CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
CURSO DE ENGENHARIA DE COMPUTAÇÃO

ROTULAGEM AUTOMÁTICA DE MÚSICAS: UMA ABORDAGEM UTILIZANDO LSTM

GUILHERME FERNANDES MARCHEZINI

Orientador: Rogério Martins Gomes
Centro Federal de Educação Tecnológica de Minas Gerais

BELO HORIZONTE
NOVEMBRO DE 2018

GUILHERME FERNANDES MARCHEZINI

ROTULAGEM AUTOMÁTICA DE MÚSICAS: UMA ABORDAGEM UTILIZANDO LSTM

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Computação do Centro Federal de Educação Tecnológica de Minas Gerais, como requisito parcial para a obtenção do título de Bacharel em Engenharia de Computação.

Orientador: Rogério Martins Gomes
Centro Federal de Educação Tecnológica de Minas Gerais

CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
CURSO DE ENGENHARIA DE COMPUTAÇÃO
BELO HORIZONTE
NOVEMBRO DE 2018

Agradecimentos

Gostaria de agradecer aos meus pais Ronaldo e Vanessa que me criaram e estiveram comigo em toda essa caminhada no CEFET-MG, apoiando minha decisão de curso, cobrando nos estudos e por me darem condições de chegar aonde estou. Agradeço meu irmão Rafael, não só pelos diversos momentos de descontração, mas também por servir de inspiração para eu buscar o meu melhor.

Agradeço toda a minha família, tios, tias, primos e primas pelos conselhos, ensinamentos de vida e também pelo reconhecimento que sempre me deram.

Agradeço à insticuição CEFET-MG, funcionários e professores. Essa instituição incrível, com pessoas muito capacitadas que me abriu grandes oportunidades de crescimento pessoal e profissional, me dando uma excelente formação acadêmica. Um agradecimento especial ao meu professor e orientador Rogério, por ter me aconselhado e ajudado neste trabalho, dedicando uma atenção especial à minha escrita.

Agradeço a todos meus amigos, aqueles que perdi contato, aos que conheci nessa caminhada, aos que sempre estiveram aqui, porque todos tiveram papel importante na minha graduação. Agradeço ao Mattar, Lauro, Vitor, Dornas e Daniel, por me ajudarem nas matérias e pelos momentos de diversão. Agradeço ao Daniel, não apenas por ter sido minha dupla na maioria das matérias, mas também pelo grande amigo que vem sendo nesses anos. Agradeço ao Bonifácio e Gabriel, por serem grandes amigos meus desde antes de entrar na faculdade e me acompanharam em todos esses anos. Um agradecimento especial à minha grande amiga e namorada Jéssica por me ajudar nesse trabalho, ouvir com interesse minhas idéias e meu progresso, me apoiando em momentos de dificuldade.

Gostaria de fazer um agradecimento a todas bandas que produziram as músicas que me ajudaram nesse trabalho, não apenas sendo o foco da pesquisa, mas também por todos os ensinamentos que me trouxeram. Foram essas músicas que me fizeram enxergar o quão longe já cheguei, as guerras que venci, pensar em voz alta, 'vitorioso e orgulhoso'. Foram também essas músicas que estiveram comigo e me fizeram aprender que eu não deveria me cobrar demais, pois eu não sou um super herói, eu não sou perfeito, eu sou somente eu, um ser humano. E quando as coisas começavam a dar errado sempre existiam melodias que me acompanhavam, músicas que me fizeram questionar se deveria levantar meus punhos e amaldiçoar os céus ou fechar os olhos e perceber que é apenas a vida. Músicas 'perfeitas' que me fizeram 'sorrir', falando da minha 'biografia', sobre 'ser humano' e da 'minha guerra interna', músicas que foram 'terapia', me fizeram 'dançar e dançar', 'mataram' meu tédio e me fizeram 'ter fé em mim'. Obrigado músicos por suas músicas, que essa arte continue trazendo felicidade, ensinamentos e conforto para todos nós, para que

possamos ouvir mais vezes ’essa música salvou minha vida’.

Por fim agradeço a Deus, por ter me dado a oportunidade de passar esses anos incríveis aprendendo, conhecendo pessoas, crescendo pessoal e espiritualmente e por toda força que Ele me deu para superar os momentos de dificuldade.

”Desencorajado, levante seus olhos, quando sentir vontade de desistir, quando disserem que não pode ser feito, cabe a você mostrar porque estão errados!” (Memphis May Fire, Legacy)

Resumo

A música é uma forma de arte composta de um padrão de sons e ritmos feitos por instrumentos, vozes e computadores ou uma combinação desses. Dessa forma, cada música possui suas características próprias, sejam elas de gênero (rock, emo), instrumentos presentes (violão, piano), informações emocionais (triste, feliz) etc, que determinam sua classificação ou rótulo. O grande número de músicas existentes, bem como sua diversidade, inviabilizam a rotulação manual, criando a necessidade de automatizar esse processo. Sendo assim, esse trabalho propõe a modelagem e desenvolvimento de um método de rotulação automática de música utilizando um tipo especial de rede neural recorrente chamada de *Long Short Term Memory* (LSTM), capaz de aprender padrões temporais de curto e longo prazo. Na construção do modelo, foram utilizadas as bases *Million Song Dataset*, composta pelas características das músicas, e pela *Last.fm*, composta pelos respectivos rótulos. Essas bases, foram divididas em dois conjuntos, treino e teste, seguindo a divisão pré-estabelecida pela base de rótulos. A análise de desempenho utilizada foi a AUC (*Area Under the ROC Curve*). Neste trabalho, foram realizados dois tipos de experimentos com o objetivo de se avaliar o método. No primeiro teste, o número de camadas escondidas será variada com o objetivo de verificar o efeito do aumento da complexidade da rede na predição dos rótulos das músicas que ainda não foram classificadas. No segundo teste, por sua vez, será avaliado o efeito do treinamento nos resultados obtidos quando se aumenta o número de épocas de treinamento. Os resultados mostraram que quando o número de épocas é mantido constante e o número de camadas é aumentado, a rede passa a apresentar uma melhora de desempenho até um certo número de camadas, quando a rede passa a apresentar um efeito de *overfit*. Por sua vez, um aumento no número de épocas produz um incremento no desempenho do sistema para redes mais complexas, com um número maior de camadas, devido a redução do efeito de subtreinamento ou *underfit*.

Palavras-chave: LSTM, rótulagem automática, música.

Abstract

Music is an art composed of a pattern of sounds and rhythms made by instruments, voices, and computers or a combination of these. In this way, each song has its characteristics, whether they are genres (rock, emo), type of instruments (guitar, piano), emotional information (sad, happy), etc. This information determines its classification or label. The large number of songs, as well as their diversity, make manual labeling unfeasible, creating the need to automate this process. Therefore, this work proposes to model and develop a method of automatic labeling of music using a particular type of recurrent neural network called Long Short Term Memory (LSTM), able to learn short- and long-term temporal patterns. In the construction of the model, the bases of the Million Song and Last.fm, composed by the characteristics of the songs and their respective labels were used. These bases were divided into two sets, training, and test, following the division pre-established by the base of labels. The used metric was the AUC (Area Under the ROC Curve). In this work, two types of experiments were carried out to evaluate the method. In the first test, the number of hidden layers will be varied to verify the effect of increasing the complexity of the network in predicting the labels of songs that have not yet been classified. In the second test, the effect of training on the results obtained will be evaluated when the number of epochs is increased. The results showed that when the number of training epochs is kept constant, and the number of layers is increased, the network will show a performance improvement up to a certain number of layers when the network has an overfit effect. In turn, an increase in the number of training epochs produces an improvement in the performance of the system for more complex networks, with a higher number of layers, due to the reduction of the undertraining or underfit effect.

Keywords: LSTM, auto-tagging, music.

Lista de Figuras

Figura 1 – Exemplo de *Fade-out* 13
Figura 2 – Exemplo de *Fade-in* 13
Figura 3 – Diferentes de tipos de barras utlizadas para escrita 13
Figura 4 – Modo Ionian 13
Figura 5 – Exemplo de espectrograma 14
Figura 6 – Célula da rede recorrente 15
Figura 7 – Célula da rede recorrente em diferentes passos de tempo 15
Figura 8 – Estrutura detalhada da célula da rede recorrente 16
Figura 9 – Estrutura detalhada da célula LSTM 17
Figura 10 – Estado da célula rede LSTM 17
Figura 11 – Camada do portão do esquecimento 18
Figura 12 – Camada do portão de entrada 18
Figura 13 – Camada de atualização 18
Figura 14 – Camada do portão de saída 19
Figura 15 – Fluxograma da metodologia proposta 20
Figura 16 – Organização da matriz de entrada 25

Lista de Tabelas

Tabela 1 – Comprativo de resultados variando o número de camadas 28
Tabela 2 – Comprativo de resultados variando o número de épocas 29
Tabela 3 – Comprativo de resultados de diversos métodos 30

Lista de Abreviaturas e Siglas

API	*Application Programming Interface*
AUC	*Area Under the ROC Curve*
BDS	*Boosted Decision Stump*
CEC	Carrosséis de Erro Constante
CRNN	*Convolutional Recurrent Neural Network*
EM	*Expectation-Maximization*
ENS	*Echo Nest Song*
ENT	*Echo Nest Timbre*
FCN	*Fully Convolutional Network*
GMM	*Gaussian Mixture Model*
LSTM	*Long Short Term Memory*
MFC	*Mel-Frequency Cepstrum*
MFCC	*Mel-Frequency Cepstral Coefficients*
MSD	*Million Song Dataset*
ReLU	*Rectified Linear Unit*
RNN	*Recurrent Neural Network*
SGD	*Stochastic gradient descent optimizer*
STFT	*Short Time Fourier Transform*
SVM	*Support Vector Machine*
TanH	Tangente Hiperbólica

Sumário

1 – Introdução **1**
- 1.1 Redes Neurais Artificiais 2
- 1.2 *Long Short Term Memory* 3
- 1.3 Motivação e Justificativa 3
- 1.4 Objetivos 4
  - 1.4.1 Objetivos Gerais 4
  - 1.4.2 Objetivos Específicos 4
- 1.5 Organização do Trabalho 4

2 – Trabalhos Relacionados **6**
  - 2.0.1 *Base de Rótulos* 9

3 – Fundamentação Teórica **11**
- 3.1 Características musicais 11
- 3.2 Redes Neurais Recorrentes 14
- 3.3 Long Short-Term Memory 16

4 – Metodologia **20**
- 4.1 *Revisão bibliográfica* 20
- 4.2 *Base de dados* 21
  - 4.2.1 *Million Song Dataset* 21
  - 4.2.2 *Base de Rótulos* 22
- 4.3 Desenvolvimento 23
  - 4.3.1 *Extração e tratamento dos dados* 23
  - 4.3.2 *Rede LSTM* 24
- 4.4 Experimentos 27
  - 4.4.1 Teste variando o número de camadas escondidas 27
  - 4.4.2 Teste variando o número de épocas 27

5 – Análise e Discussão dos Resultados **28**
- 5.1 Teste variando o número de camadas escondidas 28
- 5.2 Teste variando as épocas 28
- 5.3 Comparação com a literatura 30

6 – Considerações finais e trabalhos futuros **31**

Referências **33**

Apêndices 35

APÊNDICE A – Descrição do *Million Song dataset* 36

APÊNDICE B – Rótulos Utilizados 39

1 Introdução

Música, segundo Rationality (1999), pode ser definida como sendo "um padrão de sons feito por instrumentos musicais, vozes, computadores ou uma combinação destes, com a intenção de dar prazer às pessoas que estão escutando". Apesar de somente no século passado a música ter se tornado acessível e presente nos diversos contextos, ela já fazia parte da sociedade desde a antiguidade. De fato, cientistas evolucionistas acreditam que a música desempenha um papel importante na vida dos humanos desde a pré-história, devido à sua propriedade de coordenar emoções, facilitar a comunicação de mensagens, além de motivar as pessoas a se identificarem com um determinado grupo. Algumas dessas características citadas podem ser vistas quando analisamos a história da música.

A história da música estuda a sua evolução, bem como o contexto cultural e social de cada época. Dessa forma, a música pode ser dividida em 6 períodos musicais, cada qual com o seu estilo particular. Como mostrado por Estrella (2018 (accessed October 08, 2018)), o primeiro período foi o Medieval, com os cantos gregorianos e os cantos simples. As músicas eram somente permitidas dentro das igrejas e não envolviam instrumentos musicais no acompanhamento da melodia. O segundo período foi o renascimento, que trouxe uma revolução na forma como a música era criada e percebida. Nesse período, eram utilizados instrumentos musicais e até 6 partes de vozes. Um terceiro e importante momento, chamado Barroco, ocorreu quando as melodias passaram a ser acompanhadas por Harmonias. Nessa época, surgiu um estilo que ficou amplamente conhecido, a ópera. O período clássico, ou o quarto período, foi caracterizado por músicas de melodias e formas mais simples como as sonatas. Nessa época, a música se tornou mais popular, com o acesso da classe média, devido a simplicidade das letras e tendo o piano como um dos instrumentos mais utilizados pelos compositores. O período romântico, considerado o quinto, foi caracterizado por músicas que contavam histórias ou tentavam expressar uma ideia. Esse momento teve como destaque o início da utilização de instrumentos de sopro, como a flauta e o saxofone, bem como foi caracterizado por melodias mais complexas e dramáticas. Por fim, o século 20 trouxe muitas inovações na maneira como a música era apresentada e apreciada. Além de experimentar novas formas de fazer música ocorreu, neste período, o uso de tecnologias para melhorar as composições. A música eletrônica e o jazz são bons exemplos que surgiram nesse período.

Pela história da música é possível perceber que diversos estilos ou gêneros foram criados durante o desenvolvimento da sociedade. Além disso, a história mostra diferentes características que marcaram cada um desses estilos, ou seja, a presença de canto, de instrumentos de corda e de sopro, entre outros. Esse tipo de caracterização, chamada de rotulagem musical, também é utilizada atualmente com o intuito de estabelecer uma

determinada classificação para as músicas.

Rótulos musicais são utilizados para fornecer informação de alto nível sobre as músicas, ou seja, informações emocionais (triste, feliz), de gênero (rock, emo) e de instrumentação (guitarra, piano), que facilitem, sem a necessidade de escutá-las individualmente, a escolha dos estilos musicais de preferência. Sendo assim, por facilitar a pesquisa por novas músicas que tenham características similares, esses rótulos também têm sido utilizados pelos sistemas de recomendação.

A tarefa de rotulação de músicas é altamente complexa devido à grande quantidade de características e do grande número de músicas existentes. Dessa forma, o processo de rotulagem é altamente custoso, principalmente, porque é realizado de forma individual ou por um grupo de pessoas. Assim, visando resolver esse problema, surgiu a área de rotulação automática de música, que consiste basicamente na geração de rótulos através da análise dos sinais de áudio. Este processo consiste no treinamento de um classificador por meio de uma base de treino, onde as músicas já foram previamente rotuladas. Após o treinamento, é esperado que o classificador tenha sido capaz de extrair as relações entre músicas e rótulos. Dessa forma, propor uma rotulagem adequada para as músicas que não foram previamente treinadas ou classificadas.

Assim, como o problema de rotulagem de músicas está relacionado a problemas de reconhecimento de padrões e devido ao fato da música também poder ser vista como uma informação temporal, esse trabalho utilizará, como classificador, um tipo especial de rede neural artificial recorrente, a LSTM (*Long Short Term Memory*), que possui grande capacidade de aprender padrões temporais.

1.1 Redes Neurais Artificiais

O estudo sobre neurocomputação teve seu princípio no trabalho desenvolvido por McCulloch e Pitts (1943). Neste trabalho, os autores mostraram que mesmo redes neurais simples podiam, na teoria, computar qualquer função aritmética ou lógica. Apesar da proposta e da grande popularidade do artigo, foi somente em 1957 que surgiu o primeiro neuro-computador (Mark I) de sucesso (IBMCORPORATION, 2015 (accessed June 28, 2018)). Este computador, criado por Frank Rosenblatt, Chanles Wigtman entre outros, foi utilizado inicialmente no reconhecimento de padrões. Segundo o pesquisador da Universidade de Helsinki, Teuvo Kohonen, uma rede neural artificial é definida como: "Uma rede massivamente paralela de elementos interconectados e suas organizações hierárquicas que estão preparadas para iterar com objetos do mundo real do mesmo modo que um sistema nervoso biológico faz".

As principais características, segundo Zambiasi (2002), dessas redes são:

1 Capacidade de 'aprender' através de exemplos e de generalizar este aprendizado de forma a reconhecer elementos similares, que não foram apresentados no conjunto de exemplos (treinamento);
2 Bom desempenho em tarefas pouco ou mal definidas, onde falta o conhecimento explícito de como resolvê-las;
3 Robustez à presença de informações falsas ou ausentes;
4 Pode fornecer informações sobre quais padrões selecionar em função do grau de confiança apresentado;
5 Tolerância à falha.

1.2 *Long Short Term Memory*

Long Short Term Memory (LSTM) é um modelo de redes neurais artificiais que leva em consideração a informação no decorrer do tempo. Em (GERS; SCHMIDHUBER; CUMMINS, 1999), os autores dizem que a LSTM não falha em aprender padrões com diferenças de tempo maiores que 5 - 10 divisões de tempo. Além disso, esse trabalho também mostra que a LSTM pode aprender relações com mais de 1000 divisões de tempo discreto aplicando fluxo de erro constante através de "carrosséis de erro constante"(CECs), dentro de unidades espaciais chamadas células. O algoritmo de aprendizagem da LSTM ocorre no espaço e no tempo e sua complexidade computacional por passo e por peso são de O(1).

1.3 Motivação e Justificativa

A rotulagem de músicas é um processo realizado, majoritariamente, rotulando os artistas. No entanto, devido à enorme quantidade de variantes dos estilos musicais, o processo de rotulagem feita dessa forma se mostrou ineficiente. Vários artistas, apesar de terem um estilo preferencial, ao longo de suas carreiras, acabam por trafegar por diferentes gêneros. Além disso, devido a grande complexidade e subjetividade dessa tarefa, muitas informações que poderiam contribuir na definição dos estilos musicais, acabam sendo negligenciados.

Dessa forma, automatizar este processo surge como uma nova maneira de se identificar os estilos musicais. Ou seja, com a automatização, o processo muda da avaliação pessoal, feita por um determinado especialista, para uma avaliação que é realizada sobre a música propriamente dita. Isto se justifica devido ao número atual de produções e lançamentos, o que torna inviável, em termos técnicos e financeiros, a avaliação feita por profissionais. Além disso, o processo automático pode apresentar uma maior consistência, por utilizar diversas características da música, diminuindo a subjetividade das atribuições.

A geração de rótulos mais completos são úteis na pesquisa por músicas em um determinado repositório, por permitir a filtragem por características. Dessa forma, os sistemas de recomendação são beneficiados quando a identificação das relações entre as produções, bem como das relações entre usuários e músicas, são melhoradas.

Este trabalho, portanto, irá comparar o método de LSTM com outros métodos propostos na literatura, apresentando suas vantagens e desvantagens na solução do problema de rotulagem automática. As redes LSTM são redes que foram projetadas para lidarem com dados temporais cuja a ordem e a sucessão da informação influenciam no resultado final gerado pela mesma. Essa abordagem se justifica pela declaração de Morais (2012): "Música é a arte de combinar os sons simultaneamente e sucessivamente, com ordem, equilíbrio e proporção dentro do tempo".

1.4 Objetivos

Essa seção irá apresentar os objetivos gerais e específicos do trabalho.

1.4.1 Objetivos Gerais

Conhecido o problema de rotulagem de música, o presente trabalho tem como objetivo geral elaborar um algoritmo capaz de aprender os padrões musicais existentes e gerar automaticamente rótulos para músicas ainda não rotuladas.

1.4.2 Objetivos Específicos

Esse trabalho terá como objetivos específicos:

1. Desenvolver um algoritmo para extrair da base de dados de músicas as principais características a serem utilizadas nesse trabalho;
2. Desenvolver um modelo de análise musical que seja independente do tamanho das músicas;
3. Obter um classificador, baseado em LSTM, que faça predição de rótulos baseado em dados reais de músicas;
4. Analisar a premissa de que redes mais complexas fornecem melhores resultados.

1.5 Organização do Trabalho

Este trabalho está organizado seguindo a seguinte divisão: No Capítulo 2 são apresentados e explicados os trabalhos relacionados ao tema dessa pesquisa. O Capítulo 3 expõe os fundamentos teórico-conceituais importantes para o entendimento do trabalho. O Capítulo 4 mostra a metodologia utilizada no desenvolvimento desse trabalho. Os expe-

rimentos realizados são, por sua vez, mostrados no Capítulo 5. Finalmente, o Capítulo 6 apresenta a conclusão deste trabalho, bem como propostas de continuidade.

2 Trabalhos Relacionados

Neste capítulo serão discutidos alguns dos trabalhos encontrados na literatura científica que possuem relação com o presente projeto de modo a salientar o estado da arte sobre o tema abordado.

O algoritmo proposto por Tingle, Kim e Turnbull (2010) consiste em aprender um modelo de distribuição de mistura de gaussianas (*Gaussian Mixture Model* - GMM) nas características de espaço de áudio. O funcionamento consiste, primeiramente, em executar o algoritmo *expectation-maximization* (EM) para aprender os parâmetros do modelo de mistura de gaussianas para cada música no conjunto de treino. Depois do treinamento, para cada rótulo, é usado o *Mixture Hierarchies EM* para combinar os GMMs de cada música que conseguiu um resultado correto no treino. Em seguida, é feita a predição dos rótulos pertencentes a cada música do conjunto de teste. Como cada GMM específico é relativo à apenas um rótulo, é necessário que sejam executados todos os GMMs para todas as músicas. A saída do algoritmo é uma representação multinominal do vocabulário para aquela música.

O modelo proposto por Mandel e Ellis (2008) é um classificador *Support Vector Machine* (SVM) para rotulação automática de músicas (*music autotagging*). Assim, como tipicamente é usado o SVM em problemas de multiclasses, o algoritmo divide um modelo SVM para cada termo, definindo se a música possui ou não aquele rótulo. Após treinar cada modelo SVM com músicas que possuem e não possuem um rótulo, utiliza-se o algoritmo de *Platt scaling* para aproximar a probabilidade de pertinência do termo baseado na distância do ponto da música no hiperplano até a linha de separação de classes. Diferentes classes de SVM podem ser utilizadas no algoritmo. Os autores utilizaram em sua comparação o modelo linear e a função de base radial (*radial basis function*) como função de Kernel.

Eck et al. (2008) propuseram um modelo de *Boosted Decision Stump* (BDS). O algoritmo *Decision Stump* constrói uma árvore de decisão com apenas uma característica e um limite, ou seja, se o valor for abaixo do limite ele irá para uma classe e se for maior ou igual irá para a outra. O modelo *Boosted Decision Stumps*, por sua vez, é uma versão deste algoritmo que gera vários *decision stump* e escolhe a classe por um método de votação. Utilizando o *AdaBoost* cria-se iterativamente um *decision stump* que minimiza o erro de classificação da base de treino. Por esse método não resolver problemas multiclasses é necessário que se utilize da técnica um contra todos, sendo um classificador para cada rótulo. É possível utilizar *Platt Scaling* para gerar probabilidades de pertinência ao invés de uma decisão binária. Ao final, foi criado um modelo BDS que relacionou as músicas e termos, gerando um *Platt Scaling* que determinou as probabilidades de cada rótulo para

cada música.

Tingle, Kim e Turnbull (2010) compararam três métodos, GMM,SVM e BDS, utilizando uma base de dados chamada *Swat10k* e o *Echo Nest Music Api*. A base possui um conjunto de músicas com ID e suas devidas rotulações, enquanto a API utilizada possui um conjunto de características de áudio que foram extraídas de várias músicas e de seus respectivos IDs. O autor relaciona os IDs das músicas e da API (*Application Programming Interface*) para obter as características musicais que são utilizadas como entrada dos algoritmos. O algoritmo que obteve o melhor resultado foi o GMM utilizando as características *Echo Nest Timbre* (ENT). Este algoritmo foi significativamente melhor que os outros dois que utilizaram as características *Echo Nest Song* (ENS). Ambos os tipos de características foram descritos no artigo e representam as músicas de maneira diferente, de forma que o seu uso como entrada para alguns algoritmos não fazem sentido sem o devido tratamento, justificando, desta forma, o uso de tipos diferentes de entrada entre os algoritmos.

Choi, Fazekas e Sandler (2016) propuseram um algoritmo de rotulação automática baseado em Redes Completamente Convolucionais (*Fully Convolutional Networks* - FCN). Estas redes são a *deep convolutional networks* que possuem apenas as camadas convolucionais. Esse algoritmo maximiza as vantagens das redes convolucionais, pois reduz o número de parâmetros através do compartilhamento de pesos e faz com que as características aprendidas sejam invariante à localização no plano de espectrogramas tempo-frequência. Este algoritmo, por exemplo, possui vantagens sobre características geradas manualmente ou estatisticamente agregadas, pois ele permite a rede modelar as estruturas harmônicas e temporais dos sinais de áudio. Na arquitetura proposta pelos autores, são utilizadas de três a sete camadas convolucionais combinadas com camadas de subamostragem (*subsampling*), resultando em uma redução do tamanho do mapeamento de características para $1x1$ e fazendo todo o processo completamente convolucional. Em sequência é utilizado *kernels* convolucionais $2D$ para considerar as relações harmônicas locais.

Para avaliar o funcionamento do algoritmo proposto foram utilizadas as bases de dados *MagnaTagATune* e a *Million Song Dataset* (MSD) (BERTIN-MAHIEUX et al., 2011a). Para o caso da *MagnaTagATune* foram feitas duas comparações, a primeira comparou o funcionamento do próprio algoritmo, sendo a principal diferença o tipo de característica de entrada utilizado. As condições testadas foram:

a Redes convolucionais variando de três a cinco camadas e utilizando como entrada o *mel-spectogram*;
b Redes convolucionais com quatro camadas e a entrada sendo STFT (*Short Time Fourier Transform*);
c Redes convolucionais com quatro camadas e a entrada sendo MFCC (*Mel-frequency cepstral coefficients*);

Nesse último caso, houve uma mudança na arquitetura, pois foram utilizadas quatro camadas *feed-foward* ao invés de $2D$ e *pooling*. A métrica de comparação foi AUC (*Area Under ROC*). O experimento mostrou que as estruturas utilizando quatro e cinco camadas convolucionais e entrada *mel-spectogram* obtiveram melhor desempenho, tendo a rede de quatro camadas alcançado um melhor resultado. Todas as outras estruturas apresentaram resultados significativamente piores. O segundo experimento comparou a FNC com quatro camadas convolucionais com algoritmos presentes na literatura, utilizando também a AUC como métrica de avaliação. Os autores consideraram que os algoritmos foram competitivos, pois seus valores eram muito similares. Apesar do algoritmo proposto ter obtido um desempenho um pouco maior que os demais, não é possível afirmar a sua superioridade, uma vez que os resultados obtidos estavam bem próximos.

O segundo teste levou em consideração apenas os algoritmos propostos no artigo, pois na época a base de dados era nova e ainda não existiam outros trabalhos com resultados publicados para que fosse realizada uma comparação mais efetiva. Foi utilizada, nesse experimento, a base de dados do MSD juntamente com os rótulos disponibilizados pela base de dados do *Last.fm*. Foram selecionados os 50 rótulos mais frequentes e os testes realizados utilizaram entre três e sete camadas convolucionais tendo a AUC como métrica. Os resultados mostraram que a rede com 6 camadas convolucionais superou as outras arquiteturas, mas a diferença para o caso com apenas 5 foi de apenas 0,003 AUC. A conclusão do teste foi mostrar que redes com maior número de camadas conseguem se aproveitar melhor da grande quantidade de dados em relação às redes mais simples e menos profundas.

Choi et al. (2017) propuseram um rede neural recorrente convolucional (CRNN) para rotulação automática de músicas. A parte convolucional da rede é utilizada para extração de características locais enquanto a parte recorrente é utilizada para sumarização temporal das características extraídas. Nesse trabalho, foram utilizados três tipos diferentes de redes convolucionais (K1C2, K2C1, K2C2) com entradas de tamanho 96x1366 (*mel-frequency band x time frame*) e canal único. A rede K1C2 possui 4 camadas de rede convolucional seguidas de 2 camadas totalmente conectadas, as camadas convolucionais alternam entre 1D e *max-pooling*. Cada elemento da saída da quarta camada convolucional codifica uma característica para cada banda. Elas alimentam as camadas completamente conectadas, que agem como classificador. A rede K2C1 possui 5 camadas convolucionais que são seguidas por 2 camadas completamente conectadas. A primeira camada convolucional (96x4) utiliza um *kernel* 2D que é aplicado sobre toda banda de frequência. Na sequência, é criada, de forma alternada, camadas convolucionais de uma dimensão e *max-pooling*. A saída da última camada da rede convolucional alimenta as camadas completamente conectadas que atuam como classificador. A rede K2C2 possui cinco camadas convolucionais com *kernels* 3x3 e *max-pooling*. Essa rede reduz o tamanho das características mapeadas para 1x1 na camada final, uma vez que cada característica cobre toda a entrada ao invés de cada

banda de frequência, como ocorre em K1C1 e K2C2. As CRNN podem ser descritas como uma rede neural convolucional modificada de forma que a última camada convolucional é substituida por camadas de redes neurais recorrentes. Nessa arquitetura, são utilizadas uma camada convolucional 3x3 e 4 camadas de *max-pooling*. A saída das camadas convolucionais alimenta duas camadas de rede recorrente que utiliza *gated recurrent units* para sumarizar padrões temporais.

Os testes feitos utilizaram o *Million Song Dataset* com os rótulos extraídos do *last.fm*. As redes foram treinadas para prever os 50 rótulos mais frequentes, utilizando trechos de música que variam de 30 a 60 segundos. Entretanto, para serem utilizados como entrada do algoritmo, foram utilizados somente 29 segundos de cada música, bem como foram realizadas algumas transformações. Os autores concluiram que a rede proposta CRNN apresentou resultados comparáveis às da rede K2C2, mas teve um melhor desempenho para um mesmo número de parâmetros. A K2C2, por sua vez, foi mais rápida.

2.0.1 *Base de Rótulos*

O interesse crescente por rotulagem automática de música levou a comunidade a criar bases de dados com as informações necessárias para que pudessem ser desenvolvidos, testados e comparados algoritmos de rotulagem automática.

O primeiro *dataset* a ser utilizado amplamente pela comunidade foi o CAL500 e é constituído por 500 músicas de artistas diferentes. Nessa base, cada música recebeu um rótulo de 3 estudantes utilizando 174 termos possíveis. Mesmo tendo sido amplamente usada, essa base possui poucos dados, dificultando a utilização de algoritmos de aprendizado, que normalmente necessitam de uma base mais ampla para que possam extrair os padrões existentes. Tingle, Kim e Turnbull (2010) apontam como problema também a subjetividade dos termos utilizados.

Law e Ahn (2009) apresentaram o MagnaTagTune, uma base de dados que consiste em 6362 músicas de 269 artistas diferentes. Utilizando 188 diferentes termos, as músicas foram rotuladas por usuários utilizando um *music annotation* chamado *Tagatune*. *Music annotation* é um programa em que os jogadores descrevem a música, utilizando rótulos, durante o curso do jogo. Para um rótulo ser considerado válido dois jogadores devem, independentemente, entrar com ele quando estiverem ouvindo a mesma música. Tingle, Kim e Turnbull (2010) analisaram que um dos maiores problemas deste *dataset* é a simplicidade dos termos, pois eles são colocados no decorrer do jogo de forma a encorajar a utilização de palavras menores. Outro problema, é a baixa variedade de artistas, visto que as músicas dos mesmos artistas tendem a ter sonoridades similares, não refletindo a realidade dos sistemas nos quais *autotagging* deve ser usado.

Tingle, Kim e Turnbull (2010) citam que uma das melhores fontes para ser utilizada

como base para treinamento de algoritmos de rotulagem é o Last.fm. Esta base, utiliza um sistema de rótulo social, que pode ser adicionada à música pelos usuários. Por ser um site com muitos usuários, sua base de dados possui um grande número de músicas e artistas, perdendo, no entanto, em relação à subjetividade dos termos. Isso ocorre, pois os usuários comumente não são treinados e não conhecem termos técnicos para descrever uma música, utilizando termos como feliz ou triste, por exemplo. Essas descrições, as vezes são muito subjetivas e são mais difíceis de se prever do que informações técnicas, como gênero ou atributos da música.

3 Fundamentação Teórica

Nesse capítulo serão discutidos conceitos fundamentais para um melhor entendimento deste trabalho. A primeira subseção descreve os dados que serão utilizados como entrada do algoritmo, mostrando quais informações estão presentes, bem como seu significado. A subseção seguinte descreve os fundamentos das redes recorrentes, enquanto a última subseção descreve a rede LSTM.

3.1 Características musicais

As músicas podem ser analisadas de diversas formas, pois a maneira como os humanos escutam está relacionada tanto à parte física quanto a percepção subjetiva do som. Essa relação permite que sejam extraídas características diversas, cada qual com um significado e uma importância própria. Assim, mesmo que se possa compreender o significado de cada uma dessas características em separado, a música não pode ser analisada de forma segmentada, mas como uma composição de todos esses elementos. Dentre as diversas características existentes, serão destacadas somente aquelas relativas ao trabalho, quais sejam (Jehan e DesRoches (2011)):

- *Onset*: início de uma nota musical ou de um som.
- *Tatum*: representa o trem de pulso regular mais baixo que um ouvinte intuitivamente infere do tempo de eventos musicais percebidos (segmentos). É aproximadamente equivalente à divisão de tempo que mais coincide com os *Onsets*.
- Batida: unidade básica de tempo de uma peça de música, por exemplo, a cada *tick* de um metrônomo. Sendo assim, é o padrão rítmico regular da música. São, basicamente, múltiplos de *tatums*.
- Tempo: Batidas por minuto (BPM). Na terminologia musical, tempo é a velocidade ou ritmo de uma determinada peça e deriva diretamente da duração média da batida.
- Duração: Duração da música em segundos.
- *Fade*: Aumento ou diminuição gradual no nível de sinal de um áudio. Quando uma música é reduzida gradualmente até o silêncio, o seu final é chamado de *Fade-out* (Figura 1) e quando aumenta gradativamente a partir do silêncio, o seu começo é chamado de *Fade-in* (Figura 2).
- Barra: uma barra (ou medida) é um segmento de tempo definido como um determinado número de batidas. Dividir a música em barras fornecem pontos de referência regulares para identificar locais dentro de uma música. Também é utilizado para tornar a escrita mais fácil de seguir (Figura 3), uma vez que cada barra de símbolos pode ser lida e reproduzida como conjunto. Deslocamentos de barra também indicam

downbeats, ou seja, a primeira batida da medida.

- Assinatura de Tempo: Tempo estimado global de uma faixa. A assinatura de tempo é uma convenção de notação para especificar quantas batidas estão em cada barra (ou medida).
- Chave: identifica a tríade tônica, o acorde, maior ou menor, que representa o ponto final de descanso de uma peça.
- Volume: atributo da sensação auditiva que indica quais sons podem ser ordenados em uma escala que vai de silenciosa a alta. A relação dos atributos físicos do som com a percepção do volume consiste em componentes físicos, fisiológicos e psicológicos. O estudo do volume aparente está incluído no tópico da psicoacústica e emprega métodos da psicofísica.
- Tom: propriedade perceptual dos sons que permite a sua ordenação em uma escala relacionada com a frequência. Tom é a qualidade que permite julgar os sons como superiores ou inferiores no sentido associado a melodias musicais e só pode ser determinado em sons que tenham uma frequência clara e estável para distinguir o ruído.
- Timbre: também conhecido como tonalidade ou qualidade sonora da psicoacústica. É a qualidade sonora percebida de uma nota musical, som ou tom. O timbre distingue diferentes tipos de produção de som, como vozes de um coro, instrumentos musicais, instrumentos de corda, de sopro e de percurssão. Também permite que os ouvintes distingam diferentes instrumentos na mesma categoria. As características do som que determinam a percepção do timbre incluem espectro e envelope.
- Modo: qualquer uma das várias maneiras de ordenar as notas de uma escala de acordo com os intervalos que elas formam com a tônica, fornecendo assim uma estrutura teórica para a melodia. Um modo é o vocabulário de uma melodia; especifica quais notas podem ser usadas e indica quais delas têm importância especial. Destas, há duas notas principais: a final, na qual a melodia termina, e a dominante, que é o centro secundário. A Figura 4 mostra o modo Ionian, atualmente conhecido como a escala maior, composta por notas naturais começando em C.
- *Sections*: As seções são divididas a cada *Onset* e cada uma contém suas próprias descrições de tempo - modo, assinatura de tempo e volume sonoro.

Além das características em si, ainda existem formas de se realizar tranformações em algumas dessas informações para que possam ser computacionalmente trabalhadas. Na seção de trabalhos relacionados, foram mostradas pesquisas que utilizaram Redes Convolucionais em cima de tranformações das características para gerar diferentes tipos de espectrograma. As transformações citadas podem ser entendidas como:

- *Spectogram*: Um espectrograma é uma representação visual do espectro de frequências de um som, ou outro sinal, a medida que eles variam no tempo. A Figura 5 mostra um exemplo de espectrograma.

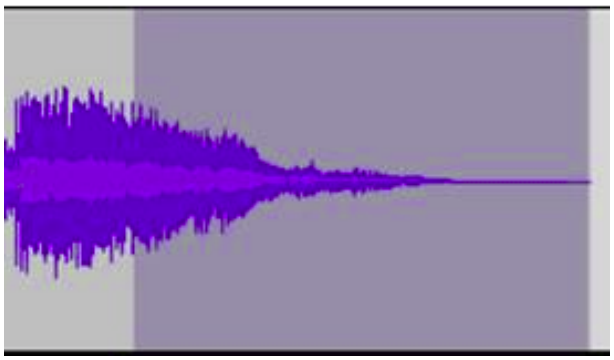


Figura 1 – Exemplo de *Fade-out*

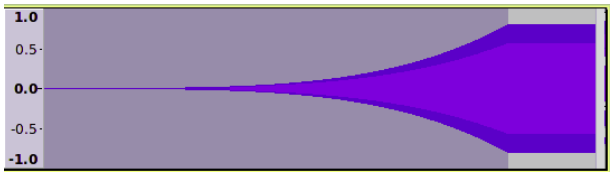


Figura 2 – Exemplo de *Fade-in*

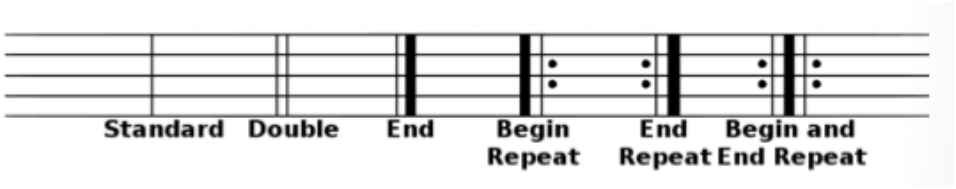


Figura 3 – Diferentes de tipos de barras utlizadas para escrita

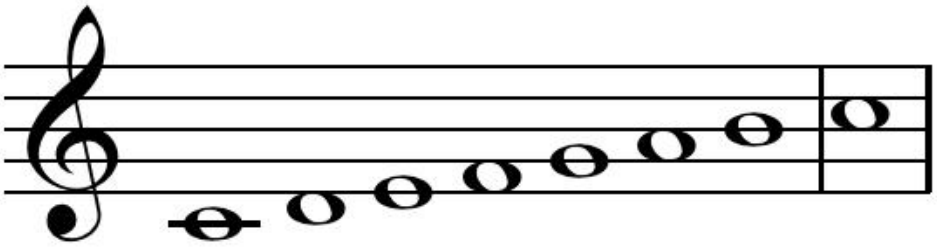


Figura 4 – Modo Ionian

- *Mel Scale*: A escala Mel vem de *melody*, melodia em inglês, e é uma escala de comparação de tons. Xu et al. (2004) mostraram a fórmula para converter a frequência (hertz) em mels: $m = 2595log_{10}(1 + \frac{f}{700})$, e explicam que, ela é uma distribuição linear nos alcances menores e logarítmica nos alcances maiores, similiar às características fisiológicas do ouvido humano.
- *Mel-frequency cepstrum* (MFC): Uma representação do espectro de potência de curto prazo de um som. É baseado em uma transformação cosseno linear do logaritmo do espectro de potência em uma escala mel.

- *Mel-frequency cepstral coefficients* (MFCCS): Sahidullah e Saha (2012) resumem essa característica como sendo um processo de filtragem dos sons por uma série de filtros triangulares, representados na escala Mel.
- *Short Time Fourier Transform* (STFT): Uma transformada relacionada a Fourier usada para determinar a frequência sinusoidal e o conteúdo de fase de seções locais de um sinal, conforme ele muda ao longo do tempo. O procedimento para calcular os STFTs é dividir um sinal de tempo mais longo em segmentos mais curtos de igual comprimento e depois calcular a transformada de Fourier separadamente em cada segmento mais curto. Isso revela o espectro de Fourier em cada segmento mais curto.

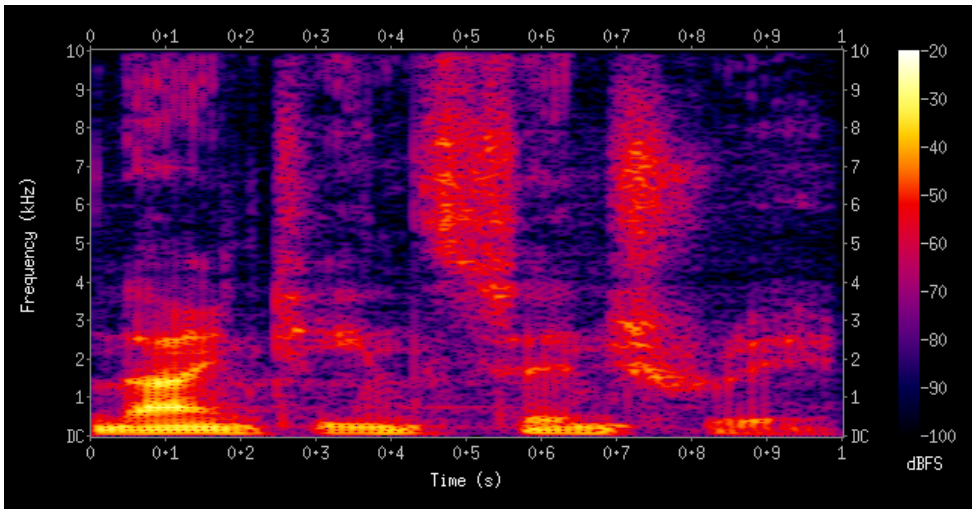


Figura 5 – Exemplo de espectrograma

3.2 Redes Neurais Recorrentes

O tutorial BRITZ (2015 (accessed May 10, 2018)) mostra o que são e como funcionam as Redes Neurais Recorrentes. Redes neurais tradicionais não possuem a capacidade de tratar informações sequenciais, pois tratam as entradas e saídas como se fossem independentes. Essa forma de tratar os dados muitas vezes é ineficiente, pois, por exemplo, em predição de palavras em uma frase é necessário saber o conjunto de palavras antecedentes. Por outro lado, as redes recorrentes foram projetadas de forma a utilizar as informações passadas, ou seja, a saída da rede depende de computações anteriores. A Figura 6 mostra uma célula de rede recorrente no qual o *loop* que retroalimenta a rede tem a funcionalidade de passar a informação computada em um passo de tempo para o próximo. A Figura 7 mostra o *loop* da rede recorrente no tempo, ou seja, está sendo mostrada uma mesma célula em diferentes passos de tempo. Dessa forma, torna-se possível visualizar o fluxo temporal de dados.

Por exemplo se estamos considerando uma sequência de 5 valores ou uma frase de 5 palavras, a rede será composta de 5 camadas, sendo uma camada para cada palavra.

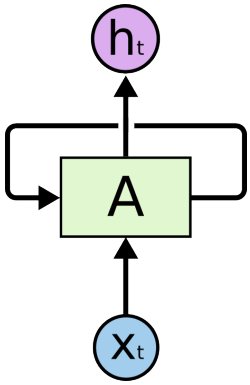


Figura 6 – Célula da rede recorrente

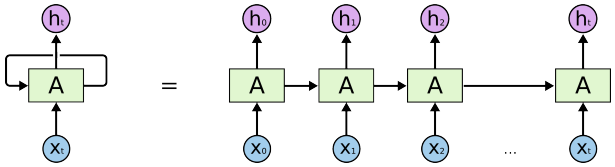


Figura 7 – Célula da rede recorrente em diferentes passos de tempo

Levando em consideração o esquema da Figura 8 a computação que ocorre na rede neural recorrente (RNN) é feita como a seguir:

- x_t é a entrada para o tempo t. Por exemplo, x_0 poderia ser o primeiro valor da sequência ou a primeira palavra de uma frase enquanto x_1 seria a segunda palavra.
- s_t é a parte escondida da rede no tempo t. Ela pode ser entendida como a memória da rede, pois s_t é calculada baseada nos estados escondidos anteriores seguindo a fórmula:

  $$s_t = f(Ux_t + Ws_{t-1}) \tag{1}$$

  A função f normalmente é não linear tais como tangente hiperbólica (*tanh*) ou Rectified Linear Unit (ReLU). O s_{-1}, que é utilizado para computar o primeiro estado escondido, normalmente é inicializado com todos os valores 0.
- o_t é a saída no tempo t. Por exemplo, no caso da frase, se quisermos prever a próxima palavra em uma frase, o_t seria um vetor com a probabilidade de cada palavra. Nesse caso, a função de ativação utilizada seria o *softmax*. Vale ressaltar que a função de ativação utilizada nessa parte é dependente do que se espera da saída, de forma que normalmente é utilizada uma função sigmoidal para classificação, *softmax* para previsão e *Relu* para regressão.

É interessante notar que o_t é calculado utilizando apenas o estado de s_t. Entretanto, esse estado reflete as informações capturadas dos estados passados. Também é importante ressaltar que nos diagramas mostrados, cada passo de tempo gera uma saída. No entanto, isso nem sempre é necessário, uma vez que é possível adaptar a rede para levar em consideração apenas os últimos resultados. Por exemplo, ao tentar identificar qual o tema

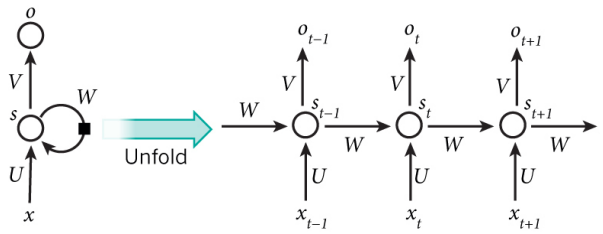


Figura 8 – Estrutura detalhada da célula da rede recorrente

do texto, não é necessário que, a cada palavra, seja nos dada uma resposta, pois espera-se apenas o resultado final.

As redes recorrentes possuem um problema na forma como foram arquitetadas e, por isso, normalmente não conseguem capturar informações de muitos passos de tempo anteriores, como mostrado por Bengio, Simard e Frasconi (1994). Sendo assim, foram desenvolvidas alterações dessa estrutura, principalmente a arquitetura da rede LSTM, com o objetivo de solucionar esse problema.

3.3 Long Short-Term Memory

As redes neurais conhecidas como *Long Short-Term Memory* (LSTM) são um tipo especial de RNN adaptada para aprender dependências de longo termo. Elas foram introduzidas por Hochreiter e Schmidhuber (1997), que mostram que uma das propriedades da LSTM é ter processamento local no espaço e no tempo. Ser local no espaço significa que a complexidade por passo de tempo e peso do LSTM independe do tamanho da rede. Ser local no tempo indica que a atualização é feita continuamente. Desta forma, essas redes não necessitam de múltiplos passos de tempo, dependendo apenas do passo mais recente. Esses dois atributos foram possíveis, pois apesar do LSTM seguir o mesmo princípio de repetição da rede recorrente, o funcionamento interno da célula foi modificado. A célula possui uma estrutura mais complexa, pois é composta de quatro partes que interagem de forma específica no lugar da função de ativação *tangh* ou *ReLu* das Redes Recorrentes.

Olah (2015 (accessed May 10, 2018)) explicou a estrutura do LSTM, bem como seu funcionamento. As informações seguintes são baseadas neste tutorial.

O diagrama apresentado na Figura 9 mostra a estrutura interna de uma rede LSTM. Nesse diagrama, os círculos rosa são operações pontuais, tais como adição vetorial. Os retângulos amarelos são, por sua vez, camadas de ativação em que são aprendidos os pesos. Quando uma linha se separa e se torna duas, significa que o conteúdo está sendo copiado e sendo utilizado em dois locais diferentes. No caso em que duas linhas se juntam em uma única linha, significa que o conteúdo está sendo concatenado.

O estado da célula, uma das principais partes do LSTM, é a linha horizontal que

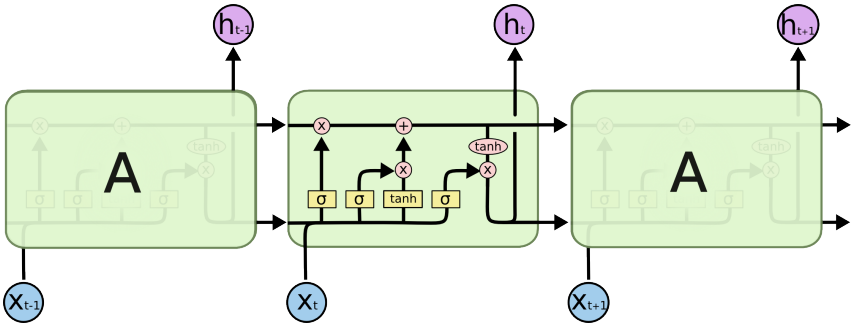


Figura 9 – Estrutura detalhada da célula LSTM

corta a parte superior da célula mostrada na Figura 10. O estado da célula atravessa a célula com apenas poucas interações lineares, de forma que é possível que a informação passe pela célula sem alterações. O LSTM consegue, como foi dito, fazer alterações nas informações do estado de célula, sendo reguladas pelos chamados portões. Estes portões são compostos por uma camada de rede neural com a função de ativação sigmoidal e uma operação de multiplicação pontual. A saída da função sigmoidal é um valor entre 0 e 1 que indicando quanto de cada componente deve passar. Sendo assim, quando o valor for 0 significará que nenhuma das informações será passada e quando for 1 todas as informações irão para frente.

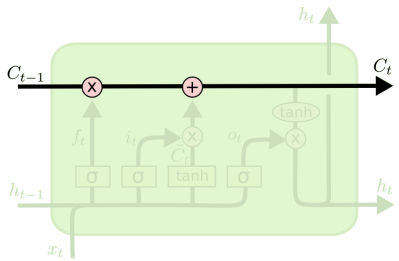
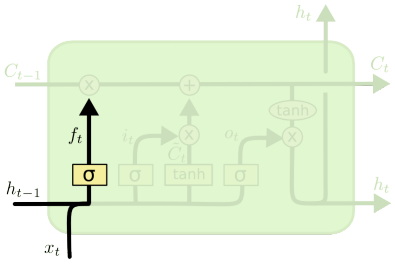


Figura 10 – Estado da célula rede LSTM

A primeira parte da rede LSTM analisa quanto da informação recebida do estado de célula no instante anterior será utilizado. Essa decisão é feita pela primeira camada sigmoidal chamada 'Camada do portão do esquecimento' (Figura 11). Esse portão tem como entrada H_{t-1} e X_t, que correspondem à saída do passo anterior e a entrada do passo atual. Dessa forma, a saída F_t utiliza essas duas informações, que são valores entre 0 e 1 referentes a cada valor do estado de célula, definindo a quantidade de informação passada que será mantida nos estágios seguintes.

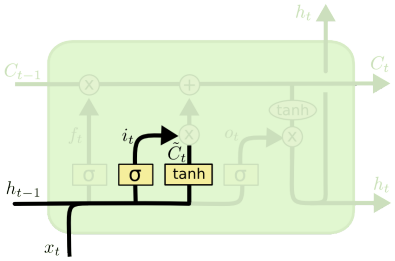
A parte seguinte do LSTM decide qual informação da entrada deverá ser adicionada ao estado de célula. Primeiro, a camada sigmoidal chamada de 'Camada do portão de entrada' (Figura 12) decide quais valores serão atualizados, tendo como base os valores da saída do passo anterior combinado com a entrada atual. A função *tangh*, em seguida, constroi novos valores candidatos, C_t, que podem ser combinados com a saída da camada



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figura 11 – Camada do portão do esquecimento

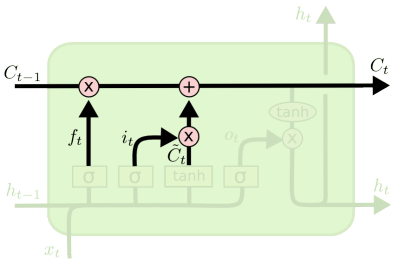
de portão de entrada. Na próxima ação, esses dois valores são combinados, por uma operação de multiplicação pontual, para criar um estado de atualização.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figura 12 – Camada do portão de entrada

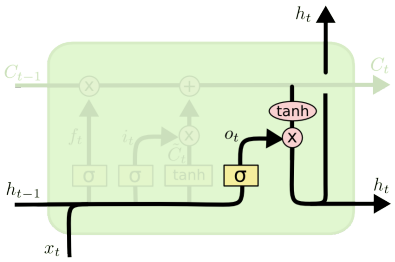
Por fim, o estado de célula antigo de C_{t-1} é atualizado para o novo estado de célula C_t. Isso será feito combinando as saídas que foram geradas nos passos anteriores. O estado anterior C_{t-1} é multiplicado por F_t e depois é adicionado o valor relativo ao estado de atualização. Desta forma, são adicionadas as informações anteriores que se deseja manter com as informações novas que se pretende atualizar (Figura 13).



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figura 13 – Camada de atualização

A saída desta célula, nesse passo de tempo, será uma versão filtrada do seu estado de célula (Figura 14). Primeiro, utiliza-se H_{t-1} e X_t na função sigmoidal para decidir quais partes do estado de célula formarão a saída. Depois, o estado de célula atual, C_t, é passado por uma função de *tanh* que irá forçar os valores a ficarem entre -1 e 1, multiplicando, logo após, pelo resultado da camada sigmoidal. Dessa forma, é gerado o H_t que é a saída desse passo de tempo.



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figura 14 – Camada do portão de saída

4 Metodologia

Esse trabalho compreende uma pesquisa experimental aplicada na área de inteligência artificial voltada para música. A metodologia proposta consiste nas etapas mostradas na Figura 15 e são descritas a seguir:

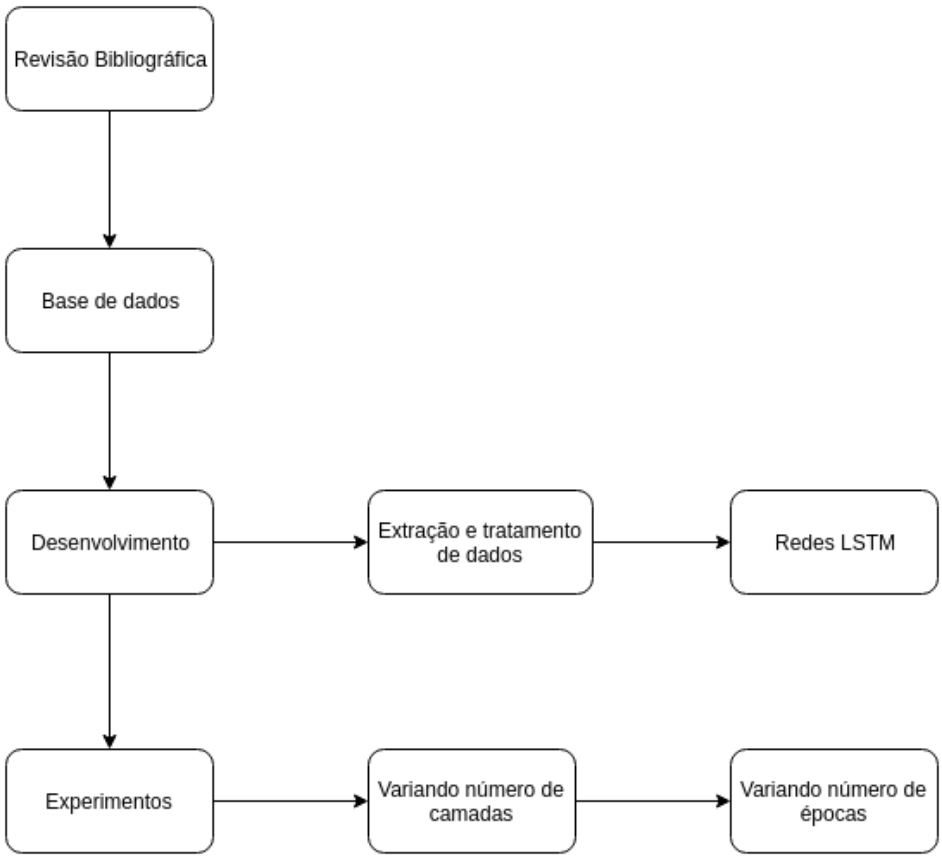


Figura 15 – Fluxograma da metodologia proposta

4.1 *Revisão bibliográfica*

Para ter um melhor entendimento do problema e das possíveis formas de resolvê-lo foi feito um estudo dos artigos que estavam relacionados ao mesmo. Por existir uma grande variedade de trabalhos na área, apenas aqueles que apresentavam mais informações e explicavam de forma clara a metodologia foi apresentado no Capítulo 2.

4.2 *Base de dados*

Essa seção apresenta as bases de dados utilizadas como entrada do algoritmo proposto, bem como as principais características extraídas das músicas. Da mesma forma, apresenta as bases de dados de rótulos que são utilizadas no treinamento e teste do algoritmo, mostrando, de forma breve, suas vantagens e desvantagens.

4.2.1 *Million Song Dataset*

A base de dados utilizada como entrada foi a *Million Song Dataset* (MSD) desenvolvida por (BERTIN-MAHIEUX et al., 2011a). Essa base é constituída por um milhão de músicas populares, e contempla tanto informações de metadados quanto características de análise de áudio que foram extraídas utilizando a *Echo Nest API*. Os dados, nessa base, são divididos em arquivos de forma que um arquivo representa uma faixa de música. Faixa, nesse contexto, significa uma versão (lançamento) de uma música feita por um artista, de forma que a mesma música pode ter diferentes versões e ser reproduzida por diversos artistas, mas a faixa é uma versão única. As informações destes quatro itens (faixa de música, música, lançamento, artista) estão em todos os arquivos, apresentando, dessa forma, certa redundância. Entretanto, o maior volume de dados, relativo às análises de áudio, é único. É importante citar que apesar dos dados serem bem completos, alguns campos podem não estar presentes em algumas músicas, como por exemplo, a não existência da localização de todos os artistas. Bertin-Mahieux et al. (2011b) descrevem as características da base de dados e, com base nesse artigo, será revisada as principais informações que serão utilizadas nesse trabalho, quais sejam:

1. 280 GB de dados
2. 1.000.000 Músicas/arquivos
3. 44,745 Artistas únicos
4. 7.643 Termos únicos (Rótulos do *Echo Nest*)
5. 2.321 Rótulos únicos do *musicbrainz*
6. 43.943 Artistas com pelo menos um termo
7. 2.201.916 Relações de similaridade assimétricas
8. 515.576 Faixas de músicas datadas começando de 1922

Os dados são armazenados usando o formato HDF5, pois o mesmo trabalha de forma eficiente com informações heterogêneas, tais como audios com *arrays* de tamanhos distintos, nomes do tipo *string* etc. As principais características acústicas armazenadas nesse formato são o tom, o timbre e o ruído. A *Echo Nest Analyze API*, por sua vez, providencia essas informações para cada 'segmento' usualmente delimitados por *note onset* ou qualquer outra descontinuidade no sinal. *Note onset* refere-se à detecção do instante em que um evento discreto é iniciado em um sinal musical (ZHOU; REISS, 2010).

Cada música tem sua descrição dividida em 56 campos, cada um deles contendo informações referentes a dados ou metadados da música ou do artista. Todos os campos estão descritos no Apêndice A.

A seguir serão descritos apenas os campos utilizados nesse trabalho, quais sejam:

1 *segments loudness max*: representa o valor, em db, do pico de maior intensidade de cada segmento;
2 *segments loudness max time*: descreve o deslocamento dentro do segmento de pontos de intensidade máxima;
3 *segments loudness max start*: valor em dB no *onset*;
4 *segments pitches*: o conteúdo é dado por um vetor *'chroma'*, que corresponde a 11 classes de tom C, C#, D até B. Essas classes possuem valores que variam de 0 a 1 e descrevem a dominância relativa de cada tom na escala cromática. Os vetores são normalizados para 1, considerando a dimensão mais forte. Dessa maneira, sons ruidosos provavelmente serão representados por um vetor com valores próximo de 1, enquanto tons puros serão representados por uma posição com valor próximo a 0;
5 *segments start*: informação de quando começa cada segmento/*onset*;
6 *segments timbre*: timbre é uma qualidade musical que distingue diferentes tipos de instrumentos musicais ou vozes. É uma noção complexa também utilizada para cores, texturas e é derivada do formato de segmento da superfície spectro-temporal, independente do tom ou da sonoridade. Essa característica é um vetor que inclui 11 valores centrados em 0. Esses valores são abstrações de alto nível da superfície espectral, ordenada por grau de importância. Para completar, a primeira dimensão representa a média do ruído do segmento, a segunda representa o brilho, a terceira está mais próximo do nivelamento do som, a quarta representa a força do som etc;
7 *segments confidence*: medida de confiança de um segmento.

A partir dessas 7 características, foram obtidas 27 informações que serão utilizadas como entrada da rede LSTM, visto que o timbre e o tom (*segments timbre* e *segments pitches*) são compostos por 11 valores cada um.

4.2.2 *Base de Rótulos*

A base de rótulos utilizada nesse trabalho teve que ser escolhida de forma separada, pois, apesar de algumas bases de dados já virem com as características das músicas e seus respectivos rótulos, a base MSD apresenta somente as características. No entenato, por ser uma base com grande quantidade de informações musicais, outras bases com informações complementares são usadas e fazem uso dos IDs (identificadores únicos) das músicas para gerarem a combinação. O *Last.fm* e o *Top MAGD dataset* podem ser consideradas bases de dados de referência para o caso de rotulagem. Além dessas bases, existem também aquelas que são utilizadas em sistemas recomendação, como o *Taste*

Profile subset, e aquelas utilizadas para extração da letra da música, como o *musiXmatch dataset*.

Apesar das restrições citadas no Capítulo 2, nesse trabalho é utilizada a base de dados *Last.fm*, por ser bastante extensa e por levar em consideração rótulos subjetivos sobre a descrição da música. Essas informações são interessantes, pois permitem categorizar as músicas, de forma mais completa, do que utilizar apenas o gênero musical.

A base de dados do *Last.fm* consiste nas informações de rótulos e artistas relacionados. Entretanto, as informações sobre os artistas relacionados não serão utilizadas nesse trabalho. A base utilizada possui as seguintes características:

1. 943347 Combinações de dados com o MSD. Ou seja, possui informação de rótulos e/ou artistas similares;
2. 505216 Músicas presentes no MSD com pelo menos um rótulo;
3. 522366 Rótulos únicos;
4. 8598630 Pares ou combinações de músicas com um rótulo.

4.3 Desenvolvimento

O desenvolvimento do algoritmo se baseia, basicamente, na extração e tratamento dos dados, bem como na construção das redes. A extração dos dados é necessária, pois a base de dados possui um conjunto de informação muito maior do que aquele que será utilizado nesse trabalho. Além disso, o tratamento é necessário, devido ao fato da rede utilizada exigir um formato específico.

As redes *LSTM* utilizadas possuem muitos parâmetros que podem ser definidos no momento de sua criação, gerando um grande número de combinações e possibilidades de testes. Com base nisso, alguns destes parâmetros foram escolhidos levando em consideração a natureza do problema, enquanto outros foram alterados de forma empírica.

4.3.1 *Extração e tratamento dos dados*

As informações retiradas da base de dados foram todas aquelas que possuiam dados temporais, não sendo utilizadas informações estáticas, tais como nome do artista etc. As informações utilizadas foram:

1. *segments loudness max time*
2. *segments loudness max start*
3. *segments loudness max start*
4. *segments start*
5. *segments timbre*
6. *segments pitches*

7. *segments confidence*

Essas informações somam 27 vetores, pois o *segments timbre* e o *segments pitches* são compostos de 11 vetores cada um. As características restantes são representadas por apenas um valor.

As músicas na base possuem tamanhos diferentes, devido ao fato das informações estarem sendo extraídas a cada *onset*. Além disso, o *onset* não é uma medida temporal fixa, uma vez que as músicas têm tempos de duração diferentes. Essa diferença de tamanho não é um problema para as redes LTSM, pois elas são locais no tempo. Entretanto, na biblioteca Keras, a rede precisa ser alimentada com uma matriz tridimensional fixa. Dessa forma, não é possível que a matriz tenha profundidade diferente para cada música. Para solucionar esse problema, foi necessário dividir as músicas que possuíam o mesmo tamanho (em *onset*) para alimentar a rede de forma iterativa.

Para fazer esse tratamento, a extração foi dividida em duas partes. No primeiro momento, é montado o dicionário de tamanhos. Isso é feito extraindo os *segments confidence* de todas as músicas, medindo o seu tamanho e verificando na base de saída se a música possui pelo menos um dos 50 rótulos (descritos no Apêndice B). Caso a música não possua rótulo, a mesma é descartada. Assim, todas as músicas não descartadas, são utilizadas para construir um dicionário que relaciona os IDs das músicas com seus respectivos tamanhos, sendo o tamanho a chave do dicionário.

Na segunda parte, é percorrido o dicionário criado, retirando os IDs de músicas com mesmo tamanho. De posse dos IDs, são extraídas as características, concatenando-as em uma matriz tridimensional, em que a profundidade é a informação no tempo, as colunas são características e as linhas são diferentes músicas (Figura 16). Além disso, são extraídos os rótulos e criada uma matriz bidimensional, na qual as linhas representam cada uma das músicas e as colunas a informação dos rótulos registrados para cada uma delas.

A próxima etapa é realizar o treinamento da rede utilizando a matriz tridimensional como entrada e a bidimensional como saída. Após treinada, o melhor modelo de rede, juntamente com a base de dados de teste, é utilizado na fase de validação da rede LSTM. Assim, as predições feitas são concatenadas em uma matriz e, ao final, é medida a eficácia das predições, por meio da métrica *ROC AUC Score*, de duas formas: por amostra e por *micro*, que leva em consideração todas as amostras e o desbalanceamento de classe.

4.3.2 *Rede LSTM*

A rede foi projetada com 27 neurônios de entrada, pois a base de dados possui 27 características numéricas, e 50 neurônios de saída, com função de ativação sigmoidal, cada um correspondendo a um rótulo. A função objetivo *'Approximation to the Wilcoxon-Mann-Whitney Statistic'* foi utilizada para otimizar a aproximação da área sob a curva *ROC*

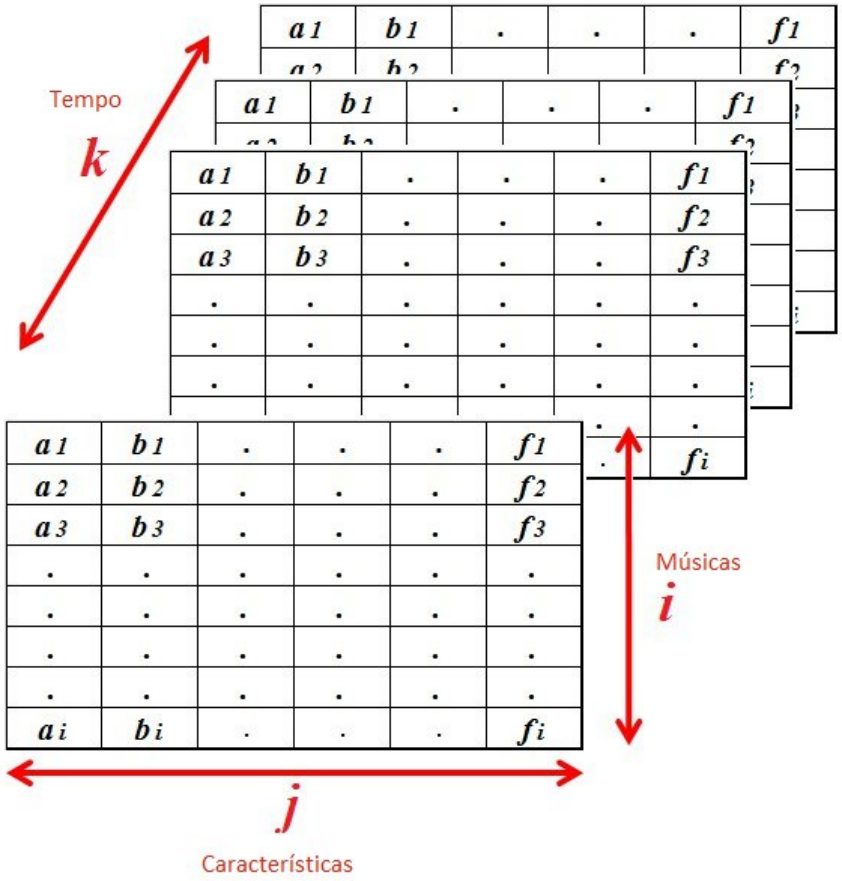


Figura 16 – Organização da matriz de entrada

(*ROC AUC Score*). O otimizador utilizado, por sua vez, foi o *Stochastic gradient descent optimizer* (SGD) com taxa de aprendizado de 0.01, momento igual a 0.9, *Nesterov* ativado e decaimento de aprendizagem igual a 0.

As configurações, que foram fixadas, controlam o funcionamento central da rede. Dessa forma, a rede recebe, para cada passo, as 27 características de cada música. Os 50 neurônios de saída com função sigmoidal fazem com que seja previsto se cada rótulo, separadamente, apareceu ou não. Isso significa que, para cada música, ao fim de todos os passos de tempo, serão previstos todos os rótulos que ela possui, podendo ser um ou mais. Isso é necessário, pois na base uma mesma música pode possuir vários rótulos, como por exemplo, uma mesma música pode ter os seguintes rótulos: Rock, *Love* e *male vocalists*, mostrando que o seu gênero é rock, o tema seria amor e que o vocalista é homem.

Utilizar a *'Approximation to the Wilcoxon-Mann-Whitney Statistic'* como função objetivo significa que para cada época a rede estará tentando minimizar a função, o que reflete na maximização dos resultados obtidos pela métrica *ROC AUC Score*. Fawcett (2006) diz em seu trabalho que "o AUC de um classificador é equivalente à probabilidade de um classificador classificar uma amostra, aleatoriamente selecionada, como uma instância positiva maior que uma instância negativa". Dessa forma, o método verifica o desempenho do

modelo levando em consideração a taxa de falsos e verdadeiros positivos. Isso é importante, pois normalmente as músicas possuem poucos rótulos e o objetivo do modelo é prever corretamente o máximo de rótulos que a música realmente possui. Sendo assim, otimizar a *ROC AUC Score* também é uma forma de lidar com desbalanceamento de classe, caso desse trabalho, em que alguns rótulos são muito mais frequentes que outros.

O otimizador *Stochastic gradient descent optimizer* (SGD), também conhecido como gradiente descendente incremental, é um método iterativo para otimizar funções objetivo diferenciáveis, ou seja, ele é uma aproximação estocástica da otimização do gradiente descendente. Estocástica, pois as amostras são aleatoriamente escolhidas ou misturadas, diferente do gradiente descendente padrão em que amostras são utilizadas como um grupo ou na ordem que foram passadas.

O otimizador SGD possui uma série de parâmetros que ajustam a forma como é realizada a otimização da função de custo escolhida e que precisam ser configurados, por exemplo, a taxa de aprendizado é um hiperparâmetro que controla quanto é ajustado os pesos da rede neural em relação ao gradiente de perda. O momento, por sua vez, é uma técnica utilizada para evitar que o aprendizado fique preso em mínimos locais e convirja para o mínimo global. Ele funciona fazendo uma segunda atualização do peso, somando o valor do momento (0.9 nesse trabalho) multiplicado pela taxa de atualização anterior. *Nesterov*, outro parâmetro de configuração, é uma modificação do momento, ou seja, ao invés de ser atualizado o peso utilizando o gradiente descendente e depois adicionado o momento, com *Nesterov*, primeiro é adicionado o momento para depois ser calculado o gradiente descendente da nova posição. Por fim tem-se a taxa de de decaimento que atua diminuindo a taxa de aprendizado. Isso significa que a cada passo a taxa de aprendizado é menor e após algumas iterações chega a zero. Essa taxa foi ajustado no valor igual a zero para evitar que o aprendizado terminasse antes que fosse utilizada toda base de treinamento e tamém para não afetar a análise do aumento do número de épocas no desempenho do sistema.

A rede LSTM foi construída utilizando a biblioteca Keras, sem qulaquer tipo de alteração das funções de ativação padrão da rede neural, a fim de se verificar o desempenho do sistema para os valores comumente utilizados nas mais diferentes aplicações. Outros parâmetros, por sua vez, foram configurados com valores fixos, com o objetivo de diminuir o número de variações e combinações possíveis. Desta forma, todas as camadas escondidas da rede foram configuradas com 32 neurônios.

O número de camadas escondidas, por sua vez, foi escolhido como um parâmetro variável. Esse parâmetro tem como consequência a variação da complexidade da rede, tornando-a mais profunda. Espera-se que redes mais complexas ou com um maior número de camadas apresente um maior poder de generalização. Outro parâmetro que também foi definido como variável foi o número de épocas.

4.4 Experimentos

Nesse trabalho, serão realizados dois tipos de experimentos com o objetivo de se avaliar o método. No primeiro teste, o número de camadas escondidas será variada com o objetivo de verificar o efeito do aumento da complexidade da rede na predição dos rotúlos das músicas que ainda não foram classificadas. No segundo teste, por sua vez, será avaliado o efeito do treinamento nos resulltados obtidos quando se aumenta o número de épocas de treinamento.

4.4.1 Teste variando o número de camadas escondidas

Nesse teste, todos os parâmetros da rede foram mantidos fixos, enquanto o número de camadas escondidas foi crescendo progressivamente até o valor máximo de cinco camadas. Cabe destacar que o número de neurônios de cada uma das camadas adicionadas tem também tamanho fixo.

O número de épocas escolhido para esse teste foi de apenas 1 época. A escolha desse valor pode ser explicada, pois o custo computacional para treinar um número maior de épocas é muito elevado, de forma que seria necessário, além de um tempo maior de processamento, de uma infraestrutura melhor que a disponível para esse trabalho. De qualquer forma, o treinamento da rede é dividida em *batches* por tamanho e em diferentes arquivos e, devido ao número elevado de exemplos, cerca de 330 mil, a rede está, na verdade, realizando diversos passos no processo de otimização da função custo, mesmo quando somente uma época é utilizada.

Os testes foram feitos utilizando, na LSTM, camadas com 32 neurônios. A rede foi treinada de uma a cinco camadas verificando a precisão da predição considerando as métricas *AUC micro* e *AUC sample*.

4.4.2 Teste variando o número de épocas

Nesse teste, todos os parâmetros foram mantidos fixos, enquanto foram variados o número de épocas de treinamento. A análise foi feita de uma à cinco épocas no sentido de se verficar o efeito que o aumento das mesmas pode provocar nas redes com diferentes números de camadas. Assim, devido ao alto custo computacional, foram escolhidas somente três configurações de camadas na realização dos testes. Ou seja, foram escolhidas a rede com o menor (uma) e o maior (cinco) números de camadas, bem como a rede que apresentou o melhor desempenho no experimento descrito na seção Seção 4.4.

5 Análise e Discussão dos Resultados

Este capítulo apresenta detalhadamente os dois tipos de testes realizados e os resultados obtidos com o modelo deste projeto. Ao final, é analisado e discutido os resultados encontrados.

5.1 Teste variando o número de camadas escondidas

A Tabela 1 mostra que, em todos os casos, o resultado foi consideravelmente maior que 0.50, para as métricas AUC *micro* e *sample*, indicando que a rede está aprendendo os padrões musicais. Da mesma forma, é possível observar que a medida que número de camadas foi aumentando os resultados obtidos também apresentaram uma melhora. No entanto, esta melhora só pode ser observada até um limite de três camadas, a partir da qual, a rede, apresentou uma piora considerável. Esse resultado mostra que tornar a estrutura mais complexa indefinidamente não reflete em uma melhoria direta dos resultados.

O resultado de três camadas com a melhor estrutura, nas condições dadas, pode ser explicado pelo fato de estar sendo utilizado um número pequeno de épocas para o treinamento. Dessa forma, a estrutura é complexa o suficiente para encontrar, de maneira mais adequada, as características da música, quando comparadas com apenas uma ou duas camadas. Ao mesmo tempo, as relações são mais simples do que aquelas obtidas com quatro e cinco camadas, apresentando um menor *underfit*. *Underfit* ocorre quando a rede fica muito genérica durante a fase de treinamento, de forma que não consegue aprender suficientemente bem os padrões existentes.

5.2 Teste variando as épocas

O teste variando o número de épocas pode ser justificado, visto que nos testes anteriores o número de épocas foi considerado um dos possíveis motivos dos resultados não terem sido adequados para todos os casos. Assim, testes utilizando cinco épocas para

Nº camadas/Métrica	AUC micro	AUC sample
1	0.6051	0.6018
2	0.6597	0.6963
3	0.6963	0.7229
4	0.6121	0.6348
5	0.5890	0.5856

Tabela 1 – Comprativo de resultados variando o número de camadas

Nº camadas/Nº épocas	1		5	
	AUC micro	AUC sample	AUC micro	AUC sample
1	0.6051	0.6018	0.5623	0.5893
3	0.6963	0.7229	0.5840	0.6064
5	0.5890	0.5856	0.6251	0.6371

Tabela 2 – Comprativo de resultados variando o número de épocas

as redes com uma, três e cinco camadas foram realizados. Os testes com uma e cinco camadas foram feitos por serem, respectivamente, o caso mais simples e complexo testado. O teste com três camadas foi realizado, por sua vez, devido ao fato desse experimento ter apresentado o melhor resultado quando o número de camadas foi alterado. Assim, como no experimento descrito na seção 5.1, o número de neurônios para cada camada foi mantido fixo (32 neurônios).

A Tabela 2 mostra que no caso de poucas camadas, o aumento de épocas resultou em resultados piores para as predições. Isso provavelmente ocorreu, pois para redes menores e portanto mais simples, um aumento no número de épocas pode provocar um efeito conhecido por *overfit*. *Overfit* ocorre quando, durante a fase de treinamento, a rede decora os padrões de treino, perdendo a capacidade de prever adequadamente padrões diferentes, ou seja, perde a capacidade de generalização.

O caso com maior número de camadas apresentou uma melhora quando o número de épocas foi aumentado, fortalecendo a ideia que, para um número menor de épocas, ocorreu o efeito de *underfit*. Esse resultado também fortalece a hipótese de que o aumento da complexidade da rede, bem como o número de épocas, poderia produzir um sistema que fosse capaz de realizar a rotulagem de músicas de maneira mais adequada.

O número de épocas apresentado na Tabela 2 é significativamente maior do que o utilizado na seção 5.1. Mesmo assim, pode-se observar que com cinco camadas o resultado ainda foi baixo devido ao *underfit*. Apesar de não ser o número ideal para a realização do teste, esse valor permitiu mostrar os efeitos causados quando o número de épocas é incrementado. Apesar do resultado ser válido para redes com dimensões e complexidade reduzidas, novos experimentos, com uma maior número de camadas e épocas, são necessários no sentido de validar, de maneira mais sólida, as conclusões descritas anteriormente. No entanto, testes mais complexos não puderam ser realizados devido à falta de infraestrutura computacional necessária ao desenvolvimento das simulações dentro do CEFET-MG.

Métodos	AUC
Rede LSTM 3 camadas	0.723
2016, *FCN-3*	0.786
2016, *FCN-4*	0.808
2016, *FCN-5*	0.848
2016, *FCN-6*	0.851
2017, *K1C2*	0.839
2017, *K2C2*	0.856
2017, *CRNN*	0.861

Tabela 3 – Comprativo de resultados de diversos métodos

5.3 Comparação com a literatura

A Tabela 3 mostra o comparativo dos resultados obtidos por meio da métrica *AUC sample* entre a rede LSTM com 3 camadas, proposta nesse trabalho, e 4 redes completamente convolucionais (com três, quatro, cinco e seis camadas), apresentadas no trabalho de Choi, Fazekas e Sandler (2016)). Além disso, exibe os resultados obtidos pelas redes K1C2, K2C2 e as redes recorrentes convolucionais propostas por Choi et al. (2017). A tabela mostra, portanto, que o resultado apresentado nesse trabalho não está na mesma faixa dos melhores valores obtidos na literatura.

Na verdade, a expectativa era que os resultados dessa pesquisa fossem comparáveis aos dos outros trabalhos apresentados. Assim, como mostrado em (CHOI et al., 2017), para se chegar aos resultados finais apresentados nos artigos, foram feitos diversos estudos sobre os efeitos das mudanças de parâmetros no desempenho dos sistemas propostos. A mesma metodologia também deveria ter sido adotada para as redes LSTM, mas, devido a falta de infraestrutura, não puderam ser executados.

6 Considerações finais e trabalhos futuros

Nesse trabalho, foi proposta a criação de uma ferramenta que gerasse rótulos automaticamente para músicas. Para isso, foi realizada uma revisão da literatura sobre as informações musicais presentes nos dados e também sobre os métodos já propostos para solucionar esse problema. Isso permitiu propor a utilização das redes LSTM como uma solução alternativa, bem como estudar seu comportamento.

Foram estudados os impactos no que tange o número de épocas e o número de camadas no desempenho da rede LSTM. Pôde-se concluir que quando o número de épocas é mantido constante e o número de camadas é incrementado, a rede passa a apresentar uma melhora de desempenho até um certo número de camadas. A partir de um determinado valor, o desempenho cai devido ao fato da rede estar subtreinada, apresentando um efeito de *underfit*.

Outro teste realizado comparou o efeito do número de épocas para redes com uma, três e cinco camadas. Nesse estudo, foi possível verificar que um aumento no número de épocas de treinamento, para os casos de redes com uma e três camadas, conduziu a resultados inferiores quando foram considerados apenas uma época de treinamento. Por sua vez, foi possível verificar que para redes com cinco camadas um aumento no número de épocas produziu um aumento no desempenho do sistema. Nos casos de redes com uma e três camadas um aumento no número de épocas aumentou o efeito de supertreinamento ou *overfit*, enquanto no caso de cinco camadas a melhora se justificou pela redução do efeito de subtreinamento ou *underfit*. Esses testes mostraram que essas modificações podem trazer um grande impacto no resultado e que um estudo mais aprofundado sobre essas alterações pode proporcionar resultados superiores ao encontrado.

É possível concluir também que apesar dos resultados não terem sido próximos aos apresentados da literatura o trabalho gerou boas contribuições, tanto teórica, com estudo de um método diferente de geração dos rótulos, como prática, ao colocar um método que utiliza informações temporais númericas ainda não exploradas na literatura. Da mesma forma, esse trabalho também mostrou a viabilidade da utilização das redes LSTM para solução do problema, abrindo novas possiblidades de estudo.

Como sugestão de trabalhos futuros propõe-se o estudo das variações do método proposto, como a utilização das redes LSTM bidirecionais baseadas em atenção (ZHOU et al., 2016). Outra proposta possível seria combinar camadas de redes LSTM com camadas de rede *feedfoward*, como proposto por Makris et al. (2017), na composição de música

rítimica. Por fim, outra possibilidade seria se fazer um estudo da utilização mais direta de combinações entre as redes LSTM e as redes *feedforward*, utilizando além das informações temporais, que foram utilizadas nesse trabalho, as informações estáticas das músicas, tais como duração, rótulo do artista etc.

Referências

BENGIO, Y.; SIMARD, P.; FRASCONI, P. Learning long-term dependencies with gradient descent is difficult. **IEEE transactions on neural networks**, IEEE, v. 5, n. 2, p. 157–166, 1994. Citado na página 16.

BERTIN-MAHIEUX, T. et al. The million song dataset. In: **Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)**. [S.l.: s.n.], 2011. Citado 2 vezes nas páginas 7 e 21.

BERTIN-MAHIEUX, T. et al. The million song dataset. In: **Ismir**. [S.l.: s.n.], 2011. v. 2, n. 9, p. 10. Citado na página 21.

BRITZ, D. **Recurrent Neural Networks Tutorial, Part 1 – Introduction to RNNs**. [S.l.], 2015 (accessed May 10, 2018). <http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/>. Citado na página 14.

CHOI, K.; FAZEKAS, G.; SANDLER, M. Automatic tagging using deep convolutional neural networks. **arXiv preprint arXiv:1606.00298**, 2016. Citado 2 vezes nas páginas 7 e 30.

CHOI, K. et al. Convolutional recurrent neural networks for music classification. In: IEEE. **Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on**. [S.l.], 2017. p. 2392–2396. Citado 2 vezes nas páginas 8 e 30.

ECK, D. et al. Automatic generation of social tags for music recommendation. In: PLATT, J. C. et al. (Ed.). **Advances in Neural Information Processing Systems 20**. Curran Associates, Inc., 2008. p. 385–392. Disponível em: <http://papers.nips.cc/paper/3370-automatic-generation-of-social-tags-for-music-recommendation.pdf>. Citado na página 6.

ESTRELLA, E. **A Beginner's Guide to Music History**. [S.l.], 2018 (accessed October 08, 2018). <https://www.thoughtco.com/music-history-101-2455857>. Citado na página 1.

FAWCETT, T. An introduction to roc analysis. **Pattern recognition letters**, Elsevier, v. 27, n. 8, p. 861–874, 2006. Citado na página 25.

GERS, F. A.; SCHMIDHUBER, J.; CUMMINS, F. Learning to forget: Continual prediction with lstm. IET, 1999. Citado na página 3.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, MIT Press, v. 9, n. 8, p. 1735–1780, 1997. Citado na página 16.

IBMCORPORATION. **IBM's ASCC introduction**. [S.l.], 2015 (accessed June 28, 2018). <http://www-03.ibm.com/ibm/history/exhibits/markI/markI_intro.html>. Citado na página 2.

JEHAN, T.; DESROCHES, D. The echo nest analyzer documentation. **Prepared by: September**, v. 2, 2011. Citado na página 11.

LAW, E.; AHN, L. V. Input-agreement: a new mechanism for collecting data using human computation games. In: ACM. **Proceedings of the SIGCHI Conference on Human Factors in Computing Systems**. [S.l.], 2009. p. 1197–1206. Citado na página 9.

MAKRIS, D. et al. Combining lstm and feed forward neural networks for conditional rhythm composition. In: SPRINGER. **International Conference on Engineering Applications of Neural Networks**. [S.l.], 2017. p. 570–582. Citado na página 31.

MANDEL, M. I.; ELLIS, D. P. W. Multiple-instance learning for music information retrieval. In: **ISMIR**. [S.l.: s.n.], 2008. Citado na página 6.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The bulletin of mathematical biophysics**, Springer, v. 5, n. 4, p. 115–133, 1943. Citado na página 2.

MORAIS, M. V. G. Álgebra dos tons. **Brasília. Disponível em:< http://www. ucb. br/sites/100/103/TCC/22008/MarcosViniciusGomesMorais. pdf>. Acesso em**, v. 25, 2012. Citado na página 4.

OLAH, C. **Understanding LSTM Networks**. [S.l.], 2015 (accessed May 10, 2018). <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>. Citado na página 16.

RATIONALITY. **The Cambridge Dictionary of Philosophy**. [S.l.]: Cambridge University Press, 1999. Citado na página 1.

SAHIDULLAH, M.; SAHA, G. Design, analysis and experimental evaluation of block based transformation in mfcc computation for speaker recognition. **Speech Communication**, Elsevier, v. 54, n. 4, p. 543–565, 2012. Citado na página 14.

TINGLE, D.; KIM, Y. E.; TURNBULL, D. Exploring automatic music annotation with "acoustically-objective"tags. In: **Multimedia Information Retrieval**. [S.l.: s.n.], 2010. Citado 3 vezes nas páginas 6, 7 e 9.

XU, M. et al. Hmm-based audio keyword generation. In: SPRINGER. **Pacific-Rim Conference on Multimedia**. [S.l.], 2004. p. 566–574. Citado na página 13.

ZAMBIASI, S. P. **Introdução a Redes Neurais Artificiais**. 2002. Citado na página 2.

ZHOU, P. et al. Attention-based bidirectional long short-term memory networks for relation classification. In: **Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)**. [S.l.: s.n.], 2016. v. 2, p. 207–212. Citado na página 31.

ZHOU, R.; REISS, J. D. Music onset detection. **Machine Audition: Principles, Algorithms and Systems: Principles, Algorithms and Systems**, IGI Global, p. 297, 2010. Citado na página 21.

Apêndices

APÊNDICE A – Descrição do *Million Song dataset*

1 *analysis sample rate*: taxa de amostragem do uso de áudio;
2 *artist 7digitalid*: ID do *7digital.com* ou -1 quando não existir;
3 *artist familiarity*: estimativa algorítmica do quão conhecido o artistas é;
4 *artist hotttnesss*: estimativa algorítmica do quão falado o artista tem sido ultimamente;
5 *artist id*: *Echo Nest ID*;
6 *artist latitude*: latitude;
7 *artist location*: nome do local do artista;
8 *artist longitude*: longitude;
9 *artist mbid*: ID do *musicbrainz.org*;
10 *artist mbtags*: rótulos do artista dado pelo *musicbrainz.org*;
11 *artist mbtags count*: número de rótulos dado para o artista pelo *musicbrainz*;
12 *artist name*: nome do artista;
13 *artist playmeid*: ID do playme.com, ou -1 quando não existir;
14 *artist terms*: rótulos do artista dado pelo *Echo Nest*;
15 *artist terms freq*: frequência do rótulo para aquele artista retirado do *Echo Nest*;
16 *artist terms weight*: peso dado a cada rótulo pelo *Echo Nest*;
17 *artist id*: *Echo Nest ID*;
18 *audio md5*: código *hash* do áudio;
19 *bars confidence*: medida de confiança da barra que corresponde à primeira batida;
20 *bars start*: início de uma barra. Uma barra (ou medida) é um segmento de tempo definido como um determinado número de batidas. Deslocamentos de barra também indicam *downbeats*, ou seja, a primeira batida da medida;
21 *beats confidence*: medida de confiança da batida que correspode a um conjunto de *tatums*;
22 *beats start*: início de uma batida, ou seja,resultado de um rastreamento de batidas. Batida é a unidade básica de tempo de uma música, ou seja, a cada *tick* de um metrônomo. As batidas são tipicamente múltiplas de *tatums*. *Tatum* é equivalente à divisão de tempo fixa que mais coincide com inícios de *onset*;
23 *danceability*: valor de 0 a 1 gerado por algoritmo que define o quão dançante é a música;
24 *duration*: duração da música em segundos precisamente calculado por um decodificador de áudio;
25 *end of fade in*: tempo em segundos que começa a música;

26 *energy*: valor de 0 a 1 que define a energia da música do ponto de vista do ouvinte;
27 *key*: a chave geral estimada de uma faixa. A chave identifica a tríade tônica, o acorde, maior ou menor, que representa o ponto final de descanso de uma música;
28 *key confidence*: medida de confiança da chave musical;
29 *loudness*: o volume total de uma faixa em decibeis (dB). Valores de volume no analisador são calculados em uma faixa inteira e são úteis para comparar a intensidade relativa de segmentos de faixas. Volume é a qualidade de um som que é correlato psicológico primário da força física (amplitude). O volume é representado por três dados: o valor em dB em um textitonset (*loudness start*), o valor em dB de um pico (*loudness max*) e o valor em dB relativo ao último segmento da faixa de música (*loudness end*);
30 *mode*: indica modalidade (maior ou menor) de uma faixa, ou seja, o tipo de escala da qual seu conteúdo melódico é derivado;
31 *mode confidence*: medida de confiança do *mode*;
32 *release*: nome do album;
33 *release 7digitalid*: ID do album no 7digital.com ou -1 quando não existir;
34 *sections confidence*: medida de confiança de um seção;
35 *sections start*: *onset* de cada seção. As seções são definidas por grandes variações no ritmo ou timbre, por exemplo, refrão, verso, ponte, solo de guitarra etc. Cada seção contém suas próprias descrições de tempo, tecla, modo, assinatura de tempo e volume sonoro;
36 *segments loudness max*:representa o valor, em db, do pico de maior intensidade de cada segmento;;
37 *segments loudness max time*: descreve o deslocamento dentro do segmento de pontos de intensidade máxima;
38 *segments loudness max start*: valor em dB no *onset*;
39 *segments pitches*: o conteúdo é dado por um vetor 'chroma', que correspondendo a 11 classes de tom C, C#, D até B. Essas classes possuem valores que variam de 0 a 1 e descrevem a dominância relativa de cada tom na escala cromática. Os vetores são normalizados para 1 considerando a dimensão mais forte. Dessa maneira, sons ruidosos provavelmente serão representados por um vetor com valores próximo de 1, enquanto tons puros serão representados por uma posição com valor próximo 0;
40 *segments start*: informação de quando começa cada segmento/*onset*;
41 *segments timbre*: timbre é uma qualidade musical que distingue diferentes tipos de instrumentos musicais ou vozes. É uma noção complexa também utilizada para cores, texturas e é derivada do formato de segmento da superfície spectro-temporal, independente do tom ou da sonoridade. Essa característica é um vetor que inclui 11 valores centrados em 0. Esses valores são abstrações de alto nível da superfície espectral, ordenada por grau de importância. Para completar, no entanto, a primeira

dimensão representa a média do ruído do segmento, o segundo representa o brilho, o terceiro está mais próximo do nivelamento do som, o quarto representa a força do som etc;

42 *similar artists*: ID do *Echo Nest* para artistas relacionados;

43 *song hotttnesss*: estimativa algorítmica da relevância na atualidade;

44 *song id*: ID da música no *Echo Nest*;

45 *start of fade out*: tempo em segundos que a música acaba;

46 *tatums confidence*: medida de confiança de *tatum*, descritos como eventos musicais percebidos dentro de um segmento;

47 *tatums start*: quando começa um *tatums*.

48 *tempo*: o tempo estimado global de uma faixa em batidas por minuto (BPM). Na terminologia musical, tempo é a velocidade ou ritmo de uma determinada música e deriva diretamente da duração média da batida;

49 *time signature*: tempo estimado global de uma faixa. A assinatura de tempo é uma convenção de notação para especificar quantas batidas estão em cada barra (ou medida);

50 *time signature confidence*: medida de confiança do *time signature*;

51 *title*: nome da música;

52 *track id*: ID da faixa no *Echo Nest*;

53 *track 7digitalid*: ID da faixa no *7digital.com*;

54 *year*: ano de lançamento da música pelo *MusicBrainz* ou 0 quando não existir;

55 *segments confidence*: medida de confiança de um segmento.

APÊNDICE B – Rótulos Utilizados

1 *rock*
2 *pop*
3 *alternative*
4 *indie*
5 *electronic*
6 *female vocalists*
7 *favorites*
8 *Love*
9 *dance*
10 *00s*
11 *alternative rock*
12 *jazz*
13 *beautiful*
14 *singer-songwriter*
15 *metal*
16 *chillout*
17 *male vocalists*
18 *Awesome*
19 *classic rock*
20 *soul*
21 *indie rock*
22 *Mellow*
23 *electronica*
24 *80s*
25 *folk*
26 *british*
27 *90s*
28 *chill*
29 *american*
30 *instrumental*
31 *punk*
32 *oldies*
33 *seen live*
34 *blues*
35 *hard rock*

36 *cool*
37 *Favorite*
38 *ambient*
39 *acoustic*
40 *experimental*
41 *Favourites*
42 *female vocalist*
43 *guitar*
44 *Hip-Hop*
45 *70s*
46 *party*
47 *country*
48 *easy listening*
49 *sexy*
50 *catchy*