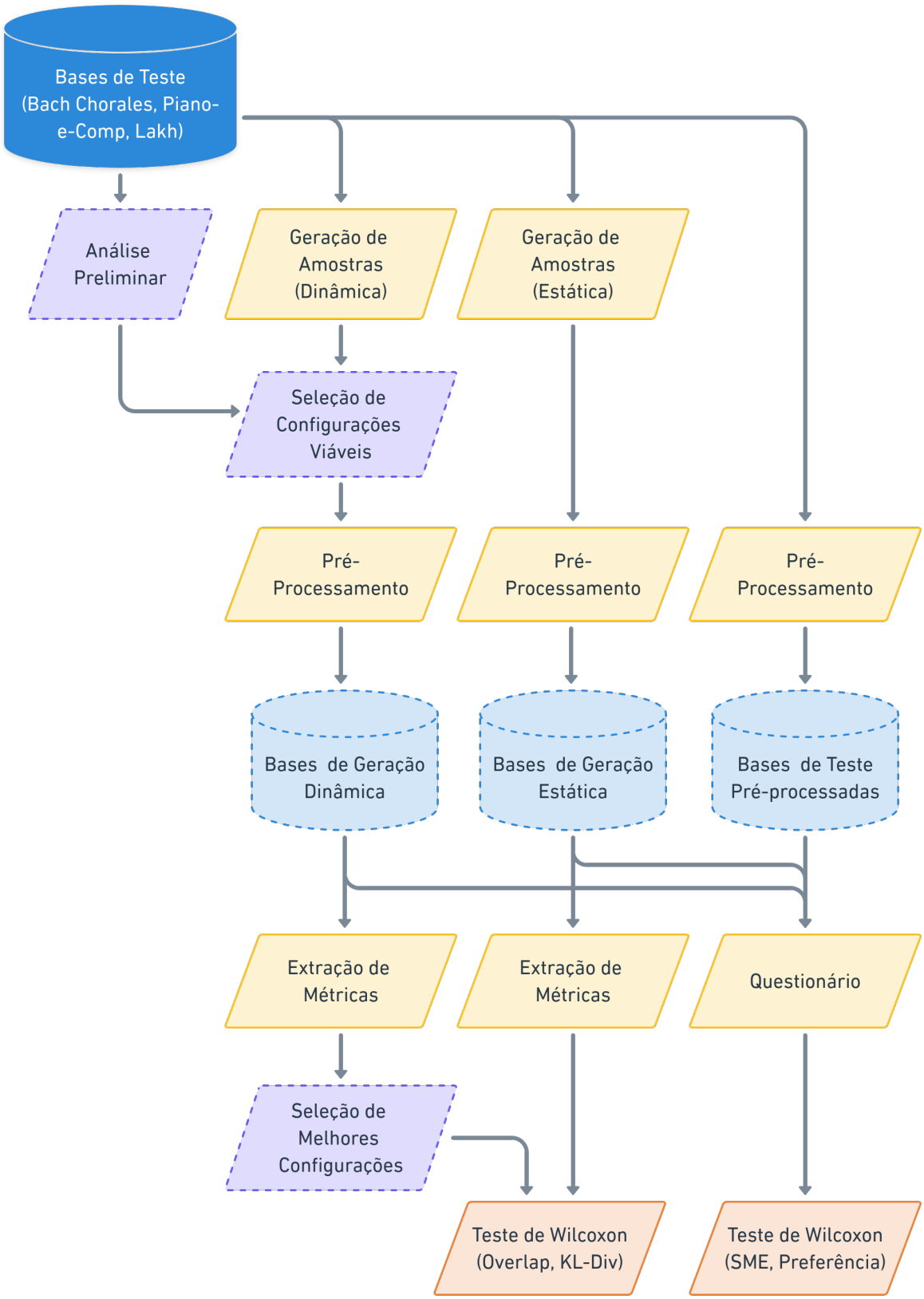


Figura 4.1: Fluxograma do Experimento

Para a avaliação subjetiva, conduz-se uma pesquisa na forma de um formulário composto de duas partes. Uma variante do Teste de Turing é usada para aferir o impacto da adaptação sobre a facilidade com que amostras geradas são detectadas quando comparadas a amostras reais. Além disso, um estudo de preferência é conduzido para avaliar seu impacto sobre a qualidade percebida da saída musical dos modelos.

4.1 Fase de Geração de Amostras

A análise proposta baseia-se em amostras das duas modalidades de geração (GE e GD) para cada um dos modelos estudados (*MelodyRNN*, *PolyphonyRNN*, *PerformanceRNN* e *PianorollRNN-NADE*). Cada modelo gerará continuações baseadas em amostras de sua própria base de treinamento. As continuações resultantes serão então extraídas em arquivos MIDI para uso durante as demais fases avaliativa.

Para cada base de treinamento T, foi selecionado um subconjunto amostral B_T contendo 8 elementos para servirem como entrada para os processos de geração. B_T é então usado tanto para GE, que resulta numa base de amostras de saída B_{GE}, quanto para GD sob cada configuração c, gerando B^c_{GD}. O espaço de configurações de GD para avaliação inclui comprimentos de entrada e saída escalares inteiros variando independentemente no intervalo $I_c = [1 : 6]$, em compassos. Dado o comprimento limitado das amostras, configurações acima de 6 compassos de saída perderiam seu caráter iterativo, fugindo à proposta da adaptação. Assim, 36 diferentes configurações foram avaliadas por modelo.

Buscando minimizar o viés aleatório do processo de geração, gerou-se $n_e = 4$ amostras para B_{GE} por amostra de entrada, totalizando 32 saídas por modelo. Além disso, o processo de geração como um todo foi repetido $r = 8$ vezes, visando melhorar a convergência métrica para cada teste. O número de elementos no conjunto final de amostras geradas $C_{saída}$ é calculado pela equação 4.1.

$$\begin{aligned} |C_{saída}| &= |C_{modelos}| \times |C_{amostras}| \times (n_e + r \times |I_c|^2) \\ &= 4 \times 6 \times (4 + 8 \times 6^2) \\ &= 7008 \end{aligned} \tag{4.1}$$

Pode haver inconsistências na formatação dos arquivos contidos entre bases de teste diferentes. Por conta disso, uma etapa breve de pré-processamento é conduzida sobre as amostras de saída, garantindo que seja possível extrair as métricas de maneira uniforme. Além disso, essa etapa também remove o trecho inicial alimentado aos modelos e remove qualquer excesso das amostras, garantindo que não ultrapassem o comprimento desejado.

4.2 Estudo de Viabilidade de Uso em Tempo Real

O objetivo dessa fase é determinar quais configurações de cada modelo podem ser usadas para GD, para garantir o caráter dinâmico do processo. Isso equivale a encontrar as configurações estudadas sob as quais cada modelo é capaz de gerar informação musical continuamente, conforme é consumida. Para isso, toma-se como base o BPM médio de cada base de testes, fator que determina o tempo que cada compasso dura, e portanto o limite de tempo que um MS predispõe para geração de cada compasso.

Para todo processo de AM é necessária uma base de dados de treinamento. Os modelos estudados foram treinados sobre três bases distintas. *PerformanceRNN* emprega a base *Piano-e-Competition* composta de trechos de composições clássicas (polifônicas) sendo executadas por instrumentistas reais. Os modelos *PolyphonyRNN* e *PianorollRNN-NADE* foram treinados sobre a mesma base, *Bach Chorales*, composta de composições polifônicas transcritas para MIDI do autor Johann Sebastian Bach.

Apesar de existirem bases de dados comuns em MIR, é comum que autores desenvolvam as próprias bases para o treinamento dos modelos [57, 24]. O modelo *MelodyRNN* emprega uma base própria, não disponibilizada pelos autores, que afirmam que o modelo foi treinado sobre uma base de melodias (camadas monofônicas) de música popular. Para estudá-lo, será adaptada uma base de dados de música popular chamada *Lakh* [71], extraindo apenas as notas de um instrumento monofônico por amostra, aproximando o comportamento da base proposta pelos autores.

Bach Chorales, *Piano-e-Competition* e *LAKH* são exemplos de bases polifônicas populares em MIR. São compostas integralmente por arquivos MIDI e usadas para diversas tarefas, incluindo o treinamento de MSs para geração musical [8, 17, 9, 52, 53] .

Os MSs selecionados são treinados especificamente para a geração de música cuja fórmula de compasso seja 4/4, havendo exatamente 4 batidas por compasso. Amostras com fórmulas de compasso diferentes são então removidas das bases de dados escolhidas para permitir uma avaliação justa dos processos de geração. Tendo isso em vista, extraindo-se o BPM de cada arquivo na base, é possível calcular então a média $\overline{BPM}$ e por fim a duração média por compasso d_c através da equação 4.2.

$$d_c = \frac{60 \times 1000}{(\overline{BPM} \times 4)} ms/Bar \tag{4.2}$$

Os valores médios de BPM ($\overline{BPM}$) encontrados por meio desse cálculo variam entre 88BPM e 120BPM através das três bases analisadas. Como resultado, as durações médias por compasso d_c variam entre 2000ms/Bar e 2811.29ms/Bar, conforme ilustra a Tabela 4.1.

Tabela 4.1: Características Médias das Bases de Dados Analisadas

Base de Dados	Batidas/min	Batidas	Compassos	$\bar{d_c}(ms)$
Bach Chorales	88.02	77.85	19.46	2811.29
Lakh	115.34	411.16	102.79	2213.69
Piano-e-Competition	120.00	1145.17	286.29	2000.00

Observa-se para a base *Bach Chorales* a maior duração média por compasso, com aproximadamente 2.8s. Comparada à mesma medida para a base *Piano-e-Competition* (2s), infere-se que modelos treinados sobre a primeira dispõem em média de maior tempo para geração. Isso significa que o espaço de configurações viáveis para um modelo hipotético seria mais amplo quando treinado sobre *Bach Chorales*, possivelmente demonstrando melhores métricas por conta disso. Por tratarmos de modelos diferentes nesse estudo, não é possível fazer uma comparação direta da mesma forma.

4.3 Avaliação de Geração Dinâmica

O resultado mais importante desse estudo é o impacto da adaptação para GD de MSs voltados a GE. Assim, tomaremos os modelos sob GE como base de comparação: medições objetivas e subjetivas serão feitas sobre cada MS, antes e depois de sua adaptação para GD através do sistema proposto. A variação entre as medições feitas será então avaliada. Caso não se perceba diferença significativa entre ambos, concluímos que não há prejuízo no uso dos modelos adaptados para GD, com relação a suas versões originais, e portanto a técnica proposta é viável para promover um MS para GD sem prejuízo ao resultado musical. Por fim, para determinar se há diferença estatisticamente significativa entre as modalidades de geração, será conduzido um Teste de Wilcoxon [72] ao fim de cada etapa avaliativa: sobre as distâncias entre as métricas extraídas, na fase objetiva, e sobre cada componente da fase subjetiva.

4.3.1 Avaliação Objetiva

O objetivo desa fase de avaliação é constatar se estruturalmente há diferença para um dado modelo M entre sua GD adaptada e sua GE original. Isso é feito com base nas 12 métricas propostas por Yang, et. al [21]. As métricas não dizem respeito à qualidade musical de cada amostra, mas cumprem um papel descritivo, servindo então como referência para comparar a semelhança entre conjuntos distintos de amostras, através dos indicadores de distância *Overlap* e *KL-Divergence*, ambos valores escalares contínuos.

O primeiro passo dessa fase é a extração da cada métrica m a partir de cada base sintética (B_T, B_{GE} e B_{GD}) e a aproximação de suas respectivas FDPs. Para quantificar a distância entre as amostras geradas sob GE e as amostras de teste, calcula-se o *Overlap* $O_m\langle GE, T\rangle$ e a *KL-Divergence* $KL_m\langle GE, T\rangle$ de cada métrica m entre B_{GE} e B_T. O mesmo é feito para GD, resultando nos indicadores de distância $O_m\langle GD, T\rangle$ e $O_m\langle GD, T\rangle$.

Por fim, aplica-se um Teste de Wilcoxon para verificar se a distância métrica entre as saídas dos modelos e suas respectivas bases de testes é impactada pela adaptação proposta. O teste é aplicado duas vezes: uma para calcular o $w-valor$ entre $O\langle GE, T\rangle$ e $O\langle GD, T\rangle$ e seu respectivo $p-valor$, e outra para fazer o cálculo análogo entre $KL\langle GE, T\rangle$ e $KL\langle GD, T\rangle$. Caso ambos os z-valores sejam positivos e possuam p-valores acima de 0.05, então não há diferença significativa entre as distâncias pré e pós-adaptação para GD, rejeitando-se então a hipótese de que a adaptação proposta incorra em um impacto negativo à qualidade sonora dos modelos estudados.

4.3.2 Avaliação Subjetiva

Em paralelo à análise objetiva, uma avaliação subjetiva é indispensável para validar a musicalidade dos resultados obtidos. Uma solução comum na literatura é um teste conhecido como Teste de Turing Subject Matter Expert (SME) [73], variante do teste original onde um *expert* em um determinado campo de conhecimento é desafiado a distinguir amostras reais de artificiais para um determinado tipo de dado. Em adição a isso, empregamos um questionário simples de preferência, onde solicita-se que o candidato manifeste sua preferência por cada amostra.

Para realizar essa avaliação, foi construído um formulário selecionando-se amostras aleatoriamente entre as bases B_T, B_{GE} e B_{GD}, contendo mesmo número de amostras reais, geradas estaticamente e geradas dinamicamente. Os arquivos MIDI selecionados foram renderizados para amostras de áudio, fornecidas aos candidatos da pesquisa. O questionário foi enviado a 35 candidatos, prevendo quatro diferentes graus de instrução musical, em ordem crescente: Leigo, Aprendiz, Técnico ou Informalmente Experiente e Graduado ou Pós-Graduado em Música, sendo considerados como *experts* as duas últimas categorias.

Feito o Teste de Turing SME, são calculados os índices percentuais de acertos por categoria, e conduzido um teste de Wilcoxon sobre os percentuais referentes às amostras geradas estaticamente e dinamicamente, respectivamente. Por fim, o mesmo processo será repetido para as notas médias de cada modelo, com o objetivo de constatar se a variação de modalidade de geração impacta sua qualidade subjetiva. Assim, é possível avaliar então o impacto da técnica proposta sobre ambas a verossimilhança e a qualidade das amostras geradas.

Capítulo 5

Resultados

Esse capítulo apresenta os resultados do experimento proposto e divide-se em três seções: As seções 5.1 e 5.2 discutem o resultados obtidos durante as fases de avaliação objetiva e subjetiva, respectivamente. A última seção explora os resultados dos testes estatísticos realizados sobre ambas as etapas. A seção 5.1 discute os resultados encontrados através da avaliação objetiva dos modelos, apresentando as configurações viáveis para uso em GD, os valores de distância encontrados entre as métricas de GD e as métricas de teste e comparando-os por fim com os valores equivalentes obtidos através de GE.

Em seguida, a seção 5.2 apresenta os resultados subjetivos encontrados. Primeiro, discute-se o teste SME e as taxas de acerto dos usuários na detecção de amostras artificiais, e então são apresentados os valores de qualidade musical percebidos pelos participantes da pesquisa. Concluindo o capítulo, a seção 5.3 apresenta e discute os testes estatísticos conduzidos sobre ambas as etapas de avaliação para aferir a diferença entre GE e GD na tarefa de geração automática de música.

5.1 Avaliação Objetiva

O resultado final da avaliação objetiva é a comparação entre as medidas de distância $O_m\langle M_c, T\rangle$ e $KL_m\langle M_c, T\rangle$, obtidos entre os dados de teste T e a melhor configuração c de cada dado modelo M com relação a cada métrica m de interesse. Esse valor é construído dos resultados intermediários referentes às etapas anteriores da metodologia avaliativa. O primeiro é a análise de viabilidade, indicando quais configurações analisadas são inviáveis num contexto de Geração Dinâmica. Em seguida, para as configurações viáveis, são encontrados os valores obtidos de distância $O_m\langle M, T\rangle$ e $KL_m\langle M, T\rangle$ por métrica m com relação à base de teste B_T. A média do *Overlap* $O\langle GD, T\rangle$ por modelo determina então a melhor configuração C de GD para aquele modelo. Finalmente, é possível fazer a

comparação lado a lado entre o melhor resultado de GD, concluindo essa etapa da da análise

As próximas subseções discutem os valores obtidos em cada etapa descrita: 5.1.1 expõe as configurações viáveis encontradas, 5.1.2 apresenta as métricas obtidas para as configurações viáveis e 5.1.3 conclui a análise de resultados objetivos, comparando os indicadores de GD obtidos contra seus correspondentes em GE.

5.1.1 Viabilidade de Configurações em Tempo Real

A duração média por compasso d_c aferida na base de treinamento de cada modelo estabelece o limite de tempo para a geração de um compasso d_g, que varia de acordo com a configuração escolhida. Assim, a viabilidade de cada configuração é determinada por modelo comparando-se d_g e d_c. A Figura 5.1 mostra as curvas observadas para a variável tempo de geração d_g, indicada em ms no eixo vertical da figura. O eixo horizontal indica a configuração usada para GD: o valor superior indica o comprimento da janela de entrada, em compassos, e o inferior, o comprimento da janela de saída.

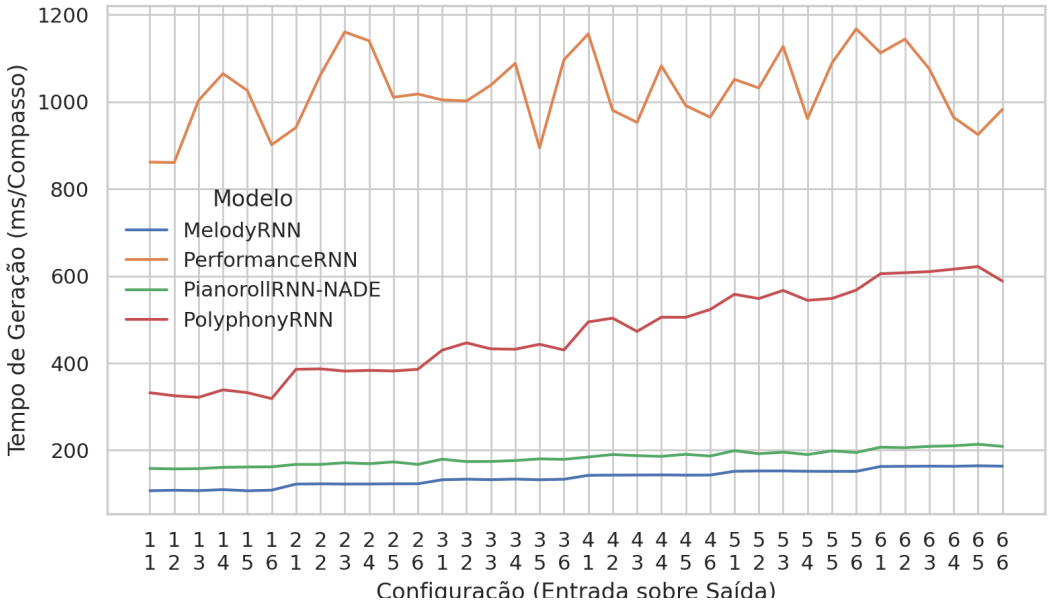


Figura 5.1: Tempo de Geração Por Configuração

Observa-se para três dos quatro modelos (*PolyphonyRNN*, *PianorollRNN-NADE* e *MelodyRNN*) um comportamento crescente de d_g com em função da configuração escolhida. Especificamente, o comprimento da janela de entrada parece dominar assintoticamente o comprimento da janela de saída, sendo portanto o parâmetro mais determinante

para d_g. Constata-se que o efeito do comprimento da janela de saída em d_g, caso haja, é basicamente irrelevante.

Dito isso, observa-se que nenhuma configuração dentro do espaço definido ultrapassa o limiar estabelecido pela própria base de dados: os tempos de geração máximos d_g observados para *PianorollRNN-NADE* e *PolyphonyRNN* aproximam-se de $600ms$ e $200ms$, respectivamente, para um valor $d_c = 2811.29ms$. De maneira semelhante, o tempo de geração do *MelodyRNN* não supera $200ms$ para um limite de $2213.69ms$, e por fim os valores máximos para o *PerformanceRNN* estão no intervalo $[1000 : 1200]ms$, sendo seu limite de tempo igual a 2000ms.

Em outras palavras, o espaço de configurações definido como escopo do estudo é menor que o espaço de configurações viáveis. Primeiro, isso indica que todas podem ser usadas para geração dinâmica sem apresentar problemas para valores de BPM próximos à média de suas respectivas bases de treinamento. Segundo, indica também que configurações adicionais podem ser estudadas: apesar de potencialmente benéfico com relação à estrutura da saída, e portanto às métricas finais, é possível que um aumento excessivo agrave indesejavelmente o efeito de esquecimento. Além disso, tamanhos maiores de janelas de saída implicam em maiores segmentos contíguos para um mesmo contexto musical, o que não é necessariamente desejável para amostras com trocas de contexto frequentes.

5.1.2 Análise de Distância entre Métricas

Feita a pré-seleção de configurações viáveis, analisa-se as amostras geradas com relação ao conjunto de métricas estabelecido. Para cada configuração, calcula-se então o *Overlap* e a *KL-Divergence* por métrica contra seu próprio conjunto de teste. Para determinar a melhor configuração de cada MS, serão considerados os valores de *Overlap*, ilustrados nas Figuras 5.2 e 5.3.

As Figuras representam os valores encontrados de *Overlap* para cada métrica discutida para os conjunto amostrais de GD com relação às respectivas base de testes, na forma de um mapa de calor onde o eixo vertical indica a configuração usada, o horizontal indica os diferentes modelos estudados e cada célula contém o valor de *Overlap* obtido de cada métrica usando o respectivo modelo e configuração.

Por tratar-se de métricas independentes e desprovidas de qualquer hierarquia, não há uma métrica chave que determine a superioridade de configurações específicas em relação às demais. Também não se observa facilmente um padrão que relacione as configurações testadas aos valores de distância medidos. Dessa forma, é difícil distinguir de algum possível ruído experimental o efeito da seleção de configurações distintas. Métricas como *PS* por exemplo mostram valores próximos para basicamente todas as configurações, enquanto outras métricas como *NC/Bar* mostram um grau mais expressivo de variação.

Figura 5.2: Overlap Métrico por Modelo, por Configuração

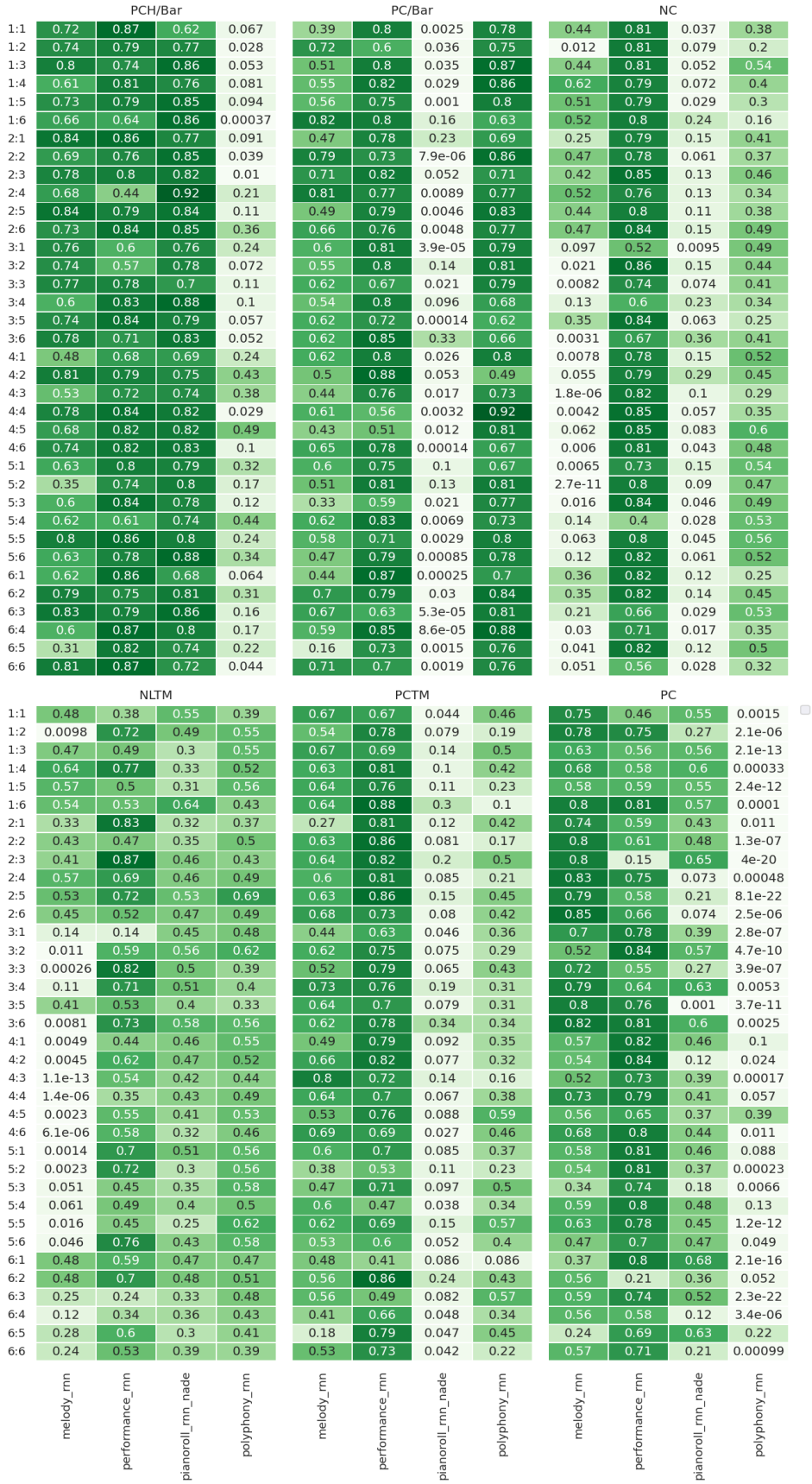
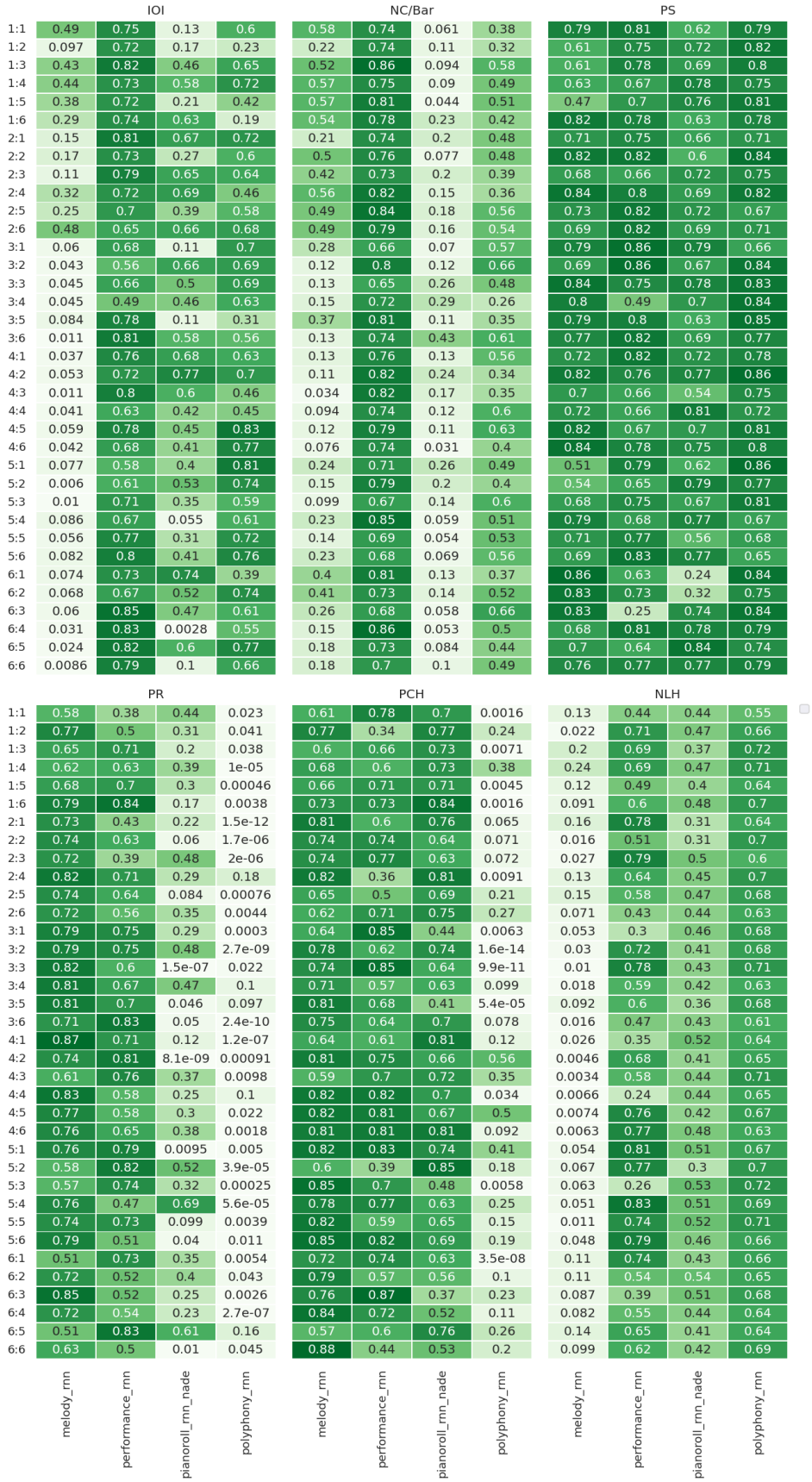


Figura 5.3: Overlap Métrico por Modelo, por Configuração



Dito isso, é possível observar algumas configurações que se destacam com relação a outras: A configuração C_5^6, por exemplo, minimiza o *Overlap* de uma das métricas PC, PC/Bar, NC e PCTM para o *MelodyRNN*, enquanto C_5^3 gera um efeito semelhante para o *PianorolLRNN-NADE*. Da mesma forma, há configurações ideais para cada modelo, que em média maximizam o *Overlap* $O\langle GD, T\rangle$ - essas configurações são então selecionadas e expostos na Tabela 5.1.

É desejável que os valores de *Overlap* $O_m\langle A, B\rangle$ entre duas bases A e B seja o mais alto possível para cada dada métrica m. Esse indicador tende a apresentar menos *outliers* e a concentrar-se no intervalo contínuo $[0 : 1]$ de forma mais estável que a *KL-Divergence*. Assim, a média entre os valores de $O\langle GD, T\rangle$ será usada para determinar a configuração ideal de cada modelo, que deve então maximizar esse valor, indicando maior verossimilhança das amostras geradas.

Modelo	Entrada Ideal (Compassos)	Saída Ideal (Compassos)
MelodyRNN	2	4
PerformanceRNN	4	2
PianorollRNN-NADE	3	6
PolyphonyRNN	4	5

Tabela 5.1: Configuração Ideal por Modelo

A Figura 5.4 apresenta os valores de *KL-Divergence* e *Overlap* por métrica para a base amostral de GD de cada MS com respeito às próprias bases de teste. Cada cor diferente indica um MS distinto sob sua configuração ideal de *GD*. O eixo vertical é dividido para representar cada uma das 12 métricas, e o eixo horizontal representa o valor escalar para cada um dos indicadores de distância para cada métrica e modelo: *KL-Divergence* à esquerda e *Overlap* à direita.

Observa-se valores baixos de *KL-Divergence*, e com dispersão relativamente baixa entre os modelos, enquanto os valores de *Overlap* métrico apresentam um grau maior de variação. Os trabalhos que propõem cada MS não realizam uma avaliação semelhante, por isso será empregada como base de comparação para os modelos adaptados o próprio modelo em GE. Os valores de $O\langle GD, T\rangle$ variam por modelo, situando-se principalmente na faixa entre 0.4 e 0.8. Isso indica um resultado bom: a variável correspondente $O\langle GE, T\rangle$ para GE situa-se no intervalo $[0.5 : 0.9]$, concentrando-se em torno de 0.8.

Destaca-se o modelo *PerformanceRNN*, cuja versão adaptada apresenta valores consistentemente altos de $O\langle GD, T\rangle$, na faixa $[0.6 : 0.9]$. Isso indica que suas amostras devem ser mais verossímeis que as de outros modelos, ou seja, mais difíceis de identificar comparadas às amostras de teste. Em contraste, o *PianorollRNN-NADE* mostra performance visivelmente inferior uma vez adaptado, com valores de $O\langle GD, T\rangle$ mais baixos e disper-

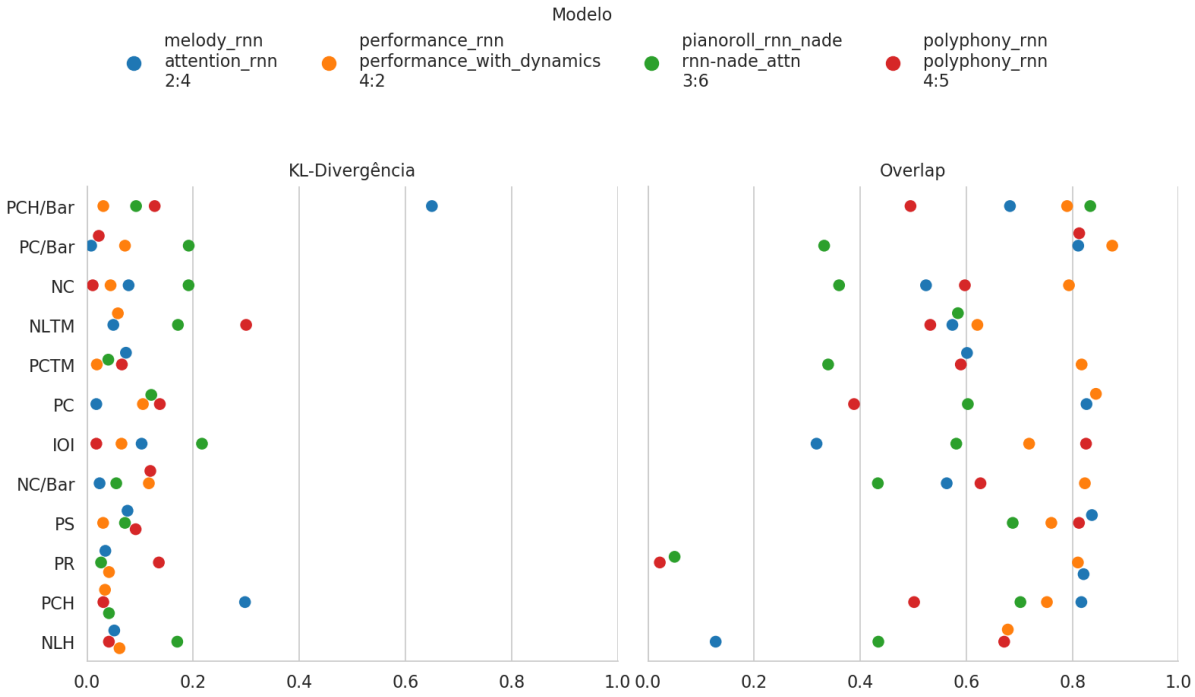


Figura 5.4: Pontuação Métrica por Modelo - Melhores Configurações

sos. Mesmo com uma quantidade razoável de amostras em análise, esse *Overlap* ainda apresenta um grau considerável de dispersão, o que pode acontecer em razão do tamanho limitado do espaço amostral.

Observa-se também uma inconsistência nas métricas relativas aos modelos *PolyphonyRNN* e *PianorollRNN*. Apesar de ambos haverem sido treinados com a mesma base de dados e para tarefas semelhantes, mostram valores discrepantes de *Overlap* para algumas métricas, como *PCH/Bar*, *PC/Bar*, *PC*, e *PR*. Além de surpreendente, visto que a saída original de ambos os modelos é semelhante, esse resultado aponta para a a possibilidade que a implementação do sistema introduza algum grau de erro aos modelos, distorcendo de alguma forma a estrutura da saída desses dois modelos.

5.1.3 Comparação com Métricas de Referência

A última etapa da avaliação objetiva consiste na comparação das distâncias aferidas para cada métrica em GD com seu correspondente em GE. A Figura 5.5 e a Tabela 5.2 dispõem os valores dos indicadores de distância relativos a cada métrica, agrupados por modelo.

A Tabela divide-se em oito colunas, agrupadas em pares por MS, e representa em ordem a GD e GE para cada MS de interesse. A metade superior da tabela representa os valores aferidos de *KL-Divergence*, e a inferior, os *Overlaps* encontrados.

Modelo	MelodyRNN		Performance RNN		Pianoroll RNN-NADE		Polyphony RNN	
Geração	Din.	Est.	Din.	Est.	Din.	Est.	Din.	Est.
Métrica	KL-Divergence							
IOI	0.103	0.025	0.065	0.032	0.217	0.06	0.018	0.196
NC	0.078	0.011	0.044	0.012	0.191	0.072	0.011	0.123
NC/Bar	0.024	0.02	0.117	0.041	0.055	0.053	0.119	0.279
NLH	0.051	0.053	0.061	0.009	0.17	0.068	0.041	0.702
NLTM	0.05	0.161	0.058	0.013	0.171	0.026	0.3	0.171
PC	0.018	0.03	0.105	0.019	0.121	0.009	0.137	0.114
PC/Bar	0.008	0.619	0.072	0.007	0.192	0.028	0.022	0.073
PCH	0.298	0.314	0.034	0.047	0.041	0.063	0.031	0.148
PCH/Bar	0.65	0.584	0.031	0.029	0.092	0.297	0.128	0.014
PCTM	0.073	0.283	0.018	0.02	0.04	0.01	0.066	0.109
PR	0.035	0.091	0.041	0.255	0.026	0.015	0.135	0.05
PS	0.076	0.018	0.03	0.061	0.071	0.025	0.092	0.004
Métrica	Overlap							
IOI	0.318	0.611	0.718	0.755	0.581	0.797	0.826	0.572
NC	0.524	0.726	0.793	0.804	0.36	0.809	0.597	0.557
NC/Bar	0.563	0.596	0.823	0.814	0.433	0.819	0.627	0.623
NLH	0.128	0.817	0.678	0.782	0.434	0.001	0.671	0.039
NLTM	0.574	0.755	0.621	0.7	0.584	0.441	0.532	0.349
PC	0.827	0.663	0.844	0.824	0.603	0.011	0.388	0.128
PC/Bar	0.811	0.499	0.875	0.772	0.332	0.729	0.813	0.6
PCH	0.817	0.816	0.752	0.721	0.702	0.769	0.502	0.697
PCH/Bar	0.682	0.459	0.79	0.861	0.834	0.353	0.495	0.439
PCTM	0.601	0.76	0.817	0.784	0.34	0.782	0.59	0.636
PR	0.821	0.801	0.81	0.705	0.05	0.031	0.022	0.037
PS	0.837	0.877	0.76	0.781	0.688	0.794	0.812	0.787

Tabela 5.2: Proximidade entre Amostras Geradas e Dados de Treinamento

Na Figura, os eixos vertical e horizontal representam respectivamente o *Overlap* e a *KL-Divergence* de cada métrica para cada modelo sob cada forma de geração, em comparação à própria base de teste. Cada uma das 12 cores representa uma métrica, e a forma de cada ponto denota sua respectiva modalidade de geração: triângulos simbolizam GE e círculos representam GD, conectados por métrica usando uma linha sólida, para melhor visualização.

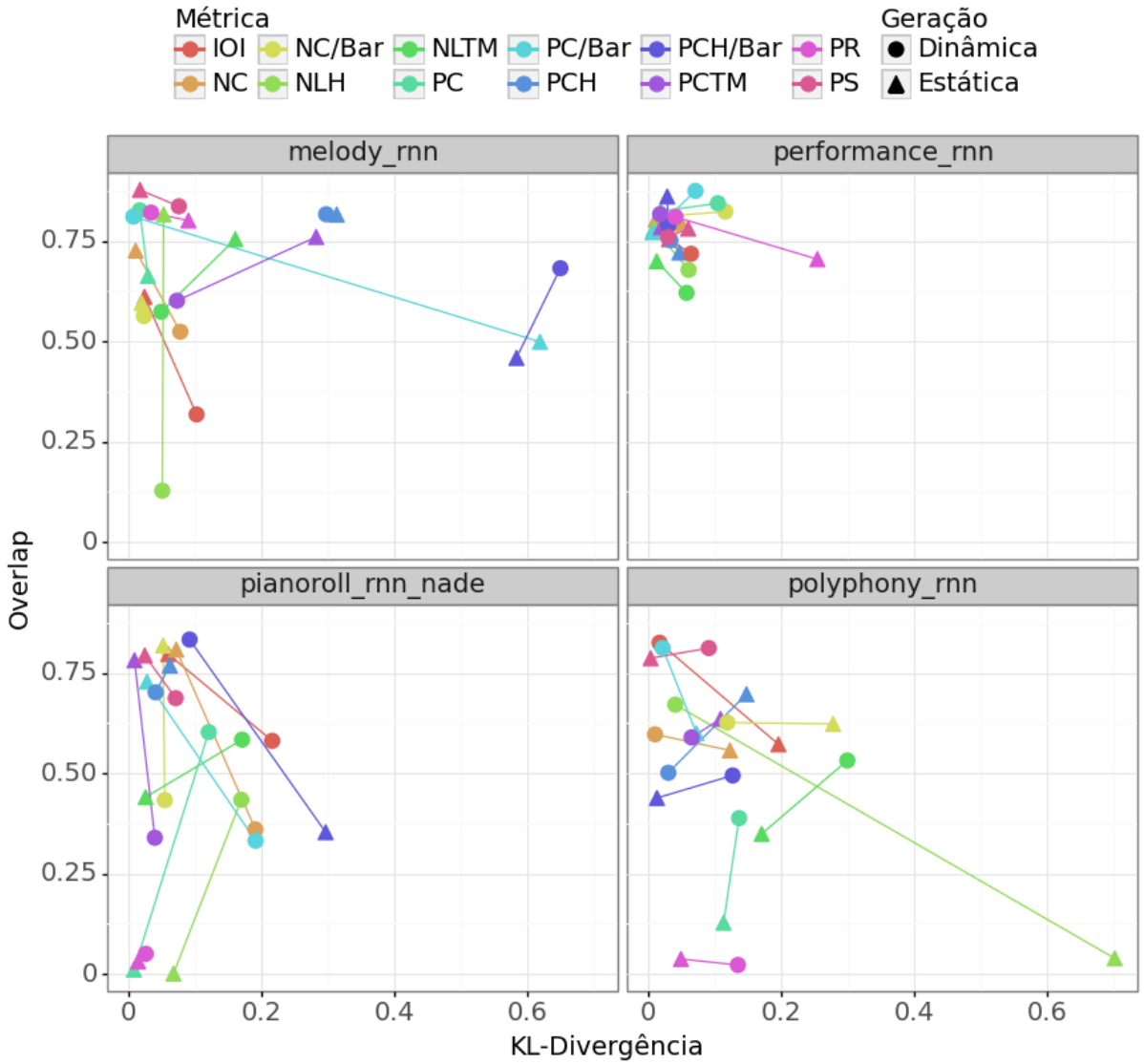


Figura 5.5: Comparação entre Geração Dinâmica e Estática

Confirmando o resultado exposto na figura 5.4, o modelo *PerformanceRNN* mostra uma concentração intensa dos pontos de dados na extremidade superior esquerda do diagrama. A proximidade entre pontos circulares e triangulares reflete claramente a semelhança entre GE e GD, e portanto o sucesso da adaptação proposta. Outros mode-

los apresentam dispersões maiores, sendo *PianorollRNN-NADE* o pior adaptado: nele, observa-se uma concentração dos pontos de GE na parte superior direita, enquanto os pontos de GD distribuem-se na faixa média do gráfico. Isso aponta para uma relação entre a adaptação desse modelo e a perda de verossimilhança das amostras. Os demais modelos apresentam distribuições igualmente dispersas e sobrepostas, não se observando uma relação de causa e efeito entre a adaptação e a qualidade das amostras.

Um resultado secundário dessa avaliação é a comparação entre os modelos originais para geração de amostras musicais verossimilhantes, cuja qualidade é indicada no gráfico pela distribuição dos pontos de GE. Os dados apontam que *PerformanceRNN* (e possivelmente também o *PianorollRNN-NADE*) apresenta maior robustez para essa tarefa que os demais, gerando amostras de maior *Overlap* e menor *KL-Divergence*.

5.2 Avaliação Subjetiva

As Figuras 5.6 e 5.7 mostram as respostas dos participantes da avaliação subjetiva. Em ambas as figuras, a cor de cada barra indica a base de origem das respectivas amostras: vermelho indica amostras de GE (B_{GE}), verde indica GD (B_{GD}) e em azul estão representadas as amostras de teste (B_T). Por fim, as barras são agrupadas horizontalmente por grau de instrução musical, indicando as quatro categorias previstas: Leigo, Aprendiz, Técnico (ou Informalmente Experiente) e Graduado ou Pós-Graduado em Música A Figura 5.6 ilustra o resultado do teste SME, indicando no eixo vertical a porcentagem de acerto dos participantes da pesquisa na tarefa de distinguir entre amostras reais e amostras sintéticas.

Os valores mais altos obtidos para os dados de GD em comparação às amostras de GE indicam ser mais fácil detectar amostras geradas através da adaptação proposta, independente do grau de formação do ouvinte. Observa-se nas amostras de GD a presença de artefatos sonoros resultantes do processo de síntese de informação musical, que facilitam sua identificação. Minimizar esses artefatos mostrou-se surpreendentemente desafiador, tanto por conta do conjunto complexo de requisitos imposto pelo *framework* adaptativo quanto pelo caráter imprevisível da saída dos modelos.

O outro aspecto da avaliação subjetiva é ilustrado através da Figura 5.7, que resume as respostas relativas à qualidade musical percebida pelos participantes em amostras de cada conjunto apresentado. O eixo vertical na figura representa a qualidade subjetiva numa escala de 0 a 3, e as barras pretas indicam a variância nas respostas obtidas.

Conforme esperado, dados os resultados expostos na figura anterior, as amostras de GD apresentam qualidade musical inferior às de GE, mesmo a diferença sendo aparentemente

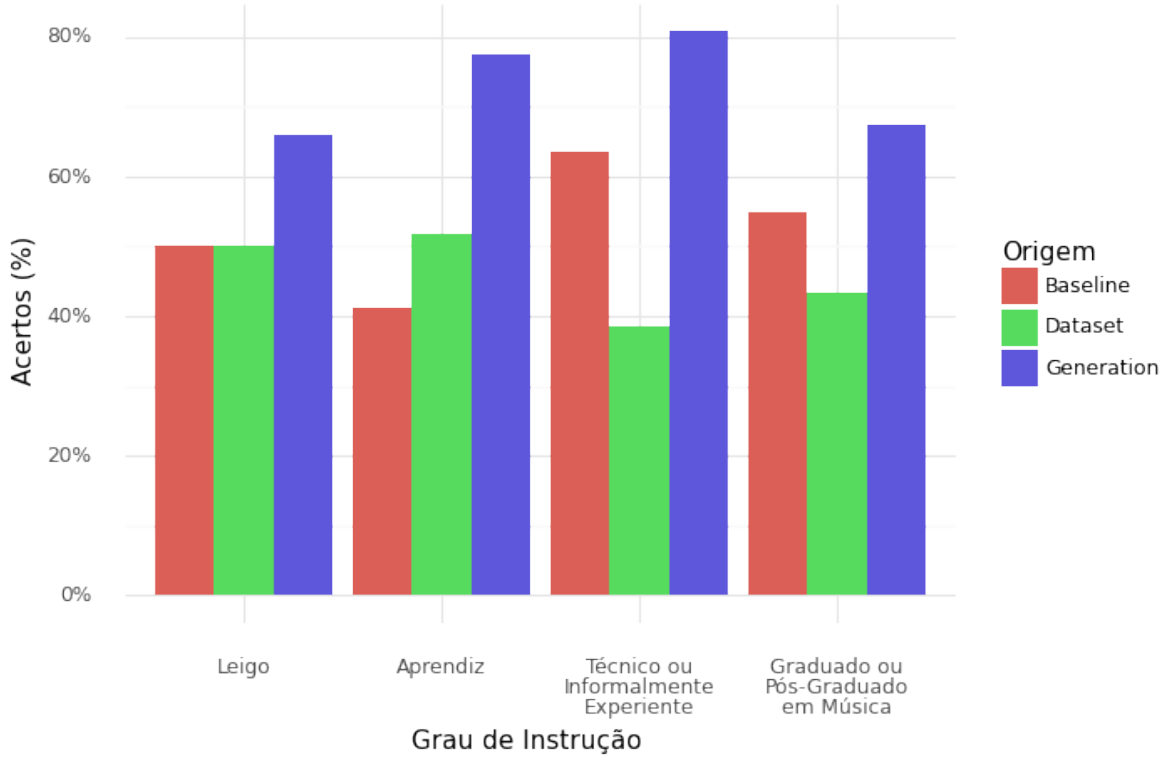


Figura 5.6: Teste Subject Matter Expert Turing

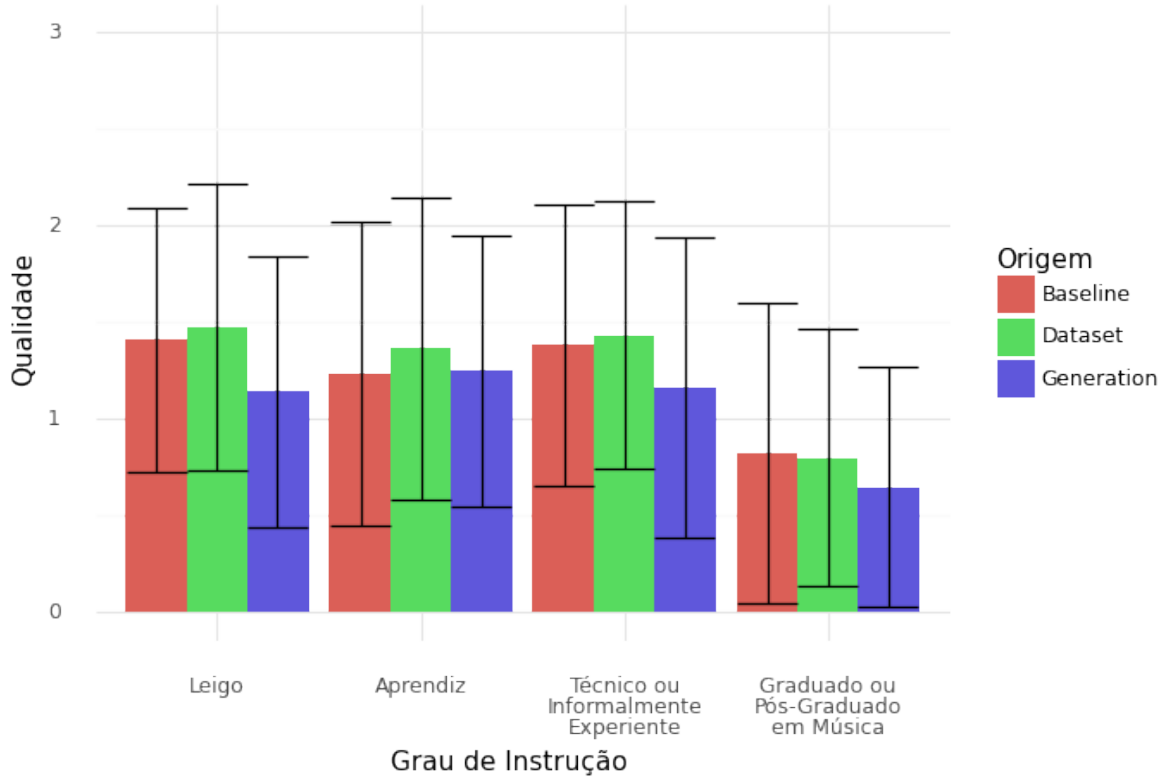


Figura 5.7: Qualidade Percebida das Amostras

pequena. Apesar disso, a alta dispersão dos dados compromete a precisão dos dados, introduzindo um valor de erro estatístico altíssimo.

5.3 Análise Estatística dos Resultados

Para concluir a avaliação da técnica proposta, avalia-se os dados obtidos das avaliações objetiva e subjetiva através de um Teste de Wilcoxon, para determinar se há diferença estatisticamente relevante entre a GE original e a GD adaptada.

Aplicando o teste sobre os resultados objetivos, obtém-se para os valores de *Overlap* z-valor $z = 524.5$ e num p-valor $p = 0.5148$. O caráter fortemente positivo do z-valor, em combinação ao p-valor alto, rejeitam a hipótese de que haja diferença no *Overlap* métrica em função da modalidade de geração. Semelhantemente, os valores obtidos do teste para a *KL-Divergence* ($z = 583$, $p = 0.959$) não indicam causalidade entre a aplicação da técnica e a qualidade do resultado musical.

Modelo	z-valor	p-valor
MelodyRNN	114.0	0.32
PerformanceRNN	121.0	0.42
PianorollRNN-NADE	133.0	0.64
PolyphonyRNN	125.0	0.49
Geral	2240.0	0.75

Tabela 5.3: Teste de Wilcoxon: Avaliação Objetiva

Quando aplicado sobre os dados da avaliação subjetiva, o teste rejeita a hipótese de que as gerações estática e dinâmica sejam equivalentes, com relação ambos ao Teste de Turing ($z = 486$, $p = 0.019$) e à análise de preferência ($z = 798$, $p = 0.009$). Ou seja, o experimento afirma que a técnica proposta de fato prejudica em algum nível o resultado sonoro gerado, com relação tanto à preferência geral quanto à sua verossimilhança. A pesquisa subjetiva contemplou 35 participantes, considerando-se *experts* o universo de entrevistados cujo grau de instrução seja pelo menos técnico ou informalmente experiente.

A divergência entre os resultados das etapas avaliativas é intrigante e mostra novamente o desafio de quantificar qualidade musical: mesmo constatada sob o atual estado-da-arte, a semelhança estrutural entre amostras não determina o comportamento da preferência humana por elas.

Capítulo 6

Conclusões

Nesse trabalho, propôs-se um *framework* para a adaptação de MSs de GE para GD. Quatro MSs foram selecionados e adaptados, servindo como base para a avaliação da técnica proposta. Construiu-se um sistema para implementação da técnica proposta e conduziu-se um experimento para avaliar seus resultados. Analisou-se por fim o impacto da avaliação sobre a qualidade musical dos modelos através de uma metodologia avaliativa robusta baseada no estado-da-arte em avaliação musical objetiva e subjetiva.

Os resultados mostraram promessa na adaptação apesar de ainda incorrer numa diferença perceptível na qualidade da saída dos modelos estudados. A implementação proposta ainda não é trivialmente generalizável enquanto solução para improviso dinâmico baseado em AM, mas funciona bem para casos particulares. Isso, além de estabelecer uma importante prova de conceito, é também um passo significativo em direção a uma solução geral.

Entendemos haver nesse trabalho duas contribuições principais. Cientificamente, em comparação com resultados anteriores, o experimento proposto oferece uma forma mais ampla e profunda de avaliar modelos para geração dinâmica de música, tarefa para a qual metodologias avaliativas são escassas e limitadas. De um ponto de vista tecnológico, a abordagem modular da solução e sua implementação na forma de sistema são contribuições inéditas que permitem maior controle sobre o processo generativo baseado em AM, além de melhor integração com ferramentas existentes.

6.1 Contribuição Científica

Esse trabalho propõe alguns conceitos úteis para o estudo do improviso auxiliado por AM. A distinção clara entre as modalidades de geração estática e dinâmica, além do próprio conceito de um SGDM e a definição dos requisitos funcionais dessa natureza de sistema são contribuições que podem basear futuras explorações em direção à composição

interativa humano-máquina em tempo real. Por fim, a avaliação do impacto de diferentes tamanhos de janelas no processo de geração segmentada de música, e a definição de sua viabilidade é um conceito novo que pode inspirar futuros estudos e sistemas.

Como discute o Capítulo 2, a avaliação de resultados musicais é desafiadora por si só, não havendo até então sequer uma metodologia voltada especificamente à geração dinâmica. Mostramos também ser possível uma adaptação de modelos pré-existentes para esse fim, validando também a aplicação da metodologia proposta por Yang, et al (2020) [21] na avaliação de modelos independentemente treinados sobre bases de dados distintas.

6.2 Contribuição Tecnológica

Além de provar um conceito importante, a implementação do sistema oferece um benefício ao campo de Criatividade Artificial, na medida em que não há outro sistema hoje que conforme simultaneamente com os quatro aspectos gerais propostos para um SGDM. A modularidade da arquitetura proposta traz diversos benefícios para a engenharia do sistema, sendo possível por exemplo implementar testes automatizados para diversos componentes, reutilizar lógicas de codificação e decodificação de informação sonora e estender o conjunto de controles do sistema, de forma simples e consistente através das suas diferentes interfaces.

6.3 Trabalho Futuro

Há um leque bastante amplo de possibilidades para a evolução desse trabalho, tanto pela natureza do sistema enquanto prova de conceito quanto pela quantidade de etapas e parâmetros distintos das quais o experimento se constitui, que podem ser explorados em estudos futuros com diferentes objetivos. Em primeiro lugar, é possível usar a técnica proposta para a adaptação de mais MSs e para a avaliação de seus resultados na tarefa de GD de música. Isso é benéfico em dois aspectos: um científico, permitindo-se a comparação desses modelos, e outro prático, habilitando seu uso como ferramentas de improviso interativo. O *framework* proposto pode ser também elaborado para permitir o treinamento de MSs sob GD (ao invés de apenas adaptar modelos de GE), o que pode promover um ganho de qualidade para a tarefa.

É possível também que ao minimizar o custo temporal de geração, a otimização de modelos de AM permita o uso de configurações adicionais, possivelmente gerando resultados melhores. O uso de diferentes modelos de forma intercambiável abre margem para uma possível investigação do uso de técnicas como *Meta-Learning* para esse fim, para

recomendar os modelos mais adequados a uma composição para cada dado contexto, com base em sua informação musical. De um ponto de vista arquitetural, o desacoplamento entre o sistema e os respectivos módulos pode ser melhorado, não parecendo haver um fator limitante externo, de forma que seria possível inclusive o emprego de linguagens de programação independentes, tanto na construção dos modelos quanto dos SGDMs. A abstração de *comandos* usada como base para a construção da interface do sistema permite que futuras interfaces sejam construídas de maneira mais simples e voltadas a cenários específicos. Dito isso, apesar de expor de forma acessível seu controle interno, a interface em si deixa a desejar, e pode ser elaborada para o futuro.

Um último caminho para a expansão desse trabalho seria o estudo de sua eficácia para diferentes áreas que exijam síntese dinâmica de alguma forma de mídia, como imagens, vídeo ou texto - áreas nas quais IA tem mostrado resultados promissores.

Referências

[1] Horsley, I. e Ernest T. Ferand: *Improvisation in nine centuries of western music : an anthology with a historical introduction*. Notes, página 119, 1962. 1

[2] Gorow, Ron: *Hearing and writing music: professional training for today's musician*. September Publishing, 2002. 1, 7

[3] Bailey, Derek: *Improvisation: Its Nature and Practice In Music*. Da Capo Press, 1993. 1, 3

[4] Hernandez-Olivan, Carlos e José Ramón Beltrán: *Music composition with deep learning: A review*. CoRR, abs/2108.12290, 2021. 1, 11, 12

[5] Chomsky, Noam: *Syntactic structures*. De Gruyter Mouton, 2009. 1

[6] Hakimi, Shunit Haviv, Nadav Bhonker e Ran El-Yaniv: *Bebopnet: Deep neural models for personalized jazz improvisations*. Em *Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR 2020, Montreal, Canada, October 11-16, 2020*, páginas 828–836, 2020. 1

[7] Hiller Jr, Lejaren A e Leonard M Isaacson: *Musical composition with a high-speed digital computer*. Journal of the Audio Engineering Society, 6(3):154–160, 1958. 1

[8] Huang, Cheng-Zhi Anna, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu e Douglas Eck: *Music transformer: Generating music with long-term structure*. Em *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 1, 2, 3, 9, 12, 13, 19, 30

[9] Simon, Ian e Sageev Oore: *Performance rnn: Generating music with expressive timing and dynamics*. `https://magenta.tensorflow.org/performance-rnn`, 2017. 1, 2, 9, 12, 13, 15, 19, 30

[10] Rodriguez, Jose David Fernández e Francisco J. Vico: *AI methods in algorithmic composition: A comprehensive survey*. CoRR, abs/1402.0585, 2014. 1, 3, 9, 17

[11] Müller, Meinard: *Fundamentals of Music Processing - Audio, Analysis, Algorithms, Applications*. Springer, 2015. 2, 6

[12] Research, Magenta: *Magenta studio*. `https://magenta.tensorflow.org/studio/ableton-live/`, 2019. 2, 3

[13] Goodfellow, Ian, Yoshua Bengio e Aaron Courville: *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`. 2, 9, 11, 12

[14] Sutskever, Ilya, Oriol Vinyals e Quoc V. Le: *Sequence to sequence learning with neural networks*. CoRR, abs/1409.3215, 2014. 2

[15] Latif, Siddique, Heriberto Cuayáhuitl, Farrukh Pervez, Fahad Shamshad, Hafiz Shehbaz Ali e Erik Cambria: *A survey on deep reinforcement learning for audio-based applications*. CoRR, abs/2101.00240, 2021. 2

[16] Sutskever, Ilya, Oriol Vinyals e Quoc V. Le: *Sequence to sequence learning with neural networks*. CoRR, abs/1409.3215, 2014. 2

[17] Jiang, Nan, Sheng Jin, Zhiyao Duan e Changshui Zhang: *Rl-duet: Online music accompaniment generation using deep reinforcement learning*. CoRR, abs/2002.03082, 2020. 3, 9, 13, 19, 30

[18] Biles, John: *Genjam: A genetic algorithm for generating jazz solos*. julho 1994. 3

[19] Ji, Shulei, Jing Luo e Xinyu Yang: *A comprehensive survey on deep music generation: Multi-level representations, algorithms, evaluations, and future directions*. CoRR, abs/2011.06801, 2020. 3, 4, 9, 20

[20] Ens, Jeffrey e Philippe Pasquier: *MMM : Exploring conditional multi-track music generation with the transformer*. CoRR, abs/2008.06048, 2020. 3

[21] Yang, Li Chia e Alexander Lerch: *On the evaluation of generative models in music*. Neural Computing and Applications, 32, 2018. 3, 4, 7, 17, 21, 31, 46

[22] Pearce, Marcus, David Meredith e Geraint Wiggins: *Motivations and methodologies for automation of the compositional process*. Musicae Scientiae, 6, 2002. 3

[23] Hadjeres, Gaëtan e Frank Nielsen: *Anticipation-rnn: enforcing unary constraints in sequence generation, with application to interactive music generation*. Neural Comput. Appl., 32(4):995–1005, 2020. 3

[24] Huang, Yu-Siang e Yi-Hsuan Yang: *Pop music transformer: Generating music with rhythm and harmony*. CoRR, abs/2002.00212, 2020. 3, 19, 30

[25] Dannenberg, Roger: *A brief survey of music representation issues, techniques, and systems*. Computer Music Journal, 17, 1998. 6

[26] Khulusi, R., J. Kusnick, C. Meinecke, C. Gillmann, J. Focht e S. Jänicke: *A survey on visualizations for musical data*. Computer Graphics Forum, 39(6):82–110, 2020. 6

[27] Oore, Sageev, Ian Simon, Sander Dieleman, Douglas Eck e Karen Simonyan: *This time with feeling: Learning expressive musical performance*. CoRR, abs/1808.03715, 2018. 7

[28] Hewitt: *Music Theory for Computer Musicians Bk/Cd*. Hal Leonard Corp, 2008. 7

[29] McKay, G.F. e F.L. Mckay: *Creative Orchestration: A Project Method for Classes in Orchestration and Instrumentation*. GFM Publishing Company, 2016, ISBN 9781537236131. https://books.google.com.br/books?id=DQgsvgAACAAJ. 8

[30] Huang, Cheng-Zhi Anna, Curtis Hawthorne, Adam Roberts, Monica Dinculescu, James Wexler, Leon Hong e Jacob Howcroft: *The bach doodle: Approachable music composition with machine learning at scale*. CoRR, abs/1907.06637, 2019. 8, 19, 20

[31] Benetatos, Christodoulos: *Bachduet: A deep learning system for human-machine counterpoint improvisation*. 2020. 8, 13, 20

[32] Rossi, Nathália Angela Fragoso: *Acaso e indeterminação no processo de composição: um diálogo com a obra de john cage*, 2015. 9

[33] Ames, Charles: *The markov process as a compositional model: A survey and tutorial*. Leonardo, 22:175–187, 1989. 9

[34] Franceschelli, Giorgio e Mirco Musolesi: *Creativity and machine learning: A survey*. CoRR, abs/2104.02726, 2021. 9

[35] Loy, D.: *Musicians make a standard: the midi phenomenon*. Computer Music Journal, 9:8–26, 1985. 9, 13

[36] Waite, Elliot: *Generating long-term structure in songs and stories*. https://magenta.tensorflow.org/2016/07/15/lookback-rnn-attention-rnn, 2016. 9, 19

[37] Roads, C.: *The Music Machine: Selected Readings from Computer Music Journal*. MIT Press, 1989. 9

[38] Wright, Matthew e Adrian Freed: *Open soundcontrol: A new protocol for communicating with sound synthesizers*. Em *Proceedings of the 1997 International Computer Music Conference, ICMC 1997, Thessaloniki, Greece, September 25-30, 1997*. Michigan Publishing, 1997. 10

[39] Allombert, Antoine, Myriam Desainte-Catherine e Gérard Assayag: *Iscore: A system for writing interaction*. Em *Proceedings of the 3rd International Conference on Digital Interactive Media in Entertainment and Arts*, página 360–367. Association for Computing Machinery, 2008. 10

[40] McCartney, J.: *Supercollider, a new real time synthesis language*. Em *ICMC*, 1996. 11

[41] Mclean, Alex: *Making programming languages to dance to: Live coding with tidal*. setembro 2014. 11

[42] Hutchings, Patrick e Jon McCormack: *Adaptive music composition for games*. CoRR, abs/1907.01154, 2019. 11, 19

[43] Murphy, Kevin P.: *Machine learning - a probabilistic perspective*. Adaptive computation and machine learning series. MIT Press, 2012. 11

[44] Russell, Stuart J. e Peter Norvig: *Artificial Intelligence - A Modern Approach, Third International Edition*. Pearson Education, 2010. 11

[45] Radford, Alec, Luke Metz e Soumith Chintala: *Unsupervised representation learning with deep convolutional generative adversarial networks*. Em Bengio, Yoshua e Yann LeCun (editores): *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. 11

[46] Liebman, Elad e Peter Stone: *Artificial musical intelligence: A survey*. CoRR, abs/2006.10553, 2020. 11

[47] French, Robert: *Catastrophic forgetting in connectionist networks*. Trends in cognitive sciences, 3:128–135, 1999. 12

[48] Jaques, Natasha, Shixiang Gu, Richard E. Turner e Douglas Eck: *Tuning recurrent neural networks with reinforcement learning*. CoRR, abs/1611.02796, 2016. 12

[49] Torfi, Amirsina, Rouzbeh A. Shirvani, Yaser Keneshloo, Nader Tavaf e Edward A. Fox: *Natural language processing advancements by deep learning: A survey*, 2020. 12, 17

[50] Hochreiter, Sepp e Jürgen Schmidhuber: *Long short-term memory*. Neural computation, 9:1735–80, 1997. 12

[51] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser e Illia Polosukhin: *Attention is all you need*. CoRR, abs/1706.03762, 2017. 12, 16

[52] Research, Magenta: *Polyphony rnn*. `https://github.com/magenta/magenta/tree/main/magenta/models/polyphony_rnn`, 2016. 13, 16, 30

[53] Research, Magenta: *Pianoroll rnn-nade*. `https://github.com/magenta/magenta/tree/main/magenta/models/pianoroll_rnn_nade`, 2017. 13, 16, 30

[54] Pachet, François: *The continuator: Musical interaction with style*. Journal of New Music Research, 32(3):333–341, 2003. 14

[55] Grachten, Maarten: *Jig: an approach to computational jazz improvisation*. 2001. 14

[56] Haki, Behzad e Sergi Jordà: *A bassline generation system based on sequence-to-sequence learning*. Em *NIME*, 2019. 14

[57] Elliot Waite, Douglas Eck, Adam Roberts e D. Abolafia: *Project magenta: Generating long-term structure in songs and stories*, 2016. 14, 30

[58] Yang, Li Chia, Szu Yu Chou e Yi Hsuan Yang: *Midinet: A convolutional generative adversarial network for symbolic-domain music generation*. 2017. 15

[59] Wu, Jian, Changran Hu, Yulong Wang, Xiaolin Hu e Jun Zhu: *A hierarchical recurrent neural network for symbolic melody generation*. 2018. 15

[60] Boulanger-Lewandowski, Nicolas, Yoshua Bengio e Pascal Vincent: *Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription*, 2012. 16, 17

[61] Lin, Chin Yew: *Rouge: A package for automatic evaluation of summaries*. Em *ACL 2004*, 2004. 17

[62] Papineni, Kishore, Salim Roukos, Todd Ward e Wei Jing Zhu: *Bleu: a method for automatic evaluation of machine translation*. outubro 2002. 17

[63] Geisser, S.: *Predictive inference: An introduction*. 2017. 18

[64] Castro, Pablo Samuel: *Performing structured improvisations with pre-trained deep learning models*. CoRR, abs/1904.13285, 2019. 19, 20

[65] Huang, Cheng-Zhi Anna, Tim Cooijmans, Adam Roberts, Aaron C. Courville e Douglas Eck: *Counterpoint by convolution*. CoRR, abs/1903.07227, 2019. 19

[66] Arpaci-Dusseau, Remzi H.: *Operating systems: Three easy pieces*. login Usenix Mag., 42(1), 2017. 19, 22

[67] Parikh, Tejus: *Iris: artificially intelligent real-time improvisation system*. Tese de Mestrado, Emory University, 2003. 19

[68] Louie, Ryan, Andy Coenen, Cheng Zhi Huang, Michael Terry e Carrie J. Cai: *Novice-ai music co-creation via ai-steering tools for deep generative models*. Em *CHI '20: CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, April 25-30, 2020*, páginas 1–13. ACM, 2020. 20

[69] Bazin, Théis e Gaëtan Hadjeres: *NONOTO: A model-agnostic web interface for interactive music composition by inpainting*. CoRR, abs/1907.10380, 2019. 20

[70] Boettiger, Carl: *An introduction to docker for reproducible research, with examples from the R environment*. CoRR, abs/1410.0846, 2014. 25

[71] Raffel, Colin: *Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching*. 2016. 30

[72] Demšar, Janez: *Statistical comparisons of classifiers over multiple data sets*. Journal of Machine Learning Research, 7:1–30, 2006. 31

[73] Feigenbaum, Edward A.: *Some challenges and grand challenges for computational intelligence*. 2003. 32