ANDRÉ MIGUEL SIKORA MARQUES

RECONHECIMENTO DE TABLATURA MUSICAL ATRAVÉS DE REDES NEURAIS CONVOLUCIONAIS

(*versão pré-defesa, compilada em 5 de dezembro de 2024*)

Trabalho apresentado como requisito parcial à conclusão do Curso de Bacharelado em Ciência da Computação, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Computação*.

Orientador: André Luis Vignatti.

CURITIBA PR

2024

RESUMO

Tablatura é uma notação musical utilizada para músicos aprenderem músicas em instrumentos de corda como violão e guitarra. A transcrição de tablaturas é um processo manual que demanda tempo e esforço de um músico. Algumas abordagens atuais utilizam o GuitarSet, que é um dataset que contém gravações de violão anotadas, para treinar uma rede neural convolucional que transcreve automaticamente tablaturas a partir de um arquivo de áudio. Esse artigo expõe algumas dessas abordagens, e mostra como a inclusão de um outro instrumento (especificamente, bateria) nos arquivos de áudio afeta o resultado da predição da rede neural convolucional.

Palavras-chave: Redes neurais convolucionais. Transcrição automática de tablaturas. GuitarSet.

ABSTRACT

A tablature is a type of musical notation used by musicians to learn songs on stringed instruments such as the guitar. Transcribing tablatures is a manual process that requires time and effort from a musician. Some modern approaches use GuitarSet, a dataset of annotated guitar recordings, to train a convolutional neural network that automatically transcribes tablatures from an audio file. This paper presents some of these approaches, and shows how the inclusion of another instrument (specifically, drums) to the audio files affects the results of the convolutional neural network's predictions.

Keywords: Convolutional neural networks. Automatic tablature transcription. GuitarSet.

LISTA DE FIGURAS

1.1 Como a tablatura deve ser interpretada em uma guitarra. 7
1.2 Um exemplo de como uma partitura pode ser interpretada de duas formas diferentes. 8
1.3 Um exemplo de uma forma menos eficiente de se tocar essa partitura. A versão inferior é menos eficiente pois precisa de mais movimentos do instrumentista. . . 9
1.4 Estrutura da *CNN* proposta por Humphrey e Bello (2014). 10

2.1 Um exemplo de arquitetura de rede neural convolucional [O'Shea e Nash (2015)] 11
2.2 Estrutura da *CNN* proposta por Wiggins e Kim (2019). 14

LISTA DE TABELAS

2.1 Métricas de estimação multitom. 15
2.2 Métricas de estimação de tablaturas. 15

5.1 Resultados do experimento de 2019 19
5.2 Resultados da repetição do experimento 19
5.3 Resultados do experimento modificado. 19

SUMÁRIO

1	**INTRODUÇÃO**	**7**
1.1	MOTIVAÇÃO	7
1.2	TRABALHOS RELACIONADOS	8
1.3	RESULTADOS	10
1.4	ESTRUTURA DO TRABALHO	10
2	**CONCEITOS PRELIMINARES**	**11**
2.1	*CONSTANT-Q*	11
2.2	REDES NEURAIS CONVOLUCIONAIS	11
2.3	*GUITARSET*	12
2.4	*.JAMS*	12
2.5	*TABCNN*	12
2.5.1	*Downsampling*	12
2.5.2	Transformação de áudios em imagens	13
2.5.3	Pré processamento das anotações	13
2.5.4	Rede Neural	14
2.5.5	Métricas	14
3	**PROPOSTA**	**16**
4	**METODOLOGIA**	**17**
5	**RESULTADOS.**	**19**
6	**CONCLUSÃO E TRABALHOS FUTUROS.**	**21**
	REFERÊNCIAS	**22**

1 INTRODUÇÃO

1.1 MOTIVAÇÃO

Uma tablatura é uma notação musical que indica onde colocar os dedos (e implicitamente a nota), ao invés de quais notas devem ser tocadas. Normalmente é utilizada para instrumentos trasteados, como violão, guitarra, baixo, etc., embora ainda possa menos comumente ser utilizada para instrumentos não trasteados, como piano.

Na tablatura, cada linha representa uma corda do instrumento, e cada número representa uma casa no braço do instrumento (Figura 1.1).

Figura 1.1: Como a tablatura deve ser interpretada em uma guitarra.

A tablatura proporciona uma visualização mais eficiente de como tocar uma música no violão, pois a partitura mostra somente a nota que deve ser tocada, já a tablatura mostra a nota e o local onde deve ser pressionada no braço do instrumento, pois no violão uma mesma nota pode ser tocada em diferentes lugares do braço (Figura 1.2).

Para instrumentos nos quais uma determinada nota só pode ser tocada em um único lugar, como o piano, a partitura é uma notação suficiente. Porém, em instrumentos como o violão existe mais de um possível local para uma determinada nota. Nesse caso partituras podem não ser muito eficientes, pois escolher um local inadequado para uma nota pode afetar o timbre, ou ser uma forma menos eficiente de tocar o instrumento (Figura 1.3).

Escrever manualmente uma tablatura é um processo demorado, que exige conhecimentos musicais e possivelmente compreender como funciona um *software* de escrita de tablaturas, como o *TuxGuitar* ou o *Guitar Pro* por exemplo.

Para que seja possível escrever a tablatura, é preciso que o músico saiba identificar que notas está ouvindo, o que pode ser um problema para iniciantes. Utilizando a guitarra como exemplo, em músicas mais simples, a identificação pode ser mais fácil por motivos como a música ser mais lenta, existirem menos efeitos (reverberação, *delay*, etc) sobre o instrumento,

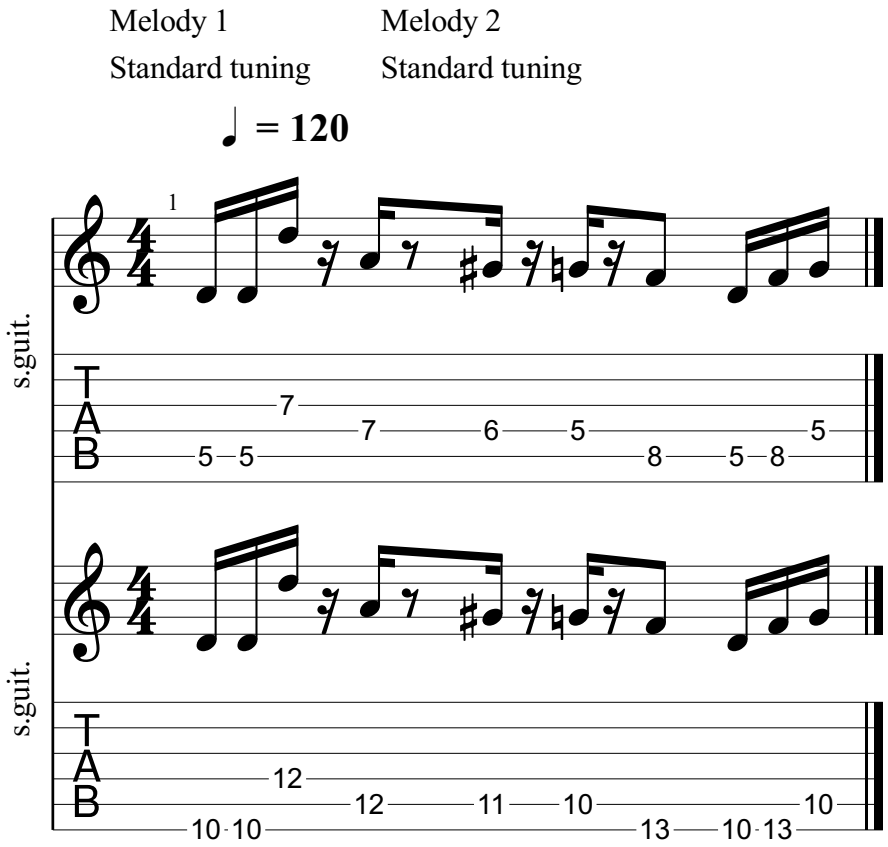

Figura 1.2: Um exemplo de como uma partitura pode ser interpretada de duas formas diferentes.

uso de acordes mais comuns, entre outros. Em músicas mais complexas, porém, o oposto ocorre. Ou seja, a música pode ser mais rápida, muitos efeitos, acordes mais raros, etc.

Para músicos menos experientes, uma ferramenta que resolva esse problema facilita o aprendizado de novas músicas. Já profissionais como professores de música também se beneficiam disso, pois diminui o tempo que precisam transcrever músicas. A proposta desse trabalho é mostrar algumas abordagens de inteligência artificial para transcrição automática de músicas, e como a presença de mais instrumentos afeta a predição.

1.2 TRABALHOS RELACIONADOS

Em 2014, *Humphrey* e *Bello* utilizaram redes neurais convolucionais para identificar acordes em uma música, utilizando *datasets* de música pop e dos *Beatles*. Estes *datasets* contém anotações de quais acordes estão sendo tocados em um determinado instante em uma música. Como representação do áudio, foi escolhida a transformada *Constant-Q*, por essa transformada ser linear em relação ao tempo e ao tom, e também com o intuito de reduzir a dimensionalidade dos dados.

A CNN proposta foi definida como duas camadas convolucionais, uma camada afim, e uma camada *6D-Softmax* (Figura 1.4).

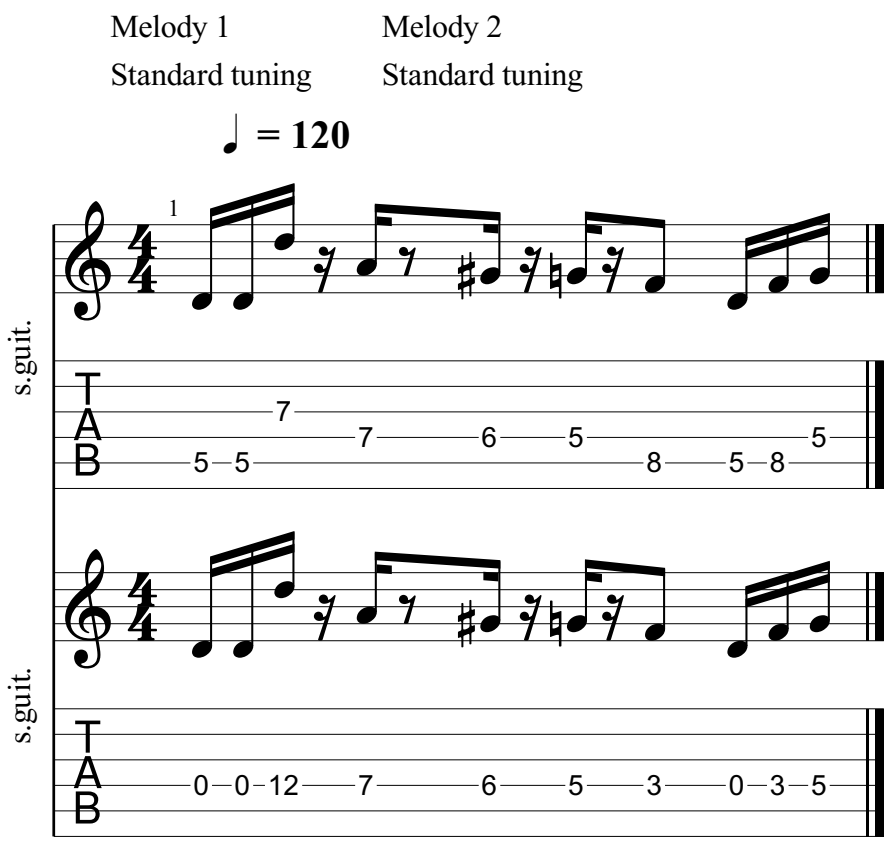

Figura 1.3: Um exemplo de uma forma menos eficiente de se tocar essa partitura. A versão inferior é menos eficiente pois precisa de mais movimentos do instrumentista.

Em 2019, *Wiggins* e *Kim* também utilizaram redes neurais convolucionais, mas desta vez com o intuito de transcrever uma tablatura identificando onde no braço do violão determinada nota foi tocada, e utilizando o *GuitarSet*, que é um dataset de performances solo de violão. De acordo com os autores, foi a primeira tentativa de utilizar *CNN* para esse objetivo.

Assim como *Humphrey* e *Bello*, utilizaram a *Constant-Q* para representar o áudio, pelo motivo de que o eixo de frequências é linearmente espaçado em relação ao tom, o que permite que a rede aprenda de forma independente do tom.

A estrutura da rede neural (chamada *TabCNN*) é parecida com o *AlexNet* [Krizhevsky et al. (2012)] e o *VGGNet* [Simonyan e Zisserman (2015)], uma série de convoluções seguidas por uma operação de *sub-sampling*, terminando em conexões de camada densa. A ideia é que as convoluções servem como extratores de características, aprendendo coeficientes de filtros espaciais que resultam em um mapa de características, e as camadas densas servem como um classificador para processar essas características. Como o esse trabalho se baseia nessa rede neural, essa é melhor explicada na seção 2.5.4.

Em 2023, *Cwitkowitz et al* expandiram a ideia de 2019, aprofundando o modelo e mudando a camada de predição de tablatura.

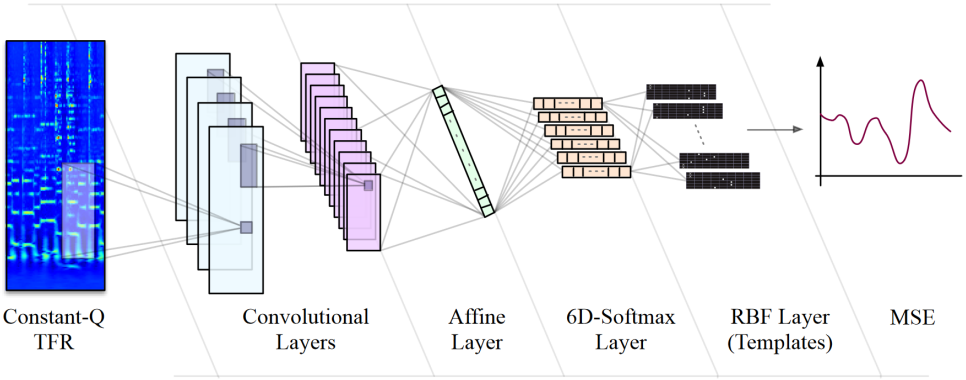

Figura 1.4: Estrutura da *CNN* proposta por Humphrey e Bello (2014).

Ao contrário dos autores acima, utilizaram a *Harmonic Constant-Q*, que calcula múltiplas *Constant-Q* antes de juntá-las em uma única representação. Com isso, espera-se que o modelo consiga explorar informações harmônicas de forma mais eficiente.

O *FretNet*, que é como batizaram seu modelo, consiste de três blocos, cada um possui duas camadas convolucionais seguidas de normalização e ativação *ReLU*. No primeiro bloco, as camadas convolucionais contém 16 filtros, no segundo 32, e no terceiro 48. Em todos os blocos, as camadas convolucionais utilizam um *kernel* de tamanho 3×3.

Após o segundo e o terceiro bloco, é utilizado *max pooling* com *kernel* e *stride* 2, e *dropout* é aplicado com taxas 0,5 e 0,25 respectivamente. Os vetores que emergem disso alimentam três predições diferentes: tablatura, sem desvios, e sem *onsets*. Cada uma dessas consiste de uma camada totalmente conectada, uma ativação *ReLU*, e uma última camada totalmente conectada.

1.3 RESULTADOS

Após a inclusão de bateria nos arquivos de áudio do *GuitarSet*, que é um *dataset* de gravações de violão anotadas, todas as métricas obtiveram um resultado pior em relação ao trabalho original de *Wiggins* e *Kim*. Notavelmente, o *recall* caiu mais de 19% em duas métricas.

1.4 ESTRUTURA DO TRABALHO

Primeiramente definimos quais as ferramentas utilizadas, especificamente a transformada *Constant-Q*, redes neurais convolucionais, e o *TabCNN*. Além disso, o *dataset* utilizado, e os arquivos *.jams*, que são utilizados pelo *GuitarSet* para armazenar as informações de cada nota sendo tocada.

Após isso, explicamos qual a proposta para modificar o *GuitarSet* e ver como o *TabCNN* lida com as mudanças, e como essas mudanças foram feitas. Por último apresentamos os resultados, comparando-se o *dataset* sem modificações com a versão com as modificações propostas, e as conclusões decorrentes disso.

2 CONCEITOS PRELIMINARES

2.1 *CONSTANT-Q*

A transformada *Constant-Q* transforma séries de dados em domínio de frequência. É relacionada à transformada de *Fourier* [Brown (1991)]. Quando a *Constant-Q* é aplicada a um áudio, é possível utilizá-la para representar as frequências em relação ao tempo através de um gráfico.

A *CQT* é considerada melhor que a *FFT* para esta aplicação por ter uma dimensionalidade menor, por diminuir o tamanho do eixo de frequência por meio de espaçamento linear da frequência em relação ao tom musical [Wiggins e Kim (2019)].

2.2 REDES NEURAIS CONVOLUCIONAIS

Redes neurais são compostas por camadas de nodos, e uma camada é de entrada, uma ou mais “escondidas”, e por fim uma camada saída. Cada nodo está conectado a outro nodo e possui um peso e um *threshold*. Se a saída é um valor acima do *threshold*, o nodo é ativado e propaga dados à próxima camada (Figura 2.1).

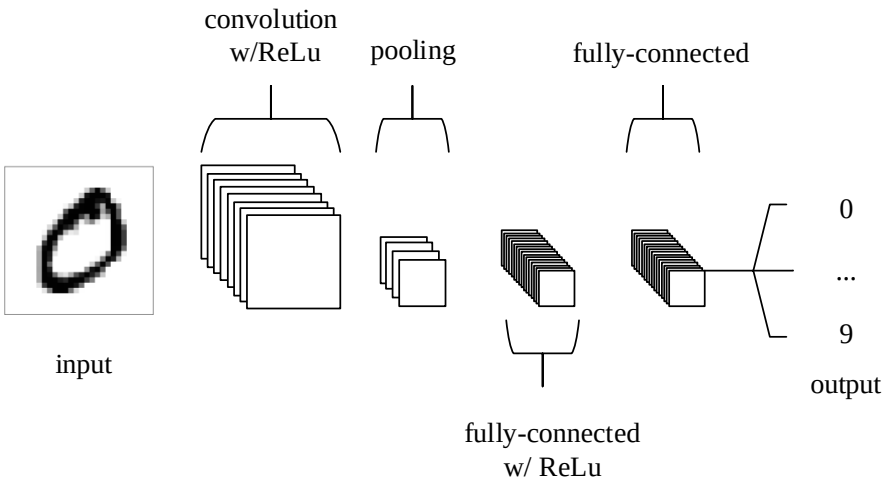

Figura 2.1: Um exemplo de arquitetura de rede neural convolucional [O’Shea e Nash (2015)]

Uma classe específica de redes neurais são as convolucionais. Uma rede neural convolucional é um algoritmo de aprendizado de máquina que aprende através de filtros sucessivos.

Uma convolução é um produto cartesiano entre duas matrizes, uma *input* e uma *kernel*, normalmente 3x3, 5x5, ou 7x7 (embora outros valores também seja possíveis). O *kernel* "desliza"pela matriz *input* e constrói outra imagem a partir do *pooling* extraído. O método de *pooling* mais popular é o *max pooling*, que seleciona o maior valor na vizinhança do *kernel*. Frequentemente são feitas convoluções sucessivas em um *input*.

Como a convolução é uma operação linear, após a convolução são utilizadas camadas de não-linearidade, como por exemplo *ReLU* (unidade linear retificada), que transforma *outputs* negativos em zero.

A *CNN* é uma rede neural ideal para tarefas de visão computacional. Como o arquivo de áudio é transformado em uma imagem, torna-se uma ferramenta ideal para essa tarefa.

2.3 *GUITARSET*

O *GuitarSet* é um dataset composto por arquivos de áudio e arquivos *.jams*, que foi criado a partir das gravações de seis instrumentistas tocando violão. Os músicos foram instruídos a improvisar de duas formas diferentes. A primeira os autores chamam de *comp*, quando o musicista é instruído a tocar acordes, e segunda *solo*, quando é instruído a improvisar um solo baseado no que fez na parte anterior [Xi et al. (2018)].

As gravações foram feitas através de um microfone e de um captador hexafônico montado no instrumento. As anotações foram coletadas automaticamente através do áudio desse captador, o que as torna mais precisas, já que é possível captar o áudio de cada corda separadamente. No total, o dataset tem uma duração de pouco mais de 3 horas, abrangendo diversos gêneros musicais.

2.4 *.JAMS*

Os arquivos *.jams* são arquivos de texto similares a arquivos *.json* que contém diversas informações relevantes para a tarefa de recuperação de informações musicais [Humphrey et al. (2014)]. No *GuitarSet* são incluídas informações como acordes, batidas, ritmo, contorno do tom, entre outras. Para este trabalho são particularmente relevantes as informações de a qual valor *MIDI* corresponde uma nota que está sendo tocada em um determinado instante.

2.5 *TABCNN*

O *TabCNN* é um algoritmo desenvolvido por *Wiggins* e *Kim* que utiliza redes neurais convolucionais para extrair tablaturas a partir de um arquivo de áudio. O algoritmo utiliza o *GuitarSet* como base, então utiliza arquivos de áudio, e arquivos *.jams* no treinamento. O funcionamento do algoritmo é explicado nas subseções abaixo.

2.5.1 *Downsampling*

Os arquivos de áudio do *GuitarSet* tem um tamanho desnecessariamente grande, portanto, antes de extrair-se a representação visual do áudio, é feito um *downsampling* (redução da taxa de amostragem) de 44100Hz para 22050Hz. Devido à Frequência de *Nyquist*, isso nos permite representar sinais até a frequência de 11025Hz [Leis (2013)].

Esse *downsampling* introduz uma pequena distorção ao áudio, porém como a maior nota produzida pelo violão do *dataset* é B5 (que tem como frequência fundamental 988Hz), presume-se que não se perde muita informação acima dos 11025Hz. A intenção do *downsampling* é reduzir a dimensionalidade do *input*. Com isso, diminui-se o volume de dados.

2.5.2 Transformação de áudios em imagens

Como pretende-se utilizar redes neurais convolucionais, é preciso utilizar uma imagem como entrada, ao invés dos arquivos de áudio. Para converter os áudios do *dataset* em uma representação visual, existem diversas transformadas possíveis. Essas transformadas convertem um arquivo de áudio em um vetor de números complexos, e esses números complexos são mapeados para a imagem que é utilizada como entrada na *CNN*.

A mais comum é a *Transformada Rápida de Fourier* (*STFT*), porém esta não é a ideal para redes neurais convolucionais, pois tem uma alta dimensionalidade. Além disso, uma representação linearmente espaçada permite que a rede aprenda características independentes do tom. Para isso foi utilizada a transformada *Constant-Q* (*CQT*), que reduz a dimensionalidade do eixo de frequência ao espaçar compartimentos de frequência linearmente em relação à nota musical [Wiggins e Kim (2019)].

A *CQT* exige alguns parâmetros, foi executada com 192 *bins* (compartimentos), abrangendo 8 oitavas (um total de 96 notas). Isso totaliza 24 *bins* por oitava, ou 2 *bins* por semitom. O *hopsize* utilizado foi 512 amostras, que corresponde a 43 *frames* por segundo. Foi utilizada uma janela de contexto deslizante de 9 *frames* para gerar as imagens, incluindo *padding* com zeros dos dois lados para que cada clipe tenha uma janela de contexto completa de 9 *frames*. Com isso, as amostras de input tem tamanho 192×9 a cada *frame*.

2.5.3 Pré processamento das anotações

Cada anotação de tom é amostrada ao *frame rate* de 43 *frames* por segundo. Como as anotações não são valores *MIDI* exatos, é necessário um arredondamento ao inteiro mais próximo. Para descobrir em qual casa do instrumento a nota foi tocada, é subtraído da anotação o valor *MIDI* da corda solta na qual a anotação foi encontrada. Por exemplo, se o valor *MIDI* encontrado foi 50 na corda 5, o valor *MIDI* da corda 5 solta é 45, logo a nota foi tocada na casa 5 da corda 5.

Se o valor dessa subtração é 0, significa que a corda foi tocada solta (sem pressionar uma casa). Como no violão do dataset existem 19 casas, são necessárias 21 classes para cada casa, pois precisa-se incluir a corda solta, e a corda não sendo tocada. Como são 6 cordas no instrumento, o total de classes passa a ser 6×21 a cada *frame*.

2.5.4 Rede Neural

A *CNN* proposta foi construída com três camadas convolucionais, sendo cada camada seguida por uma ativação *ReLU* (unidade linear retificada). A Primeira camada possui 32 filtros com *kernel* de tamanho 3×3, e as subsequentes 64 filtros de mesmo tamanho. Essas camadas são seguidas de uma camada *Max Pooling* de tamanho 2×2, seguida de uma camada densa de dimensão 128, e mais uma ativação *ReLU*. Isso é seguido de uma camada densa de dimensão 126, que é remodelada para 6×21 (6 cordas e 21 classes de notas, incluindo nenhuma nota e a corda solta), e por fim uma ativação *Softmax*. (Figura 2.2).

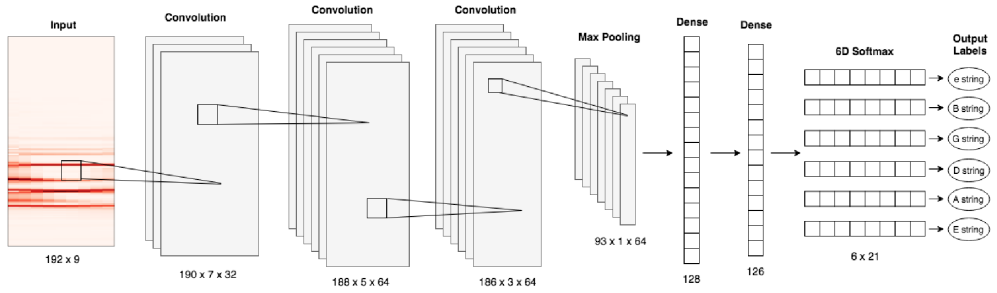

Figura 2.2: Estrutura da *CNN* proposta por Wiggins e Kim (2019).

A função de perda é computada através da soma da entropia-cruzada categórica de cada corda, e é utilizado o algoritmo de otimização *ADADELTA*. A taxa de aprendizado é 1,0, e o tamanho do mini *batch* é 128 amostras de treinamento. A rede neural é treinada por 8 épocas, e utiliza validação cruzada em 6 *folds* (um para cada músico do dataset), utilizando 5 dos 6 instrumentistas para treinar, e 1 dos 6 para testar.

2.5.5 Métricas

Para avaliar a performance do algoritmo, os autores utilizam métricas previamente estabelecidas pela literatura para detecção multitom (tabela 2.1), e como anterior ao artigo não existiam métricas para avaliar a performance da estimação de tablaturas, propõe novas métricas que avaliam isso (tabela 2.2).

Título	Como é calculado	O que mede
Precisão Multitom	O número de tons identificados corretamente divididos pelo número total de tons preditos	Quão frequentemente tons identificados estão corretos
Recall Multitom	O número de tons identificados corretamente divididos pelo número total de tons *ground truth*	Quão frequentemente tons existentes no sinal são detectados pelo sistema
F-score Multitom	Média harmônica da precisão e *recall* multitom	Performance geral para estimação de tons

Tabela 2.1: Métricas de estimação multitom.

Título	Como é calculado	O que mede
Precisão da Tablatura	O número de combinações casa-corda identificadas corretamente divididas pelo número total de combinações casa-corda preditas	Quão frequentemente a tablatura identificada está correta
Recall da Tablatura	O combinações casa-corda identificadas corretamente divididas pelo número total de combinações casa-corda *ground truth*	Quão frequentemente uma tablatura existente no sinal é detectada pelo sistema
F-score da Tablatura	Média harmônica da precisão e *recall* da tablatura	Performance geral para estimação de tablatura
Taxa de Desambiguação da Tablatura (TDR)	O número de combinações casa-corda corretamente identificadas divididas pelo total de tons corretamente identificados	Quão frequentemente tons identificados são atribuídos à tablatura correta

Tabela 2.2: Métricas de estimação de tablaturas.

3 PROPOSTA

O *GuitarSet* contém gravações de apenas um violão, portanto não é viável utilizá-lo para treinar uma rede neural convolucional para transcrever músicas que sejam feitas por uma banda com vários membros. Por exemplo, uma banda pode ter um cantor, baixista, e baterista, além do músico tocando violão.

Isso faz com que a rede neural convolucional tenha que fazer uma filtragem mais minuciosa do que é uma nota sendo tocada por um violão em um sinal de áudio. Portanto utilizar uma *CNN* treinada apenas no *GuitarSet* para tentar transcrever uma música com tantos instrumentos simultâneos não obterá um resultado tão satisfatório.

Neste trabalho pretende-se verificar como a adição de mais instrumentos pode afetar a predição do *TabCNN*. Isso será feito através da inclusão de percussão aos arquivos de áudio do *GuitarSet*. Como este consiste de um número limitado de ritmos (especificamente, todas performances do *dataset* foram feitas em um de 27 ritmos abrangendo desde 68 BPM — batidas por minuto — até 200 BPM), é possível criar um pequeno número de percussões que seguem o mesmo ritmo e mesclá-las aos arquivos de áudio de forma programática.

Ao treinar a rede neural novamente com esses áudios modificados, espera-se que a *CNN* obtenha um desempenho inferior, pois anteriormente a rede neural precisava identificar as notas apenas de uma gravação de um violão, e agora precisa distingui-las de uma gravação que também possui uma bateria, ou seja, o áudio é mais "sujo"e tem mais diferenças entre um exemplo de uma nota e outro.

4 METODOLOGIA

Antes de se fazer qualquer modificação ao *dataset*, o *TabCNN* foi treinado novamente sem nenhuma modificação, para garantir que os resultados modificados possam ser comparados com o do artigo original de *Wiggins* e *Kim*.

A implementação original, porém, não estava obtendo os resultados esperados, possivelmente devido à sua idade. Então foi necessário contactar os autores, os quais sugeriram uma outra implementação feita em 2022 [Cwitkowitz (2022)].

Após isso os resultados obtidos foram suficientemente próximos do original, então foi possível dar início às modificações. Primeiramente, foi preciso definir quais ritmos seriam necessários para englobar o *dataset* completo. Isso foi feito através de um pequeno *script* em *Python*.

```
import jams
import os

annotations = os.listdir("data/GuitarSet/annotation")
bpm_list = []

for annotation in annotations:
      jams_file = jams.load(f"data/GuitarSet/annotation/{annotation}")
      if jams_file.annotations["tempo"][0]["data"][0][2] not in bpm_list:
            bpm_list.append(jams_file.annotations["tempo"][0]["data"][0][2])

print([int(bpm) for bpm in bpm_list])
print(len(bpm_list))
```

Com isso, obtiveram-se 27 ritmos diferentes, entre 68 BPM e 200 BPM. Após isso, foram criadas 27 percussões diferentes. As percussões foram criadas no *Reaper* (um *Digital Audio Workstation*), utilizando o *Drummic'a*, *VST* (*Virtual Studio Technology*) de bateria gratuito da *Sennheiser*, escolhendo-se arbitrariamente *presets* de afinação e de percussão.

Cuidado foi tomado para que o volume da bateria não ficasse muito alto em relação ao violão, isso foi feito escolhendo alguns áudios do GuitarSet e mesclando-os ainda no *Reaper* com a percussão criada, e diminuindo o volume da percussão, antes de salvar o áudio isolado da percussão.

Após criar uma percussão para cada BPM, a sincronização com os arquivos do dataset foi feita com um *script* em *Python* utilizando *FFmpeg*.

```
import subprocess
import os

if not os.path.exists("data/GuitarSet_with_drums"):
      os.makedirs("data/GuitarSet_with_drums")

GuitarSet = os.listdir("data/GuitarSet/audio/audio_mic")

for audio in GuitarSet:
      file_ext = audio.split(".")[-1]
      if file_ext == "wav":
            bpm = audio.split("-")[1]
            print(f"Merging {audio} with {bpm}_bpm.wav ...")
            ffmpeg = f"ffmpeg -i data/drum_samples/{bpm}_bpm.wav -i
                              data/GuitarSet/audio/audio_mic/{audio}
                              -filter_complex
                              amix=inputs=2:duration=shortest
                              data/GuitarSet_with_drums/{audio}"
            subprocess.run(ffmpeg, shell=True)
```

Após isso a rede neural foi treinada novamente com esses novos arquivos de áudio, e os resultados comparados com a versão original. A rede neural foi treinada em um computador com uma *CPU* Intel i5-3470 3.20GHz com 4 núcleos, *GPU Nvidia RTX* 3060, e 16GB de *RAM*. O treinamento durou pouco mais de três dias.

5 RESULTADOS

A repetição do experimento obteve resultados similares ao original, e o experimento com o dataset modificado para incluir percussão obteve resultados piores em todas as métricas, como esperava-se. O experimento original de *Wiggins* e *Kim* obteve os seguintes resultados (tabela 5.1).

Multitom			Tablatura			
Precisão	*Recall*	*F-score*	Precisão	*Recall*	*F-score*	TDR
0,900	0,764	0,826	0,809	0,696	0,748	0,899

Tabela 5.1: Resultados do experimento de 2019

Para garantir que o código feito por *Wiggins* e *Kim* ainda funciona, tendo em vista que foi feito em 2019 e reimplementado em 2022 por um outro autor, o experimento foi executado novamente, ainda com o dataset inalterado. A repetição do experimento, sem modificação dos arquivos de áudio, obteve os seguintes resultados (tabela 5.2).

Multitom			Tablatura			
Precisão	*Recall*	*F-score*	Precisão	*Recall*	*F-score*	TDR
0,913	0,749	0,819	0,814	0,683	0,739	0,905

Tabela 5.2: Resultados da repetição do experimento

Os resultados são suficientemente similares, e estão todos dentro do desvio padrão descrito no artigo de *Wiggins* e *Kim*, o que torna viável a implementação das modificações. No experimento modificado proposto por esse artigo obtiveram-se os seguintes resultados (tabela 5.3).

Multitom			Tablatura			
Precisão	*Recall*	*F-score*	Precisão	*Recall*	*F-score*	TDR
0,878	0,559	0,673	0,730	0,483	0,574	0,838

Tabela 5.3: Resultados do experimento modificado

Como esperado, todas as métricas obtiveram uma queda, sendo a mais drástica o *recall*, que caiu 20% para a tablatura e 19% para o multitom. Tamanha queda mostra que o sistema tem uma dificuldade maior em detectar tons existentes no sinal quando se incluem mais instrumentos.

Curiosamente, a precisão multitom não obteve uma queda não tão brusca, estando apenas 3,5% pior. Isso mostra que a inclusão de percussão nos arquivos de áudio não afetou em muito a classificação do tom nos casos em que o sistema conseguiu identificar que uma nota estava ocorrendo no sinal.

Ainda assim, a precisão da classificação quanto a tablatura obteve uma queda de 8,4%, o que indica que com o dataset modificado o sistema identifica um tom corretamente, mas atribui-o a uma tablatura incorreta com uma frequência um pouco maior.

Uma possível solução para esses resultados pouco satisfatórios seria a inclusão de mais dados ao *dataset*. Além do artifício imediato de gravar mais performances de violão, uma possibilidade é a ideia de Humphrey e Bello (2014), que utilizam de *pitch shifting* aleatoriamente para melhor distribuir a variância de notas.

6 CONCLUSÃO E TRABALHOS FUTUROS

Neste trabalho foi apresentado como modificações aos arquivos de áudio do *GuitarSet* afetam o resultado de um sistema de transcrição de tablatura. O experimento indica que as modificações afetam todas as métricas, principalmente a detecção de tons no sinal.

Para trabalhos futuros, a ideia imediata seria incluir outros instrumentos, por exemplo baixo, e ver como isso afeta os resultados. Uma outra ideia possível seria transformar arquivos de tablaturas (por exemplo, arquivos *.gp*) no formato *.jams*, e criar um dataset que utiliza músicas e suas respectivas tablaturas para utilizar como entrada no *TabCNN*.

REFERÊNCIAS

Brown, J. (1991). Calculation of a constant q spectral transform. *Journal of the Acoustical Society of America*, 89:425–.

Cwitkowitz, F. (2022). Automatic music transcription (amt) tools. `https://github.com/cwitkowitz/amt-tools/tree/master`. Acessado em 07/11/2024.

Cwitkowitz, F., Hirvonen, T. e Klapuri, A. (2023). Fretnet: Continuous-valued pitch contour streaming for polyphonic guitar tablature transcription. Em *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 30, página 1–5. IEEE.

Humphrey, E., Salamon, J., Nieto, O., Forsyth, J., Bittner, R. e Bello, J. (2014). Jams: A json annotated music specification for reproducible mir research. Em *ISMIR*.

Humphrey, E. J. e Bello, J. P. (2014). From music audio to chord tablature: Teaching deep convolutional networks toplay guitar. Em *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, páginas 6974–6978.

Krizhevsky, A., Sutskever, I. e Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Em *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, página 1097–1105, Red Hook, NY, USA. Curran Associates Inc.

Leis, J. W. (2013). *Digital signal processing using MATLAB for students and researchers*. Wiley.

O'Shea, K. e Nash, R. (2015). An introduction to convolutional neural networks.

Simonyan, K. e Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition.

Wiggins, A. e Kim, Y. (2019). Guitar Tablature Estimation with a Convolutional Neural Network. Em *Proceedings of the 20th International Society for Music Information Retrieval Conference*, páginas 284–291. ISMIR.

Xi, Q., Bittner, R. M., Pauwels, J., Ye, X. e Bello, J. P. (2018). Guitarset: A dataset for guitar transcription. Em *International Society for Music Information Retrieval Conference*.

(Wiggins e Kim, 2019) (Humphrey e Bello, 2014) (Cwitkowitz et al., 2023) (Krizhevsky et al., 2012) (Simonyan e Zisserman, 2015) (O'Shea e Nash, 2015)