Transcrição Automatizada de Bateria

Thiago Martin Poppe [1], **Flavio Vinicius Diniz de Figueiredo** [1]

[1]Instituto de Ciências Exatas – Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte - MG - Brasil

tmartinpoppe@gmail.com, flaviovdf@gmail.com

Abstract. *The existing literature for automatic drum transcription is mainly focused on estimating the playing times of each piece of the drum kit. Moreover, no studies were found that deal with the output's presentation to an end user, typically a musician. In this sense, we propose the presentation of these results in the form of sheet music. To do this, we use the estimated playing times from a recurrent neural network, along with meta information relevant to the song, such as the BPM and time signature, to automatically generate the sheet music of an input signal. The product of this study proved to be promising, enabling new research fronts in this area.*

Resumo. *A literatura existente para a transcrição automática de bateria é concentrada principalmente na estimação dos tempos de toque de cada peça do kit da bateria. Sendo assim, nenhum estudo foi encontrado que trata da apresentação dos resultados para um usuário final, tipicamente um músico. Neste sentido, propomos a apresentação destes resultados na forma de partituras. Para tal, utilizamos os tempos de toque estimados a partir de uma rede neural recorrente, juntamente com meta informações pertinentes à música, como o BPM e a fórmula de compasso, para gerar automaticamente a partitura do sinal de entrada. O produto deste estudo provou-se promissor, viabilizando novas frentes de pesquisa voltadas para essa área.*

1. Introdução

A transcrição de músicas utilizando a partitura pode ser aplicada a diversos instrumentos musicais, e é uma tarefa árdua que exige muito conhecimento de teoria musical. O foco da pesquisa desenvolvida durante os Projetos Orientados em Computação 1 e 2 (POC 1 e POC 2), restringiu o escopo da transcrição de partituras para instrumentos percussivos, especificamente para a bateria. O principal objetivo foi de cobrir o passo, ainda não feito na literatura, entre a tarefa de *Automatic Drum Transcription* (ADT) e a escrita de uma partitura. Isso foi feito a partir da criação de uma rede neural recorrente, usando como inspiração os artigos [Vogl et al. 2016] e [Southall et al. 2016], tendo como entrada o espectrograma da gravação de uma bateria e como saída uma representação da estimação dos tempos das batidas das peças da bateria através de uma função de ativação, indicando em um tempo t a probabilidade de termos um toque em alguma das 3 peças da bateria (chimbal, caixa e bumbo), ou combinações delas. Posteriormente, foi feito uma conversão dessa saída utilizando uma extensão do algoritmo proposto no POC 1 para escrevermos a partitura final da gravação transcrita, como mostrado na Figura 1.

Por conta de instrumentos percussivos produzirem sons de altura indefinida, ou seja, que não são capazes de serem descritos a partir de uma frequência de maior destaque, é necessária uma análise de intervalos de frequências, que podem se sobrepor, para

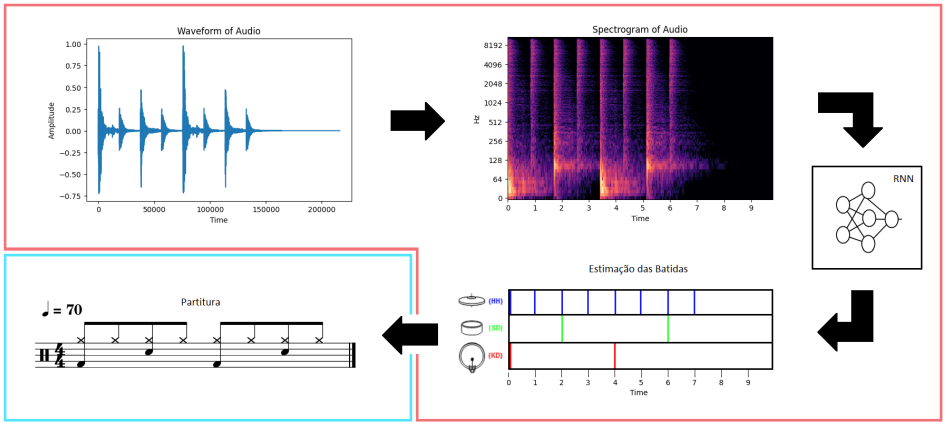

Figura 1. Etapas do POC 1 (em azul) e POC 2 (em vermelho)

estimar os instrumentos que estão sendo tocados. A Figura 2 ilustra o problema citado anteriormente, bem como a dificuldade em estimar com precisão a duração das notas na bateria, uma vez que elas não possuem sustentação sonora, fazendo com que notas com diferentes durações soem, na maior parte das vezes, da mesma forma. No espectrograma da esquerda temos dois toques na caixa da bateria e na direita duas notas tocada no piano, sendo que o primeiro toque/nota contém a duração de 1 tempo (semínima) e o segundo a duração de 1/4 do tempo (semicolcheia). Podemos perceber que, enquanto na bateria temos uma alta dispersão das frequências ao longo do eixo y; no piano, conseguimos perceber em maior destaque uma frequência específica e também outras frequências acima dela, denominadas de série harmônica.

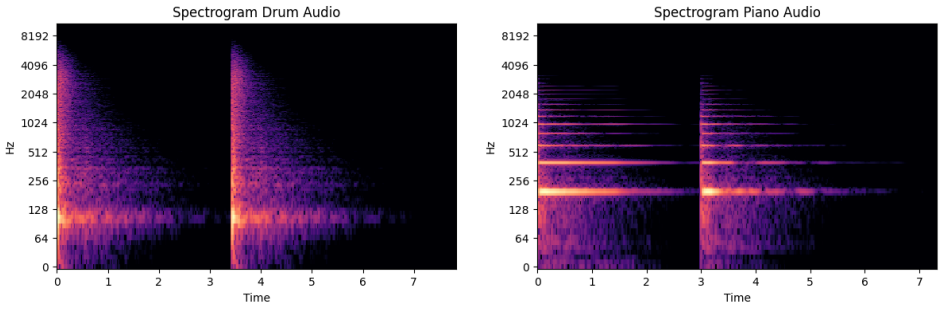

Figura 2. Espectrograma de sons na bateria e no piano, respectivamente

Dadas as dificuldades citadas anteriormente, a proposta de um método para a transcrição automatizada de partituras para a bateria é bastante benéfica, principalmente para auxiliar nos estudos de músicos iniciantes que ainda não possuem domínio da teoria musical necessária para a escrita de partituras. Além disso, a representação da transcrição via partitura trará mais valor para os resultados obtidos nos estudos referentes à tarefa de ADT, uma vez que tais transcrições poderão ser estudadas, tocadas e compartilhadas entre músicos por utilizarem um vocabulário que estão habituados.

O restante do trabalho será estruturado da seguinte forma: a seção 2 oferece uma análise de trabalhos relacionados; a seção 3 apresenta a metodologia utilizada, incluindo detalhes das bases de dados utilizadas e a solução proposta. Por fim, experimentos e resultados podem ser encontrados na seção 4, seguidos por uma conclusão na seção 5.

2. Trabalhos Relacionados

Nesta seção, serão apresentados trabalhos relacionados com os estudos feitos ao longo do desenvolvimento do POC 2, mais especificamente com as soluções propostas para o problema de transcrição automatizada de bateria.

Um dos primeiros estudos realizados para resolver a tarefa de ADT [Schloss 1985] foi baseado na utilização da representação do áudio da bateria como uma onda, juntamente com a detecção de eventos *onset* [Müller 2015a]. Ao combinar essas duas informações, o método computacional resultante foi capaz de detectar batidas simples de gravações possuindo apenas o áudio de uma bateria como entrada. Atualmente, os estudos referentes à tarefa de ADT focam em expandir o número de instrumentos presentes na gravação de entrada, tratando assim de polifonias (textura musical com duas ou mais linhas independentes) ao invés de monofonias, propondo métodos baseados em dois paradigmas: *Non-Negative Matrix Factorization* (NMF) e *Deep Neural Networks*.

A utilização do paradigma baseado em NMF [Müller 2015b] consiste em tratar a decomposição espectral do sinal de entrada, representado como um espectrograma através da Transformada de Fourier de Tempo Curto (STFT) [Müller 2015c]. A decomposição se dá em duas funções espectrais base: os *templates* e as ativações, como observado na Figura 3. Intuitivamente, os *templates* podem ser interpretados como o espectro médio das frequências de cada peça da bateria; enquanto que as ativações descrevem onde e quão intenso eles serão. A família de algoritmos NMF origina diversos outros métodos para a transcrição automatizada de bateria, como o *Partially-Fixed NMF* [Wu and Lerch 2015] e o *Probabilistic Latent Component Analysis* (PLCA) [Benetos et al. 2014].

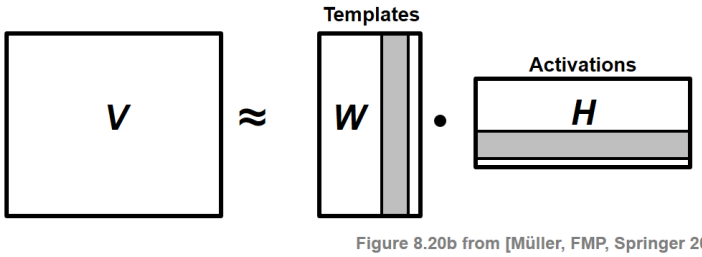

Figura 3. Decomposição NMF da matriz V (extraído de [Müller 2015b])

Em contraste com a abordagem feita por NMF [Müller 2015b], o paradigma que utiliza Redes Neurais Profundas processa o espectrograma do sinal como uma série temporal através de uma janela deslizante de tamanho fixo, onde cada fragmento extraído serve como entrada para um modelo neural. Para cada fragmento, a saída da rede será a probabilidade de termos um toque em uma determinada peça, resultando assim em uma função de ativação similar à matriz de ativações do método NMF [Müller 2015b], como podemos observar na Figura 4.

O modelo estado da arte para esse paradigma são as Redes Neurais Recorrentes, ou RNNs, sendo os artigos [Vogl et al. 2016] e [Southall et al. 2016] os pioneiros a utilizarem esse modelo de aprendizagem profunda no contexto de transcrição automatizada de bateria. A utilização de uma RNN para a tarefa de ADT demonstrou ser bastante promissora, principalmente por conta das RNNs levarem em consideração características temporais durante seu treinamento, o que é um ponto interessante já que músicas pos-

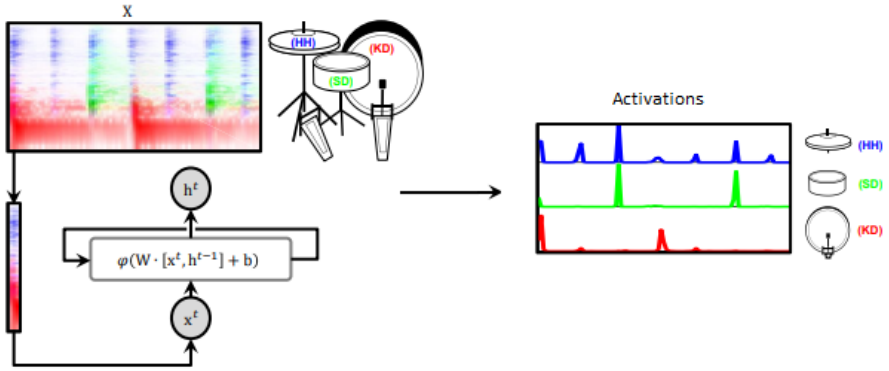

Figura 4. Transcrição utilizando redes neurais recorrentes (adaptado de [Wu et al. 2018])

suem tendências que se repetem ao longo do tempo, como podemos observar em motivos e refrãos [Müller 2015d].

No artigo [Vogl et al. 2016], os autores propõem 4 arquiteturas de redes neurais recorrentes para realizar a estimação dos tempos de toque de cada peça da bateria, sendo elas:

1. *Basic RNN*: Uma arquitetura formada por uma RNN com 1 camada oculta contendo 200 neurônios;

2. *Backward RNN*: A mesma arquitetura do modelo anterior, porém dessa vez o espectrograma é processado no sentido inverso. A principal hipótese dos pesquisadores com essa arquitetura era de verificar se haveria algum ganho no resultado se a rede utilizasse a sustentação do som das peças ao seu favor;

3. *Bidirectional RNN*: Uma arquitetura formada por 1 camada oculta com 100 neurônios, onde a predição de um tempo t_i é feita utilizando tanto a entrada no tempo t_i quanto os resultados das camadas ocultas nos tempos t_{i-1} e t_{i+1}. Em outras palavras, essa arquitetura também utiliza o futuro para fazer as predições. Há uma série de restrições com essa arquitetura, como o fato de precisarmos carregar de antemão todo o espectrograma do sinal, inviabilizando aplicações *online*;

4. *Time-Shifted RNN*: Similar à primeira arquitetura, porém as anotações do tempo são deslocadas por um valor ϵ pequeno. Isso faz com que o modelo utilize, em partes, informações do futuro para predizer o tempo atual sem precisar carregar todo o espectrograma do sinal, permitindo assim simularmos o poder de redes recorrentes bidirecionais em aplicações *online*.

Já no artigo [Southall et al. 2016], os autores propõem apenas duas arquiteturas, similares às arquiteturas *Basic* e *Bidirectional RNN* vistas acima, variando apenas os hiperparâmetros da rede, como número de neurônios, número de camadas ocultas e funções de ativação entre as camadas ocultas. Porém, o principal diferencial do artigo mencionado para o anterior, foi a proposta de treinar uma RNN separada para cada peça da bateria ao invés de uma RNN para a bateria como um todo, facilitando assim a adição e remoção de peças durante o processo de transcrição, como podemos observar na Figura 5.

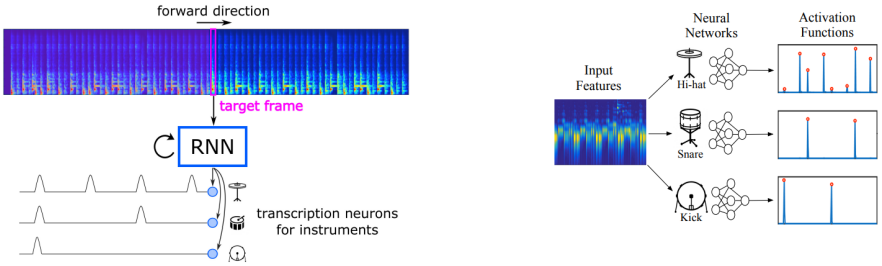

Figura 5. Arquiteturas propostas por [Vogl et al. 2016] (esquerda) e [Southall et al. 2016] (direita), adaptadas dos mesmos artigos

3. Metodologia

Na seção 3.1, iremos explicar um pouco mais em detalhes a base de dados utilizada para o projeto. Já nas seções 3.2 e 3.3, iremos abordar o pré-processamento feito sobre o sinal de entrada; bem como a arquitetura proposta para este trabalho. Além disso, iremos apresentar na seção 3.4 um algoritmo para a transformação da saída da rede em partitura.

3.1. Bases de dados utilizada

Inspirado nos estudos de [Vogl et al. 2016] e [Southall et al. 2016], utilizamos a base de dados IDMT-SMT-Drums para realizar o treinamento e avaliação da rede neural profunda desenvolvida ao longo do POC 2. A base de dados apresenta áudios de bateria compostos apenas por sons de 3 peças da bateria: chimbal, caixa e bumbo; bem como anotações dos tempos de toque referentes à cada peça em formato XML.

Concomitantemente, foi utilizada também a base de dados MDBDrums [Southall et al. 2017], subconjunto da base MedleyDB [Bittner et al. 2014], para avaliar a capacidade de generalização da arquitetura proposta, uma vez que essa base de dados apresenta áudios com uma variabilidade maior de estilo musicais, além de áudios de bateria com mais peças e também gravações polifônicas.

3.2. Pré-processamento do sinal de entrada

Seguindo os artigos mais recentes da área [Wu et al. 2018], utilizamos como *feature* para os modelos o espectrograma do sinal de entrada, calculado através da Transformada de Fourier de Tempo Curto (STFT) [Müller 2015c]. Porém, a depender das frequências associadas com cada peça da bateria, podemos acabar com espectrogramas que não transmitem muitas informações sobre o sinal de entrada, como observado na primeira linha da Figura 6. Para suprir essa deficiência de informação, será computado o logaritmo dos valores do espectrograma, tornando assim as frequências mais evidentes na imagem final.

Além das transformações mencionadas anteriormente, será aplicado uma escala logarítmica sobre o eixo vertical do espectrograma (eixo das frequências), resultando assim no *pipeline* final observado na Figura 6. Essa transformação faz com que as frequências fiquem igualmente espaçadas, realçando assim as frequências mais baixas. Todo o *pipeline* de pré-processamento expresso até então pode ser computado durante a Transformada de Fourier através de algumas transformações como a *Constant-Q Transform*, que foi utilizada no trabalho através de uma implementação disponibilizada pela biblioteca LibROSA [McFee et al. 2015].

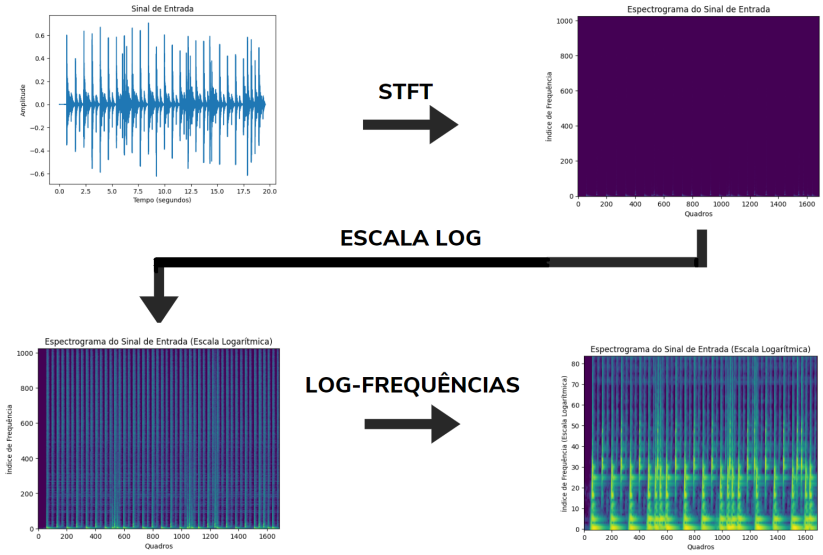

Figura 6. *Pipeline* de pré-processamento completo

Algo interessante que podemos notar a partir do pré-processamento proposto é o mapeamento quase direto das frequências para as funções de ativação (Figura 7). Além disso, podemos perceber a presença de sobreposição das frequências, principalmente da caixa com as demais peças, tornando a tarefa de transcrição mais difícil.

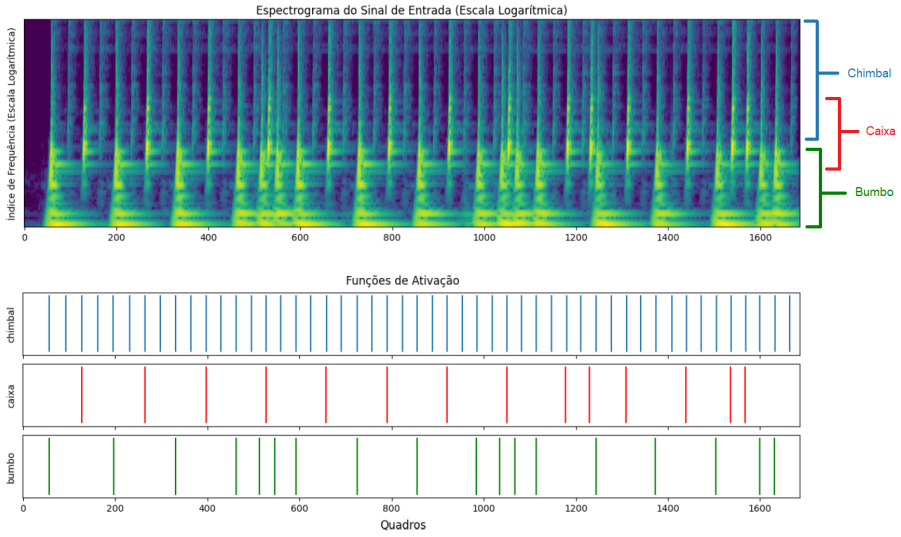

Figura 7. Mapeamento do espectrograma para as funções de ativação do chimbal (azul), caixa (vermelho) e bumbo (verde)

3.3. Arquitetura proposta

Para esse projeto, foram implementadas 3 redes neurais recorrentes inspiradas em [Vogl et al. 2016], com o objetivo de replicar os principais resultados obtidos no artigo. As arquiteturas serão explicadas em maiores detalhes a seguir:

1. 1L-RNN: uma RNN simples com apenas 1 camada oculta contendo 200 neurônios, similar à rede *Basic RNN* do artigo.

2. 2L-RNN: uma RNN com 2 camadas ocultas contendo 50 neurônios cada. Essa rede foi proposta pelo mesmo autor em outro artigo [Vogl et al. 2017], focado na transcrição de bateria em músicas polifônicas;

3. bRNN: uma RNN bidirecional com apenas 1 camada oculta contendo 100 neurônios, similar a rede *Bidirectional RNN* do artigo.

4. Observação: A saída de todas as redes descritas acima é formada por uma camada *fully-connected* com 3 neurônios de saída. O primeiro neurônio está associado com toques no chimbal, o segundo com toques na caixa e o terceiro com toques no bumbo.

O problema de transcrição foi considerado como um problema de classificação binária para cada peça da bateria. Sendo assim, utilizamos a *Weighted Binary Cross-Entropy* como função de perda. A versão balanceada foi utilizada já que a frequência da classe positiva (ocorrência do toque) é bem baixa se comparada à classe negativa. Além disso, uma *Sigmoid* foi utilizada na saída das redes para converter os valores preditos na probabilidade de ter ocorrido um toque em dado momento t.

3.3.1. Treinamento das redes

Redes neurais recorrentes são famosas por terem o problema de *vanishing gradient*, ou seja, uma diminuição do valor do gradiente ao longo da correção dos pesos e *bias* da rede, uma vez que o algoritmo de *backpropagation*, mais especificamente a versão *backpropagation through-time*, deve percorrer uma rede muito profunda para realizar as atualizações. Sendo assim, optamos por utilizar o algoritmo *truncated backpropagation through-time* [Williams and Peng 1998] para mitigar esse efeito de "desaparecimento do gradiente". Para isso, a rede foi treinada utilizando subsequências de 100 quadros do espectrograma de entrada, garantindo também um treinamento mais rápido, uma vez que precisamos processar menos dados para atualizar os parâmetros da rede em uma iteração.

O otimizador utilizado para o treinamento da rede foi o RMSProp com uma taxa de aprendizagem de 0.001, sendo dividida pela metade a cada 7 épocas de treinamento. Além disso, 80% da base de dados IDMT-SMT-Drums (cerca de 76 gravações) foi utilizada como treinamento, dos quais 10% dos dados foram utilizados como conjunto de validação (cerca de 8 gravações). O treinamento do modelo é prematuramente terminado ao atingirmos 10 épocas sem melhorias significativas na função de perda no conjunto de validação. Para garantirmos que o treino começará de forma certa, inicializamos os pesos do modelo seguindo uma distribuição uniforme $\mathcal{U}(-0.01, 0.01)$ e o *bias* em 0.

Tanto os hiperparâmetros da rede, quanto a forma de treinamento e avaliação dos modelos, foram adaptados diretamente do artigo [Vogl et al. 2016].

3.3.2. Pós-processamento da saída

Dado que a saída da rede indica a probabilidade de ter ocorrido um toque em dado momento t, podem existir alguns pontos nas funções de ativação que não possuem uma probabilidade alta, indicando assim possíveis falso positivos, como podemos observar na

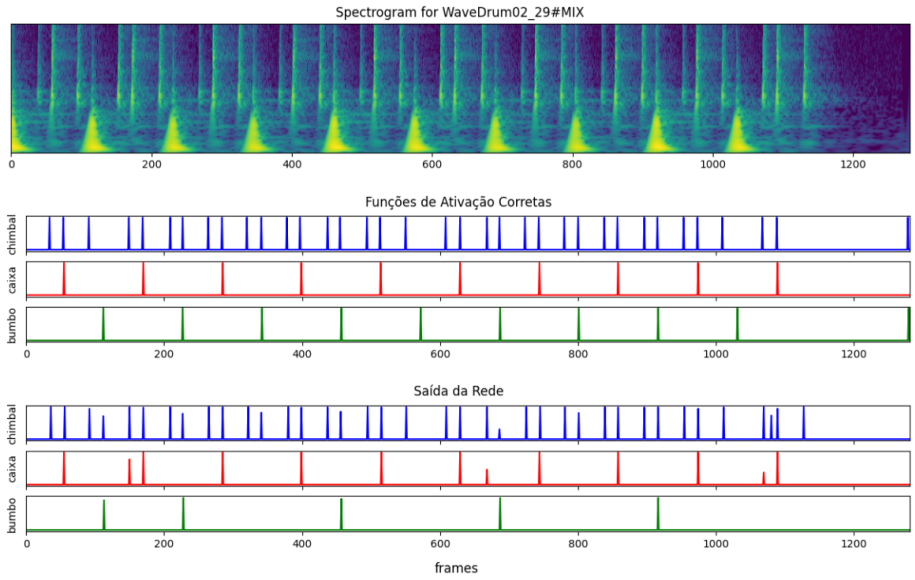

Figura 8. Espectrograma de um áudio de bateria (primeira imagem), funções de ativação corretas (segunda imagem) e a predição da RNN Bidirecional (terceira imagem)

Figura 8. Para filtrar esses pontos da predição final, utilizaremos um algoritmo de *peak-picking* [Böck et al. 2012] para selecionar picos nas funções de ativação que respeitam as seguintes restrições, onde n representa o n-ésimo pico, $F(n)$ a função de ativação e δ um valor real no intervalo $[0, 1]$:

(i) $F(n) = \max(F(n - pre\_max : n + post\_max))$

(ii) $F(n) \geq mean(F(n - pre\_avg : n + post\_avg)) + \delta$

(iii) $n - n_{lastpeak} > T_{min}$

Em resumo, as restrições indicam que um pico será considerado como parte da predição final se: (i) ele for o ponto máximo dentro de uma janela deslizante de tamanho determinado por pre_max e $post\_max$; (ii) maior ou igual do que a média de uma outra janela deslizante, de tamanho determinado por pre_avg e $post\_avg$, mais um limiar δ; e (iii) ocorrido após um tempo mínimo T_{min} do último pico. Os valores para os parâmetros do algoritmo foram escolhidos de maneira empírica, sendo $pre\_max = 50ms$, $post\_max = 50ms$, $pre\_avg = 50ms$, $post\_avg = 50ms$, $\delta = 0.25$ e $T_{min} = 50ms$.

3.4. Transformação das funções de ativação para partitura

O objetivo principal a ser alcançado com os trabalhos desenvolvidos ao longo do POC 1 e 2 é de preencher o espaço existente na literatura entre a representação da transcrição através de funções de ativação e a partitura. Para isso, foi feita uma união os trabalhos feitos no POC 2 com os estudos do POC 1, um algoritmo simples para a escrita de uma partitura baseada nos tempos de toque de uma peça da bateria.

Dado que o trabalho do POC 1 restringia o escopo apenas para transcrição da caixa da bateria, propusemos uma extensão do método para transcrição da bateria de forma completa. A extensão consiste nos seguintes passos:

1. Utilizar o algoritmo proposto no POC 1 para escrever a partitura de cada peça da bateria de forma independente. Isso é possível mesmo que o algoritmo tenha sido pensado apenas no contexto da caixa da bateria, uma vez que a saída da rede para cada peça é uma representação dos seus tempos de toque;

2. União das partituras escritas no passo anterior, se atentando ao fato de que se duas peças forem tocadas ao mesmo tempo, a figura de tempo utilizada para escrever as peças na partitura final será igual à menor figura de tempo das peças tocadas. Isso é apenas uma convenção usada para a escrita de partituras para bateria, visando aumentar principalmente a sua legibilidade.

Podemos visualizar uma iteração da etapa de união do algoritmo estendido na Figura 9. No tempo destacado, temos a presença de um toque no chimbal em colcheia e um toque na caixa em semínima. Como o toque no chimbal possui uma figura de tempo menor do que o toque na caixa, ambos toques serão escritos na partitura final em colcheia. Esse exemplo realça o fato das notas da bateria não possuírem uma sustentação bem definida, gerando uma certa ambiguidade sonora, já que o mesmo som pode ser escrito utilizando figuras de tempo diferentes.

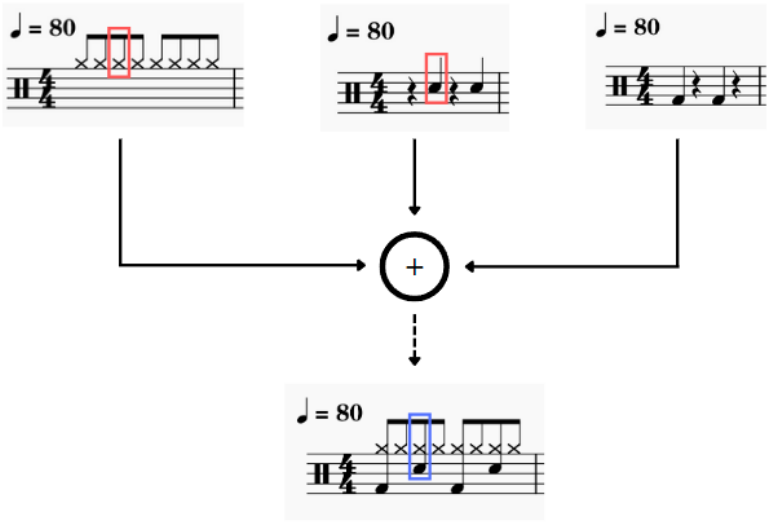

Figura 9. Exemplo de uma iteração da etapa de união do algoritmo estendido

4. Análises e Experimentos

As arquiteturas neurais descritas na seção 3.3 foram implementadas usando a linguagem de programação Python 3.7.13. Também foram utilizadas as bibliotecas LibROSA [McFee et al. 2015] para processamento digital de áudio e PyTorch [Paszke et al. 2019] para implementação, treinamento e avaliação das redes neurais. Os experimentos foram executados tanto localmente, em uma máquina com sistema operacional Windows 10, através do WSL (*Windows Subsystem for Linux*), com um processador Intel Core i7 7ª geração, 16GB de memória RAM e 1TB de HD; quanto através do Google Colab.

4.1. Métricas e método de avaliação

Para avaliação dos modelos apresentados na seção 3.3 utilizamos a métrica F1-Score, que representa a média harmônica da precisão e revocação, sendo muito utilizada em problemas de classificação por providenciar uma noção geral da relação dessas duas métricas. Tanto a precisão quanto a revocação foram calculadas utilizando as fórmulas padrões da literatura, porém a contagem dos valores necessários para computarmos as métricas foi feita através da análise de um intervalo temporal de $50ms$ em torno de cada predição feita. Em outras palavras, a verificação de falso positivos, falso negativos, entre outros valores necessários, foi feita verificando se tanto a predição quanto a ativação anotada estavam situadas no mesmo intervalo temporal. Essa modificação foi necessária devido à erros em torno de $50ms$ não afetarem o resultado final da transcrição por estarem situados em um intervalo de tempo muito curto, quase imperceptível.

A avaliação foi feita tanto sobre o conjunto de teste dos dados da base IDMT-SMT-Drums, ou seja 20% dos dados (cerca de 19 gravações); quanto sobre a base de dados MDBDrums [Southall et al. 2017] (cerca de 23 gravações). O cálculo da métrica F1-Score foi feita sobre cada peça da bateria de forma individual, computando posteriormente a média dos resultados para avaliar a performance dos modelos sobre a bateria como um todo.

4.2. Resultados principais

Através das Tabelas 1 e 2 podemos perceber que o modelo que resultou na melhor métrica foi a RNN Bidirecional (bRNN), assim como apontado pelos artigos [Vogl et al. 2016] e [Southall et al. 2016]. Os demais modelos tiveram uma performance razoável, dando maior destaque para a RNN com 2 camadas ocultas (2L-RNN) ter superado a RNN Bidirecional durante a estimação dos tempos do bumbo, indicando que uma estratégia interessante seria de treinar modelos diversos para a predição de cada peça da bateria individualmente, como proposto pelos estudos do artigo [Southall et al. 2016].

Modelo	Chimbal F1-Score [%]	Caixa F1-Score [%]	Bumbo F1-Score [%]	F1-Score Final [%]
2L-RNN	63.12	86.63	95.16	81.64
1L-RNN	70.31	87.29	94.99	84.20
bRNN	92.48	94.50	91.87	92.95

Tabela 1. Avaliação no conjunto de dados IDMT-SMT-Drums

O mesmo comportamento geral do resultado observado na Tabela 1 pode ser observado na Tabela 2, exceto pelo melhor modelo para a predição do bumbo ser a RNN com 1 camada oculta (1L-RNN). Porém, a performance de todos os modelos teve uma queda considerável de quase 65%, um comportamento similar observado também no artigo [Vogl et al. 2016] a partir da avaliação dos modelos propostos utilizando a base de dados ENST-Drums. Uma provável explicação desse resultado está relacionado com os tipos de áudios presentes nas duas bases de dados utilizadas. Enquanto que a base IDMT-SMT-Drums foca em áudios de bateria compostos por 3 peças (chimbal, caixa e bumbo) e poucos estilos musicais (Real, Wave e Techno); a base MDBDrums [Southall et al. 2017] busca uma maior diversificação das gravações, possuindo áudios de bateria compostos por kits mais completos, como a presença de tons, rides e crashes, o que pode ter comprometido a predição da rede devido à alta sobreposição de frequências presente nas gravações.

Além disso, a base MDBDrums [Southall et al. 2017] possui uma gama maior de estilos musicais, variando desde o rock e metal até o reggae e jazz, introduzindo assim maior variabilidade nos sons emitidos pela bateria nas gravações.

Modelo	Chimbal F1-Score [%]	Caixa F1-Score [%]	Bumbo F1-Score [%]	F1-Score Final [%]
2L-RNN	21.62	54.95	77.37	51.31
1L-RNN	26.15	56.34	82.93	55.14
bRNN	34.97	65.37	82.91	61.08

Tabela 2. Avaliação no conjunto de dados MDBDrums

A hipótese do resultado anterior pode ser validada ao observarmos a Tabela 3, contendo os resultados das predições para o gênero musical *rockabilly* presente na base MDBDrums [Southall et al. 2017]. Mesmo a gravação contendo a presença de tons e crashes, a rede conseguiu obter uma performance similar à observada na base de dados IDMT-SMT-Drums provavelmente devido a bateria da gravação possuir uma afinação e tocabilidade (*playstyle*) similar ao que temos na base usada para o treinamento.

Modelo	Chimbal F1-Score [%]	Caixa F1-Score [%]	Bumbo F1-Score [%]
2L-RNN	52.79	97.95	100.00
1L-RNN	52.69	97.96	98.04
bRNN	94.44	100.00	93.88

Tabela 3. Avaliação do gênero musical *rockabilly* (base MDBDrums)

5. Conclusão

Nesse trabalho, utilizamos técnicas baseadas em aprendizagem profunda para a transcrição de partituras a partir de áudios de bateria. Os resultados obtidos demonstraram que as técnicas aplicadas foram satisfatórias, sobretudo devido à qualidade das estimações dos tempos de toque nas bases IDMT-SMT-Drums e MDBDrums [Southall et al. 2017].

Conclui-se que o objetivo do trabalho foi alcançado, abrindo portas para trabalhos futuros, como a geração de partituras para músicas contendo ainda mais peças de bateria. Além disso, podemos estender o trabalho feito no POC 1 considerando também alguns rudimentos, como é o caso do *flam*, durante a transformação das funções de ativação para a partitura; bem como estudar outros métodos de transformação entre os dois domínios, como algum também baseado em aprendizagem profunda, fazendo com que a saída da rede seja uma representação do som transcrito em partitura ao invés de uma função de ativação.

Referências

Benetos, E., Ewert, S., and Weyde, T. (2014). Automatic transcription of pitched and unpitched sounds from polyphonic music.

Bittner, R., Salamon, J., Tierney, M., Mauch, M., Cannam, C., and Bello, J. (2014). Medleydb: A multitrack dataset for annotation-intensive mir research.

Böck, S., Krebs, F., and Schedl, M. (2012). Evaluating the online capabilities of onset detection methods.

McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., and Nieto, O. (2015). librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, volume 8.

Müller, M. (2015a). *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*, chapter 6.1. Springer Publishing Company, Incorporated, 1st edition.

Müller, M. (2015b). *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*, chapter 8.3.1. Springer Publishing Company, Incorporated, 1st edition.

Müller, M. (2015c). *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*, chapter 2.1.4. Springer Publishing Company, Incorporated, 1st edition.

Müller, M. (2015d). *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*, chapter 4.1. Springer Publishing Company, Incorporated, 1st edition.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Schloss, W. A. (1985). On the automatic transcription of percussive music - from acoustic signal to high-level analysis. Master's thesis, Stanford University, Stanford, CA.

Southall, C., Stables, R., and Hockman, J. (2016). Automatic drum transcription using bi-directional recurrent neural networks. In *ISMIR*.

Southall, C., Wu, C.-W., Lerch, A., and Hockman, J. (2017). MDB drums: An annotated subset of medleydb for automatic drum transcription. In *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*.

Vogl, R., Dorfer, M., and Knees, P. (2016). Recurrent neural networks for drum transcription. In *ISMIR*.

Vogl, R., Dorfer, M., and Knees, P. (2017). Drum transcription from polyphonic music with recurrent neural networks. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 201–205.

Williams, R. and Peng, J. (1998). An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural Computation*, 2.

Wu, C.-W., Dittmar, C., Southall, C., Vogl, R., Widmer, G., Hockman, J., Müller, M., and Lerch, A. (2018). A review of automatic drum transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(9):1457–1483.

Wu, C.-W. and Lerch, A. (2015). Drum transcription using partially fixed non-negative matrix factorization with template adaptation.