Universidade Federal da Bahia
Escola Politécnica
Departamento de Engenharia Elétrica
Programa de Pós-Graduação em Engenharia Elétrica

PROPOSTA DE UM MÓDULO AUDITIVO ARTIFICIAL PARA UM ROBÔ ASSISTIVO

Autor: Márcio Luiz Lima de Oliveira
Orientadores: Jés J. F. Cerqueira
Eduardo F. Simas Filho

Dissertação submetida ao Programa de Pós-Graduação em Engenharia Elétrica, para preencher os requisitos à obtenção do Título de Mestre em Engenharia Elétrica

Salvador - BA, Julho, 2018

Márcio Luiz Lima de Oliveira

PROPOSTA DE UM MÓDULO AUDITIVO ARTIFICIAL PARA UM ROBÔ ASSISTIVO

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal da Bahia

Universidade Federal da Bahia

Departamento de Engenharia Elétrica

Programa de Pós-Graduação em Engenharia Elétrica

Orientador: Jés J. F. Cerqueira

Coorientador: Eduardo F. Simas Filho

Salvador

2018

Ficha catalográfica elaborada pelo Sistema Universitário de Bibliotecas (SIBI/UFBA),
com os dados fornecidos pelo(a) autor(a).

Lima de Oliveira, Márcio Luiz
PROPOSTA DE UM MÓDULO AUDITIVO ARTIFICIAL PARA UM ROBÔ ASSISTIVO / Márcio Luiz Lima de Oliveira. -- Salvador, 2018.
102 f.

Orientador: Jés de Jesus Fiais Cerqueira.
Coorientador: Eduardo Furtado de Simas Filho.
Dissertação (Mestrado - Programa de Pós-Graduação em Engenharia Elétrica) -- Universidade Federal da Bahia, Departamento de Engenharia Elétrica, 2018.

1. audição artificial. 2. MFCC. 3. multilateração. 4. redes neurais artificiais. I. de Jesus Fiais Cerqueira, Jés. II. Furtado de Simas Filho, Eduardo. III. Título.

MÁRCIO LUIZ LIMA DE OLIVEIRA

Proposta de Um Módulo Auditivo Artificial Para Um Robô Assistivo

Dissertação apresentada à Universidade Federal da Bahia, como parte das exigências do Programa de Pós-Graduação em Engenharia Elétrica, para a obtenção do título de *Mestre*.

APROVADA em: 12 de Julho de 2018.

BANCA EXAMINADORA

Prof. Dr. Jés de Jesus Fiais Cerqueira
Orientador - UFBA

Prof. Dr. Eduardo Furtado de Simas Filho
Coorientador - UFBA

Prof. Dr. Paulo Cesar Machado de Abreu Farias
UFBA

Prof. Dr. Antonio Carlos Lopes Fernandes Junior
UFBA

Prof. Dr. Dirceu de Freitas Piedade Melo
IFBA

AGRADECIMENTOS

Aos meus pais, João e Gracil por todo o apoio para que eu pudesse chegar aqui e por serem meus exemplos a quem eu admiro muito, de forma incondicional por toda minha vida.

Aos meus irmãos, João, Ricardo e Marina por sempre estarem lá pra me ajudar quando eu precisava. A João por me ajudar com as gravações e por me emprestar equipamentos de áudio, a Ricardo por ter me ajudado com a estrutura e correção da dissertação, a Marina por toda a força necessária. Não conseguiria ter terminado sem a ajuda de vocês.

A minha namorada, Kristina, por ter me aguentado quando eu estava de mau humor por conta do excessivo trabalho e madrugadas acordado na frente de um computador escrevendo códigos.

Ao professor Jés, meu orientador por ter me apoiado e ajudado desde o início. Sem sua ajuda eu realmente não teria conseguido chegar até onde cheguei.

Ao professor Eduardo, meu orientador por também ter me ajudado e compartilhado de todo o seu conhecimento comigo.

Ao professor Augusto Loureiro por todo o apoio quando precisei.

À CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), pela concessão da bolsa durante o período do mestrado.

À UFBA (Universidade Federal da Bahia) por toda a sua trajetória de excelência como universidade.

À todos meus amigos, colegas e familiares que de alguma forma me ajudaram quando precisei e contribuíram para meu crescimento pessoal e acadêmico.

ARTIGOS DO AUTOR

1. Oliveira, M.; Cerqueira, J.; Simas F., E. Simulation of an Artificial Hearing Module for an Assistive Robot. *Intelligent Systems Conference (IntelliSys)*, p. 1-7, 2018.

RESUMO

Neste trabalho foi proposta a simulação e construção de um módulo auditivo artificial para um robô assistivo de baixo custo com os objetivos de identificar: (i) o locutor; (ii) que emoções são transmitidas pelo seu discurso; (iii) a posição do locutor no espaço. Para a resolução dessa tarefa, diferentes técnicas para a realização das três tarefas propostas são analisadas, sendo as técnicas mais pertinentes: os coeficientes Cepstrais de Frequência em Mel (MFCCs) como técnica de extração de características da voz, redes neurais artificiais (RNAs) como classificador e multilateração (MLAT) como técnica de localização da origem sonora. Foram feitas análises experimentais com diferentes configurações para se chegar nos resultados apresentados no decorrer dessa pesquisa, que foram, de forma geral, positivo e mostram que o módulo proposto foi factível.

Palavras-chave: audição artificial, MFCC, multilateração, redes neurais artificiais

ABSTRACT

It is proposed in this work the simulation and construction of an artificial hearing module for a low-cost assistive robot with the objectives of identifying: (i) the speaker; (ii) what emotions are conveyed by his or her speech; (iii) where the speaker is in space. In order to make this work, different techniques for the accomplishment of the proposed tasks are analyzed, with the most pertinent techniques being: Mel Frequency Cepstral Coefficients (MFCCs) as feature extraction technique, artificial neural networks (ANN) as classifier and multilateration (MLAT) as technique for locating a sound source. Experimental analyzes were done with different configurations to achieve the results presented in the course of this research, which were, in general, positive and show that the proposed module was feasible.

Keywords: artificial hearing, artificial neural networks, MFCC, multilateration

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ANFIS	*Adaptive Neuro Fuzzy Inference System*
BBB	BeagleBone Black
DOA	*Direction Of Arrival*
Emo-DB	*Berlin Emotion Database*
FCM	*Fuzzy C-mean*
GMM	*Gaussian Mixture Models*
IA	Inteligência Artificial
LVQ	*Learning Vector Quantization*
MEDC	*Mel-Energy spectrum Dynamic Coefficients*
MFCC	*Mel Frequency Cepstral Coefficients*
MFSC	*Mel Frequency Spectrum Coefficients*
ML	*Machine Learning*
MLAT	Multilateração (*Multilateration*)
MLF	*Multi-Layer Feedforward*
MLP	*Multilayer Perceptron*
MSE	*Mean Squared Error*
Rprop	*Resilient Backpropagation*
RNA	Redes Neurais Artificiais
SAVEE	*Surrey Audio-Visual Expressed Emotion*
SVM	*Suport Vector Machine*
TDE	*Time Delay Estimate*
TDOA	*Time Difference Of Arrival*

LISTA DE SÍMBOLOS

α	Constante do filtro passa-alta
ΔC	Coeficientes de delta
$\Delta^2 C$	Coeficientes de delta-delta
Γ	Função de mapeamento
μ	Vetor com MFCC, delta e delta-delta
η	Fator da *Resilient Backpropagation*
θ	Probabilidade
ϑ	Coeficiente limiar de um neurônio
ξ	Potencial de um neurônio
ω	Coeficiente de peso de um neurônio
$\mathbf{A}$	Matriz auxiliar para o cálculo por multilateração
a	Elemento de $\mathbf{A}$
b	Vetor auxiliar para o cálculo por multilateração
C	*Mel Frequency Cepstral Coefficients*
Emo	Emoção
F	Frequência em Hertz
f	Função (genérica)
$f(\xi)$	Função sigmoide
L	Locutor
M	Frequência em Mel
MSE	*Mean squared error*
p	Posição conhecida
q	Posição calculada

t	Tempo
v	Velocidade do som
X	Sinal de áudio
x	Coeficiente de viés de um neurônio
w	Janela de Hamming
Y	*Fast Fourier Transform*

SUMÁRIO

1 INTRODUÇÃO

Vivemos em um mundo cada vez mais moderno e globalizado. Nesse contexto, novas tecnologias e processos aparecem a cada novo instante. A partir do desenvolvimento de novas técnicas, do avanço da tecnologia e do barateamento de componentes eletrônicos, hoje é possível desenvolver robôs assistivos com custos cada vez menores. Tais robôs precisam ter diferentes módulos que se comuniquem de maneira coordenada. É um trabalho multidisciplinar, envolvendo robótica, psicologia, processamento de sinais e computação, entre outros.

A robótica assistiva está avançando de várias maneiras distintas, sendo usada em muitos casos diferentes, como, por exemplo, com crianças, idosos, portadores de necessidades especiais e adultos. Para o design de um robô assistivo deve considerar que ele tenha interações mais parecidas com as que os seres humanos teriam entre si, de tal forma que não estranharíamos o robô uma vez que ele fosse introduzido em um novo ambiente (TAKAYAMA; DOOLEY; JU., 2011). Desta forma, quando um módulo de audição artificial é elaborado, terá que ser capaz de realizar algumas tarefas que os seres humanos realizam, de forma natural, através da audição, de uma maneira análoga (porém nunca igual).

A capacidade de interação através da voz é uma das principais formas de comunicação entre os seres humanos. Para que haja interações através de voz, é preciso também que haja a recepção e interpretação dos sons. A audição humana é capaz de executar diferentes tarefas, como alertar quando há perigos, reconhecer sons familiares e, principalmente, para uso social. No entanto, esta comunicação é versátil e nem sempre precisa ser feita com o uso de vocábulos. Muitas vezes, os diálogos podem ser feitos sem o uso de quaisquer palavras. Desta forma, um robô social que interage com humanos precisa de algo semelhante a um ouvido, capaz de desempenhar algumas funções. Para a audição artificial, o som atinge os microfones e é interpretado. Diferentes interpretações podem ser feitas pelo módulo auditivo. Cada locutor possui em sua fala várias informações relevantes, como timbre, sotaque, emoções, palavras, intonações, distância entre o locutor e o receptor, etc.

Neste trabalho, é proposto um módulo de audição artificial, que será incluído em um robô assistivo. Tal "ouvido" terá que ser capaz de, inicialmente, realizar três tarefas distintas: (i) localizar a fonte de som - sabendo em que ponto no espaço estão falando com ele (NORRDINE, 2015); (ii) identificar o locutor - saber quem está falando com ele (HASAN et al., 2004); (iii) identificar emoções estereotipadas - saber que emoção está sendo transmitida pela voz do locutor (DAHAKE; SHAW, 2016).

Ao contrário da abordagem de outros pesquisadores, como o exemplo de Valin

(2005), e do senso comum para audição, o conteúdo das palavras faladas não será levado em consideração aqui. Essas três tarefas distintas foram escolhidas porque foram consideradas essenciais para o desenvolvimento de um robô social de baixo custo que poderia interagir com adultos, crianças, idosos ou pessoas com necessidades especiais. Para o propósito da elaboração de tal módulo serão utilizadas técnicas como *Mel Frequency Cepstral Coefficients* (MFCC), redes neurais artificiais (RNA) e multilateração (MLAT).

Autores como Hasan et al. (2004) já realizaram identificação de um locutor, Dahake e Shaw (2016) já identificaram emoções na voz de locutores e Norrdine (2015) já localizaram fontes sonoras, de forma que esse trabalho foi viável. A principal contribuição deste trabalho é o projeto de um módulo capaz de realizar, ao mesmo tempo, as três tarefas citadas. Arquivos de áudio gravados previamente foram utilizados para projetar o sistema e testes experimentais foram realizados para validar a arquitetura proposta.

1.1 JUSTIFICATIVA

A audição é uma das principais formas de comunicação entre seres humanos, pois estes a utilizam de forma social. Eles precisam saber de onde falam, quem fala e o que os outros seres humanos de seu convívio social estão sentindo; tudo isso apenas através da interpretação de ondas sonoras. Essas tarefas são realizadas a todo momento sem que haja, sequer esforço para tal. Com o intuito de um avanço computacional e robótico social, é necessário que o robô consiga interpretar o áudio de uma maneira similar (porém não necessariamente igual), ao modo que humanos o fariam, no sentido de conseguir distinguir locutores, saber localizar de maneira tridimensional a origem de uma fonte sonora e o sentimento transmitido por esse locutor.

A existência de robôs sociais que consigam interagir com humanos é importante para o avanço dessa área tecnológica, principalmente nas interfaces de comunicação entre máquinas e homens. Para esses, entender a voz é essencial. A interpretação do áudio gerado é importante para a tomada de decisão de um robô complexo que possua vários módulos distintos.

Uma possível aplicação para tal robô assistivo, seria a de interagir com crianças autistas. Tais crianças muitas vezes possuem dificuldade de se comunicar de maneira oral tradicional, com o uso de palavras, porém sua voz, além de conter o seu timbre característico, também transmite emoções. Um robô social com o módulo auditivo em questão, ao interagir com tais crianças seria capaz de conseguir entendê-las de uma maneira melhor que um robô tradicional cujo o enfoque seja o entendimento puramente de palavras.

Dessa maneira, para que um robô assistivo consiga interagir com pessoas tão diferentes, é necessário que ele possua um módulo auditivo que seja capaz de identificar o locutor em questão, emoções e de onde ele fala ao invés do tradicional enfoque em

vocábulos.

1.2 OBJETIVOS

Os objetivos nesse trabalho podem ser divididos entre gerais e específicos.

1.2.1 Objetivos Gerais

Os objetivos desse trabalho foram de projetar, simular e construir um módulo auditivo artificial, para um robô assistivo, que seja capaz de realizar as três distintas funções:

- Identificação de locutor.
- Identificação de emoções estereotipadas na voz.
- Localização de uma fonte sonora.

1.2.2 Objetivos Específicos

São considerados objetivos específicos desse trabalho:

- Projetar um método para identificar um locutor utilizando diferentes técnicas e compará-las. Como exemplo, o MFCC *(Mel Frequency Cepstral Coefficient)* expandido, com delta e delta-delta como extração de características e classificação dos coeficientes obtidos por múltiplas redes neurais artificais multinível *feedfoward* com aprendizado por *resilient backpropagation*.
- Projetar um método para identificação de emoções estereotipadas na voz utilizando diferentes técnicas e compará-las, utilizando técnicas semelhantes as estudas na identificação de locutor
- Simular um método para localizar uma fonte sonora, utilizando-se de técnicas de multilateração com soluções lineares.
- Construir um módulo auditivo artificial com os resultados da etapa de projeto e os adaptando para o mundo real, utilizando um sistema embarcado, que realize as tarefas citadas anteriormente.
- Comparar os resultados obtidos em todas as etapas com o que foi encontrado por outros trabalhos relacionados.

1.3 ORGANIZAÇÃO DO TRABALHO

O restante deste documento está organizado conforme descrito à seguir:

- **Capítulo 2. Revisão Bibliográfica**

  Este capítulo terá as analises e estudos realizados por outros autores que são relevantes ou que estão na mesma área de estudo desse trabalho.

- **Capítulo 3. Fundamentação Teórica**

  Nesse capítulo, todos os assuntos que foram importantes para o desenvolvimento desse trabalho serão abordados. Sendo as partes mais importantes o cálculo do MFCC com Delta e Delta-Delta, Redes Neurais Artificias, Emoções Estereotipadas e Multilateração.

- **Capítulo 4. Módulo Auditivo Artificial Proposto**

  Nesse capítulo será abordado como o módulo será desenvolvido, mostrando as ideias e propostas para seu projeto, simulação e construção. Também aqui estará a metodologia aplicada para esse trabalho e as etapas de seu desenvolvimento.

- **Capítulo 5. Simulação e Resultados**

  A proposta desse capítulo é projetar cada função do módulo, testando diferentes configurações e simulações de maneira comparativa para obter-se os melhores resultados. Também será feita uma comparação com os resultados obtidos por outros autores.

- **Capítulo 6. Construção do Módulo e Resultados**

  Para esse capítulo, a construção do módulo será abordada, assim como seus resultados comparativos com os obtidos durante a etapa de projeto em outros trabalhos correlatos.

- **Capítulo 7. Conclusão**

  Capítulo final, onde estarão as últimas considerações desse trabalho, um resumo dos resultados obtidos e futuros trabalhos que podem ser desenvolvidos após o término do mesmo.

2 REVISÃO BIBLIOGRÁFICA

Nesse capítulo, as pesquisas e resultados de outros autores serão abordados, comentando-se suas técnicas, respostas e taxas de acerto/erro. Tais autores desenvolveram pesquisas de alta relevância para suas áreas. Serão analisados diferentes trabalhos, dois de cada área de atuação do módulo auditivo artificial.

Os autores Kinnunen, Karpov e Fr (2005) construíram um sistema em tempo real de identificação e verificação de um locutor utilizando MFCC (*Mel frequency cepstral coefficients*) e LPCC (*Linear Predictive Coding Cepstrum*) que medem o envelope espectral de curto prazo, que se correlaciona com a fisiologia do trato vocal. Utilizando um banco de vozes com 10 a 10000 locutores distintos para alimentar os seus classificadores GMM (*Gaussian Mixture Models*) e VQ (*Vector Quantization*), eles conseguiram fazer a identificação em tempo real, cujos melhores resultados foram: um erro de 0,32% com VQ e 0,95% com GMM.

No trabalho de Maesa et al. (2012), foi proposto um sistema de reconhecimento automático de locutores, com um banco contendo 450 locutores distintos. Foi utilizado MFCC (*Mel-Frequency Cepstrum Coefficient*) como técnica de extração de características e GMM (*Gaussian Mixture Model*) como classificador. Ao final foi obtido um acerto de 96,22%.

Já Kim et al. (2017) buscam estimar a emoção na voz de crianças autistas. Para tal tarefa, foi utilizado um banco de dados com diferentes vozes contento diferentes emoções, o IEMOCAP (*Interactive Emotional Dyadic Motion Capture*) (BUSSO et al., 2008). Para fazer a extração de características, eles utilizam um *open code* software chamado openSMILE. Usando seus descritores de baixo nível, os padrões temporais são extraídos como recursos para o aprendizado de máquina. Para o aprendizado, SVM (*Support Vector Machine*) foi utilizado. Para a simulação, 52,7% das vezes, o sistema deles conseguiu identificar corretamente a emoção, porém, a aplicação em tempo real obteve uma taxa de acerto muito baixa e por isso os autores não a divulgaram e alegaram que atualmente estão procurando aumentar o banco de vozes deles.

De acordo com os autores Dahake e Shaw (2016), para uma simulação que detectasse emoções na voz, foi utilizado MFCC com parâmetros estatísticos, além da intonação do locutor e utilizaram SVM (*Support Vector Machine*) como classificador com diferentes *kernels*, e o resultado de cada um deles foi avaliado ao fim do trabalho. Com *kernel* linear o acerto médio foi de 81,4%, com o quadrático foi de 87%, com o polinomial foi de 83%.

No artigo dos autores Huang, Ohnishi e Sugie (1997), eles desenvolvem um robô que localiza a fonte sonora, mesmo que esteja em outro cômodo e segue o som até sua fonte.

Para isso, eles se utilizam de 3 microfones e, com a diferença de tempo que o áudio atingiu cada microfone, conseguem calcular a distância para o objeto, conhecendo a geometria que os sensores estão localizados. Também utilizam vários filtros pra evitar eco e reverberação. Outro algorítimo utilizado pelos autores é para reproduzir o fenômeno acústico chamado de *cocktail party effect* que é a capacidade de isolar um som importante de outros ruídos desinteressantes.

Na publicação de (PEI et al., 2013), eles utilizaram um módulo Kinect (WEBB; ASHLEY, 2012) para fazer a localização de uma fonte sonora. Foi utilizado uma técnica de localização chamada de DOP (*Dilution Of Precision*) junto com TDOA (*Time Difference Of Arrival*) e TDE (*Time Delay Estimate*) is para calcular onde estava a fonte sonora. Obtiveram um erro máximo de 1,9 m (190% de erro) e um erro médio inferior a 1 metro (aproximadamente 40% de erro) na identificação.

O diferencial da proposta dessa dissertação para todos os trabalhos citados acima é que, por melhor que seja o resultado deles e por mais que haja tecnologia de ponta, eles focam apenas em uma única tarefa. Identificação de locutor, emoção ou fonte sonora. O presente trabalho aborda essas três grandes áreas de estudo e realiza projetos, simulações e construção de um módulo em todas essas distintas e grandes áreas e que, futuramente, possa ser implementado em um robô assistivo. Não foi encontrado nenhum trabalho, na revisão bibliográfica realizada, que tentasse unir todas essas técnicas para a obtenção de um módulo único. Dessa forma, esse trabalho, integrando abordagens bem documentadas, irá realizar, de uma forma diferente, a análise desses tópicos.

3 FUNDAMENTAÇÃO TEÓRICA

Nesse capítulo serão apresentados alguns conhecimentos considerados importantes com o objetivo de contextualizar e facilitar a compreensão do trabalho. Foram abordadas as principais técnicas utilizadas para o reconhecimento de quem fala, de emoções e da localização da fonte sonora.

3.1 ROBÔ ASSISTIVO

De maneira introdutória à teoria, o primeiro tópico a ser abordado será o conceito de um robô assistivo.

De acordo com Cortellessa et al. (2008) existem dois tipos de robôs assistivos. Os sociais e os físicos. Ambos tipos se baseiam no fato de que robôs assistivos possuem a principal função de ajudar humanos de alguma maneira. Robôs sociais ajudam humanos com interações sociais ao invés de físicas. Por exemplo, um robô assistivo físico pode dobrar as roupas da casa. Um robô social irá tentar demonstrar empatia e poderá, por exemplo, mimetizar emoções e suas ações podem ir além de simples comandos. Na Figura 1 se pode ver um exemplo de um robô social assistivo.

Figura 1 – Exemplo de um robô social assistivo.

Fonte: Cortellessa et al. (2008)

Nesse contexto, esse trabalho focou-se na construção de um dos módulos que constituiriam um futuro e, ainda inexistente, robô assistivo com características socais, cujo

o "ouvido" seria o módulo auditivo artificial e realizaria as funções de identificação de: (i) locutor, (ii) emoção e (iii) fonte sonora.

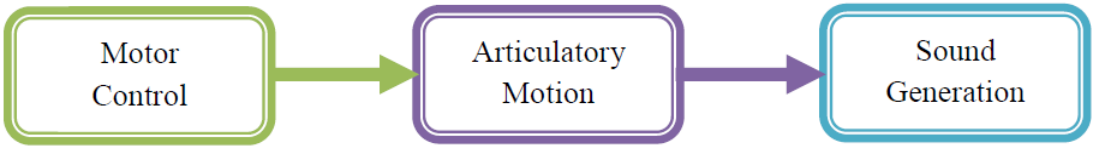
3.2 A VOZ HUMANA

Seres humanos são animais sociais. Para o desenvolvimento de atividades sociais, a utilização da voz está entre as mais importantes formas de interação, podendo ela transmitir diversas informações tais como sentimento, ideias, localização, língua falada, identidade do locutor, etc. Para tal, o aparelho vocal humano é utilizado.

De acordo com Mahendru (2014), humanos conseguem produzir sons com várias frequências diferentes que podem mudar rapidamente. Isso é possível por causa dos músculos e movimentos precisos dos órgãos envolvidos na produção da voz. O controle da voz é feito pelo cérebro através do sistema nervoso central, conectando o cérebro a diversos órgãos como o pulmão, cordas vocais, língua, mandíbula e etc. Como pode ser visto na Figura 2, o controle motor, realizado pelo cérebro, coordena diversos órgãos e movimentos articulados que, por sua vez, geram os sons vocais.

Não há dois indivíduos que soem idênticos, pois seus tratos vocais, tamanhos de laringe e outros órgãos presentes na produção de voz são diferentes. Além dessas diferenças físicas, cada pessoa tem sua maneira característica de falar, incluindo o uso de sotaque, ritmo, estilo de entonação, padrão de pronúncia, escolha de vocabulário e assim por diante (KINNUNEN; LI, 2010).

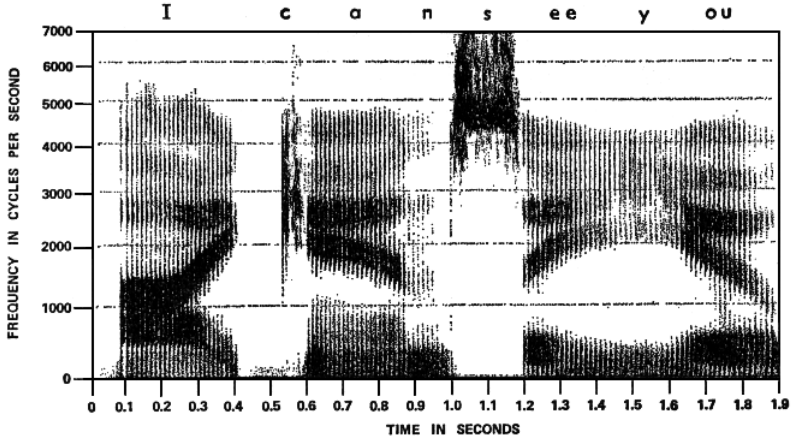
Figura 2 – Diagrama de blocos para produção da voz humana.



Fonte: Mahendru (2014)

Na Figura 3, têm-se o espectrograma da variação da frequência no tempo de uma pessoa falando a frase "*I can see you*" em inglês ou "eu vejo você" (tradução livre). É interessante notar como há uma variação na frequência produzida pelo locutor, possuindo partes com baixíssimas frequências como na produção do fonema "ou" em inglês e altas frequências na produção do som de "s".

Figura 3 – Espectrograma da variação da frequência no tempo da frase "*I can see you*".



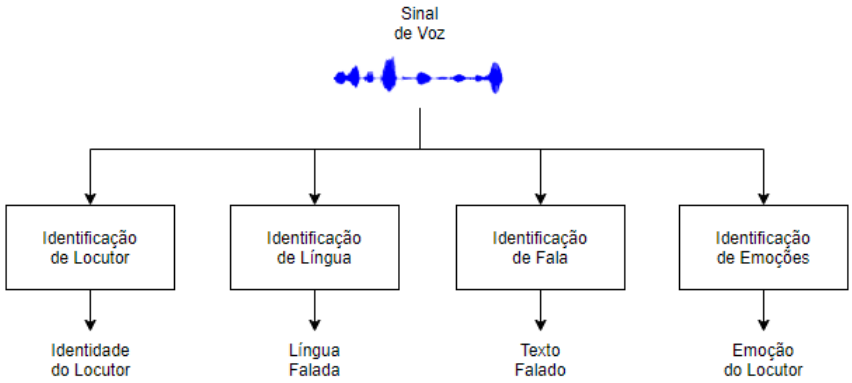
Fonte: Truax e Schafer (1999)

3.3 RECONHECIMENTO DE VOZ

O reconhecimento de locutor, assim como de emoções estereotipadas está inserido dentro de uma área mais ampla que o é o reconhecimento de voz. Algumas de suas áreas são: identificação de números, de locutor, de idioma, de sotaque, de texto. Por exemplo, pode-se aplicar um reconhecimento de idioma para saber em qual língua foi gravado um certo vídeo que pode estar disponível na internet. O reconhecimento de palavras é muito utilizado em centrais telefônicas, onde pode-se falar coisas como "cartão" e ser encaminhado para o setor de cartões de uma determinada empresa. Outra aplicação está em automação residencial, onde pode-se, por exemplo, acionar as luzes de um certo ambiente com um simples comando falado de "ligar as luzes". Reconhecimento numérico é parecido com o de palavras e é utilizado para identificação de senhas de acesso, por exemplo (SILVA, 2015). Na Figura 4, pode-se ver alguns exemplos.

Sistemas de reconhecimento de voz, podem ser divididos em dois grandes grupos: dependentes de texto e independentes de texto. Em sistemas dependentes de texto (mais utilizado por empresas, por exemplo aquelas que possuem algum tipo de telemarketing), as frases de reconhecimento são fixas ou conhecidas de antemão. Por exemplo, o usuário pode ser solicitado a falar uma sequência de números conhecidos previamente (uma senha, por exemplo). Em sistemas independentes de texto, não há restrições nas palavras que os falantes podem usar (muito utilizado na área forense e mais recententemente no reconhecimento de palavras por um celular). Assim, os enunciados de referência (o que é falado no treinamento) e a verificação (o que é proferido em uso real) podem ter conteúdo completamente diferente, e o sistema de reconhecimento deve levar em conta esses descompassos fonéticos. O reconhecimento independente de texto é o mais desafiador das duas tarefas (KINNUNEN; LI, 2010).

Figura 4 – Tipos de Identificação



Fonte: Autor

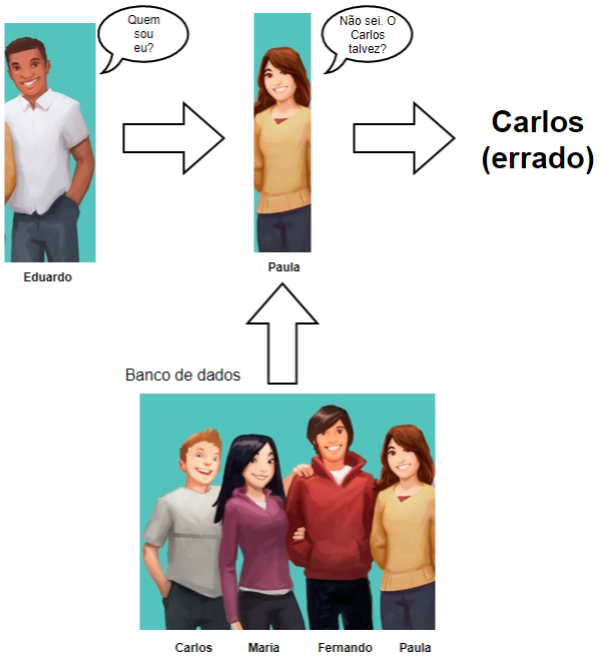
Para o escopo desse trabalho, o foco será o de reconhecimento de locutores e emoções independente do texto pronunciado pelos locutores, ou seja, não se sabe previamente quais palavras serão utilizadas pelo locutor, mas mesmo assim, deseja-se descobrir quem está falando e sua emoção.

3.3.1 Reconhecimento de um Locutor

Para que haja o reconhecimento de locutor, seja ele por um humano ou robô, é necessário que antes haja um treinamento. Por exemplo: se ninguém conhece o Eduardo em um determinado grupo de pessoas e, ao ouvirem sua voz pela primeira vez e serem indagados a dizer quem está falando, eles não saberiam dizer quem é o dono da voz com certeza e, poderiam inclusive, tentar adivinhar falando o nome de pessoas conhecidas por eles (banco de dados de pessoas conhecidas pelo grupo). Porém, se o Eduardo se introduzir ao grupo e criar um vínculo de amizade, eles gravarão sua voz, sotaque e estilo de fala. Dessa vez, ao serem indagados quem é o locutor, o grupo provavelmente conseguiria detectar que é o Eduardo. Um exemplo ilustrativo foi feito para ajudar o entendimento. Na Figura 5 o Eduardo não faz parte do grupo e não é reconhecido pelo mesmo. Após se apresentar e se enturmar e, agora, pertencer ao grupo, o mesmo o reconhece, como pode ser visto na Figura 6. Esse exemplo ilustrativo demonstra a importância da fase de treinamento do sistema antes de sua aplicação.

Em termos práticos para um módulo auditivo, durante a fase de treino, o sistema vai “aprender” a classificar as características extraídas da voz, separá-las em grupos por suas semelhanças e armazená-las em um banco de dados do sistema. Dessa forma, cada um dos grupos de seu banco de dados gerado, representariam um possível locutor que o sistema seria capaz de reconhecer.

Figura 5 – Antes do treinamento para reconhecimento de locutor.

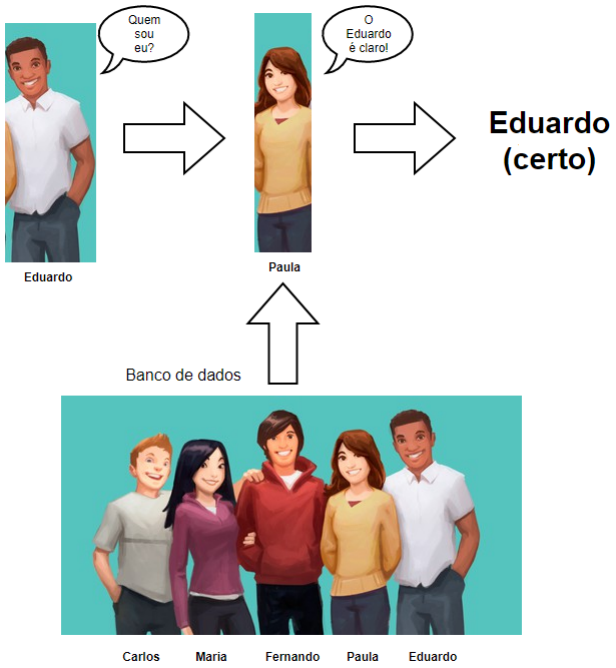


Fonte: www.lds.org modificado pelo autor

A escolha de um locutor, na prática, pode não ser tão direta como pode parecer na Figura 6. Quando as características são extraídas da voz do locutor e passam por todo o sistema, é dada uma nota de probabilidade de ser cada um dos locutores para os quais o robô foi treinado para reconhecer. Ou seja, para o robô a voz não pertence ao Eduardo, mas sim, ela tem uma maior probabilidade de ser do Eduardo do que do Fernando. As diferentes técnicas possíveis para o reconhecimento de um locutor têm em comum o objetivo de tentar aumentar a chance do locutor correto possuir uma maior nota do que os demais, ou seja minimizar a probabilidade de erros de verificação (CAMPBELL JR, 1997). Um exemplo ilustrativo pode ser visto na Figura 7.

O processo para o reconhecimento de locutor requer algumas etapas como fica explícito no fluxograma da Figura 8. Primeiramente ocorre a fase de treinamento. A voz passa por um extrator de características que transformará os sinais de áudio em vetores contendo essas características. Um classificador irá separar e "aprender" a identificar cada um dos tipos diferentes de voz e, dessa maneira, gerar um banco de dados. A segunda parte é a verificação. O sinal de áudio irá passar pela extração de características e um classificador irá dizer, baseado no banco de dados que ele possui, quem está falando. Vale a ressalva que, toda essa análise também é válida para o reconhecimento de emoções estereotipadas, cuja a diferença está que, no banco de dados, estarão emoções ao invés de locutores.

Figura 6 – Após o treinamento para reconhecimento de locutor.



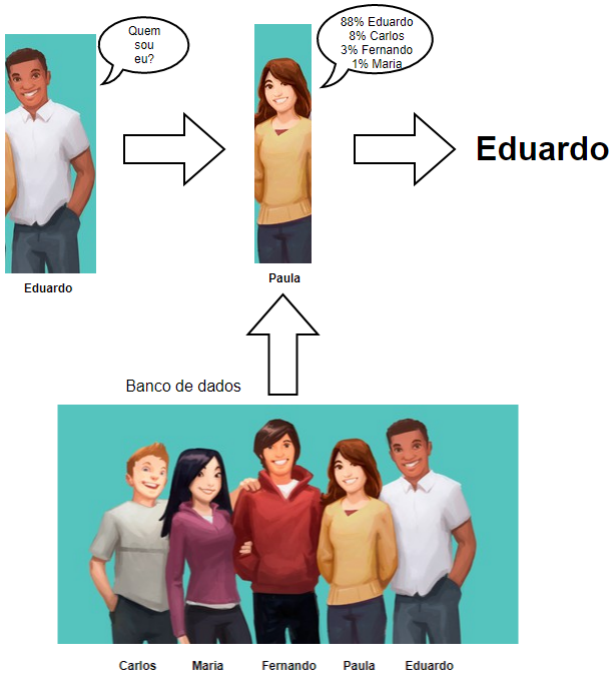
Fonte: www.lds.org modificado pelo autor

3.3.2 Extração de Características

Para que possa haver o reconhecimento de locutor por parte de um módulo artificial auditivo é necessário extrair características da voz de quem está falando. Para aplicar ferramentas matemáticas sem perda de generalidade, o sinal de fala pode ser representado por uma sequência de vetores de características. Isso significa que um trecho de fala é dividido em diversos curtos trechos de pequena duração, como por exemplo uma palavra ou curtos segmentos de 1 a 3 segundos de duração e, de cada trecho, é extraído um vetor com esses atributos. O conjunto desses vetores forma o discurso de um locutor. O objetivo da seleção de características é encontrar uma transformação para um espaço de características (onde existe o vetor de características) de dimensões relativamente baixas, que preserve as informações pertinentes para aplicações e, ao mesmo tempo, possibilite comparações significativas usando medidas simples de similaridade. Em outras palavras, o ideal é possuir um vetor com o menor tamanho possível sem prejudicar na aplicação para a qual o sistema foi projetado. Isso decorre do fato que um número maior de atributos não necessariamente significa maior precisão, além de aumentar o custo computacional do processo (CAMPBELL JR, 1997).

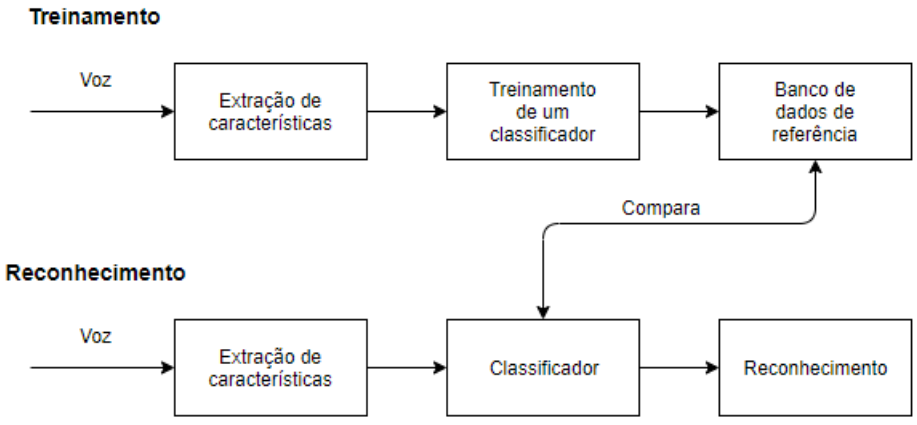
Existem várias técnicas de extração de características para sinais de voz. Geralmente elas são dividas entre: características espectrais de tempo curto (*short-term spectral features*), características de fonte da voz (*voice source features*), características espectro-

Figura 7 – Reconhecimento de locutor.



Fonte: www.lds.org modificado pelo autor

Figura 8 – Fluxograma para o reconhecimento de um locutor.



Fonte: Autor

temporais (*spectro-temporal features*), características prosódicas (*prosodic features*) e características de alto-nível (*high-level features*) (SILVA, 2015). O MFCC (*Mel Frequency Cepstrum Coeficient*), utilizado nesse trabalho, é um método de extração de características espectrais de tempo curto.

3.4 *MEL FREQUENCY CEPSTRAL COEFFICIENTS* (MFCC) COM DELTA E DELTA-DELTA

Para o reconhecimento de um locutor, é importante dividir sua fala em quadros (*frame*) e extrair características de cada um deles para que possam capturar as características específicas do locutor. Muitas dessas características foram investigadas na literatura. Os Coeficientes de Predição Linear (*Linear Prediction Coefficients* ou LPCs) receberam uma atenção especial a esse respeito, pois são derivados diretamente do modelo de produção da fala do falante. Assim também coeficientes de Predição Linear Perceptual (*Perceptual Linear Prediction* ou PLP) como estes são baseados no processamento perceptivo e auditivo humano (TOGNERI; PULLELLA, 2011).

No entanto, nas últimas décadas, as características baseadas em espectros, mais tipicamente derivadas da aplicação direta da Transformada de Fourier, tornaram-se populares. Segundo a literatura, elas são igualmente bem sucedidas quando aplicadas ao reconhecimento de um locutor. Essas características são Coeficientes Cepstrais com espaçamento de frequências Mel *(Mel-Frequency spaced Cepstral Coefficients* - MFCC) e seu sucesso surge do uso do processamento do banco de filtros espaçados em Mel (baseado na percepção humana) da Transformada de Fourier e da robustez específica (para o meio ambiente) e da flexibilidade que pode ser obtida usando a análise cepstral (TOGNERI; PULLELLA, 2011).

MFCC são coeficientes gerados pela representação de uma parcela do espectro de potência do som baseado na transformada discreta do cosseno de um espectro na escala não-linear de frequências em Mel. Eles são amplamente utilizados em reconhecimento de padrões como gêneros de músicas semelhantes ou na detecção de palavras (TOGNERI; PULLELLA, 2011).

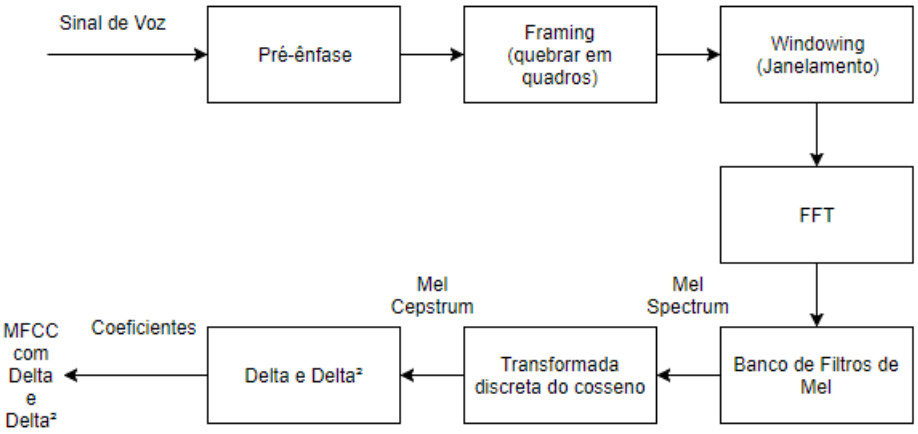
De acordo com Chakraborty (2014), Hasan et al. (2004), Martinez, Perez e Escamilla (2012), pode-se utilizar o MFCC para a identificação de locutores. Segundo Togneri e Pullella (2011), a obtenção do MFCC requer uma série de passos e equações, como pode ser visto na Figura 9. As etapas para o cálculo dos MFCC serão apresentadas à seguir.

3.4.1 Pré-ênfase

Primeiramente, passa-se o sinal da fala por um filtro cujo o objetivo é de enfatizar as altas frequências (aumentar a energia das altas frequências) para compensar o processo de produção da fala humana que tende a atenuar tais frequências. (TOGNERI; PULLELLA, 2011). Um simples filtro passa-alta de primeira ordem que pode ser utilizado:

$$X_s[n] = X_e[n] - \alpha X_e[n-1], \tag{3.1}$$

Figura 9 – Diagrama de blocos para a obtenção do MFCC.



Fonte: Autor

onde X_e é o vetor com os dados do áudio de entrada, $X_s[n]$ representa a saída e α é um valor entre $0,9$ e $1,0$ (CHAKRABORTY, 2014). Seguindo o exemplo de Togneri e Pullella (2011), foi utilizado um $\alpha = 0,97$.

3.4.2 Framing

O sinal de fala, no domínio do tempo, é dividido em segmentos de duração fixa denominados quadros (ou *frames*), de maneira que haja sobreposição de dois quadros adjacentes e, assim, nenhuma informação seria perdida. Valores de duração típicos para *frames* de sinais de áudio são de 15 ms a 40 ms. É necessário haver quadros muito pequenos para que quase nenhuma mudança estatística significativa ocorra entre o *frame* anterior e o próximo. Dessa maneira, os sinais contidos em cada um dos quadros são classificados como ondas quasi-estacionárias, permitindo, assim, que se possa fazer suas conversões para o domínio da frequência (HASAN et al., 2004).

3.4.3 Janelamento por Hamming:

Cada frame é multiplicado por uma função de janelamento (*windowing*). O janelamento é necessário para suavizar o efeito de usar um segmento de tamanho finito para a extração de características, as amostras de cada quadro (*frame*) são atenuadas nas bordas iniciais e finais a partir da multiplicação pela função janela. Como a maioria das características é de natureza espectral, a transformada de Fourier é empregada (como será visto na subseção 3.4.4) e o efeito multiplicativo da função de janela no domínio do tempo é convolutivo no domínio espectral. A função de janela cônica cria um espectro mais suave e menos distorcido (por artefatos). Sem uma função de janela específica, o

padrão resultante da operação de enquadramento teria um efeito de uma janela retangular que geraria artefatos espectrais indesejáveis. Qualquer uma das funções de janela usadas em projetos de filtro digital FIR pode ser implantada, sendo o janelamento por Hamming o mais popular (TOGNERI; PULLELLA, 2011).

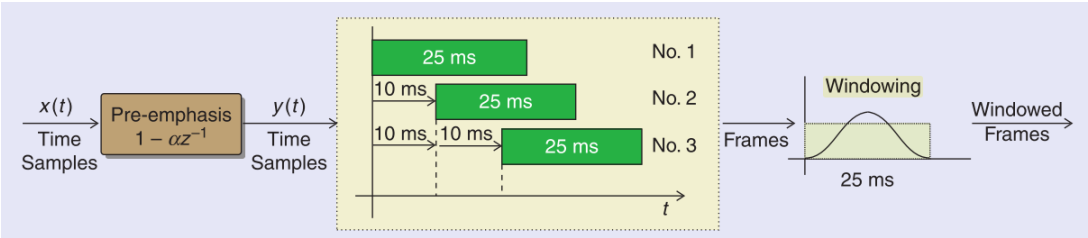
Cada um dos *frames* gerados na etapa anterior (subseção 3.4.2) passa, então, por uma Janela de Hamming, dada pela expressão matemática:

$$w(n) = 0,54 - 0,46\cos(\frac{2\pi n}{N-1}), \tag{3.2}$$

onde $0 \leq n \leq N-1$, N é número de amostras de cada *frame* e as constantes de valor $0,54$ e $0,46$ são típicos desse tipo de janela. A janela de Hann, por exemplo, utilizam duas constantes de valor igual a $0,5$.

A Figura 10 ilustra um sinal que passa pela pré-ênfase, *framing* e janelamento, onde pode-se observar, por exemplo a sobreposição de quadros adjacentes.

Figura 10 – Pré-ênfase, *framing* e janelamento.



Fonte: Togneri e Pullella (2011)

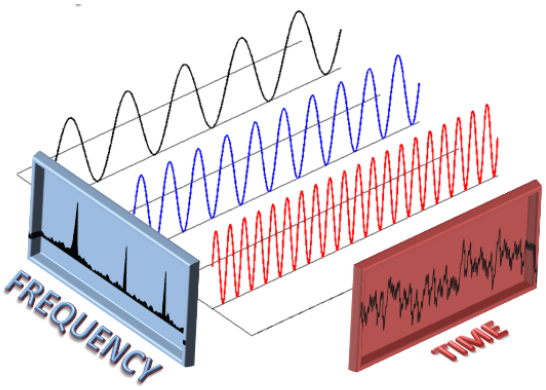
3.4.4 Transformada Rápida de Fourier (FFT)

A transformada rápida de Fourier (*Fast Fourier Transform* ou FFT) é um algoritmo computacional para a execução rápida de transformada discreta de Fourier (*Discrete Fourier Transform* ou DFT) e existem diversas técnicas e algoritmos implementados para a sua rápida execução (WAHYUNI, 2017). Um dos mais utilizados algoritmos para se executar a FFT é o algoritmo de Cooley–Tukey, cujo o pseudo-código pode ser visto na Figura 63 localizado no Anexo A.

Cada um dos *frames* que passaram através da janela de Hamming são convertidos para o domínio da frequência (espectro), usando a transformada rápida de Fourier. Se, por exemplo, uma FFT de 512 pontos é aplicada, então 256 valores espectrais complexos uniformemente espaçados de 0 a $F_s/2$ (onde F_s é a frequência de amostragem) são produzidos (ignoram-se os valores espelhados). No processamento de fala, a informação de fase é ignorada e somente a magnitude da FFT é considerada (TOGNERI; PULLELLA,

2011). A Figura 11 ilustra de forma didática a diferença entre a representação de um sinal no tempo e na frequência (espectro).

Figura 11 – Ilustração da representação de um sinal no tempo e na frequência.



Fonte: http://www.bsic.it

3.4.5 Densidade Espectral de Potência Estimada a Partir da FFT (Periodograma)

A estimativa espectral de potência baseada em periodograma é dada por:

$$P_i(k) = \frac{1}{N}|Y_i(k)|^2, \tag{3.3}$$

onde $Y_i(k)$ são os coeficientes da FFT e N é número de amostras de cada *frame* (LALITHA et al., 2015).

3.4.6 Processamento por Banco de Filtros Mel

A escala de Mel relaciona a frequência percebida, ou o *pitch*, de um tom puro com sua frequência medida real. Os seres humanos são muito melhores para discernir pequenas mudanças no tom em baixas frequências do que em altas frequências, uma vez que a a audição humana exibe compressão logarítmica (TOGNERI; PULLELLA, 2011). Um banco de filtros triangulares linearmente espaçados na frequência de Mel é utilizado para se dar maior destaque para as frequências mais importantes da voz (Figura 12). Para calcular as energias do banco de filtros de Mel, multiplica-se cada banco de filtro com o espectro de potência e soma-se os coeficientes (HASAN et al., 2004).

A conversão de frequência em Hz para frequência em Mel é dada por:

$$M = 1125\ln(1 + F/700), \tag{3.4}$$

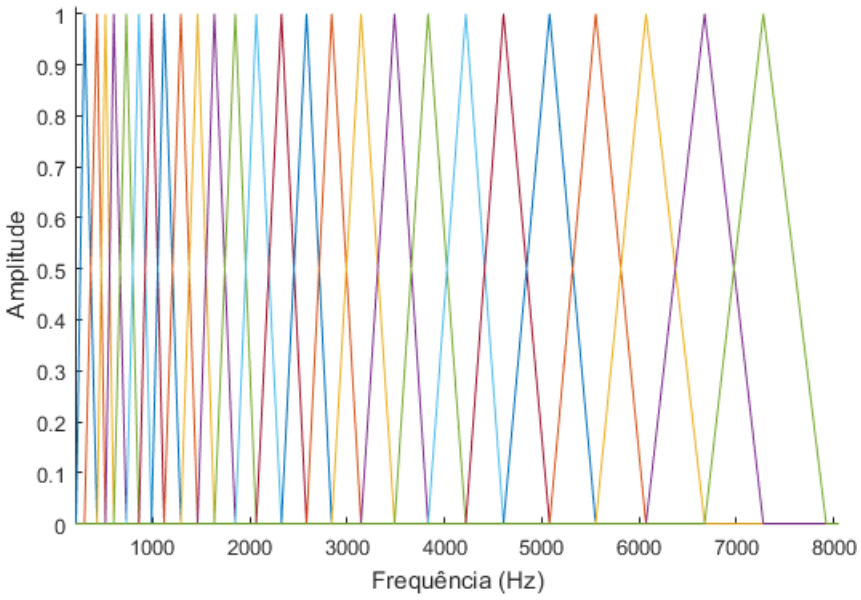
e a conversão de frequência em Mel de volta para frequência em Hz é dada por:

$$F = 700(\exp(M/1125) - 1), \tag{3.5}$$

onde F é a frequência em Hz e M é a frequência na escala de Mel (WAHYUNI, 2017).

Esse processo mapeará as potências do espectro que passou pelo banco de filtros de Mel. Isso resultará em um vetor com um número de potências igual ao número de filtros utilizados no banco. Por fim, tira-se o logaritmo natural das potências. Após passar pelos filtros de Mel, obtêm-se os coeficientes espectrais de freqüência de Mel (*Mel frequency spectrum coefficients* ou MFSC) (MUDA; BEGAM; ELAMVAZUTHI, 2010).

Figura 12 – Banco de filtros triangulares espaçados linearmente na escala de Mel.



Fonte: Autor

3.4.7 Transformada Discreta do Cosseno (DCT)

Os coeficientes espectrais de frequência de Mel (MFSC) são convertidos para um domínio semelhante ao tempo chamado quefrência (*quefrency*), onde se obtêm o cepstrum (semelhante ao espectro) usando a transformada discreta do cosseno (*discrete cosine transform* ou DCT), gerando os coeficientes cepstrais de frequência de Mel (*Mel frequency cepstral coefficients* ou MFCC). Do segundo para o décimo sexto coeficientes são mantidos (15 coeficientes). Outras configurações como a utilização de 12, 13 e 14 coeficientes também são possíveis (MUDA; BEGAM; ELAMVAZUTHI, 2010).

3.4.8 Delta e Delta-Delta

Para uma maior precisão na identificação do locutor, além dos coeficientes cepstrais de Mel (MFCC), os parâmetros delta e delta-delta são também calculados. O parâmetro delta mede a variação (primeira derivada do MFCC) da mudança da fala (ou seja, a velocidade da fala) e delta-delta mede a variação (segunda derivada do MFCC) da mudança da velocidade da fala (ou seja, a aceleração da fala) (MUDA; BEGAM; ELAMVAZUTHI, 2010). Esses outros coeficientes são adicionados ao vetor de características para adicionar informações da dinâmica da fala que podem ter sido perdidas na mudança de cada *frame* (TOGNERI; PULLELLA, 2011).

A primeira derivada, ou delta (Δ), pode ser aproximada pela expressão matemática, onde tipicamente $P = 2$ e $C[n]$ é o enésimo coeficiente da MFCC:

$$\Delta C[n] = \frac{\sum_{p=1}^{P} p(C[n+p] - C[n-p])}{2\sum_{p=1}^{P} p^2}. \tag{3.6}$$

A segunda derivada, delta-delta ou delta quadrado (Δ^2), pode ser aproximado pela mesma expressão que a anterior (Equação 3.6), só que com os elementos da primeira derivada no lugar dos coeficientes cepstrais de Mel (TOGNERI; PULLELLA, 2011, p.29):

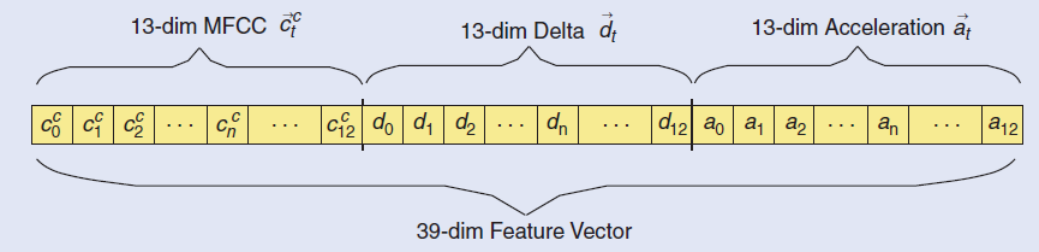
$$\Delta^2 C[n] = \frac{\sum_{p=1}^{P} p(\Delta C[n+p] - \Delta C[n-p])}{2\sum_{p=1}^{P} p^2}. \tag{3.7}$$

Depois de calcular o MFCC, delta e delta-delta, 45 coeficientes com as característica da fala são obtidos para serem usados em um classificador de reconhecimento de padrões, sendo 15 do MFCC, 15 do delta e 15 do delta-delta. Gera-se um vetor de saída cujo o resultado é a concatenação dos coeficientes ceptrais com suas derivadas temporais, tal como na Figura 13. Um exemplo desses coeficientes gerados para um mesmo locutor falando duas palavras distintas pode ser visto na Figura 14. Um outro exemplo, com as projeções 2D dos coeficientes cepstrais de Mel de dois locutores distintos pode ser visto na Figura 15. Nessa figura, cada ponto é a projeção bidimensional dos coeficientes de uma palavra. Ao todo, cada locutor falou 80 palavras. Pode-se observar que cada locutor possui um certo padrão. Esses coeficientes encontrados alimentarão um classificador, que tentará encontrar esses padrões de variação dos coeficientes e, dessa maneira, distinguir quem é o locutor falante ou a emoção transmitida por sua voz.

3.4.9 Silêncio

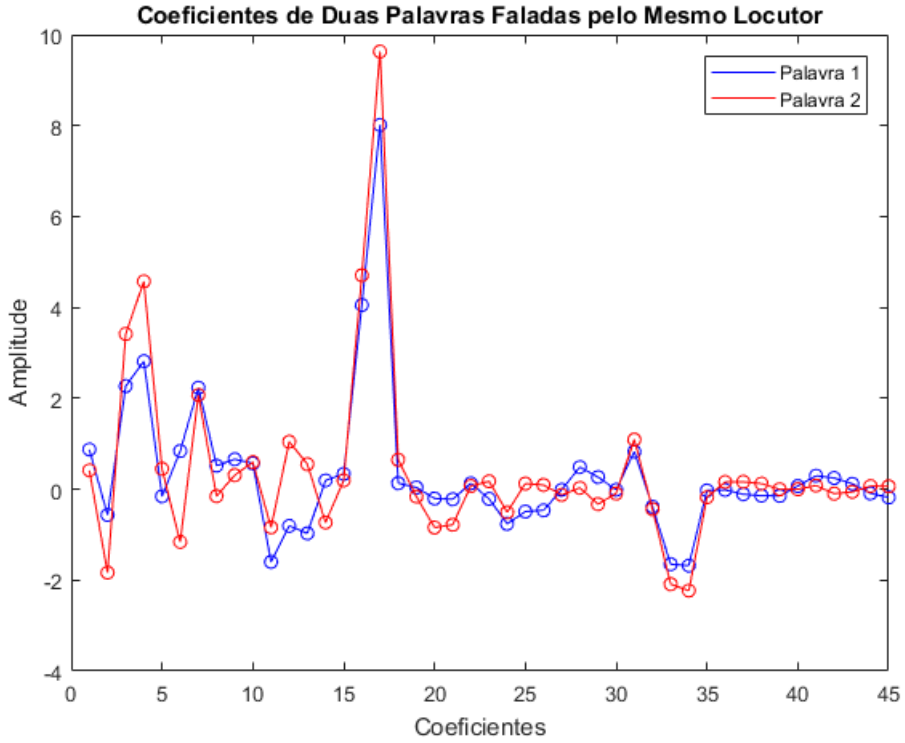
O silêncio é prejudicial na geração do vetor com características da voz do locutor, pois não possui nenhum tipo de informação relevante e atrapalham a obtenção do MFCC. Para contornar tal situação, é muito comum que seja elaborado um algorítimo de detecção

Figura 13 – Concatenação dos coeficientes do MFCC com delta e delta-delta



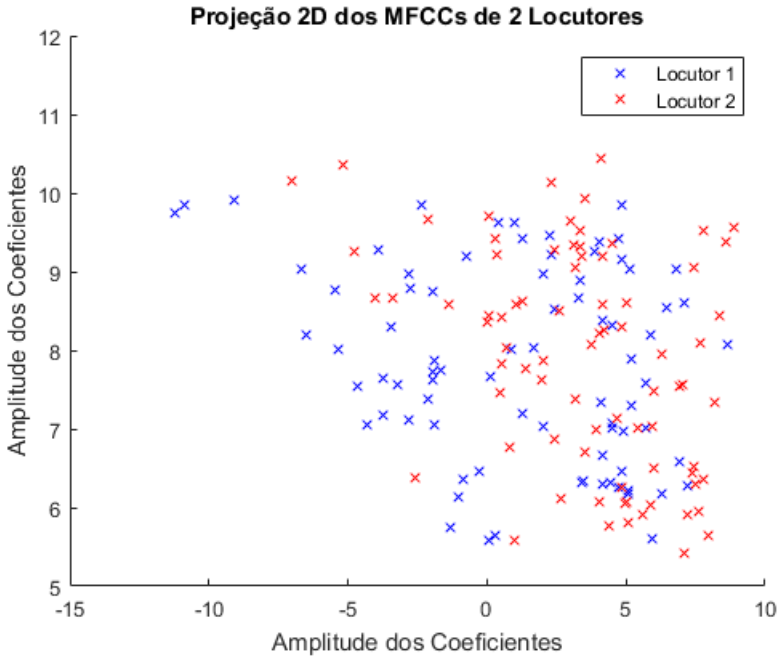
Fonte: Togneri e Pullella (2011)

Figura 14 – 45 coeficientes gerados para o mesmo locutor dizendo duas palavras distintas.



Fonte: Autor

Figura 15 – Projeção 2D dos MFCCs de dois locutores distintos falando 80 palavras.



Fonte: Autor

de silêncios no áudio, onde o mesmo será retirado, garantindo que qualquer instante do áudio possua dados da voz a ser reconhecida. Existem diferentes técnicas para realização de tal procedimento, sendo a mais simples a verificação e eliminação dos trechos onde a amplitude do áudio é muito baixa (TOGNERI; PULLELLA, 2011).

3.5 BANCO DE FALAS

Um banco falas (*speech corpus* ou *speech database* em inglês) é um banco de dados contendo arquivos ou vetores de áudios de fala, podendo ter frases completas ou simplesmente palavras. Na tecnologia de fala são usados, entre outras coisas, para criar modelos acústicos (que podem então ser usados como um mecanismo de reconhecimento de fala ou emoções, por exemplo). Na linguística, bancos de fala são usados para pesquisar fonética, análise de conversação, dialectologia entre outros (RASO; MELLO, 2014).

Um banco de falas (também chamado de banco de vozes) pode ser gravado previamente com equipamentos de alta qualidade, ou in loco. Por exemplo, se já se conhece quem serão as pessoas na identificação, ou se quem elas são não importa, essa gravação pode ser feita previamente, com maior qualidade. Se é para acrescentar um novo locutor a um sistema já existente, por exemplo, seria muito inconveniente ter que levar esse novo locutor para uma sessão de gravação com equipamentos profissionais, portanto a gravação seria feita pelo próprio módulo in loco.

Com o avanço das tecnologias de fala, cada vez mais bancos estão sendo elaborados. Qualquer pessoa que tenha microfones adequados para a gravação pode levantar seu banco de voz. Dentre os bancos de dados disponíveis, pode-se citar um banco de dados brasileiro, o Projeto ALIP (GONCALVES, 2003) desenvolvido pela Unesp. É um banco de voz que foca-se no português brasileiro falado na região de São José do Rio Preto e tem como o objetivo documentar o português falado nessa região do interior de São Paulo.

Para o caso de emoções, o banco de falas também pode ser chamado de banco de emoções (*emotion database* em inglês). Para a identificação de emoções, possuir um banco é necessário para o treinamento e avaliação do classificador. Como exemplo de banco de emoções, pode-se citar o *Surrey Audio-Visual Expressed Emotion* (SAVEE) (HAQ; JACKSON, 2011), disponível online: <http://kahlan.eps.surrey.ac.uk/savee/>, que disponibiliza 4 locutores interpretando 7 diferentes emoções falando inglês britânico: raiva, nojo, medo, alegria, neutralidade, tristeza e surpresa, gravadas com uma frequência de amostragem de 44100 Hz.

Um outro banco de emoções bastante utilizado na literatura é o *Berlin Emotion Database* (Emo-DB) (BURKHARDT et al., 2005), disponível online: <http://emodb.bilderbar.info/start.html>. Ele foi utilizado nos trabalhos dos autores Lalitha et al. (2015), Haq, Jackson e Edge (2008), Lugger e Yang (2008), entre outros. Esse banco possui 10 locutores, 5 homens e 5 mulheres, falando em alemão e interpretando 7 emoções: raiva, nojo, medo, alegria, calma, tristeza e tédio, gravado com uma taxa de amostragem de 16000 Hz.

3.6 INTELIGÊNCIA ARTIFICIAL

Muitas atividades mentais humanas tais como resolver problemas matemáticos, usar a razão, dirigir um carro, escrever programas de computador e entendimento de uma língua demandam inteligência. Nas últimas décadas, vários sistemas computacionais passaram a realizar tais tarefas. Existem sistemas computacionais que podem fazer diagnóstico de doenças, planejar a síntese de uma complexa molécula orgânica, resolver complexas equações diferenciais, analisar circuitos eletrônicos, entender (ainda que de forma limitada) o discurso humano ou ainda mesmo escrever simples códigos de computador. Pode-se dizer que tais sistemas possuem um certo grau de inteligência artificial (IA) ou *artificial intelligence* (AI) em inglês. Em outras palavras, qualquer dispositivo que perceba seu ambiente, através de sensores (reais ou simulados) e realize ações que maximizem sua chance de atingir seus objetivos com sucesso pode ser um IA (NILSSON, 1980).

De acordo com Nilsson (1980), existem diversas aplicações para as quais se usam IA. Alguns exemplos incluem: representação do conhecimento, planejamento, aprendizado, diferentes processamentos, etc. Nesse trabalho, foi focado no aprendizado, pois ele é

necessário no processo de identificação de locutores e emoções.

3.6.1 Aprendizado de Máquina

Aprendizado é o processo de adquirir novos conhecimentos declarativos, desenvolver novas habilidades motoras e cognitivas através de prática e instruções ou descobrimento de novos fatos através de observação e experimentação. Desde os primórdios da era da computação, pesquisadores têm tentado atribuir tais capacidades a computadores. Fazê-lo foi (e continua sendo) um dos mais desafiadores e fascinantes objetivos na área de inteligência artificial. O estudo e modelos computacionais de processos de aprendizado em suas múltiplas manifestações constituem a matéria de aprendizado de máquinas (*machine learning* ou ML) (MICHALSKI; CARBONELL; MITCHELL, 1983). Para a computação, aprender seria progressivamente melhorar a performance para realizar uma determinada tarefa sem haver uma programação explicita para tal.

O aprendizado de máquina geralmente se refere às mudanças nos sistemas que executam tarefas associadas à inteligência artificial. Tais tarefas envolvem reconhecimento, diagnóstico, controle de robô, predição, etc. As "mudanças" podem ser melhorias para sistemas já em execução ou síntese desde o início de novos sistemas (NILSSON, 2005).

De acordo com Deng e Li (2013), o aprendizado de máquina pode ser dividido em algumas categorias, tais como:

- Aprendizado supervisionado (*supervised learning*): O computador é apresentado com entradas de exemplo e suas saídas desejadas, dadas pelo programador, e o objetivo é aprender uma regra geral que mapeia entradas para saídas.
- Aprendizado semi-supervisionado (*semi-supervised learning*): O computador recebe apenas um sinal de treinamento incompleto: um conjunto de treinamento com algumas (muitas vezes muitas) saídas desejadas ausentes. É um caso especial do aprendizado supervisionado.
- Aprendizado ativo (*active learning*): O algoritmo de aprendizagem é capaz de consultar interativamente o usuário (ou alguma outra fonte de informação) para obter as saídas desejadas em novos pontos de dados. É um caso especial do aprendizado supervisionado.
- Aprendizado reforçado (*reinforcement learning*): Os dados de treinamento (em forma de recompensas e punições) são dados apenas como *feedback* para as ações do programa em um ambiente dinâmico, como dirigir um veículo ou jogar um jogo contra um adversário. É um caso especial do aprendizado supervisionado.
- Aprendizado não supervisionado (*unsupervised learning*): Nenhum par entrada/saída é dado ao algoritmo de aprendizado, deixando-o por conta própria para encontrar a

estrutura em sua entrada. O aprendizado não supervisionado pode ser um objetivo em si (descobrir padrões ocultos nos dados) ou um meio para um fim (aprendizado de recursos).

Para o escopo desse trabalho, apenas o aprendizado supervisionado será focado.

De acordo com Deng e Li (2013), outra forma de classificar o aprendizado de máquina é em classificadores ou regressão:

- Classificadores (*classifiers*): As entradas são divididas em duas ou mais classes e o sistema deve produzir um modelo que atribua entradas não vistas a uma ou mais dessas classes.
- Regressão (r*egression*): É um conjunto de processos estatísticos para estimar as relações entre variáveis. As saídas são contínuas e não discretas.

Nesse trabalho, apenas classificadores serão considerados.

3.6.2 Aprendizado Supervisionado

Na aprendizagem supervisionada, o conjunto de treinamento consiste em pares de entradas e saídas obtidas de uma distribuição conjunta. O objetivo do aprendizado é novamente a minimização de riscos empíricos do problema de classificação (DENG; LI, 2013).

Em outras palavras, no aprendizado supervisionado, cada exemplo é um par que consiste em um objeto de entrada (normalmente um vetor) e um valor de saída desejado (também chamado de sinal de supervisão). Um algoritmo de aprendizado supervisionado analisa os dados de treinamento e produz uma função inferida, que pode ser usada para mapear novos exemplos. Um cenário ideal permitirá que o algoritmo determine corretamente as classe para todas as instâncias. Isso requer que o algoritmo de aprendizado generalize, a partir dos dados de treinamento, situações em que hajam novos e diferentes dados de entrada.

3.6.3 Classificadores

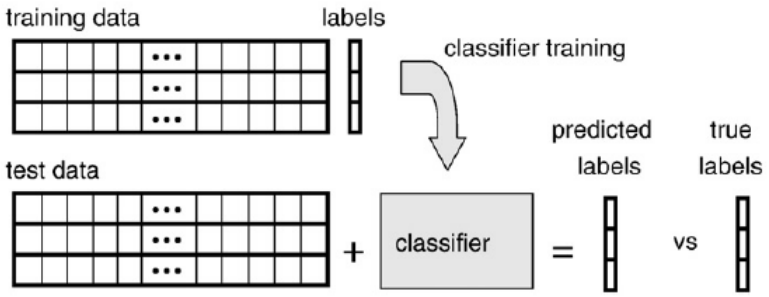
Um classificador (*classifier* em inglês) é uma função que utiliza os valores de vários recursos (variáveis independentes) em um exemplo (o conjunto de valores de variáveis independentes) e prevê a classe à qual esse exemplo pertence (a variável dependente). Uma vez treinado, o classificador pode ser usado para determinar se as características extraídas contêm informações sobre a classe do exemplo. Esse relação é testada usando o classificador treinado em um conjunto diferente de exemplos, os dados de teste. Intuitivamente, a ideia

é que, se o classificador realmente capturou a relação entre as características extraídas e as classes, ele deveria ser capaz de prever as classes de exemplos que não havia visto antes. A suposição típica para os algoritmos de aprendizado do classificador é que os exemplos de treinamento (e teste) são desenhados independentemente de uma "distribuição de exemplo"; ao julgar um classificador em um conjunto de testes, estamos obtendo uma estimativa de seu desempenho em qualquer conjunto de testes da mesma distribuição. A medida mais comumente usada para avaliar o desempenho de um classificador no conjunto de testes é sua precisão. Esta é simplesmente a divisão da quantidade de exemplos que foi corretamente prevista pela quantidade de exemplos do conjunto de teste, conforme a Equação 3.8 (PEREIRA; MITCHELL; BOTVINICK, 2009).

$$Precisão = \frac{Acertos}{Total} \tag{3.8}$$

Um exemplo ilustrado pode ser visto na Figura 16 em que um classificador aprendeu com o conjunto de treinamento, exemplos dos quais podem ser usados para prever características de um conjunto de testes. Os resultados previstos são então comparados com os verdadeiros e a precisão do classificador pode ser calculada.

Figura 16 – Treinamento e teste de um classificador.



Fonte: Pereira, Mitchell e Botvinick (2009)

Existem diversos tipos diferentes de classificadores. Como exemplo, pode-se citar: redes neurais artificiais, *support vector machines*, *linear discriminant analysis*, *learning vector quantization*, *Naive-Bayes classifier*, *quadratic classifiers*, *kernel estimation*, etc.

Para o desenvolvimento do escopo desse trabalho, existirá um foco maior em redes neurais artificiais, pois ao compará-la com o resultado obtido utilizando-se outros classificadores como *support vector machines*, por exemplo, houve um resultado superior com as redes neurais em todos os casos testados, como será visto no Capítulo 5.

3.7 REDES NEURAIS ARTIFICIAIS (RNA)

Redes neurais artificiais (RNA) ou *artificial neural networks* (ANN) em inglês são um tipo de classificador que foi inspirado no sistema nervoso central dos animais, embora sua arquitetura e elementos agora sejam completamente diferentes de sua inspiração biológica. São redes de elementos de processamento simples (chamados neurônios) operando em seus dados locais e se comunicando com outros elementos. Na arquitetura de uma rede neural, capaz de resolver problemas não-lineares, geralmente existem pelo menos 3 camadas. Uma camada de entrada, uma ou mais camadas escondidas e uma camada de saída. Todas essas camadas possuem neurônios e eles são conectados entre si. Cada neurônio se conecta pelo menos com um outro neurônio de outra camada. Cada conexão possui um peso, que será seu valor matemático que indicará o grau de importância daquela conexão para a dada rede. Em redes do tipo *feedfoward*, a informação sempre avança pra próxima camada e nunca retorna pros neurônios anteriores. São redes capazes de se adaptar sem serem diretamente programadas para isso. Eles são extremamente úteis em reconhecimento de padrões (SVOZIL; KVASNIEKA; POSPICHAL, 1997).

Em princípio, a rede neural tem o poder de um aproximador universal, isto é, pode realizar um mapeamento arbitrário de um espaço vetorial em outro espaço vetorial. A principal vantagem das redes neurais é o fato de que elas são capazes de usar algumas informações a priori desconhecidas escondidas nos dados (mas elas não são capazes de extraí-las). O processo de "capturar" a informação desconhecida é chamado de "aprendizado de rede neural" ou "treinamento de rede neural". No formalismo matemático, aprender significa ajustar os coeficientes de peso de tal maneira que algumas condições sejam satisfeitas. No treinamento por aprendizado supervisionado, a rede neural conhece o par entra/saída desejada e o ajuste dos coeficientes de peso é feito de tal forma que as saídas calculadas e desejadas sejam as mais próximas possíveis (SVOZIL; KVASNIEKA; POSPICHAL, 1997).

3.7.1 *Multilayer Perceptron*

A rede do tipo *perceptron* é o caso mais simples das redes *feedforward* por não possuírem camada escondida e por só serem capazes de resolver problemas lineares. Todos os neurônios estão interconectados entre si. *Multilayer perceptrons* (MLP) são redes semelhantes às redes *perceptrons*, mas com uma ou mais camadas escondidas, dando a elas um maior poder computacional. São redes do tipo *multilayer feedforward* (MLF) (NILSSON, 2005).

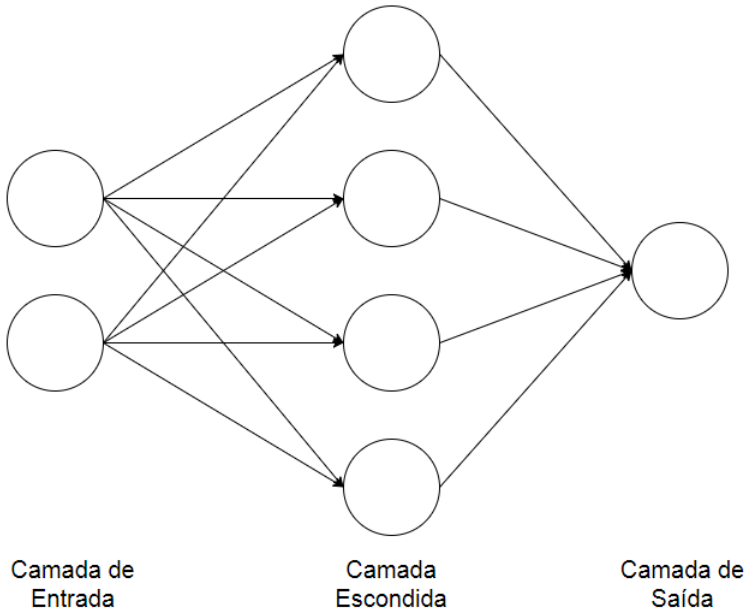
Um exemplo de função de ativação entre seus neurônios, em MLP, é a função sigmoide, que pode ser vista na Equação 3.9. Outras funções de ativação como tangente hiperbólica, softmax e etc também podem ser utilizadas. Geralmente tais redes são treinadas

usando entradas e saídas conhecidas (aprendizado supervisionado). Coloca-se a entrada na rede e observa-se o quão distante o resultado de saída ficou dos resultados gerais. Em redes com treinamento por *backpropagation*, a correção dos pesos, mediante ao erro de saída em relação ao resultado real, é feita de trás para frente, começando da última camada e corrigindo-os até a primeira (SVOZIL; KVASNIEKA; POSPICHAL, 1997).

Uma ilustração didática de um típico MLP pode ser vista na Figura 17. A equação de uma função sigmoide é dada por:

$$f(\xi) = \frac{1}{1 + e^{-\xi}}. \tag{3.9}$$

Figura 17 – Ilustração de uma típica rede neural artificial do tipo *feedforward*.



Fonte: Autor

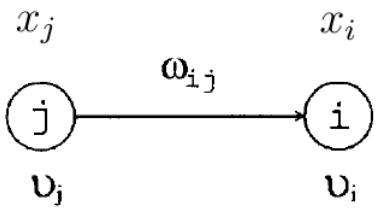
3.7.2 A matemática das RNAs

Uma rede neural MLF (*multilayer feedforward*) consiste em neurônios, que são ordenados em camadas (Figura 17). A primeira camada é chamada de camada de entrada, a última é chamada de camada de saída e quaisquer outras camadas entre essas duas são as camadas ocultas ou escondidas (SVOZIL; KVASNIEKA; POSPICHAL, 1997).

Para a descrição formal dos neurônios, podemos usar a chamada função de mapeamento Γ, que atribui para cada neurônio i um subconjunto $\Gamma(i) \subseteq V$ que consiste de todos os ancestrais do neurônio dado. Um subconjunto $\Gamma^{-1}(i) \subseteq V$ então consiste em todos os predecessores do neurônio dado i. Cada neurônio em uma camada particular é conectado com todos os neurônios na próxima camada. A conexão entre o i-ésimo e o

j-ésimo neurônios é caracterizada pelo coeficiente de peso ω_{ij} e o i-ésimo neurônio pelo coeficiente limiar ϑ_i, como pode ser vista na Figura 18. O coeficiente de peso reflete o grau de importância da conexão dada na rede neural. O valor de saída (atividade) do i-ésimo neurônio x_i é determinado pela Equação 3.10 e pela Equação 3.11. Sabe-se que:

Figura 18 – Conexão entre os neurônios i e j.



Fonte: Svozil, Kvasnieka e Pospichal (1997), modificado pelo autor

$$x_i = f(\xi_i), \tag{3.10}$$

$$\xi_i = \vartheta_i + \sum_{j \in \Gamma^{-1}(i)} \omega_{ij} x_j, \tag{3.11}$$

onde ξ_i é o potencial do neurônio i e a função $f(\xi_i)$ é a chamada função de transferência (a soma na Equação 3.11 é realizada sobre todos os neurônios j transferindo o sinal para o i-ésimo neurônio). O coeficiente limiar pode ser entendido como um coeficiente de peso da conexão com o neurônio j, formalmente adicionado, onde $x_j = 1$ (chamado de *bias* ou viés em português). Uma possível função de transferência é a sigmoide que pode ser vista na Equação 3.9 (SVOZIL; KVASNIEKA; POSPICHAL, 1997).

Na Figura 64, localizado no Anexo A pode-se observar um algoritmo para a implementação de uma RNA simples do tipo *feedfoward* com 3 camadas e função sigmoidal como ativação. Esse código foi desenvolvido em MATLAB.

3.7.3 *Resilient Backpropagation*

Resilient Backpropagation (Rprop) é um método de aprendizado supervisionado de redes neurais por *backpropagation*. Rprop leva em conta apenas o sinal da derivada parcial sobre todos os padrões (não a magnitude), e age de forma independente em cada peso. Para cada peso, se houve uma alteração da derivada parcial da função de erro total em comparação com a última iteração, o valor de atualização para esse peso é multiplicado por um fator $\eta-$ (menor que 1). Se a última iteração produzisse o mesmo sinal, o valor de atualização é multiplicado por um fator de $\eta+$ (maior que 1). Os valores de atualização

são calculados para cada peso da maneira acima e, finalmente, cada peso é alterado por sua própria atualização valor, na direção oposta da derivada parcial desse peso, de modo a minimizar a função de erro total (RIEDMILLER, 1994). O pseudo-código para implementação do Rprop pode ser visto na Figura 65 no Anexo A.

3.7.4 Função de Erros

Redes neurais artificiais são geralmente treinadas para minimizar a função de erros; como exemplos de tais funções para problemas de classificação binária, pode-se citar o erro de entropia cruzada (*cross-entropy error*) e erro quadrático médio (*Mean Squared Error* ou MSE). Na Equação 3.12, pode-se ver como se calcula o MSE.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Z_i - \widehat{Z_i})^2, \tag{3.12}$$

onde $\widehat{Z}$ é o vetor que a RNA previu, Z é o vetor com o resultado real e n é o número de amostras para as quais houve a previsão.

Dado apenas um conjunto de dados de tamanho limitado D, qualquer modelo para uma RNA baseado neste conjunto de dados será influenciado pela escolha particular de D. O desafio de construção de modelo é abstrair a distribuição subjacente da instância D particular das amostras. O problema da memorização do conjunto de dados em vez de identificar a sua distribuição subjacente é conhecido como *overfitting* (DREISEITL; OHNO-MACHADO, 2003).

3.7.5 Múltiplas Redes Neurais Artificiais

A capacidade de generalização é uma das medida do desempenho de redes neurais. A combinação de múltiplas RNAs é usada para alcançar um alto desempenho de reconhecimento de padrões. Observou-se que os conjuntos de padrões reconhecidos erroneamente por diferentes redes não necessariamente se sobrepõem. Isso sugere que diferentes redes potencialmente oferecem informações complementares sobre os padrões a serem reconhecidos e podem ser aproveitados para melhorar o desempenho geral. Em outras palavras, a ideia é não confiar em uma única rede. Em vez disso, a decisão de várias redes são combinadas para obter-se um sistema com alto desempenho. Cada rede será treinada a partir de diferentes pesos iniciais e com uma divisão diferente do primeiro conjunto de dados em conjuntos no treinamento, validação e teste. Uma das possíveis técnicas para se fundir resultados de diferentes redes é a estratégia de combinação média das saídas *(outputs average combination strategy)*. Seria uma simples média aritmética das saídas de cada uma das redes (YU; HU; BAO, 2003).

3.8 OUTROS CLASSIFICADORES

No decorrer do desenvolvimento desse trabalho, alguns outros classificadores (para além das RNAs) são testados. Nessa sessão será comentado, de forma breve, um pouco de cada um desses classificadores.

1. LVQ:

   Learning vector quantization é na computação um algoritmo de classificação supervisionado baseado em protótipo. O LVQ é a contraparte supervisionada dos sistemas de VQ (*vector quantization*). É criado um conjunto de treinamento de vetores de características e, em seguida, são agrupados em um pequeno número de classes (*codebook*). Assim pode-se computar a probabilidade de que o vector pertença ou não ao *codebook* em questão (MARTINEZ; PEREZ; ESCAMILLA, 2012).

2. FCM:

   Fuzzy C-mean é uma técnica de aprendizado supervisionado onde o FCM é uma melhoria em relação ao *clustering k-mean* tradicional, em que um determinado *cluster* pertence apenas a um *codebook* especificado. No FCM, é usado um método de *clustering* de dados bem definido, em que cada ponto de dados é associado a um *cluster* específico com algum grau de associação funcional, o que indica como os pontos de dados são agrupados em um espaço multidimensional consistindo em um número específico de grupos diferentes *clusters* (BANSAL; BHARTI, 2015).

3. ANFIS:

   Adaptive neuro fuzzy inference system é um tipo de rede neural artificial baseada no sistema de inferência *fuzzy*. Uma vez que integra as redes neurais e os princípios da lógica *fuzzy*, tem potencial para capturar os benefícios de ambos em um único ambiente. Seu sistema de inferência corresponde a um conjunto de regras IF-THEN *fuzzy* que possuem capacidade de aprendizado para aproximar funções não-lineares (JANG, 1993).

4. SVM:

   Suport vector machine é um dos classificadores usados para o reconhecimento de emoções. Ele reconhece o padrão e analisa os dados. Ele utiliza de hiper-planos para separar os dados em diferentes classes. Ele separa as características lineares e não lineares do sinal de entrada (DAHAKE; SHAW, 2016).

5. RNA com MEDC:

   Redes Neurais Artificiais com *Mel-Energy spectrum Dynamic Coefficients*. A diferença de MEDC pra MFCC é que depois de se calcular o logaritmo das energias, se computa a primeira e a segunda diferença (WATILE; ALAGDEVE; JAIN, 2017).

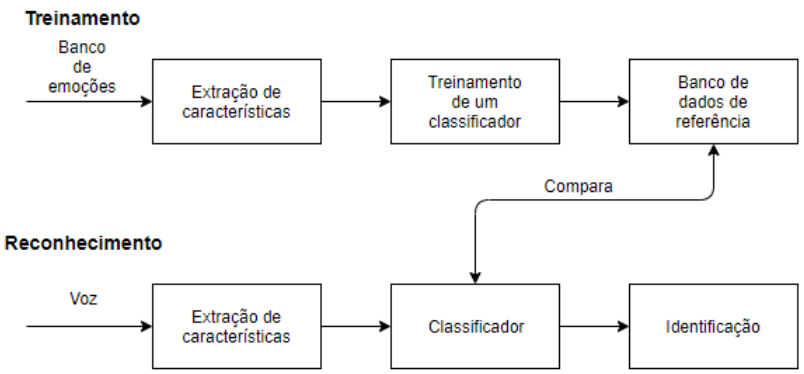
3.9 EMOÇÕES ESTEREOTIPADAS

Para a identificação de emoções estereotipadas, todas as técnicas vistas para locutor são válidas. De acordo com Demircan e Kahramanlı (2014), Lalitha et al. (2015), Dahake e Shaw (2016), MFCC, como método de extração de características, também é válido na identificação de emoções estereotipadas. Os autores Lalitha et al. (2015) e Muda, Begam e Elamvazuthi (2010) comentaram do uso de RNA para a identificação de emoções.

O nome emoções estereotipadas se dá pelo fato de que, para o treinamento do classificador, não se usam emoções que foram gravadas de maneira natural, mas sim de maneira atuada, pois seria muito difícil se conseguir, de maneira natural e com boa qualidade de áudio, tais emoções verdadeiras.

Uma das principais diferenças entre a identificação de um locutor com a de emoções é que, para identificar um locutor, é necessário ter contato prévio com o locutor para gravar sua voz e, assim, poder treinar o classificador. Isso difere da identificação de emoções estereotipadas, onde um banco de vozes com diferentes emoções é gravado previamente, sem precisar necessariamente da presença de um locutor específico. Na Figura 8 vemos como é para reconhecer um locutor e na Figura 19 para identificar uma emoção estereotipada.

Figura 19 – Fluxograma para a identificação de emoções estereotipadas.

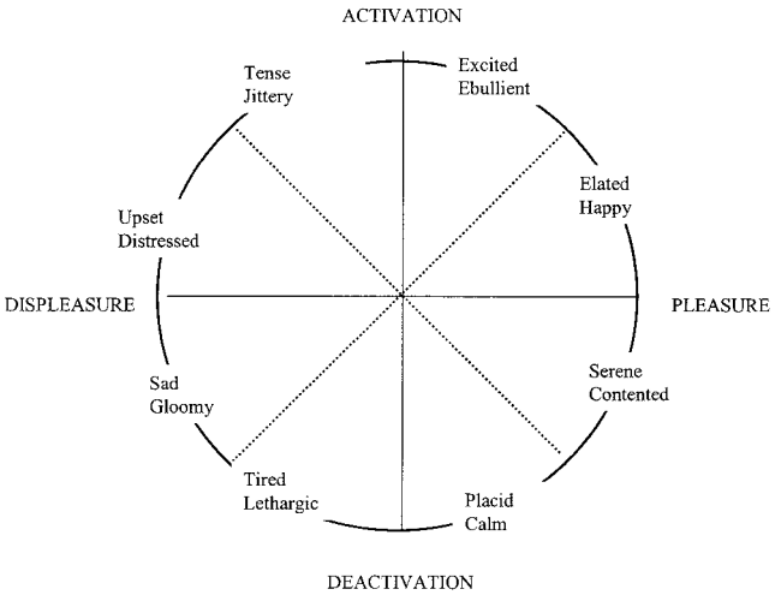


Fonte: Autor

3.9.1 *Core Affect*

A fim de classificar as emoções e, ao mesmo tempo, limitar o número de ações possíveis a serem tomadas por um robô assistivo, propõe-se usar *Core Affect* (RUSSELL, 2003), um mapa cartesiano, onde as emoções são classificadas como sendo agradável ou desagradável em seu eixo horizontal e alta excitação (ativação) ou baixa excitação (desativação) no eixo vertical (Figura 20). Assim, as inúmeras emoções humanas são

Figura 20 – *Core Affect.*



Fonte: Russell (2003)

compiladas em apenas quatro possibilidades, aumentando a precisão do classificador e reduzindo a complexidade das possíveis decisões tomadas pelo robô.

Dessa maneira, alegria seria classificado como agradável com alta excitação (primeiro quadrante), raiva e aversão seriam desagradáveis com alta excitação (segundo quadrante), medo e tristeza seriam desagradável com baixa excitação (terceiro quadrante) e calma seria agradável com baixa excitação (quarto quadrante).

Autores como Lugger e Yang (2008) usam técnicas semelhantes para poder classificar as emoções.

3.10 SISTEMAS DE NAVEGAÇÃO

A identificação da posição uma fonte sonora pode ser considerado um caso de navegação, onde não necessariamente o robô quer se localizar num meio, para poder navegar, utilizando-se de sensores, mas quer localizar o origem de um som, utilizando-se de sensores, nesse caso, microfones. Um exemplo de um sistema de navegação é a navegação hiperbólica.

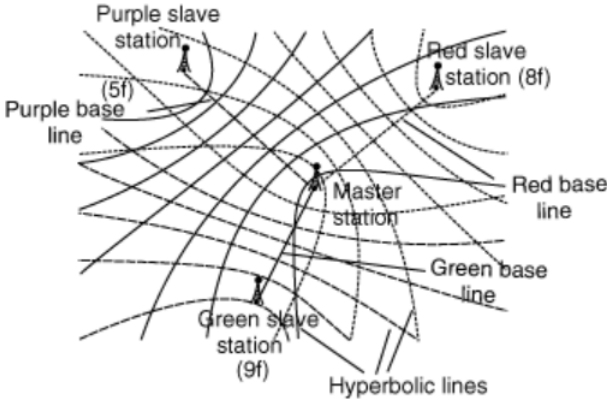
3.10.1 Navegação Hiperbólica

A navegação hiperbólica (*hyperbolic navigation* em inglês) é uma técnica bem conhecida e já é utilizada desde a Primeira Guerra Mundial. Esse tipo de navegação é alcançada quando os sinais, com uma velocidade conhecida de propagação (o som, por exemplo), são enviados por pelo menos três pontos conhecidos, e quando os tempos relativos de chegada destes sinais são medidos e interpretados por um sensor cuja a posição se quer descobrir (navegador). Os sinais podem ser transmitidos e recebidos por qualquer meio conhecido. Podem ser usados vários tipos de sinais, desde ondas contínuas até ondas moduladas e pulsos.

Considere duas estações de transmissão de posições conhecidas enviando sinais ao mesmo tempo. Se o navegador recebe esses sinais em tempos determinados, sabe-se que sua posição deve estar em algum lugar ao longo da mediatriz perpendicular da linha que conecta as estações. Se um sinal chega antes do outro, uma medição da diferença de tempo (*time difference of arrival* ou TDOA) identifica alguma outra linha de posição na qual o navegador deve estar. Estas linhas de posição são aproximadamente hipérboles esféricas. O navegador obtém uma correção encontrando suas linhas de posição em relação a dois ou mais pares de estações (PIERCE, 1946). Na Figura 21 pode-se ver um exemplo, com estações e suas linhas hiperbólicas em que poderia se encontrar o navegador baseado na diferença de tempo do receptor e conhecendo a posição das estações.

Quando o processo é feito de maneira análoga, mas inversa, onde se tem uma fonte emissora com posição desconhecida e vários estações receptoras (sensores) com posições conhecidas, pode-se encontrar a posição da fonte em questão. Essa técnica é chamada de multilateração e é um caso de navegação hiperbólica.

Figura 21 – Navegação hiperbólica com várias estações.



Fonte: www.aviation_dictionary.enacademic.com

3.11 MULTILATERAÇÃO (MLAT)

No processo de identificação da posição de uma fonte sonora, uma técnica usada frequentemente é a multilateração (*multilateration* ou MLAT). Esta técnica é amplamente utilizada na aviação e náutica com sonares e triangulações. É baseado na diferença do tempo de chegada (*Time Difference of Arrival* ou TDOA) da onda sonora para cada microfone, cujas posições são conhecidas. A partir disso, hipérboles, cujos focos estão nos microfones, são obtidas. A intersecção das hipérboles dos microfones será o ponto de origem do som (CHAN; HO, 1994). Um exemplo pode ser visto na Figura 22.

Em geral, as posições p_i, p_c e TDOA $t_i - t_c$ de um par de sensores c e i, limitam a posição do emissor de sinal no plano de um hiperboloide circular de dois lados com focos em p_i e p_c. Dado que c é o primeiro dos dois sensores a registrar um evento de sinal. O plano diretriz associado pode ser encontrado por um vetor de posição conhecido e um vetor normal. Uma vez que a diretriz é ortogonal ao vetor que aponta do sensor i para o sensor c, sabemos que o próprio $p_c - p_i$ ou o seu vetor unitário pode servir como vetor normal. Em seguida, precisamos de um vetor de posição com ponto final no plano diretriz. Um desses vetores pode ser definido pelo ponto em que a diretriz intersecta o segmento de linha entre p_i e p_c. Este ponto coloca um comprimento $v(t_i - t_c)$, onde v é a velocidade do som, distante de p_i para p_c, isto é, ele se encontra no vetor de posição (NORRDINE, 2015):

$$\frac{\vec{p_c} - \vec{p_i}}{||\vec{p_c} - \vec{p_i}||} \tag{3.13}$$

$$\vec{p_i} + v(t_i - t_c)\frac{\vec{p_c} - \vec{p_i}}{||\vec{p_c} - \vec{p_i}||} \tag{3.14}$$

Com restrições suficientes, ou seja, com sensores suficientes, podemos isolar a posição do emissor de sinal em um único ponto como na figura a seguir no espaço $\mathbb{R}^2$.

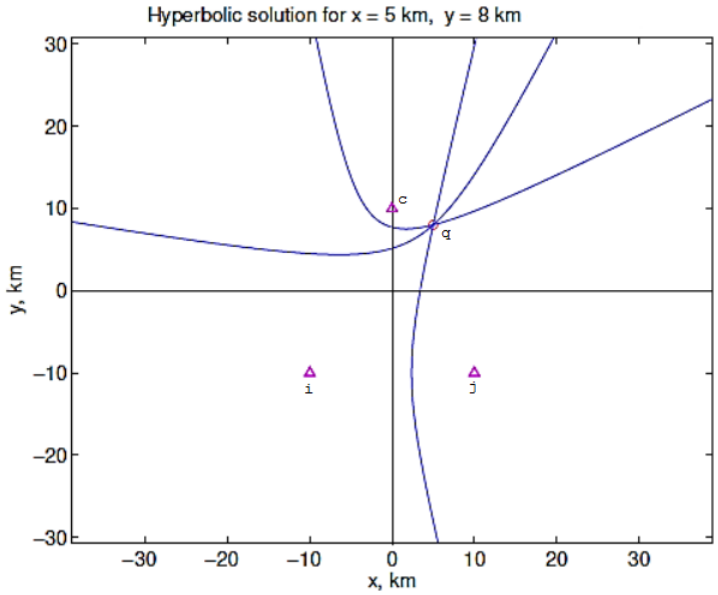
A maneira menos pesada computacionalmente de resolver o problema da multilateração é reduzí-lo à forma linear. Para determinar a localização do emissor do sinal, podemos usar três pares únicos i, j de sensores em adição a um sensor de descoberta inicial c. Isso reduz o problema para um sistema linear de três expressões e requer pelo menos cinco sensores no caso tridimensional:

$$||\vec{q} - \vec{p_i}|| = v(t_i - t_c) + ||\vec{q} - \vec{p_c}|| \tag{3.15}$$

$$||\vec{q} - \vec{p_i}||^2 = (v(t_i - t_c) + ||\vec{q} - \vec{p_c}||)^2 \tag{3.16}$$

$$||\vec{q} - \vec{p_i}||^2 = v^2(t_i - t_c)^2 + 2v(t_i - t_c)||\vec{q} - \vec{p_c}|| + ||\vec{q} - \vec{p_c}||^2 \tag{3.17}$$

Figura 22 – Três sensores gerando três hipérboles encontrando o ponto de origem do som.



Fonte: Mourabit e Badri (2015), modificado pelo autor

Introduzindo outro sensor j, pode-se eliminar o termo $||\vec{q} - \vec{p_c}||$, em que q é a posição desconhecida do emissor:

$$2 - 2||\vec{q} - \vec{p_c}|| = v(t_i - t_c) + \frac{||\vec{q} - \vec{p_c}||^2 - ||\vec{q} - \vec{p_i}||^2}{v(t_i - t_c)} \tag{3.18}$$

$$2 - 2||\vec{q} - \vec{p_c}|| = v(t_j - t_c) + \frac{||\vec{q} - \vec{p_c}||^2 - ||\vec{q} - \vec{p_j}||^2}{v(t_j - t_c)} \tag{3.19}$$

Logo:

$$v(t_i - t_c) + \frac{||\vec{q} - \vec{p_c}||^2 - ||\vec{q} - \vec{p_i}||^2}{v(t_i - t_c)} = v(t_j - t_c) + \frac{||\vec{q} - \vec{p_c}||^2 - ||\vec{q} - \vec{p_j}||^2}{v(t_j - t_c)} \tag{3.20}$$

Expandindo as definições das distâncias quadradas de cada um dos sensores e do emissor obtemos o seguinte resultado:

$$||\vec{q} - \vec{p_i}|| = (\vec{q} - \vec{p_i})^T(\vec{q} - \vec{p_i}) = \vec{q}^T\vec{q} - 2\vec{p_i}^T q + \vec{p_i}^T\vec{p_i} \tag{3.21}$$

Para os outros sensores:

$$||\vec{q} - \vec{p_j}|| = \vec{q}^T\vec{q} - 2\vec{p_j}^q + \vec{p_j}^T\vec{p_j} \tag{3.22}$$

$$||\vec{q} - \vec{p_c}|| = \vec{q}^T\vec{q} - 2\vec{p_c}^T q + \vec{p_c}^T\vec{p_c} \tag{3.23}$$

Isso leva ao resultado:

$$v(t_j - t_c) + \frac{2(\vec{p_j}^T - \vec{p_c}^T)\vec{q} + \vec{p_c}^T\vec{p_c} - \vec{p_j}^T\vec{p_j}}{v(t_j - t_c)} =$$

$$v(t_i - t_c) + \frac{2(\vec{p_i}^T - \vec{p_c}^T)\vec{q} + \vec{p_c}^T\vec{p_c} - \vec{p_i}^T\vec{p_i}}{v(t_i - t_c)} \tag{3.24}$$

Observe que os termos quadráticos sumiram e obtêm-se com um sistema linear de equações facilmente resolvível.

$$\vec{a_{ijc}}^T\vec{q} = b_{ijc}, \tag{3.25}$$

onde:

$$\vec{a_{ijc}} = 2(v(t_j - t_c)(\vec{p_i} - \vec{p_c}) - v(t_i - t_c)(\vec{p_j} - \vec{p_c})) \in \mathbb{R}^3, \tag{3.26}$$

$$\begin{aligned} b_{ijc} = v(t_i - t_c)(v^2(t_j - t_c)^2 - \vec{p_j}^T\vec{p_j}) + \\ +(v(t_i - t_c) - v(t_j - t_c))\vec{p_c}^T\vec{p_c} + \\ +v(t_j - t_c)(\vec{p_i}^T\vec{p_i} - v^2(t_i - t_c)^2), \end{aligned} \tag{3.27}$$

e $b_{ijc} \in \mathbb{R}$.

Isso pode ser expresso como:

$$\mathbf{A}q = \mathbf{b}, \tag{3.28}$$

onde q é a posição do evento sonoro e cada par i e j cria a matriz $\mathbf{A}$ e o vetor $\mathbf{b}$ definidos por $b_{ijc} \in \mathbf{b} \in \mathbb{R}^3$ e $\vec{a_{ijc}} \in \mathbf{A} \in \mathbb{R}^{3\times3}$tal que $\vec{a_{ijc}}^T$ são as linhas de $\mathbf{A}$.

Isolando q, obtêm-se:

$$q = \mathbf{A}^{-1}\mathbf{b}. \tag{3.29}$$

A partir dessas expressões matemáticas, é possível desenvolver um algorítimo de identificação de fonte sonora. É importante notar que, para a localização de uma fonte em duas dimensões, usando-se a simplificação linear, são necessários pelo menos 4 sensores. (NORRDINE, 2015).

3.12 SISTEMAS EMBARCADOS

O termo “Sistemas Embarcados” vem da tradução do inglês “*Embedded Systems*” que seria melhor traduzido como “Sistemas Embutidos”. Nesse sentido, “embarcados” tem o significado de “embutido” ou “imerso”. Dessa maneira, sistemas embarcados também podem ser tratados como “sistemas embutidos” ou até mesmo “sistemas automatizados”.

Um sistema embarcado tem interações contínuas com o dinâmico mundo exterior. Podem ser fisicamente embutidos ou puramente computacionais. Robôs móveis, controlador

de processos industriais ou geladeiras inteligentes podem ser exemplos de sistemas embarcados. É esperado que esses sistemas executem tarefas por um longo período de tempo, repetidamente processando novas informações de entrada (por sensores, por exemplo) e gerando informações de saída (atuando um motor, por exemplo) (KAELBLING, 2008).

Sistemas embarcados são tipicamente baseados em microprocessadores que são construídos para controlar uma variedade de funções e que são projetados de tal maneira que o usuário final não precise programá-los, da mesma maneira que os computadores atuais funcionam. A depender da aplicação, o usuário pode sim fazer escolhas sobre a funcionalidade do sistema. Um sistema embarcado é projetado para desempenhar uma função em particular, embora com escolhas e opções diferentes (HEATH, 2002).

Com o crescimento da demanda por microprocessadores de baixo custo para aplicações diversas, começaram a aparecer no mercado placas com um bom processamento, mas que podiam ser acessíveis tanto para o público geral como para pesquisadores, que inclusive poderiam, construir seus próprios computadores de baixo custo baseados em tais sistemas. Como exemplo de tais placas, pode-se citar o Rasberry Pi, Intel Galileo e BeagleBone Black. Um foco maior será dado ao último, pois essa foi a placa que foi disponibilizada para o desenvolvimento desse trabalho.

O entendimento de sistemas embarcados é importante para o desenvolvimento desse projeto, uma vez que um sistema será construído um módulo auditivo artificial que interagirá com o mundo real.

3.12.1 BeagleBone Black

A placa BeagleBone Black (BBB) é um hardware de baixo custo, aberto e expansível, lançado por uma comunidade de desenvolvedores patrocinada pela Texas Instruments. O tamanho da placa é pequeno o suficiente para caber na palma da mão de uma pessoa. Pode ser usada para uma variedade de projetos, desde estudantes até protótipos de sistemas incorporados reais complexos. É uma placa muito utilizada em sistemas embarcados (HE; HUANG; WOLTMAN, 2014).

O BBB apresenta um processador AM335x 1GHz ARM CortexA8, 512MB de memória RAM DDR3, 2GB de armazenamento flash *onboard* 8bit eMMC, acelerador de gráficos 3D, acelerador de pontoflutuante, 2x microcontroladores PRU 32bits, porta USB para alimentação, comunicação e USB Host, Ethernet, saída HDMI e 2x 46pin *headers* de GPIO (a configuração também pode ser vista na Tabela 1 e na Figura 23), e preço de mercado avaliado em pouco mais de US$ 50,00 (XAVIER; BARBACENA, 2015).

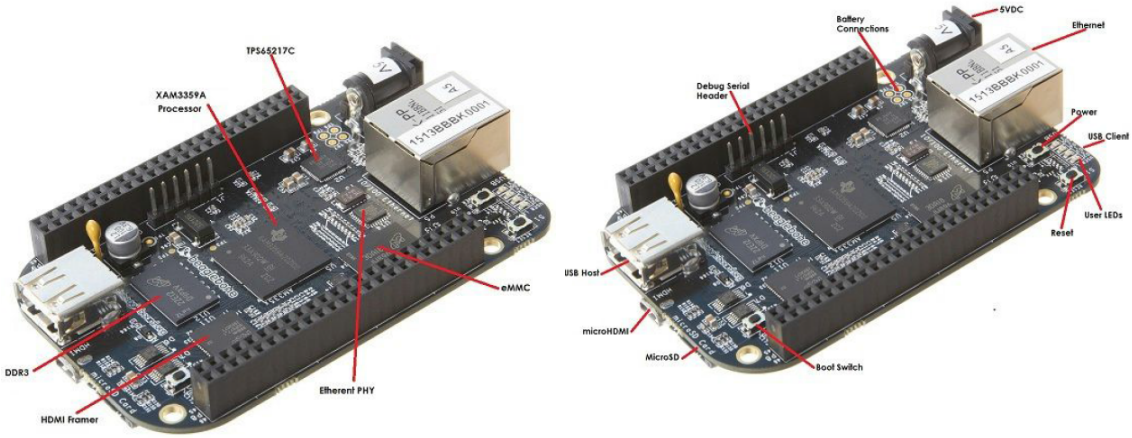
O BBB é compatível com diversos sistemas operacionais, em sua grande maioria baseados em Linux, tais como Debian, Ubuntu, Ångström, dando assim uma maior liberdade aos desenvolvedores. Linux é um sistema operacional de código aberto rodando

Tabela 1 – Descrição Técnica do BeagleBone Black

-	**BeagleBone Black**
Processador	AM3358BZCZ100, 1GHz
Saída de Vídeo	HDMI
Memória DRAM	512MB DDR3L 800MHz
Memória Flash	4GB eMMC, uSD
Onboard JTAG	Opcional
Porta Serial	Via *Header* (pinos)
Tensão e Corrente de Operação	210-460mA com 5V

Fonte: Xavier e Barbacena (2015) adaptado pelo autor

Figura 23 – Alguns componentes e conectores importantes do BeagleBone Black



Fonte: http://goo.gl/WuW2QE

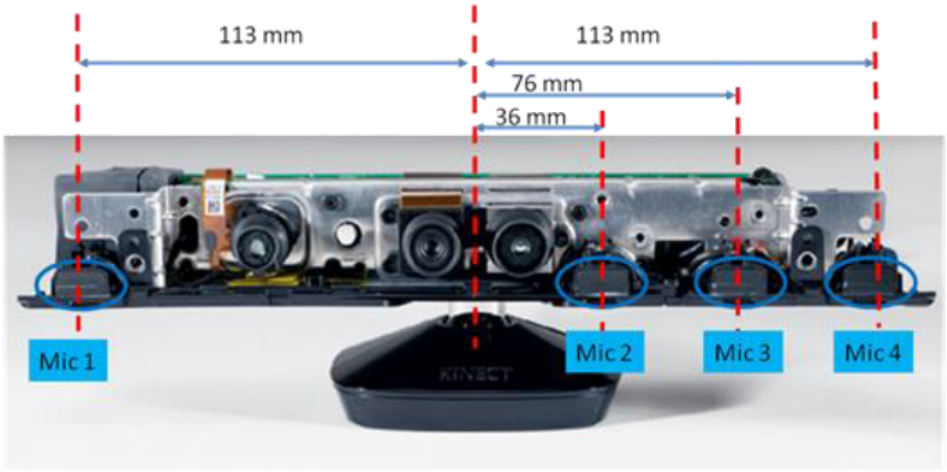
em todas as principais arquiteturas de processadores, incluindo os processadores ARM, que é, atualmente, a principal arquitetura de microcontroladores para aplicações embarcadas (XAVIER; BARBACENA, 2015).

O BeagleBone Black é bem versátil e pode ser programado de diferentes maneiras. Em sistemas operacionais baseados em Linux, em C/C++, em Python entre outros. De acordo com a página do MATLAB, eles fornecem suporte para BeagleBone desde 2015. O MATLAB *Support Package for BeagleBone Black Hardware* permite que seja feita a programação da BeagleBone pelo próprio MATLAB, podendo obter dados pelos sensores e fazer comunicação por protocolos como I2C, serial e SPI.

3.13 KINECT

O Kinect é um dispositivo de entrada de detecção de movimento (câmeras e microfones) da Microsoft para o console de videogame Xbox 360 e também para PCs com Windows, sendo também usado para pesquisas. O sistema Kinect possui um *array* de microfones (um vetor de microfones) que permite que o dispositivo realize supressão de ruído ambiente. Portanto, o jogador pode interagir com o jogo através do reconhecimento de voz. Um conjunto de microfones Kinect possui quatro microfones com processamento de cada canal de 16-bits e taxa de amostragem de até 16000 Hz. A posição do array de microfones do Kinect pode ser visto na Figura 24 (PEI et al., 2013).

Figura 24 – O array de microfones de um módulo Kinect.



Fonte: Pei et al. (2013)

4 MÓDULO AUDITIVO ARTIFICIAL PROPOSTO

Nos dias de hoje, dentro da área da robótica, ainda há muito mais enfoque em visão ou navegação por sensores do que utilização de ondas sonoras audíveis que incorporariam um "ouvido". Diante de um problema robótico como é a "audição" para robôs, resolveu-se estudar a possibilidade de desenvolver um módulo auditivo artificial para um robô assistivo. Tal módulo teria que realizar as funções de (i) identificar o locutor, (ii) identificar emoção estereotipada na voz, (iii) localizar o locutor. Diferentes técnicas foram estudadas para o desenvolvimento do mesmo. Nesse capítulo será abordada a proposta do trabalho, assim como a metodologia e o desenvolvimento de como foi a realização desse trabalho. É importante notar que o enfoque desse trabalho é na construção do módulo e não na construção do robô assistivo, sendo o robô uma motivação para o estudo realizado.

4.1 METODOLOGIA

Esta seção descreve a metodologia utilizada durante esse trabalho

- **Pesquisa e Estudo:**

  Busca de referências sobre identificação de um locutor independente de texto, identificação de emoções na fala de um locutor e localização de uma fonte sonora, identificando as técnicas utilizadas por diversos autores, destacando as que obtiveram melhores resultados e as publicações mais recentes acerca dos assuntos a serem abordados.

  Estudo dos tipos de características na fala de um locutor e as diferentes técnicas de extração das mesmas e escolha da técnica de extração que foi julgada como mais relevante para os casos de estudo; estudo do algorítimo para gerar os MFCCs com delta e delta-delta; estudo dos diferentes tipos de classificadores e escolha do tipo de classificador que foi julgado como apropriado para os casos de estudo; estudo das técnicas para se localizar uma fonte sonora e escolha de uma técnica que possa ser aplicada com microfones e ondas de som; estudo do microprocessador a ser utilizado e de como embarcar o sistema e criar o módulo.

- **Criação do banco de vozes em português:**

  Criação de um banco de dados de vozes através da gravação de palavras, em um estúdio, de 5 locutores distintos falando 100 palavras cada, com taxa de amostragem de 44100 Hz e 16 bits.

- **Implementação do código de extração do MFCCs:**

  Desenvolvimento, através do MATLAB, da extração das características da voz por meio do MFCC com delta e delta-delta que foram utilizados para gerar os vetores de entrada para ser usados nas fases de treino e testes das RNAs.

- **Implementação de redes neurais:**

  Utiliza-se das funções de redes neurais artificiais que o MATLAB fornece e o estudo de como aplicar da maneira mais efetiva as RNAs.

- **Implementação da fase de treinamento:**

  Desenvolvimento, em MATLAB, da parte de treinamento para gerar modelos com classificadores treinados para a identificação de locutores e para emoções. MFCC foi utilizado como método de extração de características e múltiplas RNAs como o classificador a ser treinado.

- **Implementação da fase de verificação:**

  Desenvolvimento, em MATLAB, da parte de teste, com declarações ou palavras novas, desconhecidas pelas redes e compará-las com o locutor/emoção alvo. Foi-se utilizado MFCCs como extração de características e RNAs como classificador treinado do sistema. Nessa fase também ocorre o cálculo da precisão das redes.

- **Comparação do resultado:**

  Comparação o resultado com outros modelos de classificadores e RNAs testados. Comparar com configurações diferentes. Comparar com o resultado obtido por outros autores.

- **Criação da simulação para localização de fonte sonora:**

  Utilização de MLAT para, localizar a fonte sonora criando uma simulação em MATLAB que faz o som se deslocar até os microfones, calcular a diferença de tempo entre cada microfone e, assim, obter a origem da fonte sonora.

- **Estudo do BeagleBone Black:**

  Estudo de como configurar o BeagleBone Black para se comunicar com o MATLAB e poder ser programado no mesmo, com a finalidade de não ter que se repetir toda a programação feita anteriormente em uma nova linguagem. Estudo de como importar e adaptar o código desenvolvido para a etapa de projeto para a placa BeagleBone Black.

- **Estudo do Kinect:**

  Estudo de como utilizar e obter dados dos sensores do módulo Kinect. Estudo de como fazer a comunicação do Kinect com o computador e como capturar dados que

possam ser interpretados pelo MATLAB para se fazer a comunicação entre o sistema embarcado e o computador.

- **Criar a tarefa de identificação de um locutor**

  Utilizando-se dos conhecimentos adquiridos na fase de projeto, criar a tarefa de identificação de um locutor para o módulo, integrando um microfone, uma placa para aplicações embarcadas e um computador.

- **Criar a tarefa de identificação de uma emoção**

  Utilizando das redes geradas pela etapa de projeto, criar a tarefa de identificação da emoção na voz do locutor para o módulo, integrando um microfone, uma placa para aplicações embarcadas e um computador.

- **Criar a tarefa de localização de uma fonte sonora**

  Com os conhecimentos adquiridos pela simulação com MLAT, criar, utilizando o Kinect, a tarefa para localizar o locutor para o módulo, integrando-o com a placa embarcada e um computador.

- **Integrar todas as tarefas**

  Integrar todas as tarefas e ter certeza que existe harmonia na integração entre Kinect, computador e BeagleBone Black.

Todos os passos acima foram executados para a implementação e execução do projeto, simulação e construção do módulo auditivo artificial.

4.2 PROPOSTA

Para a realização dos objetivos propostos, dividiu-se esse trabalho entre projeto e construção do módulo propriamente dito. A parte de projeto, foi dividido em três partes, um para cada um dos objetivos. Na parte embarcada, dividiu-se o trabalho em duas partes. A primeira seria a implementação e teste separado de cada uma das suas funções e, finalmente, implementação e teste do módulo com todas as funcionalidades em acordo.

Para a extração de características, utilizou-se dos MFCC com delta e delta-delta. Isso se deve ao fato de que a grande maioria dos trabalhos vistos que visam identificação de locutor, emoção, língua, fala (com um pequeno vocabulário) utilizam o MFCC ou alguma variação do mesmo. Como exemplo, pode-se citar os trabalhos de Hasan et al. (2004), Martinez, Perez e Escamilla (2012) e Chakraborty (2014).

Com o propósito de aprendizado e para garantir que haveria uma compatibilidade máxima do MATLAB e uma otimização do processo por completo, para o desenvolvimento

desse projeto propõe-se que o MFCC seja programado e construído do zero, sem a utilização de um programa ou código prévio, utilizando-se dos conhecimentos adquiridos que podem ser vistos na seção 3.4. Dessa maneira, garante-se o controle de todo o processo de extração das características da voz do locutor.

Em seguida, precisa-se utilizar um método para identificação dos locutores. Existem várias técnicas utilizadas na literatura para se identificar locutores. Por exemplo, Martinez, Perez e Escamilla (2012) utilizou de *Learning Vector Quantization* (LVQ), Dahake e Shaw (2016) usa *Support Vector Machine* (SVM), Maesa et al. (2012) faz o reconhecimento por *Gaussian Mixture Model* (GMM), Sória e Jr (1996) prefere usar RNA, Bansal e Bharti (2015) compara o uso de *Fuzzy C-mean* (FCM) com LVQ. Para esse trabalho, redes neurais artificiais foi escolhido, pois, conforme será visto na Tabela 2 do Capítulo 5, dentre os testes realizados, RNA teve o melhor rendimento por custo-benefício, considerando que os vetores de entrada são muito grandes e a aplicação precisa ser de baixo custo computacional.

Foi utilizado aprendizado supervisionado do tipo *resilient backpropagation* (Rprop) e função de erro do tipo *mean squared error* (MSE) com as RNAs. Para aumentar a generalização, múltiplas redes foram treinadas.

Para que os objetivos de projeto, simulação e embarcação propostos possam ser atingidos, tudo que foi implementado, foi feito utilizando-se do MATLAB 2017b e programado durante o desenvolvimento desse trabalho.

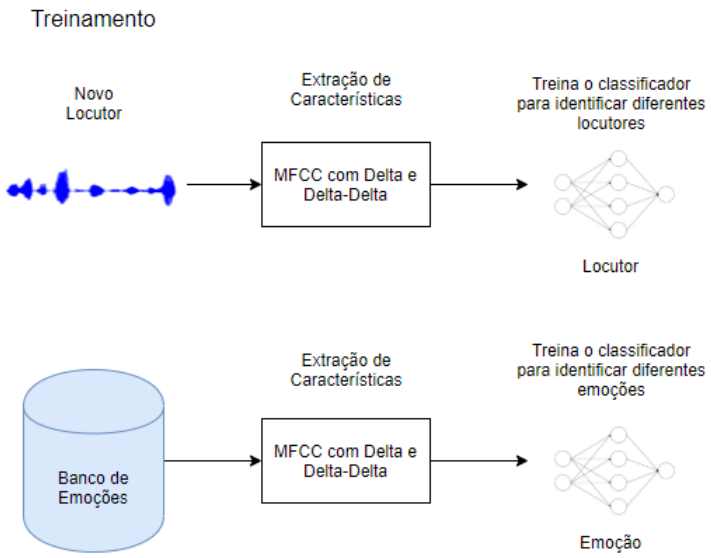
4.2.1 Proposta de Integração das Diferentes Funções do Módulo Auditivo Artificial

Para a elaboração de um módulo auditivo artificial, foi proposto uma maneira de integrar as três distintas tarefas do módulo: (i) identificar quem fala; (ii) a emoção e (iii) de onde é falado; tanto na fase de treino (que pode ser visto na Figura 25), em que se treinam as RNAs da identificação de locutor e de emoção (mas não precisa-se de um treinamento para a localização da fonte sonora), como na fase de aplicação (que pode ser visto na figura Figura 26) em que se usa o módulo propriamente dito para executar as três funções mencionadas anteriormente.

1. *Treinamento*

   Como pode ser visto na Figura 25, a localização de fonte sonora não precisa de treinamento pois não utiliza um classificador. Para o treinamento dos classificadores da identificação de locutor e de emoções, é necessário um banco de vozes (quando a gravação for feita em estúdio) ou gravar um novo locutor in loco (quando não for feita em estúdio). Dessa maneira, o "novo locutor" da figura, também pode ser substituído por um banco de falas.

Figura 25 – Fluxograma do módulo proposto na fase de treinamento.



Fonte: Autor

Nessa etapa, há a extração de características, para cada uma das funções, utilizando-se de MFCC com delta e delta-delta. Com essas características já extraídas, treinam-se classificadores distintos para cada tipo de função, podendo serem eles para locutor ou emoção.

2. *Aplicação*

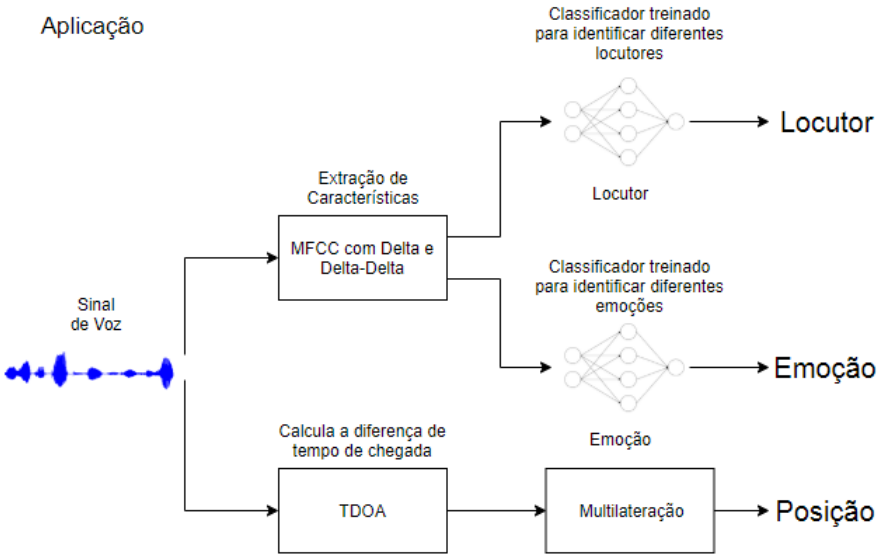
   Conforme pode ser visto na Figura 26, essa etapa teve o objetivo de identificar o locutor, a emoção e sua posição. Quando os microfones percebem a presença de um áudio, o primeiro passo é extrair informações para que se consiga fazer as identificações propostas. Para tal, calcula-se o MFCC com delta e delta-delta de um trecho do áudio que passará por classificadores distintos para que seja feita a identificação do locutor e da emoção. Ao mesmo tempo, o cálculo da diferença de tempo de chegada do áudio em cada microfone é feita (TDOA) e calcula-se a posição aproximada do locutor pelo método da multilateração (MLAT)

4.2.2 Construção e Obtenção de Bancos de Falas

Antes de se detalhar as propostas em si, é importante falar da construção e/ou obtenção dos bancos de falas. Conforme foi visto na seção 3.5, bancos de vozes são imprescindíveis no treinamento por aprendizado supervisionado de classificadores.

Para que houvesse uma melhor qualidade do áudio na parte de treinamento de identificação de um locutor e para poder manter a consistência na hora de comparar

Figura 26 – Fluxograma do módulo proposto na fase de aplicação.



Fonte: Autor

diferentes classificadores ou configurações de RNAs, um banco de fala foi elaborado. Dessa maneira, garante-se uma maior robustez e refino da rede neural se comparado com áudios gravados por celular ou microfones embutidos em computadores. As gravações foram feitas por 5 diferentes locutores num estúdio com equipamentos profissionais. Cada um dos locutores repetiu as mesmas 100 palavras, criando um banco de voz com 500 arquivos de áudio. A gravação foi feita em um estúdio com um microfone condensador Samson C01 com uma frequência de amostragem de 44100 Hz e 16 bits. A escolha das palavras foi aleatória e pode ser vista no Apêndice A. Esse banco de falas está disponível no link: <https://bit.ly/2LgfoZu>.

Para a identificação de emoções estereotipadas seria muito mais complicado de se construir um banco de emoções próprio, pois, para tal, seriam necessários atores treinados capazes de imitar diferentes emoções. Sabendo-se dessa dificuldade, preferiu-se usar bancos de emoções disponíveis para uso acadêmico de duas diferentes fontes. A primeira fonte foi o *Surrey Audio-Visual Expressed Emotion* (SAVEE) (HAQ; JACKSON, 2011), que disponibiliza 4 locutores interpretando 7 diferentes emoções falando inglês britânico: raiva, nojo, medo, alegria, calma, tristeza e surpresa, gravadas com uma frequência de amostragem de 44100 Hz. A segunda fonte foi o *Berlin Emotion Database* (Emo-DB) (BURKHARDT et al., 2005). Esse banco possui 10 locutores, 5 homens e 5 mulheres, falando em alemão e interpretando 7 emoções: raiva, nojo, medo, alegria, calma, tristeza e tédio, gravadas com uma taxa de amostragem de 16000 Hz. Ambos bancos de emoções foram citados previamente no seção 3.5.

4.2.3 Proposta da Etapa de Projeto do Módulo Auditivo Artificial

A etapa de projeto visa comparar diferentes métodos e configurações para que se obtenha um melhor resultado no universo de testes e simulações realizados, utilizando-se apenas de um computador e dos arquivos de áudio dos bancos de fala. Essa etapa pode ser divida em três tarefas distintas: (i) identificação de locutor, (ii) identificação da emoção estereotipada na voz, (iii) localização da fonte sonora.

i. **Identificação de Locutor:**

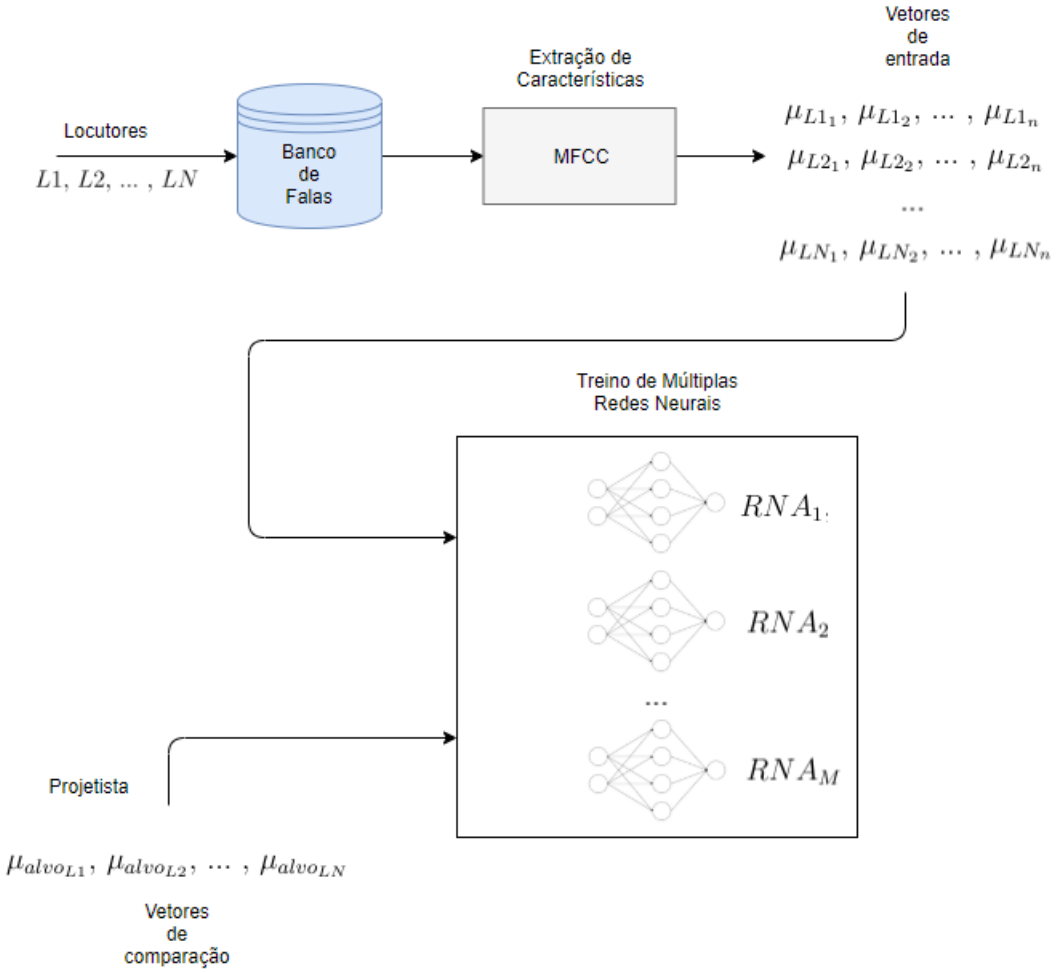
A identificação de um locutor independente de texto é composta por duas distintas fases. A primeira fase é o treinamento e nela ocorre a extração de características de um banco de dados previamente gravado. Nessa etapa, os N locutores ($L1$, $L2$, ... , LN) que tiveram suas vozes gravadas em um banco de falas, têm os seus coeficientes cepstrais de Mel com delta e delta-delta extraídos (conforme foi visto na Figura 9 do Capítulo 3), gerando um vetor (μ), que possui dimensões iguais a três vezes a quantidade de coeficientes do MFCC (conforme foi visto na Figura 13 do Capítulo 3), para cada amostra de voz. Como existem n amostras de voz, são gerados n vetores por locutor (μ_{L1_1}, μ_{L1_2}, ... , μ_{L1_n} para o $L1$; μ_{L2_1}, μ_{L2_2}, ... , μ_{L2_n} para o $L2$; ... ; μ_{LN_1}, μ_{LN_2}, ... , μ_{LN_n} para o LN). Todos os N x n vetores são gravados, de forma que a simulação não precisa gerá-los do início a cada novo teste.

Para o treinamento da rede neural, precisa-se criar um vetor alvo para cada um dos locutores ($\mu_{alvo_{L1}}$, $\mu_{alvo_{L2}}$, ... , $\mu_{alvo_{LN}}$), também de dimensão n, que diz para a rede qual é a resposta correta para cada um dos seus vetores de entrada. Esse vetor é gerado pelo projetista da rede, que conhece qual deve ser a resposta desejada pra cada um dos vetores de entrada. A partir disso, a RNA vai corrigir seus pesos e construir um modelo para aquele locutor. Para aumentar a generalização, esse processo é repetido M vezes para M redes neurais (RNA_1, RNA_2, ... , RNA_M). Todo esse processo aqui descrito, pode ser visto na Figura 27.

A segunda fase é a verificação, em que uma voz de um dos locutores precisa ser identificada (Lx). Para tal, extrai-se as características gerando um vetor do locutor de teste (μ_{Lx}). As M redes neurais dão as suas notas de probabilidade (θ) de ser cada um dos locutores ($\theta_{L1_{RNA_1}}$, $\theta_{L2_{RNA_1}}$, ... , $\theta_{LN_{RNA_1}}$; $\theta_{L1_{RNA_2}}$, $\theta_{L2_{RNA_2}}$, ... , $\theta_{LN_{RNA_2}}$; ... ; $\theta_{L1_{RNA_M}}$, $\theta_{L2_{RNA_M}}$, ... , $\theta_{LN_{RNA_M}}$). É tirada a média das notas de probabilidade (Equação 4.1). O locutor será aquele que possuir a maior nota (Equação 4.2). Isso pode ser observado na Figura 28.

$$\begin{aligned} \theta_{L1} &= \frac{\theta_{L1_{RNA_1}} + \theta_{L1_{RNA_2}} + ... + \theta_{L1_{RNA_M}}}{M}; \\ \theta_{L2} &= \frac{\theta_{L2_{RNA_1}} + \theta_{L2_{RNA_2}} + ... + \theta_{L2_{RNA_M}}}{M}; \ ... \ ; \\ \theta_{LN} &= \frac{\theta_{LN_{RNA_1}} + \theta_{LN_{RNA_2}} + ... + \theta_{LN_{RNA_M}}}{M} \end{aligned} \tag{4.1}$$

Figura 27 – Proposta do projeto do treinamento para identificação de um locutor



Fonte: Autor

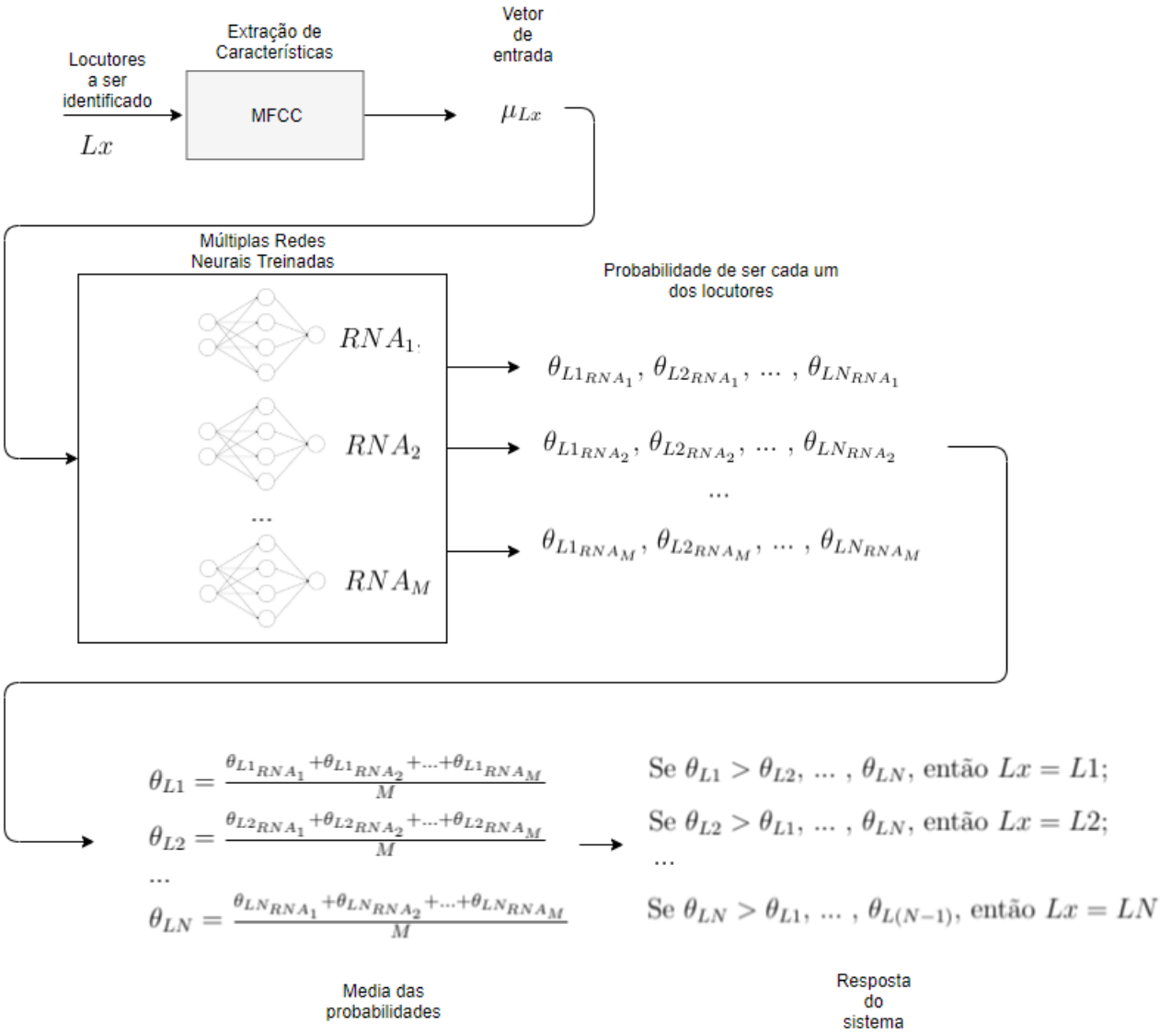
$$\begin{gathered}\text{Se } \theta_{L1} > \theta_{L2}, \ldots, \theta_{LN}, \text{ então } Lx = L1; \\ \text{Se } \theta_{L2} > \theta_{L1}, \ldots, \theta_{LN}, \text{ então } Lx = L2; \ldots; \\ \text{Se } \theta_{LN} > \theta_{L1}, \ldots, \theta_{L(N-1)}, \text{ então } Lx = LN\end{gathered} \tag{4.2}$$

O resultado obtido será comparado com os obtidos por outros métodos de aprendizado, por outras configurações da redes neurais, assim como os resultados obtidos por outros autores.

ii. **Identificação de uma Emoção Estereotipada:**

Para a identificação de uma emoção estereotipada, pode-se fazer o mesmo processo comentado acima para a identificação de um locutor, com a diferença que existiriam NN emoções (que seriam representadas por $Emo1$, $Emo2$, ... , $EmoNN$) ao invés

Figura 28 – Proposta do projeto do teste da identificação de um locutor

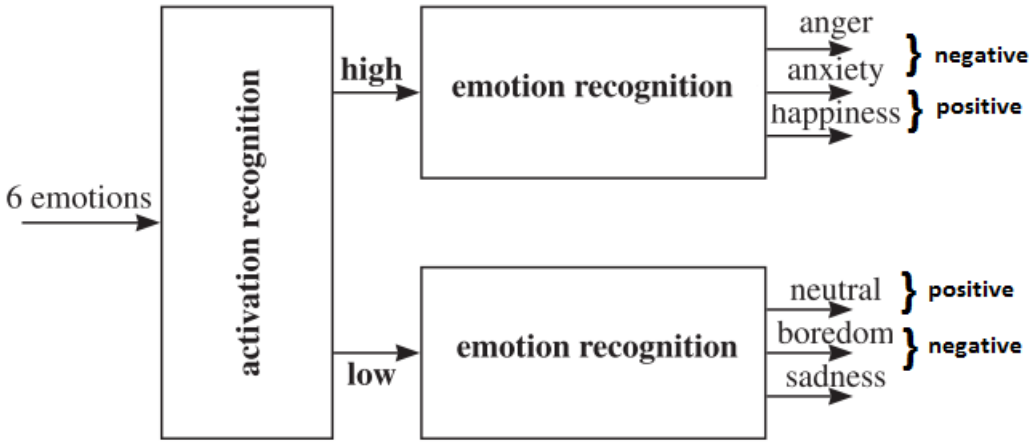


Fonte: Autor

de locutores ($L1$, $L2$, ... , LN). O resto do processo de treinamento e verificação seria igual, portanto, nesse trabalho, após obter o melhor resultado para a identificação de um locutor, utiliza-se o mesmo método adaptando-o para a utilização na identificação de emoções. Dessa maneira, tanto a Figura 27 quanto a Figura 28 são válidas também.

Para efeito de comparação, propõe-se também a construção de outros modelos de classificação e comparação dos resultados. Um outro modelo seria a identificação de emoções e agrupá-las posteriormente utilizando o *core affect* (conforme visto na subseção 3.9.1) e, dessa maneira, melhorando a resposta do sistema e restringindo o número de decisões possíveis para o robô assistivo. Um segundo modelo seria treinar

Figura 29 – RNAs em cascata



Fonte: Lugger e Yang (2008), modificado pelo autor

as RNAs para identificar os 4 quadrantes do *core affect* ao invés das emoções em si. Nesse caso, $NN = 4$, onde cada um seria um quadrante do *core affect*. Um outro modelo seria a utilização do método de RNAs em cascata conforme foi sugerido por Lugger e Yang (2008), confome a Figura 29, onde 3 conjuntos diferentes de RNAs seriam treinados, em cascata, utilizando-se de *core affect*, para aumentar sua eficiência.

Todos os resultados obtidos foram comparados entre si e, comparados com os resultados obtidos por outros autores. Também se comparou as respostas obtidas utilizando-se diferentes bancos de emoções.

iii. **Localização de uma Fonte Sonora:**

Para a simulação da localização da fonte sonora, foram utilizados 4 microfones virtuais, cujas posições são conhecidas. Considerou-se também, um ambiente onde a velocidade do som no ar é de 343 m/s. Essa simulação não considerou efeitos como ruído, eco, desconhecimento da velocidade real do som no ambiente, limitações de hardware, sensibilidade do microfone, reverberação e outros. O ambiente simulado foi hipotético e considerando condições ideais. Esse ambiente pode ser visto de uma maneira ilustrativa na Figura 30.

O método de resolução da localização de uma fonte sonora utilizado foi MLAT (multilateração), utilizando-se de TDOA que é a diferença de tempo que o evento sonoro demora para atingir cada microfone.

Utilizando-se da Equação 3.29 que diz que: $q = \mathbf{A}^{-1}\mathbf{b}$, e sabendo que i, j e c são sensores (microfones) de posições conhecidas, onde cada par i e j cria a matriz $\mathbf{A}$ e o vetor $\mathbf{b}$ definidos por $b_{ijc} \in \mathbf{b} \in \mathbb{R}^3$ e $\vec{a_{ijc}} \in \mathbf{A} \in \mathbb{R}^{3\times3}$tal que $\vec{a_{ijc}}^T$ são as linhas de

Figura 30 – Ilustração do ambiente simulado.

Fonte: Autor

A, pode-se calcular a posição da fonte sonora. Quatro microfones foram utilizados. É importante notar que os índices i e j podem assumir valores diferentes de 1 e c é o microfone que primeiro percebeu o evento sonoro. Dessa maneira, variando i e j se pode utilizar quantidades superiores a 3 microfones.

4.2.4 Proposta da Etapa de Construção do Módulo Auditivo Artificial

A proposta de construção do módulo auditivo artificial se separa nas três funções distintas, sendo estudadas de forma separada e, no teste final, de forma integrada.

Toda a construção do módulo foi feita utilizando o BeagleBone Black (BBB), que foi visto na subseção 3.12.1. Para essa etapa, utilizou-se um módulo Kinect (WEBB; ASHLEY, 2012) (visto na seção 3.13) para fazer a captura do áudio, pois havia disponibilidade de uso do mesmo. O módulo Kinect se encontra no laboratório de robótica da Universidade Federal da Bahia, por isso a gravação dos locutores foi feita com um microfone simples (Samson Q7). Porém, para fazer a integração microfone/Kinect com o BBB, foi preciso utilizar o computador, sendo que o mesmo agiria de forma intermediária entre a captura e o processamento, uma vez que o Kinect é apenas compatível oficialmente com Windows e Xbox por ter sido desenvolvido pela Microsoft, apesar de existir projetos "*open source*"

(como é o caso do OpenKinect) de portabilidade do Kinect para outras plataformas como Linux. Mesmo assim, nenhum projeto aparenta dar tanto suporte quanto o oficial da Microsoft.

Assim como na etapa de projeto, existem três tarefas distintas na etapa construção: (i) identificação de locutor, (ii) identificação de uma emoção estereotipada na voz, (iii) localização da fonte sonora.

i. **Identificação de Locutor:**

   Assim como no projeto, para a identificação de um locutor independente de texto é necessário que ocorra treinamento e verificação (teste). Dessa maneira, tanto a Figura 27 quanto a Figura 28 continuam sendo aplicadas para a construção. A maior diferença é que, ao invés de ser gravado em um estúdio para gerar um banco de dados, as gravações de cada locutor são feitas in loco, sendo armazenadas e utilizadas para gerar novas RNAs. Todos os locutores gravaram as mesmas frases e que podem ser vistas no Apêndice B. Preferiu-se gravar frases ao invés de palavras para poder capturar melhor o estilo de fala de cada locutor.

Figura 31 – Proposta para a captura e processamento de áudio para a identificação de locutor.

Fonte: www.samsungmobilepress.com, www.istockphoto.com e www.beagleboard.org; modificado pelo autor

   Para a gravação dos locutores, utiliza-se um microfone comum modelo Samson Q7 que é conectado ao computador com o mesmo conectado ao BBB, conforme pode ser visto na Figura 31. O computador faz a captura do áudio e, através do MATLAB, a comunicação entre computador e BBB para o processamento do áudio é feita. Pelo fato do BBB não possuir entrada P2 para microfones e pela dificuldade em se encontrar um microfone USB, em especial que seja compatível com o BBB, preferiu-se utilizar o computador como intermédio. Com cada novo locutor introduzido ao sistema, geram-se novas redes neurais e avalia-se o desempenho da rede.

Os resultados obtidos são então comparados com os obtidos previamente na etapa de projeto e com resultados de outros autores. Será também testada e comparada a identificação da emoção "calmo" em 5 diferentes locutores e a interpretação de 6 emoções distintas por um único locutor desconhecido pelas redes.

ii. **Identificação de uma Emoção Estereotipada:**

Para se fazer a identificação de emoções, utiliza-se o sistema treinado na fase de projeto (subseção 4.2.3). Dessa maneira, não existe uma nova fase de treinamento, apenas a etapa de verificação da voz capturada pelo microfone para se determinar qual a emoção na voz do locutor.

A aquisição do áudio é a mesma da etapa de Identificação de Locutor, feita com um microfone Samson Q7, portanto a Figura 31 é válida também para essa etapa.

Existe porém um problema associado a esse método. Para se chegar aos melhores resultados possíveis seria necessário que cada um dos locutores gravasse uma determinada frase com diferentes emoções, assim o resultado obtido seria muito mais preciso.

Os resultados obtidos são comparados com os da fase de projeto e com os da literatura.

iii. **Localização da Fonte Sonora:**

Para se localizar a origem de um som, é necessário utilizar os quatro microfones do módulo Kinect (visto na seção 3.13). Precisa-se também de um grande ambiente para que se possa fazer diferentes testes. Para tal, o laboratório de robótica da Universidade Federal da Bahia foi utilizado.

No processo de captura, utiliza-se 4 canais de áudio, cada um representado por um dos microfones. Esses áudios são gravados para que se possa fazer outros testes e comparações. Utiliza-se MLAT, mesmo método utilizado na simulação (subseção 4.2.3).

Existem, também, alguns problemas associados ao uso do módulo Kinect. A sensibilidade dos microfones dele é limitada, pois ele foi projetado para se basear mais em suas três câmeras do que em seus microfones para a localização do jogador. Outro problema é a disposição linear dos microfones. Quando se usam técnicas de localização de fonte sonora é desejável que os mesmos não estejam dispostos em uma linha única, conforme Pei et al. (2013) comentam em seu trabalho. Por fim, as distâncias entre microfones é muito pequena, o que significa que precisaria-se de mais sensibilidade por parte dos microfones, coisa que não acontece. Em outras palavras, os problemas como a taxa de amostragem limitada e o sincronismo entre a aquisição das amostras dos diferentes canais contribuem para o erro nos cálculo do TDOA para o uso da técnica de MLAT.

Os resultados obtidos foram comentados e comparados com os da simulação e da literatura. Também foi comparado o resultado obtido por esse trabalho com o de uma rotina oferecida pelo próprio MATLAB utilizando *Direction of Arrival* (DOA).

5 PROJETO E RESULTADOS

Foi proposto nesse trabalho o desenvolvimento de três funções distintas que integrariam o módulo auditivo artificial para um robô assistivo, aplicando diferentes técnicas utilizadas pela literatura e, assim, conseguir um resultado de baixo custo computacional e monetário que possa ser incorporado junto aos outros módulos do robô proposto. Por tal motivo, esse capítulo é separado em três diferentes partes, cada uma representando uma das funções que o módulo tem que executar, sendo: (i) identificação de locutor, (ii) identificação de emoções estereotipadas, (iii) localização de fonte sonora.

Primeiramente, houve uma extensa pesquisa pelas técnicas mais utilizadas e como integrar os projetos da melhor maneira possível. Várias técnicas diferentes foram testadas e, para o resultado final dos projetos, escolheram-se as que apresentaram melhores resultados. Para avaliar a eficiência de cada uma das partes do projeto, seus resultados foram comparados com os obtidos por outros pesquisadores. Uma análise de cada resultado e suas implicações será feita no decorrer das seções desse capítulo.

Tomou-se como base para a execução desse trabalho, que o melhor método para extração de características, tanto para a identificação de um locutor quanto para a de emoções, seria MFCC (com delta e delta-delta). Dessa forma, o custo computacional é reduzido, pois os parâmetros gerados para uma tarefa, podem ser utilizados para outras tarefas. Dentre as técnicas testadas, RNA foi escolhida por ser confiável, de rápida resposta e por ter o melhor desempenho dentre as que foram testadas. É debatível se a utilização de RNAs sempre produziriam as melhores soluções ou não, mas de fato é uma técnica muito utilizada pela literatura e é simples o suficiente para que possa ser executada por um microprocessador de aplicação embarcada.

Para essa etapa o MATLAB foi utilizado. Para a execução desse trabalho, achou-se necessário desenvolver todas as etapas do processo. Dessa maneira, esse trabalho não se utiliza de funções, programas ou simulações de outros autores, apenas conhecimentos prévios que foram profundamente pesquisados. Todas as partes do código, desde a interpretação do áudio até a resposta de cada teste, foram elaboradas para o desenvolvimento apenas desse trabalho.

Todas os testes realizados foram feitos em um computador com Windows 8.1, processador Intel i7-4700MQ (quarta geração) de 2.2 GHz com 8 GB de memória DDR3.

5.1 PROJETO: IDENTIFICAÇÃO DE UM LOCUTOR

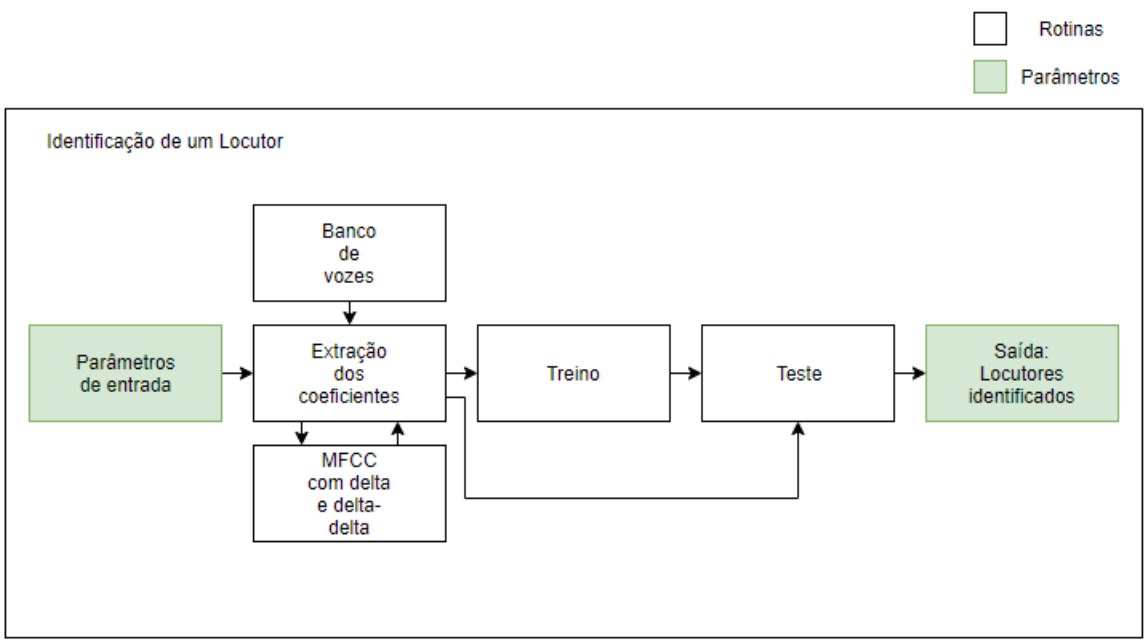
Nessa etapa, busca-se identificar quem está falando. Conforme foi descrito na subseção 4.2.3, primeiro geram-se os MFCCs, depois treinam-se redes neurais (Figura 27 do Capítulo 4) e depois avalia-se a RNA treinada (Figura 28 do Capítulo 4).

Para essa etapa do projeto, utilizou-se:

1. Para a captura dos áudios:
   - 5 locutores, cada um falou 100 palavras.
   - A frequência de amostragem dos áudios utilizados foi de 44100 Hz.
2. Para gerar os MFCCs com delta e delta-delta:
   - 1024 amostras na frequência por *frame*.
   - 50% de superposição entre *frames* adjacentes.
   - 26 filtros no banco de filtros de Mel.
   - O melhor resultado foi obtido se extraindo 45 coeficientes (15 do MFCC, 15 do delta e 15 do delta-delta).
3. Para o treinamento:
   - 80% das amostras de áudio foram utilizadas para o treinamento (400 palavras).
   - No melhor resultado, 15 RNAs em paralelo foram utilizadas
4. Para a verificação:
   - 20% das amostras de áudio foram utilizadas para a verificação (100 palavras).

Para o desenvolvimento dessa etapa, criou-se um código principal e 5 rotinas secundárias. Primeiramente, quando o código se inicia, ele lê os parâmetros de entrada (discutido acima, como quantidade de amostras por *frame*, filtros, superposição, etc). Logo em seguida, o programa chama a rotina de extração de características, que irá gerar os vetores com os coeficientes tanto para o treino quanto para a verificação. Para tal, duas outras rotinas são necessárias, uma rotina que contenha todo o banco de vozes (retirando-se os silêncios) e outra rotina que calcula o MFCC do sinal de áudio em questão. É gerado então um vetor, possuindo todos os vetores de treino. Logo após os vetores serem gerados, a rotina de treino é chamada, onde geram-se as RNAs. Em seguida, o sistema é avaliado com a última rotina. Enfim, há os resultados dos locutores que é o dado de saída dessa etapa. A representação do fluxograma dessa etapa pode ser na Figura 32. Vale a pena citar que essa estrutura vista na Figura 32 é similar a utilizada em outras partes desse

Figura 32 – Fluxograma do projeto com suas rotinas criadas para esse trabalho.



Fonte: Autor

trabalho, como para a etapa da identificação de emoções, assim como para a construção do módulo, adaptando-os da maneira necessária.

Para manter uma coerência e maior entendimento dos testes e resultados obtidos, primeiramente será abordado, nesse trabalho, o melhor resultado do qual se conseguiu chegar. A partir desse ponto, irá se comparar o melhor com outros resultados obtidos durante o desenvolvidos desse trabalho e com o de outros pesquisadores. É importante notar que, como o melhor resultado será o primeiro a ser apresentado e, posteriormente, diferentes comparações serão feitas, o desenvolvimento desse capítulo não segue a ordem cronológica que desenvolveu esta pesquisa.

Todos os códigos desenvolvidos para essa e as outras etapas de projeto estão disponíveis em: <https://github.com/marciooluizz/Modulo-Auditivo-Artificial>.

5.1.1 A melhor configuração

O melhor resultado ocorreu quando se utilizaram 45 coeficientes para a extração de características, 15 RNAs como classificadores, Rprop como aprendizado supervisionado, MSE como cálculo do erro e a função do MATLAB *fitnet* para gerar as RNAs. Para obter tal resultado, rodou-se a rotina com os melhores parâmetros (em comparativo com as outras possibilidades de parâmetros testados) 100 vezes e utilizou-se a rotina que obtivesse o melhor resultado. Das 100 vezes que a rotina rodou, a maioria das vezes a rede obteve

um acerto entre 95-96%.

Para esta configuração proposta, o melhor resultado obtido, dentre todos testados, teve uma precisão de 97%. Este resultado mostra que essa etapa projeto pode dizer qual do cinco locutores falou em 97 dos 100 casos testados. Como pode ser visto na matriz de confusão da Figura 33, houve uma identificação correta do locutor 1 em 100% das vezes, 90% do locutor 2, 100% do locutor 3, 100% do locutor 4 e 95% do locutor 5.

É importante notar que a matriz de confusão gerada utiliza 100 trechos de áudio (20 de cada locutor) que não foram utilizados pelas redes para fazer o treinamento, logo são completamente desconhecidos para o sistema. Para a matriz de confusão da Figura 33 assim como todas as outras desse trabalho, as linhas representam os valores reais e as colunas os valores preditos.

Figura 33 – Matriz de confusão do melhor resultado para a identificação de um locutor.

Matrix de Confusão	Locutor 1	Locutor 2	Locutor 3	Locutor 4	Locutor 5	Total
Locutor 1	20	0	0	0	0	20 100%
Locutor 2	1	18	1	0	0	20 90%
Locutor 3	0	0	20	0	0	20 100%
Locutor 4	0	0	0	20	0	20 100%
Locutor 5	0	0	1	0	19	20 95%
Total	21 95,23%	18 100%	22 90,9%	20 100%	19 100%	100 97%

Fonte: Autor

Conforme também pode ser visto na Figura 33, o locutor 1 foi confundido pelo locutor 2 uma vez, o locutor 2 não foi confundido por nenhum outro locutor, o 3 foi confundido pelo 2 e pelo 5, porém, nem os locutores 4 e 5 foram confundidos por nenhum outro.

Comparando esse resultado com o obtido por outros autores, chega-se a conclusão de que a precisão de 97% obtida pelo treinamento do melhor sistema é um resultado adequado e dentro dos parâmetros de outros estudos realizados por diferentes pesquisadores

(conforme será visto na subseção 5.1.4), especialmente para uma aplicação em que não se possui grandes poderes computacionais como a de um desenvolvimento de um módulo para um robô assistivo.

5.1.2 Outras Configurações e Resultado

Para que pudesse ser atingido o valor da melhor configuração visto na subseção 5.1.1, precisou-se fazer uma série de testes prévios com diferentes parâmetros, redes e classificadores. Dessa forma, será discutido aqui as outras configurações com parâmetros distintos e suas implicações em comparação com o melhor. Mesmo que o melhor resultado tenha tido 97%, é importante ressaltar que a maior parte dos sistemas gerados por essa etapa tinham um acerto entre 95 e 96%, podendo esses serem comparados como desempenhos similares ao do sistema atual.

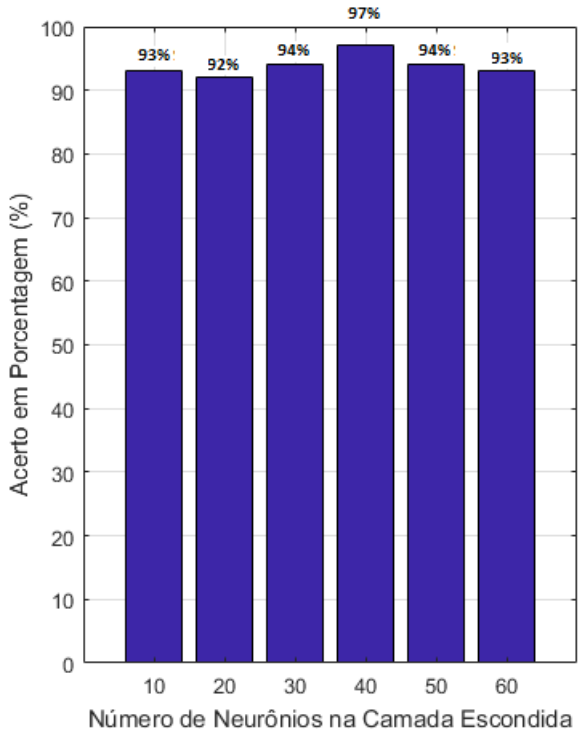
5.1.2.1 Número de Neurônios na Camada Escondida

Um dos parâmetros avaliados é a quantidade de neurônios na camada escondida. A falta de neurônios faz com que a rede não possua a generalização necessária para que tenha uma alta taxa de acerto. Porém, um número muito grande de neurônios faz com que haja um caso de *overfitting* (conforme foi visto na subseção 3.7.4). Portanto, existe um número ideal de neurônios para cada caso. De forma empírica, foi descoberto que 40 neurônios na camada escondida produzia o melhor resultado se comparado com mais ou menos neurônios, conforme pode-se ver na Figura 34. Mantendo todos os outros parâmetros das redes iguais (quantidade de coeficientes, múltiplas redes neurais, etc) e apenas variando a quantidade de neurônios das redes, obteve-se o resultado que: com 10 neurônios na camâda escondida, o acerto foi de 93%, com 20 neurônios um acerto de 92%, 30 neurônios teve 94%, 40 teve 97%, 50 teve 94% e 60 teve 93%.

5.1.2.2 Quantidade de Coeficientes no MFCC

Um dos parâmetros que podem ser alterados no sistema é a quantidade de termos do MFCC que serão mantidos (conforme visto na subseção 3.4.7). Cada coeficiente a mais mantido, geram 3 coeficientes a mais para alimentar a rede, pois ele gerará também o delta e o delta-delta. Após estes testes, chegou-se à conclusão empírica que 15 coeficientes geravam os melhores resultados na hora do treinamento das RNAs, que pode ser visto na Figura 35. Conforme feito anteriormente (na subseção 5.1.2.1), todos os outros parâmetros permaneceram iguais e apenas a quantidade de coeficientes foi alterada para se fazer essa comparação, da qual se viu que, com 12 coeficientes, a precisão era de 94%, com 13 coeficientes era de 92%, com 14 era de 94% e com 15 de 97%.

Figura 34 – Quantidade de acertos das redes com diferentes quantidades de neurônios na camada escondida.



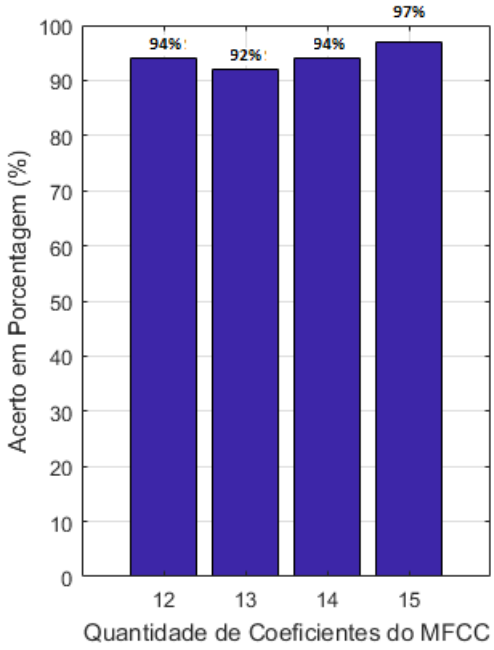
Fonte: Autor

5.1.2.3 Quantidade de RNAs

Pode-se dizer que uma boa RNA é aquela que possui uma baixa função de erro. De acordo com a literatura, quanto menor o erro, mais precisa será a rede (conforme visto na subseção 3.7.4). Portanto, teoricamente, se muitas redes forem treinadas e seus erros forem comparados, pode-se escolher a melhor (a que tiver o menor erro). Para esse teste, 50 redes neurais foram geradas e comparou-se o resultado da rede com menor função de erro desse conjunto com o resultado coletivo das 50 redes juntas (que inclui a de menor erro), que pode ser visto na Figura 36, em que 1 RNA obteve um acerto de 93%, contra 96% produzidos pelas 50 RNAs trabalhando em paralelo. Para efeito de comparação, outros testes semelhantes a este foram rodados 10 vezes e, em nenhuma delas, a melhor rede individual conseguiu ter um resultado superior ou igual ao do conjunto de várias redes. Isso foi visto na subseção 3.7.5 e acontece porque o fraco de uma rede é compensado com o forte de outra rede e, na média, produzem em conjunto um resultado de saída melhor do que o que uma única rede conseguiria. Vale a pena comentar que essas 50 RNAs usadas nesse caso não representam a melhor configuração (subseção 5.1.1), porém mostram que o resultado de múltiplas RNAs é sempre superior ao de uma única RNA.

Uma vez que foi decidido que haveria mais de uma RNAs, faltava descobrir a

Figura 35 – Quantidade de acertos das redes com diferentes quantidades de coeficientes do MFCC.



Fonte: Autor

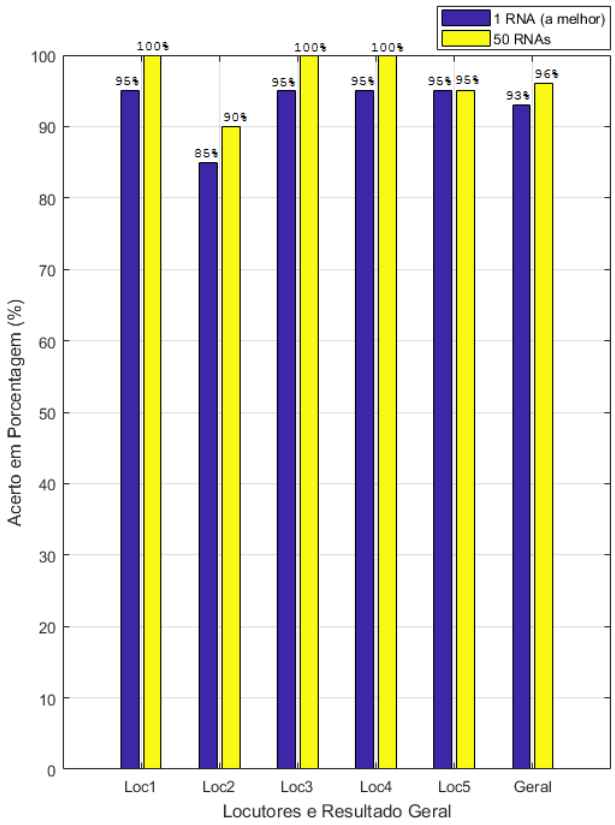
melhor quantidade de RNAs. Chegou-se ao número 15 RNAs de forma empírica, pois com esse valor, havia robustez suficiente no sistema e que, em sua maioria, as respostas de saída eram altas, entre 95 e 97%. Na Figura 37 pode-se ver o exemplo das respostas do sistema para 1 RNA (89%), 3 RNAs (93%), 8 RNAs (95%) e 15 RNAs (97%). Nota-se que o resultado com 8 RNAs pode ser considerado similar ao de 15, conforme comentado na subseção 5.1.2. Porém, testando-se várias vezes diferentes, percebeu-se que com 15 RNAs a robustez era maior pois a média dos acertos era maior. É importante citar que, ao invés do caso anterior, esses sistema com 1 RNA não foi gerado escolhendo-se a rede com menor erro, sendo simplesmente o resultado para quando se gera uma única rede.

5.1.2.4 Tipos de Redes Neurais Artificiais

O MATLAB oferece várias funções para gerar RNAs. Três diferentes funções foram testadas (*fitnet*, *patternnet* e *feedforwardnet*) e houve uma preferência pela função *fitnet*.

Todas as funções testadas são RNAs do tipo *feedfoward*. De acordo com a documentação do MATLAB, a função *feedforwardnet* gera uma rede do tipo *feedfoward* mais genérica. A função *patternnet* é um caso da *feedforwardnet*, em que os vetores de treino e de saída são sempre 0 ou 1 e é normalmente utilizada para reconhecimento de padrões. A função *fitnet* também é um caso da *feedforwardnet*, focada na regressão (o MATLAB utiliza um ajustador de curvas).

Figura 36 – Quantidade de acertos de 1 RNA em comparação com 50 RNAs.



Fonte: Autor

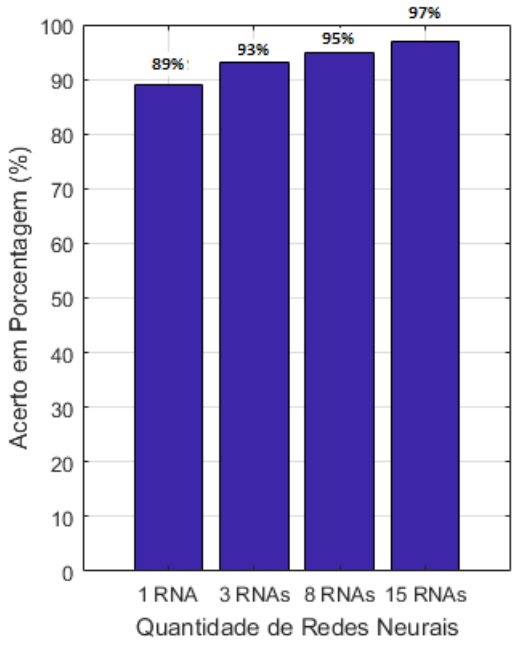
Conforme pode ser visto na Figura 38, a função *patternnet* gerou os piores resultados dentre as funções testadas (porém a mais rápida), com acertos de 93%, contra 96% da *feedforwardnet* e 97% da *fitnet*. As funções *feedforwardnet* e *fitnet* podem ser consideradas possuindo resultados similares, o que significa que qualquer uma delas produziria os resultados desejados. A rede *fitnet* foi escolhida por possuir uma leve vantagem no tempo de execução (cerca de 10% mais rápida).

5.1.2.5 Diferentes Classificadores

Para o desenvolvimento desse trabalho, outros classificadores também foram testados além de RNAs, mas nenhum obteve um desempenho equiparado ao de redes neurais artificiais. Dentre os outros classificadores testados estão *learning vector quantization* (LVQ), *fuzzy C-mean* (FCM) e *adaptive neuro fuzzy inference system* (ANFIS). Uma breve explicação sobre cada um dos classificadores testados foi vista na seção 3.8 durante a Fundamentação Teórica.

1. LVQ:

Figura 37 – Quantidade de acertos de diferentes quantidades de RNAs trabalhando em conjunto.



Fonte: Autor

Para um projeto utilizando LVQ, um resultado de 70% foi obtido. Os autores Martinez, Perez e Escamilla (2012) obtiveram 90% de acerto para 10 locutores com LVQ.

2. FCM:

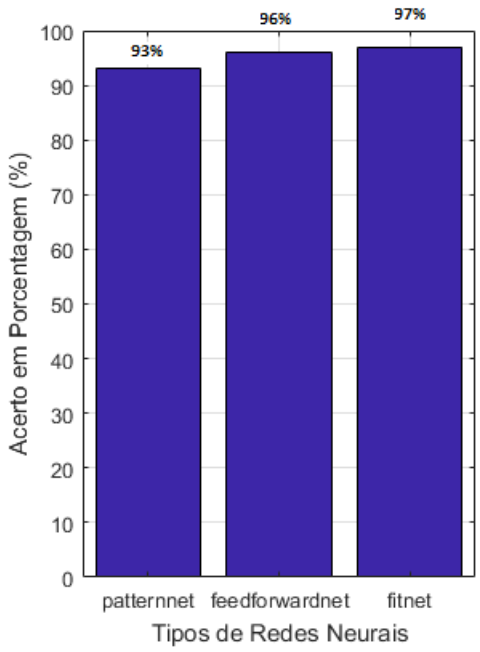
   Para um projeto utilizando FCM, apenas 44% das vezes houve a identificação correta do locutor. Alguns autores conseguiram bons resultados utilizando FCM, como é o caso de Bansal e Bharti (2015) que conseguiu 90% de precisão.

3. ANFIS:

   Apesar de ser muito promissor e ser considerado uma técnica com altos resultados, o treinamento de uma rede ANFIS é muito pesado. Quando rodando a rotina, o treinamento em um computador durou mais de uma hora e foi interrompido, pois não seria compatível com a ideia de ser utilizada numa aplicação embarcada.

Um gráfico de comparação dos experimentos desenvolvidos utilizando-se 15 RNAs (97%), LQV (70%) e FCM (44%) pode ser visto na Figura 39.

Figura 38 – Quantidade de acertos para diferentes funções para RNA do MATLAB.



Fonte: Autor

5.1.3 Tempo de execução das rotinas

Para a aplicação embarcada desse projeto, o tempo de execução tem que ser levado em consideração. Por um lado, pode-se gastar um pouco mais de tempo para se ter uma rotina melhor. Por outro lado, um treinamento que possa vir a demorar horas (como seria o caso se *deep learning* fosse usado, por exemplo) não seria adequado num módulo que estará integrado a um robô. Nessa subseção, uma análise e comparação dos diferentes tempos de execução das rotinas será feita.

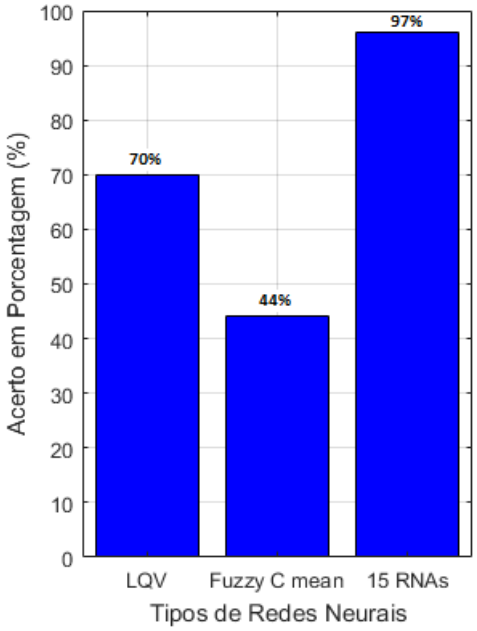
5.1.3.1 Tempo do Melhor Resultado

Novamente, antes de comparar os resultados com outros resultados, primeiramente será analisado o tempo de execução do melhor resultado (que foi visto previamente na subseção 5.1.1).

Conforme pode ser visto na Figura 40, o tempo para se executar a rotina completa é de aproximadamente 29 segundos, onde 7 segundos são gastos gerando os vetores com as características que alimentarão as RNAs, 19 segundos treinando 15 redes neurais e 3 segundos gerando o MFCC dos áudios da etapa de verificação e avaliação do sistema (gerando na saída o desempenho das redes).

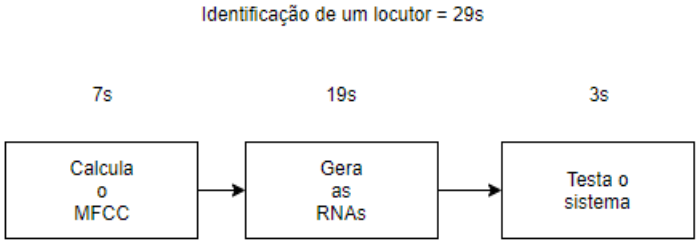
Apesar, do mais importante ser o tempo de teste/validação do sistema, ele é relativamente rápido (menos do que 4 segundos) para todos os testes realizados. Também

Figura 39 – Quantidade de acertos para diferentes classificadores.



Fonte: Autor

Figura 40 – Tempo para rodar a rotina da melhor configuração no computador.



Fonte: Autor

nota-se que todas as formas de aprendizado supervisionado necessitam de vetores com características para treiná-los, considera-se que os primeiros 7 segundos da fase de treino são razoavelmente imutáveis. O tempo de teste/validação pode ser um pouco mais rápido que 3 segundos, porém, essa resposta, não deixa de ser rápida o suficiente, portanto todos os sistemas gerados tiveram desempenhos similares na hora de gerar resultados. A parte que mais varia e que será mais examinada é o tempo para se gerar a(s) RNA(s) ou outros classificadores. Mesmo que os classificadores só sejam gerados uma única vez e que esse processo possa rodar em "*background*" numa placa embarcada, o tempo para gerá-los é um parâmetro interessante para ser analisado, pois valores muito grandes demonstram

necessidade de um alto poder computacional e valores menores demonstram uma maior otimização do sistema em comparação com outros sistemas.

5.1.3.2 Tempos de Outras Configurações

Uma análise do tempo de diferentes configurações foi feito conforme pode ser visto na Tabela 2.

1. Funções de RNA do MATLAB:

   Foi feita uma análise comparativa do tempo de execução e da quantidade de acertos entre as funções de RNA do MATLAB. *feedforwardnet* gerou as 15 RNAs em 21 segundos, *patternnet* em 13 segundos e *fitnet* em 19 segundos.

2. Diferentes aprendizados supervisionados:

   Foi feita uma comparação do aprendizado por Rprop e por Levenberg-Marquardt para a rede *fitnet* do MATLAB. Levenberg-Marquardt levou 6720 segundos pra treinar 15 redes com um acerto de 95%. Rprop gerou 15 RNAs em 19 segundos com um acerto de 97%.

3. Outros classificadores:

   Apenas 1 RNA levou 3 segundos para ser treinada, com uma taxa de acerto de 93%. LVQ levou 180 segundos para ser treinada e acertou em 70% dos casos. FCM teve um treinamento muito rápido, menos de 1 segundo, sendo a mais rápida, porém com o pior desempenho, 44% de acerto.

Tabela 2 – Tempo e acerto de diferentes classificadores

Classificador	**Tempo(s)***	**Acerto(%)**
15 RNA (fitnet Rprop)	19s	97%
15 RNA (fitnet Levenberg-Marquardt)	6720s	95%
15 RNA (patternnet Rprop)	13s	94%
15 RNA (feedfowardnet Rprop)	21s	96%
1 RNA (fitnet Rprop)	3s	93%
LVQ	180s	70%
FCM	1s	44%

*Tempo para gerar apenas o classificador durante a execução da rotina.

5.1.4 Resultado do Projeto de Identificação de um Locutor

Baseado no custo-benefício computacional de acertos e de tempo de execução para uma tarefa de um robô assistivo de baixo custo, chegou-se a conclusão que o teste que

demora 19 segundos pra gerar as redes neurais artificiais, 3 segundos para verificar a rede e possui uma taxa de acerto de 97% (subseção 5.1.1) foi o melhor resultado dentre todos os testes realizados. É um resultado que está dentro dos parâmetros desejados para uma aplicação embarcada e, ao mesmo tempo, possui uma resposta condizente com as encontradas na literatura.

Como exemplo de trabalhos de outros autores que chegaram a outros resultados, podem-se citar os trabalhos de Chauhan e Chandra (2017) obteve 96% de precisão para 5 locutores utilizando MFCC com centroide e RNA, Bansal e Bharti (2015) conseguiu 94% utilizando de MFCC e *fuzzy C mean clustering* para uma quantidade desconhecida de locutores, Martinez, Perez e Escamilla (2012) teve um acerto de 90% utilizando de MFCC e LVQ (*Linear Vector Quantization*) para um total de 10 locutores e Maesa et al. (2012) em que houve um acerto de 96,2% para 450 locutores utilizando-se de MFCC e GMM (*Gaussian Mixture Model*).

5.2 PROJETO: IDENTIFICAÇÃO DE UMA EMOÇÃO ESTEREOTIPADA

Nessa etapa da projeto, busca-se identificar a emoção na voz de quem fala. Conforme foi descrito na subseção 4.2.3, primeiro gera-se os MFCC, depois treina-se redes neurais (Figura 27 do Capítulo 4) e, por fim, avalia-se as RNAs treinadas (Figura 28 do Capítulo 4).

Para essa etapa, utilizou-se:

1. Para a captura dos áudios:
    - 6 emoções: raiva, nojo, medo, alegria, calma e tristeza.
    - Para o banco SAVEE foram utilizados 3 locutores com 30 trechos de áudio por emoção por locutor, resultando em 90 trechos de áudio por emoção (um total de 540 trecho de áudio). A frequência de amostragem da gravação desse banco foi de 44100 Hz.
    - Para o banco Emo-DB foram utilizados 10 locutores em que cada um falou uma quantidade diferente de trechos por emoção, mas cada emoção possui 92 trechos de áudio (um total de 552 trecho de áudio). A frequência de amostragem da gravação desse banco foi de 16000 Hz.
2. Para gerar os MFCCs com delta e delta-delta:
    - 1024 amostras na frequência por *frame*.
    - 50% de superposição entre *frames* adjacentes.
    - 26 filtros no banco de filtros de Mel.

- Foram extraídos 45 coeficientes (15 do MFCC, 15 do delta e 15 do delta-delta).

3. Para o treinamento:

   - 86,6% das amostras de áudio foram utilizadas para o treinamento (468 trechos de áudio) para SAVEE.
   - 84,7% das amostras de áudio foram utilizadas para o treinamento (468 trechos de áudio) para Emo-DB.
   - Foram utilizadas 15 RNAs em paralelo.

4. Para a verificação:

   - 13,4% das amostras de áudio foram utilizadas para o verificação (72 palavras) para SAVEE.
   - 15,3% das amostras de áudio foram utilizadas para o verificação (84 palavras) para Emo-DB.

Para essa etapa, escolheu-se percorrer o mesmo caminho desenvolvido pela identificação de um locutor ao invés de se começar tudo do início novamente, dessa forma pode-se utilizar as rotinas criadas anteriormente, apenas as adaptando para o caso de reconhecer emoções ao invés de locutores. Dessa forma a Figura 32 também é válida para essa etapa do projeto, uma vez que os códigos foram apenas adaptados e mudou-se o banco de vozes.

Primeiramente será feita uma análise comparativa dos resultados obtidos para as 6 distintas emoções utilizando os diferentes banco de vozes (SAVEE e Emo-DB) e do resultado de se usar os dois bancos de vozes juntos. Depois, irá se comparar formas diferentes para se aplicar o *core affect*. Por fim, uma comparação utilizando outros classificadores será feita.

Outro fator a ser considerado

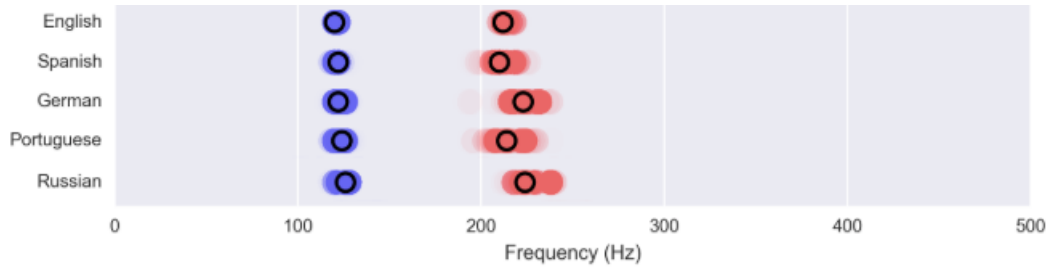
5.2.1 Diferentes Bancos de Vozes

Conforme foi visto anteriormente na subseção 4.2.2, foram utilizados dois bancos de falas, o SAVEE (*Surrey Audio-Visual Expressed Emotion*) e o Emo-DB (*Berlin Emotion Database*). Como foram utilizados os mesmos parâmetros do melhor resultado da identificação de locutor, se tornaria repetitivo fazer as mesmas análises da quantidade de coeficientes, números de RNAs treinadas e etc (que já foram feitas na seção 5.1). Portanto, sabendo-se que os mesmos parâmetros foram utilizados, foi feita uma análise comparativa entre os dois bancos de vozes a fim de se descobrir quais seriam os resultados obtidos.

No banco SAVEE foram utilizados 3 locutores, enquanto o Emo-DB 10 locutores. A quantidade de amostras de áudio, por emoção, utilizada foi similar (90 para o SAVEE

e 92 para o Emo-DB). Mesmo assim, conforme será visto a seguir, os resultados de um sistema utilizando o banco SAVEE se mostraram muito superiores se comparados a um sistema utilizando Emo-DB. O sistema com a fusão dos dois bancos de emoções possui um resultado ainda inferior ao de sistemas com os bancos separados. Isso acontece por alguns fatores, como o que classificadores possuem uma certa dificuldade em identificar só as emoções sem levar em consideração, de alguma maneira, os locutores. Outro fator importante é que diferentes línguas faladas (inglês, alemão e português no caso) possuem uma distribuição diferente de frequências em suas palavras e emoções, conforme pode ser visto na Figura 41. No Emo-DB foram usados 10 locutores e no SAVEE apenas 3. A identificação de emoções numa situação real é menor se o classificador não conhecer previamente os locutores. Por exemplo, no artigo dos autores Kim et al. (2017), o banco de vozes Emo-DB é utilizado e eles conseguem um acerto de 52,7% com esse banco (que é inferior ao resultado encontrado nesse trabalho, conforme será visto na subseção 5.2.4), porém, ao fazerem uma aplicação em tempo real, com locutores desconhecidos para o sistema deles, o acerto foi tão baixo que eles preferiram não o publicar.

Figura 41 – Diferentes "*pitches*" na frequência para diferentes línguas faladas.



Fonte: https://erikbern.com modificado pelo autor

5.2.1.1 *Surrey Audio-Visual Expressed Emotion* (SAVEE)

O projeto utilizando SAVEE possui o melhor resultado entre todas as configurações testadas para identificação de emoções. Da mesma maneira que foi feita anteriormente (na melhor identificação de um locutor), rodou-se a rotina 100 vezes e obteve-se o melhor resultado. As emoções foram corretamente classificadas em 75% dos casos conforme pode ser visto na matriz de confusão da Figura 42 (as linhas representam os valores reais e as colunas os valores preditos). Para a maior parte dos resultados, dentre as 100 vezes que a rotina rodou, os acertos ficaram entre 69% e 74%, podendo esses serem considerados resultados similares.

As RNAs conseguiram identificar a emoção raiva 67% das vezes, nojo 67%, medo 75%, alegria 58%, calma 92% e tristeza 92%. É interessante notar como há uma certa discrepância, mostrando que a RNA tem mais facilidade de identificar algumas emoções em

Figura 42 – Matriz de confusão do melhor resultado para a identificação de uma emoção estereotipada (com SAVEE).

Matrix de Confusão	Raiva	Nojo	Medo	Alegria	Calma	Tristeza	Total
Raiva	8	0	1	3	0	0	12 67%
Nojo	0	8	0	0	2	2	12 67%
Medo	1	1	9	0	0	1	12 75%
Alegria	2	3	0	7	0	0	12 58%
Calma	0	0	0	0	11	1	12 92%
Tristeza	1	0	0	0	0	11	12 92%
Total	12 67%	12 67%	10 90%	10 70%	13 85%	15 73%	72 75%

Fonte: Autor

detrimento de outras. Por exemplo, alegria é bastante confundido com emoções negativas como raiva e nojo por serem emoções fortes, equanto emoções mais tranquilas como tristeza e calma obtiveram um desempenho mais elevado.

5.2.1.2 *Berlin Emotion Database* (Emo-DB)

Apesar de possuir mais locutores, o projeto utilizando Emo-DB não conseguiu ter a mesma taxa de acerto do que utilizou SAVEE, com 64,3% de acerto, como pode ser visto na Tabela 3. Conforme feito anteriormente, a rotina foi rodada 100 vezes para se obter o melhor resultado possível.

Na comparação dos acertos de um sistema utilizando o banco Emo-DB com o de um utilizando SAVEE, pode-se ver que o que usa Emo-DB é superior apenas na identificação da emoção “alegria”, mas na média, o sistema que usa SAVEE acertou em 75% das vezes, em comparação com 64,3% de acerto de um sistema utilizando Emo-DB.

Tabela 3 – Desempenho do sistema utilizando o banco de falas Emo-DB

Emoção	**Acerto (%)**
Raiva	36%
Nojo	50%
Medo	71%
Alegria	64%
Calma	71%
Tristeza	93%
TOTAL	64,3%

5.2.1.3 Fusão dos Bancos de Vozes SAVEE e Emo-DB

Mesmo sabendo que os bancos são em línguas diferentes e com taxa de amostragem diferentes, foi feita uma tentativa de fusão dos mesmos tentando buscar uma maior generalização e robustez na classificação de emoções. Quando a fusão dos bancos foi feita, ganhou-se o dobro de amostras para análise pelas redes neurais, porém houve um aumento do número de locutores. Nota-se também que são bancos diferentes, gravados em condições diferentes. Conforme pode ser visto na Tabela 4, ao fazer a fusão e rodar a rotina 100 vezes, obteve-se um resultado inferior aos obtidos utilizando-se SAVEE ou Emo-DB, conseguindo um acerto geral de 52%. Comparado com os 64,3% e 75% de acerto de sistemas com Emo-DB e SAVEE respectivamente, percebe-se que foi um resultado inferior.

Tabela 4 – Desempenho do sistema utilizando a fusão dos bancos de falas SAVEE e Emo-DB

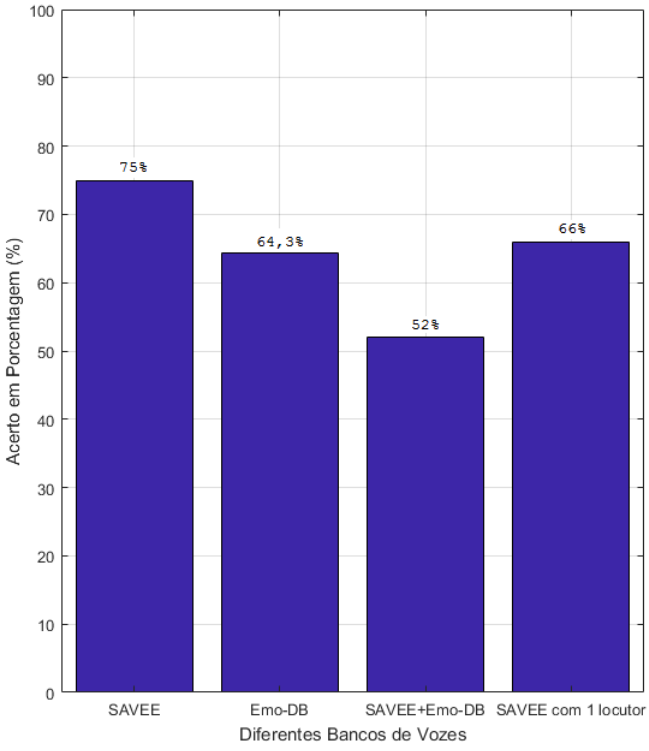
Emoção	**Acerto (%)**
Raiva	53%
Nojo	34%
Medo	44%
Alegria	59%
Calma	56%
Tristeza	66%
TOTAL	52%

5.2.1.4 SAVEE com 1 Locutor

Uma vez que uma maior quantidade de locutores gera uma rede com menor quantidade de acertos, ao diminuirmos a quantidade de locutores do banco de falas para apenas 1 (utilizando apenas um dos locutores do SAVEE), a quantidade de acertos não aumentou, pois reduziu-se em 2/3 a quantidade de amostras para treinar as redes e, dessa maneira, não há um treinamento tão bom.

Como pode ser visto na Figura 43, utilizar apenas 1 locutor teve um resultado reduzido em relação a utilizar 3 locutores do SAVEE pois existem menos amostras para o treinamento e, portanto, não consegue-se reduzir tanto a função de erro. Mas, por outro lado, mostrou um resultado superior do que utilizar os 10 locutores do Emo-DB ou os 13 locutores da combinação dos bancos de vozes.

Figura 43 – Quantidade de acertos utilizando diferentes bancos de vozes.



Fonte: Autor

5.2.2 Diferentes Configurações Utilizando *Core Affect*

Conforme foi visto na subseção 3.9.1, *core affect* é um mapa cartesiano, onde as emoções são classificadas como sendo agradável ou desagradável em seu eixo horizontal e alta excitação (ativação) ou baixa excitação (desativação) no eixo vertical (como foi visto na Figura 20 do Capítulo 3).

Propõe-se o uso do *core affect* de três maneiras diferentes para se comparar os seus distintos resultados. Primeiramente utilizará RNAs que identificam as 6 emoções e se reorganizará o resultado com o *core affect*. Depois, será calculado as RNAs utilizando o *core affect* como entrada, logo gerarão apenas 4 neurônios na camada de saída, um para cada quadrante. Por fim, utilizará o *core affect* em cascata, conforme foi visto na Figura 29

do Capítulo 4, utilizando-se de 3 sequências de RNAs em cascata. Todas os resultados apresentados nessa subseção foram gerados utilizando o banco de emoções do SAVEE, pois ele apresentou um melhor resultado conforme visto anteriormente.

5.2.2.1 Reconfiguração

Por mais contra-intuitivo que seja, o melhor resultado obtido com *core affect* foi utilizando o sistema para identificação de 6 emoções com SAVEE (que foi visto na Figura 42) e reorganizar os acertos e erros dentro das 4 possibilidades cartesianas. Dessa maneira, conseguiu-se ter um acerto de 80,5% das vezes reorganizando as emoções em 4 possibilidades, conforme pode ser visto na Figura 44 (as linhas representam os valores reais e as colunas os valores preditos), em que alegria fica no primeiro quadrante, raiva e nojo no segundo quadrante, medo e tristeza no terceiro quadrante e calma no quarto quadrante.

Figura 44 – Quantidade de acertos reconfigurando o sistema para *core affect*.

Matrix de Confusão	Desagradável Baixa	Agradável Baixa	Desagradável Alta	Agradável Alta	Total
Desagradável Baixa	21	2	1	0	24 88%
Agradável Baixa	1	11	0	0	12 92%
Desagradável Alta	2	0	19	3	24 79%
Agradável Alta	3	0	2	7	12 58%
Total	27 78%	13 85%	22 86%	10 70%	72 80,5%

Fonte: Autor

Porém, se cada eixo do *core affect* for analisado de forma independente, consegue-se um resultado ainda melhor. Esse sistema identifica corretamente em 85% das vezes se é uma emoção prazerosa ou não e em 92% das vezes se é uma emoção de alta ou baixa ativação, fazendo com que, mesmo que o robô possa errar em que quadrante está exatamente, existe uma chance maior dele acertar a ativação ou se é ou não prazeroso.

5.2.2.2 RNAs Para *Core Affect*

Gerar redes neurais artificiais especializadas em classificar as emoções em 4 quadrantes se mostrou como sendo o menos efetivo dos métodos. As RNAs têm uma dificuldade maior em reconhecer similaridades entre os dados de entrada quando são classificados dessa maneira. Gerando redes específicas para calcular os casos cartesianos gerou um acerto médio de 72%, conforme pode ser visto na Tabela 5, sendo esse um resultado levemente inferior (porém similar) ao de classificar cada emoção individualmente (75%) e bem inferior ao de reconfigurar o sistema (80,5%).

Tabela 5 – Desempenho do sistema utilizando RNAs para classificar as emoções de forma cartesiana

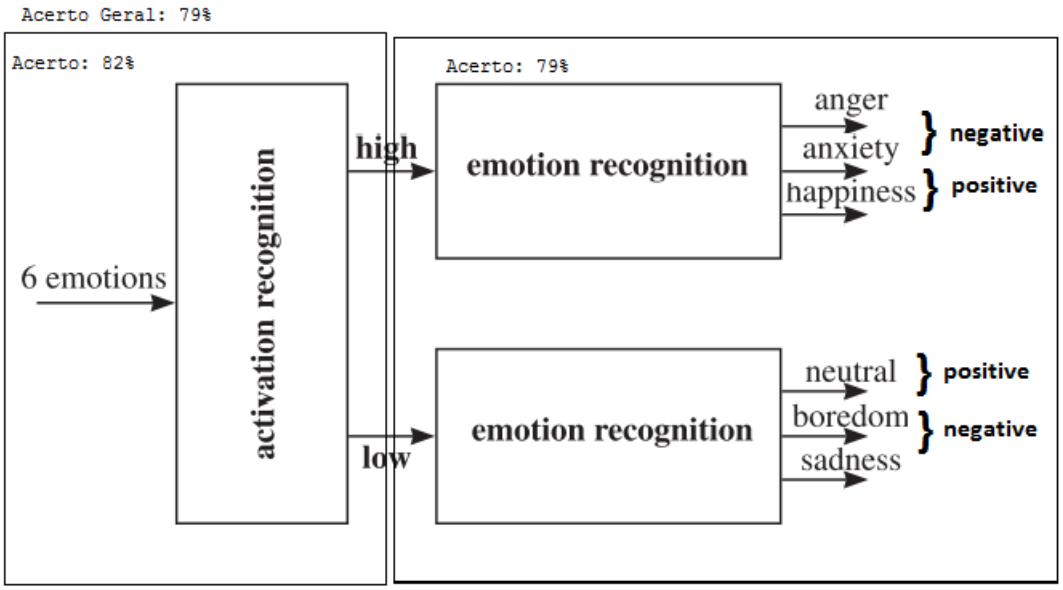
Emoção	**Acerto (%)**
Agradável Alta Ativação	42%
Agradável Baixa Ativação	83%
Desagradável Alta Ativação	71%
Desagradável Baixa Ativação	83%
TOTAL	72%

5.2.2.3 Cascata

Nesse sistema utilizam-se RNAs em cascata, utilizando-se de 3 sequências de redes neurais. Primeiramente identifica-se a ativação da emoção (alta ou baixa) e depois, separadamente, identifica-se se é uma emoção prazerosa ou não. Conforme pode ser visto na Figura 45, o sistema em cascata consegue identificar em 82% dos casos se a emoção é de alta ou baixa ativação e em 79% dos casos ela também consegue identificar se é uma emoção prazerosa ou não. Nota-se que a taxa de acerto de prazeroso ou não é a mesma do sistema porque o sistema é em cascata, ou seja, o resultado de uma parte influencia no todo.

Esse resultado de 79% é muito próximo dos 80,5% e ambos os sistemas podem ser considerados “equivalentes”, mostrando que esse é um sistema que funciona e pode ser aplicado. Por questão de que o sistema reconfigurado possui maior quantidade de acertos entre prazeroso/não prazeroso e alta/baixa ativação individualmente o fez ser escolhido.

Figura 45 – Quantidade de acertos de um sistema em cascata com *core affect*.



Fonte: Lugger e Yang (2008), modificado pelo autor

5.2.3 Diferentes Classificadores para Emoções

Para o desenvolvimento desse trabalho, outros classificadores também foram testados, para identificação de emoções, além de RNAs e MFCC com delta e delta-delta, mas nenhum obteve um desempenho equiparado. Dentre os outros classificadores testados estão *Suport Vector Machine* (SVM) e RNA treinada com MFCC com MEDC (*Mel-Energy spectrum Dynamic Coefficients*). Uma breve explicação de cada uma das técnicas utilizadas foi vista na seção 3.8 durante a Fundamentação Teórica.

1. SVM:

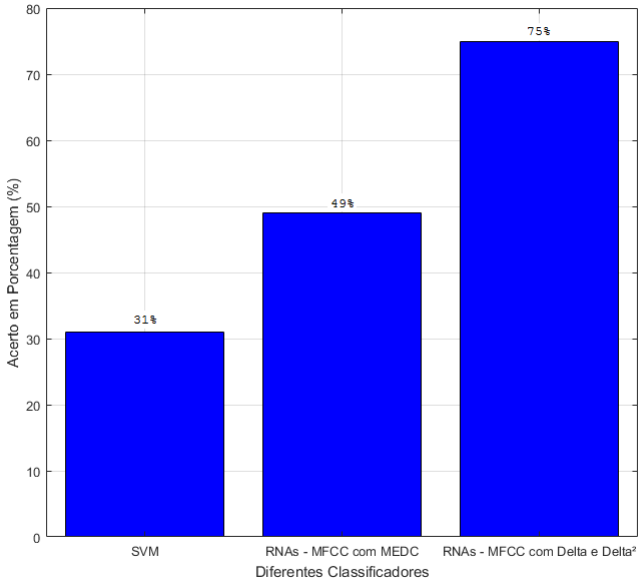
   Para esse teste se utilizou de um SVM com *kernel* polinomial, conseguiu-se um acerto de 31%.

2. RNA com MEDC:

   Utilizando RNA com MFCC com MEDC ao invés de delta e delta-delta também é utilizando na literatura para a identificação de emoções. O sistema, nesse trabalho, que utilizou MFCC com MEDC conseguiu um acerto de 49%.

Como pode ser na Figura 46, dentre todos os testes realizados nesse trabalho, os com RNAs treinadas por MFCC com delta e delta-delta obtiveram o melhor resultado.

Figura 46 – Quantidade de acertos de diferentes classificadores para identificação de emoções.



Fonte: Autor

5.2.4 Resultados do Projeto de Identificação de Emoções Estereotipadas

Baseado na quantidade de acertos de cada uma das configurações exploradas por essa seção, chegou-se a conclusão que o melhor resultado foi obtido utilizando RNAs treinadas com o banco de vozes SAVEE, tendo um acerto de 75% (subseção 5.2.1.1). Utilizando *core affect*, chegou-se a conclusão que o melhor é reconfigurar a rede mencionada anteriormente. Dessa forma, o acerto foi de 80,5%. Porém, se cada eixo for analisado de forma independente, se tem uma identificação correta em 85% e 92% das vezes (subseção 5.2.2).

Os autores Dahake e Shaw (2016) utilizaram SVM com kernel polinomial e conseguiram um acerto global de 83%, já os autores Lalitha et al. (2015) utilizaram RNA treinadas com MFCC + Cepstrum + *shifted* MFCC e conseguiram um acerto de 85%. Lugger e Yang (2008) utilizaram GMM (*Gaussian Mixture Model*) como classificador com MFCC e VQP (*Voice Quality Parameters*) e utilizaram os classificadores em cascata e, quando 2 níveis de cascada foram usados, obtiveram um acerto de 83%, porém ao utilizarem *core affect* e 3 níveis de cascata (5 classificadores em cascata), obtiveram um acerto de 88.8%. Demircan e Kahramanlı (2014) utilizaram classificador KNN (*k-Nearest Neighbor*) com MFCC e obtiveram um acerto de 50%.

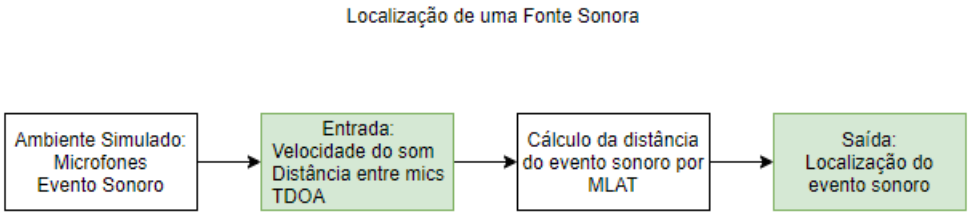
Os resultados encontrados nesse trabalho estão de acordo com o que foi encontrado na literatura.

5.3 PROJETO: LOCALIZAÇÃO DE UMA FONTE SONORA

Para essa parte da etapa de projeto, simulou-se um ambiente virtual com 4 sensores (microfones), em que o som se propagava de sua origem até os microfones em linha reta (campo de onda distante), sem obstáculos e considerando um ambiente perfeito, sem reverberação, ruídos altos ou eco. Considerou-se também que os microfones possuíam uma sensibilidade perfeita, não importando, dessa maneira, a distância entre eles. É verdade que esse ambiente virtual é bem diferente da realidade, porém, o objetivo da aplicação no robô assistivo dessa etapa é saber apenas de maneira aproximada qual a localização da pessoa, uma vez que o robô terá outros sensores, como câmeras, para aumentar a precisão da localização. Da mesma maneira, humanos têm uma noção de qual a direção do som e sua distância, mas eles não possuem certeza até haver uma verificação visual. Nota-se que nessa etapa do projeto não se utiliza de MFCC ou de técnicas de extração de características, sendo a diferença de tempo que o evento sonoro demora para atingir cada microfone o mais importante e que será utilizado no cálculo da multilateração, visto na seção 3.11.

Para essa parte do trabalho, elaborou-se um algoritmo com duas etapas, como pode ser visto na Figura 47. A primeira etapa é a de gerar um evento sonoro simulado e fazer esse evento atingir os microfones simulados. A segunda etapa é, conhecendo apenas a velocidade do som no ar, a distância entre os microfones e o TDOA (*Time Difference of Arrival*), que é a diferença de tempo em que o som atinge cada um dos sensores, calcular a origem sonora utilizando MLAT (Multilateração), conforme foi visto na seção 3.11.

Figura 47 – Fluxograma da simulação de localização de um locutor.



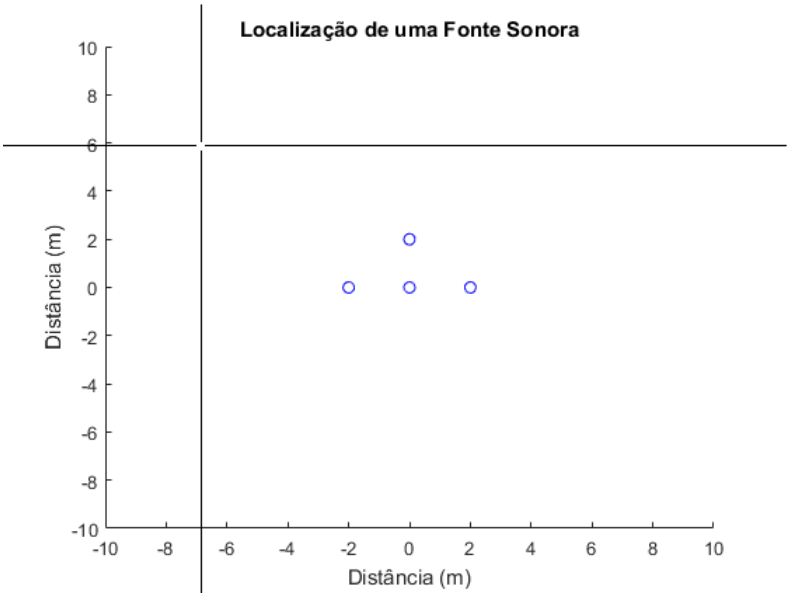
Fonte: Autor

5.3.1 Ambiente Simulado

Esse ambiente é composto por 4 microfones cujas posições são conhecidas e podem ser alteradas pelo programador. O programa deixa, então, o usuário escolher onde acontecerá o evento sonoro (conforme pode ser visto na Figura 48). Em seguida, o programa irá calcular o tempo para a onda sonora atingir cada um dos microfones sabendo-se que

a velocidade do som nesse ambiente simulado é de 343 m/s (que é a velocidade do som aproximada a 20 graus Celsius).

Figura 48 – Ambiente simulado em que o usuário escolhe onde será o evento sonoro.



Fonte: Autor

5.3.2 Localização

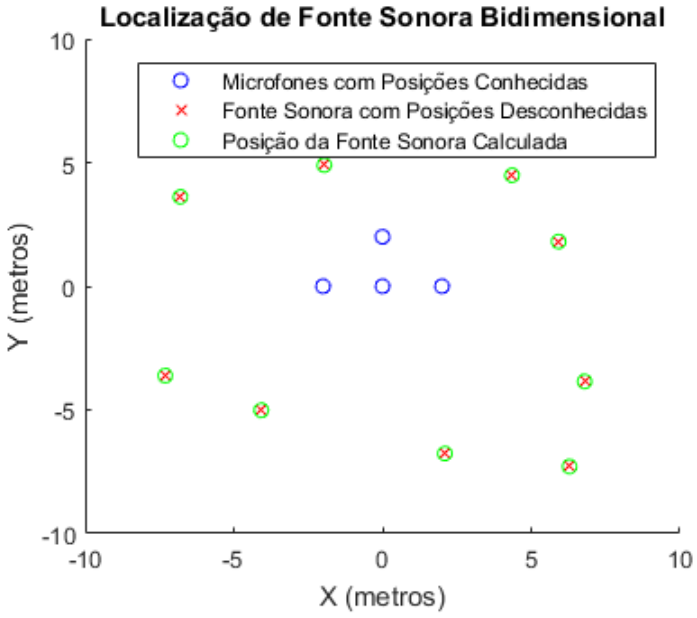
Sem conhecer a posição do evento ou qual o tempo que levou para o evento sonoro atingir os microfones, mas conhecendo a diferença de tempo que o som atingiu cada microfone (TDOA), a velocidade do som no ar naquele meio e a posição dos microfones, o programa calcula, utilizando as expressões matemáticas, que foram vistas na seção 3.11, a posição da fonte sonora.

5.3.3 Resultados da Simulação da Localização de uma Fonte Sonora

Conforme pode ser visto na Figura 49, esta simulação é muito precisa, localizando a fonte sonora 100% do tempo, embora, dificilmente em um ambiente real, essa precisão seja a mesma, já que a simulação atual é configurada em um ambiente perfeito com condições hipotéticas.

Os resultados desta simulação mostraram que a multilateração linear é uma técnica viável e que poderá apresentar resultados satisfatórios a depender das características específicas da configuração experimental utilizada.

Figura 49 – Simulação da Localização de 10 fontes sonoras em 10 momentos distintos.



Fonte: Autor

6 CONSTRUÇÃO DO MÓDULO E RESULTADOS

Para a execução do que foi proposto nesse trabalho, é necessária a construção do módulo auditivo artificial. Para tal, será utilizado o que já foi projetado previamente, com suas distintas funcionalidades. Para isso, esse capítulo será dividido em quatro partes. Uma parte para cada uma das três distintas funções do módulo, construídas e testadas de forma separadas e uma última parte, onde haverá a integração das três tarefas.

Precisou-se fazer uma extensa pesquisa sobre a utilização do BeagleBone Black (BBB) e do Kinect. Vários outros métodos foram considerados e poderiam inclusive serem mais precisos, porém seriam necessários pelo menos quatro microfones com uma boa sensibilidade e uma mesa de som que permitisse a captura de 4 canais de áudio de maneira separada (1 para cada microfone). Porém, para tal, precisaria-se de um investimento de equipamentos mais caros e não disponíveis nos laboratórios, enquanto o Kinect, além de já existir no laboratório da Universidade Federal da Bahia, possui um custo inferior, sendo assim, mais adequado para ser utilizado em um robô de baixo custo.

Em todas as etapas da construção do módulo utilizou-se o melhor resultado obtido na fase de projeto, sendo que para a identificação de uma emoção estereotipada, utilizou-se a própria rede gerada na etapa anterior.

A eficiência dos resultados obtidos na construção do módulo serão discutidas e comparadas com o que foi obtido na etapa de projeto e na literatura também.

Seguindo o que foi feito na etapa de projeto, toda a programação foi feita em MATLAB, utilizando-se do *Support Package for BeagleBone*® dentro do *Embedded Coder*® do MATLAB. Todo o código foi elaborado exclusivamente para esse trabalho, sendo essa uma das vantagens já que os códigos não precisam ser completamente reescritos entre uma etapa e outra, adaptado-os e modificado-os conforme necessidade.

6.1 CONSTRUÇÃO: IDENTIFICAÇÃO DE UM LOCUTOR

Nessa etapa, busca-se identificar quem está falando. Conforme foi descrito na subseção 4.2.3, primeiro geram-se os MFCCs, depois treinam-se redes neurais (Figura 27) e depois avalia-se a RNA treinada (Figura 28).

Para essa identificação, utilizou-se:

1. Para a captura dos áudios:

- 5 locutores, cada um falou por 90 segundos e cortou-se cada áudio em 80 trechos.
- A frequência de amostragem da captura foi de 44100 Hz.

2. Para gerar os MFCCs com delta e delta-delta:
   - Treinamento offline (não embarcado).
   - 1024 amostras na frequência por *frame*.
   - 50% de superposição entre *frames* adjacentes.
   - 26 filtros no banco de filtros de Mel.
   - 45 coeficientes (15 do MFCC, 15 do delta e 15 do delta-delta).
3. Classificador:
   - 15 RNAs em paralelo.

Para o desenvolvimento dessa étapa, o áudio falado por um locutor é capturado por um microfone conectado a um computador. O computador se comunica com o BBB para fazer o processamento. O áudio é então armazenado e, com os áudios já capturados, criam-se novas redes neurais (offline) utilizando os mesmos parâmetros do melhor resultado da etapa de projeto (subseção 5.1.1), com 15 RNAs, Rprop, MSE e *fitnet*, sendo que as rotinas criadas para tal (que foram vistas na Figura 32 do Capítulo 5) foram adaptadas para ao invés de serem reescritas completamente. O BBB devolve os resultados para o MATLAB que são exibidos em seu *prompt* para serem analisados, dessa forma pode-se acompanhar os resultados obtidos.

Para a etapa de verificação, cada locutor falou 10 vezes, totalizando 50 testes, e calculou-se qual foi a taxa de acerto das redes para cada um dos locutores e a taxa de acerto geral.

6.1.1 Resultados para Identificação de um Locutor

Aqui, foi alcançado um acerto de 98% dos casos (conforme pode ser visto na Figura 50, em que as linhas representam os valores reais e as colunas os valores preditos), que é melhor do que o resultado obtido na etapa de projeto que utilizou o banco de falas criado em um estúdio (conforme pode ser visto na Tabela 6). Para o locutor 1, houve 9 acertos de 10 (90% de acerto). Todos os outros 4 locutores obtiveram 100% de acerto, portanto, de 50 testes, 49 foram corretamente identificados.

Apesar da qualidade inferior da captura de áudio *in loco*, se comparado com a de um estúdio, acredita-se que a taxa de acerto foi maior na construção do módulo auditivo artificial porque frases inteiras foram capturadas ao invés de palavras isoladas, dando a rede uma maior capacidade de reconhecimento do estilo de fala de cada locutor.

Figura 50 – Matriz de confusão do resultado do módulo para a identificação de um locutor.

Matrix de Confusão	Locutor 1	Locutor 2	Locutor 3	Locutor 4	Locutor 5	Total
Locutor 1	9	0	0	0	1	10 90%
Locutor 2	0	10	0	0	0	10 100%
Locutor 3	0	0	10	0	0	10 100%
Locutor 4	0	0	0	10	0	10 100%
Locutor 5	0	0	0	0	10	10 100%
Total	9 100%	10 100%	10 100%	10 100%	11 90,9%	50 98%

Fonte: Autor

Tabela 6 – Comparação entre os acertos (em %) das etapas de projeto e construção do módulo para a identificação de locutor

-	**Loc. 1**	**Loc. 2**	**Loc. 3**	**Loc. 4**	**Loc. 5**	**Total**
Projeto	100%	90%	100%	100%	95%	97%
Construção	90%	100%	100%	100%	100%	98%

Os autores Kinnunen, Karpov e Fr (2005) fizeram identificação de locutor em tempo real. Um modelo com 8 locutores e utilizando-se de GMM como classificador, eles obtiveram uma taxa de acerto de 99% em seu melhor resultado com esse modelo.

6.2 CONSTRUÇÃO: IDENTIFICAÇÃO DE UMA EMOÇÃO ESTEREOTIPADA

Para a identificação de uma emoção estereotipada, utiliza-se a melhor rede treinada durante a etapa de projeto (que foi vista na subseção 5.2.4), em que se obteve um acerto de 75% com as RNAs classificando as 6 emoções possíveis e 80,5% utilizando-se de *core affect* para se reclassificar os resultados, sendo que se cada eixo cartesiano do mapa de

emoções for considerado de forma independente se obtém acertos em 85% e 92% das vezes.

Dois diferentes testes foram realizados nessa etapa. O primeiro teste foi: utilizando-se das 50 amostras de áudio capturados para a identificação de um locutor (que foi visto na seção 6.1) e sabendo que todos os 5 locutores estavam calmos, analisou-se quantas vezes as redes acertaram. O segundo teste foi: com um locutor desconhecido pela rede interpretando, ao seu modo, as 6 emoções possíveis: raiva, nojo, medo, felicidade, calma e tristeza, observou-se a quantidade de acertos das redes.

6.2.1 Cinco Locutores Interpretando Uma Emoção

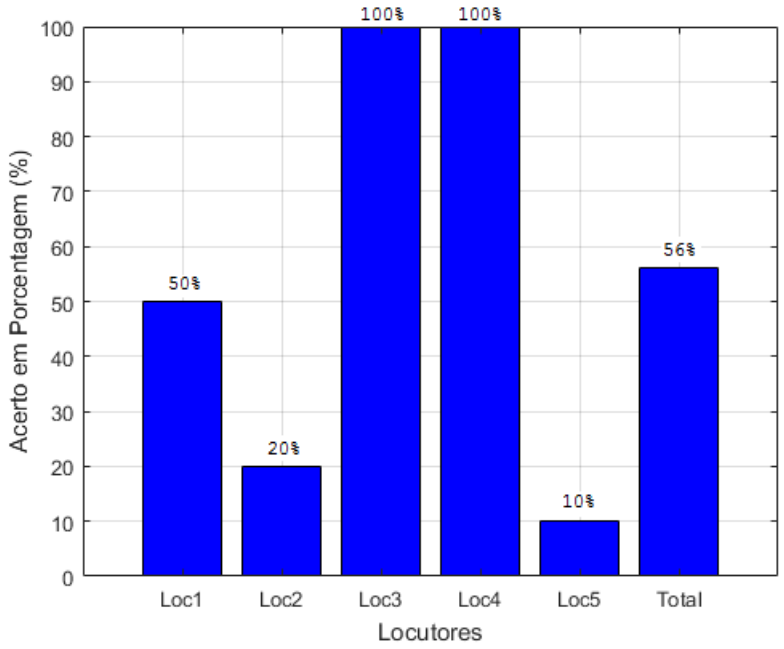
Considerando o caso em que os cinco locutores que falaram na identificação de locutor estavam calmos (pediu-se para que todos os locutores transmitissem calma em suas falas), calculou-se a taxa de acerto que a rede conseguiu identificar a emoção calma. Em 56% dos casos, conforme pode ser visto na Figura 51 e na Figura 52 (a linha representa os valores reais e as colunas os valores preditos), a rede conseguiu identificar a emoção, porém com uma taxa muito alta para alguns locutores e muito baixa para outros, o que mostra que essa taxa é diretamente ligada ao estilo de fala de cada um dos locutores. Vale a pena ressaltar que, como já foi visto anteriormente (Figura 42), calma possui uma taxa de acerto de 92% na rede que foi treinada. Isso significa que esse acerto de 56% está ligado ao tipo de emoção (que possui uma taxa de acerto mais elevada que outras emoções), mas também está ligada ao estilo de fala de cada locutor. Uma emoção com uma taxa de acerto mais baixa resultaria em menos acertos nesse teste. Pode-se perceber, mais uma vez, conforme já foi discutido anteriormente (subseção 5.2.1), que a identificação de emoções está relacionada com os locutores.

O resultado de 56% permanece inalterado quando se reorganiza as respostas utilizando *core affect*, porém, ao se analisar cada eixo independentemente, existe um acerto maior. Em 62% das vezes, houve uma identificação correta que é uma emoção de baixa ativação e em 58% das vezes, identificou-se corretamente que era uma emoção agradável, conforme pode ser visto na Tabela 7.

Tabela 7 – Comparação entre os acertos para identificação de 1 emoção por 5 locutores por diferentes interpretações

-	**Acerto**
6 emoções	56%
Core affect	56%
Ativação	62%
Agradável	58%

Figura 51 – Gráfico com a quantidade de acertos das redes para cada um dos cinco locutores transmitindo a emoção “calma”.



Fonte: Autor

Figura 52 – Acertos e erros das redes para cinco locutores transmitindo “calma” em comparação com as outras emoções.

	Raiva	Nojo	Medo	Alegria	Calma	Tristeza	Total
Calma	0	16	3	3	28	0	50 56%

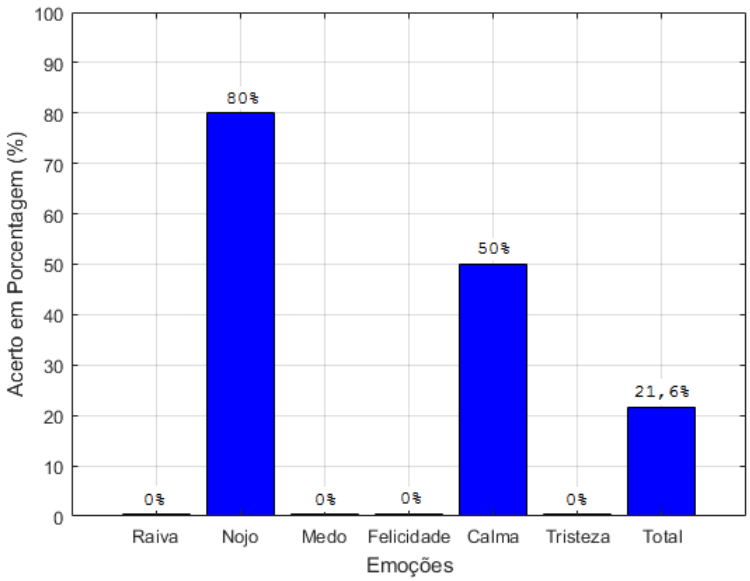
Fonte: Autor

6.2.2 Um Locutor Interpretando Seis Emoções

Considerando o caso em que um único locutor desconhecido pelas redes interpreta, de sua própria maneira, as seis diferentes emoções, calculou-se a taxa de acerto das redes. Em 21,7% dos casos as redes conseguiram identificar de maneira correta qual a emoção estava sendo transmitida, conforme foi visto no gráfico da Figura 53 e na matriz de confusão da Figura 52 (as linhas representam os valores reais e as colunas os valores preditos). Esse resultado não contradiz os outros resultados já encontrado anteriormente, mostrando outra vez como existe uma relação entre o locutor e a emoção na hora que as redes tentam identificar emoções e que o sistema acerta mais vezes para algumas emoções e erra mais

em outras.

Figura 53 – Gráfico com a quantidade de acertos das redes para seis emoções interpretadas por um único locutor.



Fonte: Autor

Quando se muda a interpretação do resultado para *core affect*, há um considerável aumento no acerto, subindo para 36,7% a taxa, conforme pode ser visto de forma comparativa na Tabela 8. Porém, ao se analisar separadamente cada um dos eixos do *core affect*, obtêm-se um acerto de 68,3% na identificação se a emoção é de alta ou baixa ativação e de 55% se a emoção é agradável ou desagradável, sendo esses últimos resultados semelhantes aos obtidos na subseção 6.2.1.

Tabela 8 – Comparação entre os acertos para identificação de 6 emoções por 1 locutor por diferentes interpretações

-	**Acerto**
6 emoções	21,7%
Core affect	36,7%
Baixa ativação	68,3%
Agradável	55%

6.2.3 Resultados para Identificação de uma Emoção Estereotipada

Conforme foi visto anteriormente, ao consideramos 5 locutores calmos, houve um acerto desta emoção em 56% dos casos, tanto para a emoção em si quanto para o *core*

Figura 54 – Matriz de confusão das redes para seis emoções interpretadas por um único locutor.

Matrix de Confusão	Raiva	Nojo	Medo	Alegria	Calma	Tristeza	Total
Raiva	0	8	0	1	1	0	10 0%
Nojo	0	8	0	0	2	0	10 80%
Medo	0	3	0	0	6	1	10 0%
Alegria	0	8	0	0	2	0	10 0%
Calma	0	5	0	0	5	0	10 50%
Tristeza	0	6	0	0	4	0	10 0%
Total	0 0%	38 21%	0 0%	1 0%	20 25%	1 0%	60 21,6%

Fonte: Autor

affect. Ao considerarmos um único locutor interpretando seis emoções, houve um acerto de 21,7%, se elevando para 36,7% quando se usou *core affect*.

Os autores Kim et al. (2017) obtiveram um acerto de 62,9% em sua simulação (menor do que os 75% encontrados pela etapa equivalente desse trabalho), porém eles não divulgaram o acerto em tempo real por causa da baixa precisão do preditor. Também, se compararmos com todos os outros resultados citados anteriormente, na subseção 5.2.4, todos os trabalhos que estão ali, assim todos os outros que foram pesquisados durante o estudo para a elaboração desse trabalho, fazem a verificação do classificador com os mesmos locutores que gravaram as emoções. Por exemplo, se existe um banco de falas com 10 locutores e 500 trechos de áudio, eles utilizam 400 trechos de áudio para treinar os classificadores e 100 para verificar, porém os que serão utilizados para testes foram gravados pelos mesmos locutores que interpretaram as emoções.

Dessa maneira, mostra-se que existe uma relação forte entre o locutor que interpreta a emoção e a identificação de emoções. Para que essa taxa de acerto fosse mais elevada, cada um dos locutores teria que interpretar diferentes emoções, o que não tornaria muito prático numa aplicação real, pois 1 minuto e meio de gravação de voz (para a identificação

de um locutor) se transformaria facilmente em 9 minutos de gravações e interpretações (para 6 emoções), perdendo portanto, um pouco da fluidez de uma aplicação real.

Por outro lado, se o lado humano for analisado, é muito difícil para pessoas desassociarem a identificação de uma pessoa com a emoção na voz, pois cada um possui uma voz distinta e o que é considerado alegria e tristeza por cada um é, de certa maneira, pessoal. Do mesmo modo, ao compararmos línguas diferentes, podem existir maneiras distintas de se entender emoções. Quando um brasileiro (que não fale alemão) ouve a língua alemã sendo falada de maneira calma e tranquila, muitas vezes pode ser interpretada como raiva.

Considerando todos esses pontos e tudo o que foi encontrado na literatura, pode-se dizer que os resultados aqui encontrados estão coerentes com o esperado.

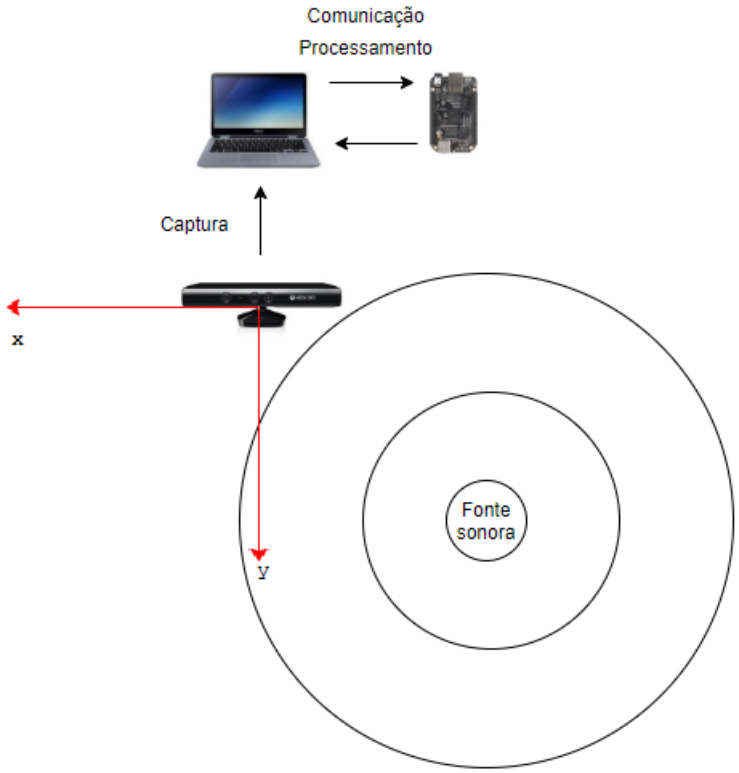
6.3 CONSTRUÇÃO: LOCALIZAÇÃO DE UMA FONTE SONORA

Para a localização de uma fonte sonora, utiliza-se um módulo Kinect (conforme já visto na seção 3.13), que possui 4 microfones acoplados em linha. A captura do áudio é feita através do Windows, pois o Kinect possui suporte para o mesmo através da plataforma Kinect SDK (https://developer.microsoft.com/pt-br/windows/kinect), versão 1.8. Com essa ferramenta, é possível utilizar os microfones do Kinect e fazer gravações com 4 canais de áudio, um para cada microfone. É feita, então, a comunicação entre o computador e o BBB, conforme pode ser visto na Figura 55.

A gravação é feita com uma taxa de amostragem de 16000 Hz por ser a taxa máxima de amostragem do Kinect (PEI et al., 2013). É realizada uma análise de amostra por amostra (são 16000 amostras por segundo de gravação) para identificar quando que existe um evento sonoro. Calcula-se a diferença de amostras até a identificação do evento sonoro para cada um dos microfones e encontra-se a diferença de tempo que o evento sonoro atingiu cada um dos microfones. Como a distância entre microfones é muito pequena e sabendo que a velocidade do som no ar é de 340 m/s, essa diferença de tempo é da ordem de 0,0001 a 0,001 segundos. Como a frequência de amostragem é de 16000 Hz, existe a captura de uma amostra cada 0,0000625 segundos, que é muito próxima da ordem da diferença de tempo citada acima. Portanto, qualquer defasagem no sincronismo dos 4 canais pode provocar erro na estimativa do TDOA. Para os testes realizados nessa etapa, considerou-se o primeiro ruído acima de um limiar mínimo permitido como sendo o evento sonoro (amplitude mínima no tempo). Por isso foi necessário ajustar a sensibilidade desse limiar algumas vezes para se chegar nos resultados que serão apresentados no decorrer dessa seção. Não foram considerados os efeitos de múltiplos caminhos, reverberação e eco durante essa etapa.

Para essa seção, serão analisados os testes e resultados obtidos, assim como uma

Figura 55 – Ilustração do teste real para localização de uma fonte sonora.



Fonte: www.samsungmobilepress.com, www.beagleboard.org e www.windowsteam.com.br; modificado pelo autor

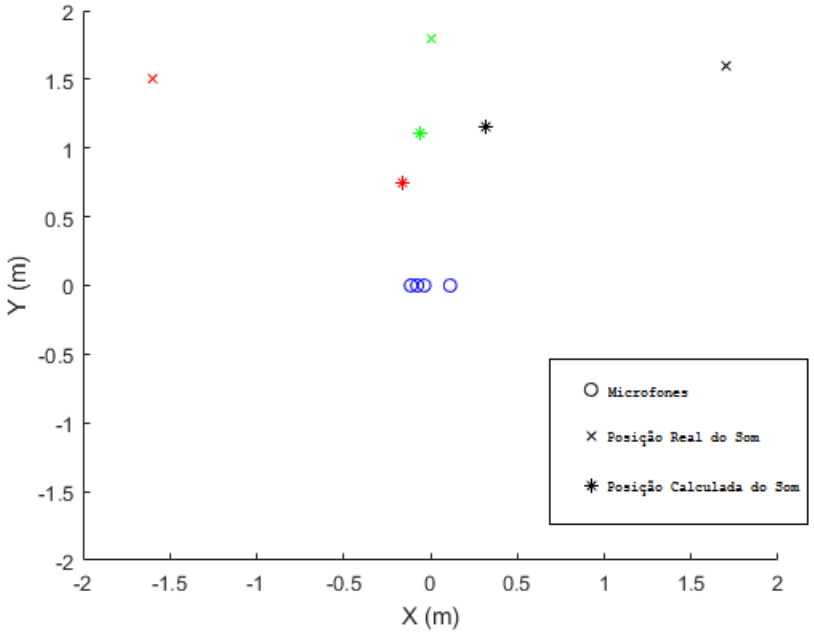
comparação com o código que o próprio MATLAB possui para o uso do Kinect e outros resultados da literatura.

6.3.1 Testes na Localização de uma Fonte Sonora

Três testes foram feitos, onde um locutor falou em posições conhecidas e o sistema calculou sua posição aproximada para cada um dos casos. Para o primeiro teste, a fonte sonora estava a uma distância de 2,2 metros com um ângulo de 136º em relação ao meio do Kinect (entre as duas câmeras frontais). Foi calculado que ele estava a 0,77 metros, com um ângulo de 102º. Isso representa um erro de 1,42 metros (64,5% de erro) e de 34,45º. Para o segundo teste, o locutor se posicionou a uma distância 1,8 metros dos microfones, com um ângulo de 90º em relação a eles. Foi calculado que ele estava a uma distância de 1,1 metros com 92,9º em relação aos microfones. Isso significa um erro de 0,7 metros (38,8% de erro) e de 2,9º. Já no terceiro teste, a origem sonora estava numa distância de 2,3 metros com um ângulo de 43,3º em relação a origem. De acordo com o cálculo realizado pelo sistema foi encontrado que o mesmo estava a 1,2 metros com um ângulo de 74,7º. Um erro de 1,1 metros (47,8% de erro) e 31,4º. O erro médio de 1,08 metros (48,5%)

e 22,9º. Tais resultados podem ser visualizados na Figura 56, onde vermelho representa o primeiro teste, verde representa o segundo teste e preto o terceiro.

Figura 56 – Resultado da localização de 3 fontes sonoras.



Fonte: Autor

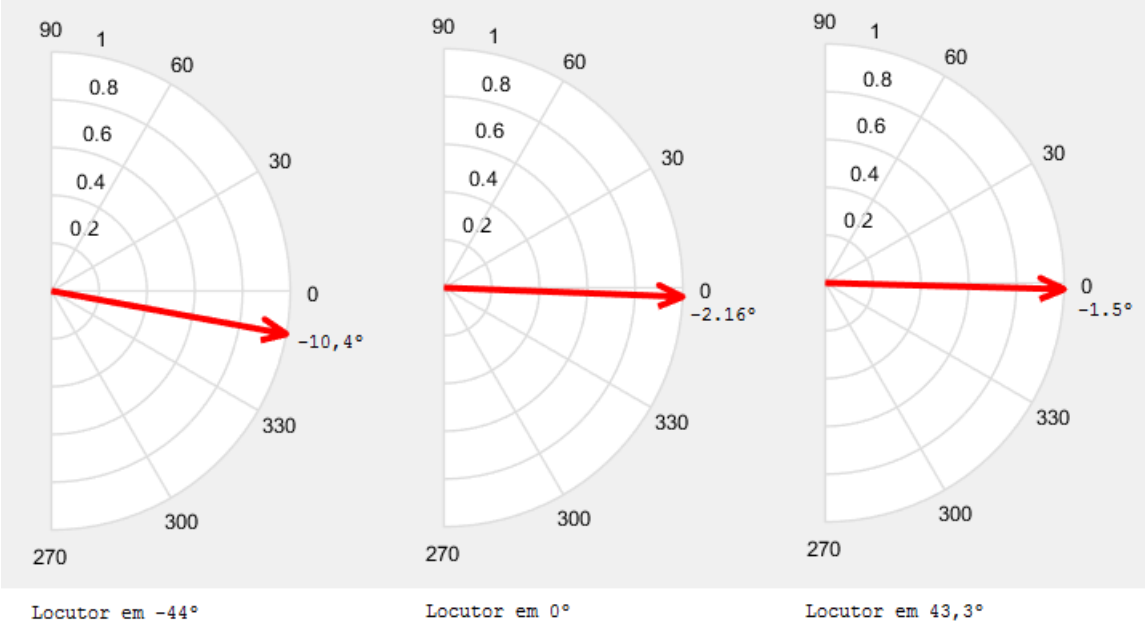
É importante ressaltar que, para atingir esse resultado, foi necessário ajustar a sensibilidade para cada um dos casos para aumentar a acurácia. Isso se dá ao fato de que os microfones do Kinect não são extremamente adequados para tal aplicação, sendo necessários microfones mais sensíveis e dispositivo de captura ou processamento de áudio com um número de entradas e saídas desejadas, por exemplo uma mesa de som. Porém, o custo do Kinect está mais próximo da realidade do robô de baixo custo.

6.3.2 Utilizando o Código do MATLAB para o Kinect

O MATLAB traz em suas versões mais novas, um código pronto chamado *Direction Of Arrival Estimation with a Linear Microphone Array*, onde o mesmo utiliza o Kinect e de uma técnica chamada *Direction Of Arrival* (DOA) para calcular a direção da fonte sonora. DOA considera o ângulo que uma onda atinge os sensores (KRISHNAVENI; KESAVAMURTHY; APARNA, 2013). Para o resultado, utilizou-se a média das direções obtidas no tempo do áudio. Para esse teste não se utilizou o BBB.

Em cada um dos três testes feitos anteriormente, o áudio (com 4 canais) foi gravado para que outros testes pudessem ser replicados nas mesmas condições. Portanto, utilizou-se aqui os mesmos áudios usados anteriormente na subseção 6.3.1. Pode-se ver o ângulo médio do código fornecido pelo MATLAB na Figura 57.

Figura 57 – Resultado do código *Direction Of Arrival Estimation with a Linear Microphone Array* fornecido pelo MATLAB para Kinect.



Fonte: Autor

O código do MATLAB errou o primeiro teste, na média, por 33,6º. O segundo teste teve um erro médio de 2,16º. O terceiro teste teve um erro de 44,8º. O erro médio foi por esse código foi de 26,8º, sendo portanto próximo do erro encontrado utilizando multilateração (o erro médio foi de 22,9º conforme foi visto na subseção 6.3.1). Porém, além dessa técnica não calcular a distância aproximada, ainda houve o erro da direção.

6.3.3 Resultado para a Localização de uma Fonte Sonora

Para o cálculo da localização de uma fonte sonora, conseguiu-se identificar corretamente se o locutor estava na frente dos microfones, a esquerda ou a direita. O erro médio aproximado foi de 1,08 metros (48,5%) e 22,9º. O resultado de 22,9º é compatível com o calculado pelo código pronto do MATLAB para o Kinect, que teve um erro médio de 26,8º. Os autores Pei et al. (2013) encontraram um erro aproximado médio de 0,8 metros (40% de erro) e um erro máximo de 2 metros quando fizeram seu experimento usando um método de localização chamada de *Dilution Of Precision* (DOP), portando compatível com o encontrado nesse trabalho.

Pode-se apontar alguns pontos que contribuíram para o erro apresentado durante a localização da fonte sonora. Além dos problemas já mencionados como distância dos microfones, ordem de grandeza do TDOA, possível defasagem entre canais e não consideração de eco, outro problema enfrentado é que foi consideram que as ondas sonoras são de campo distante, quando na verdade, para as distâncias utilizadas, ainda são ondas de

campo próximo. Mais um problema é o fato de que os microfones estão posicionados em linha (diminui a precisão do MLAT, conforme já foi visto na subseção 4.2.4).

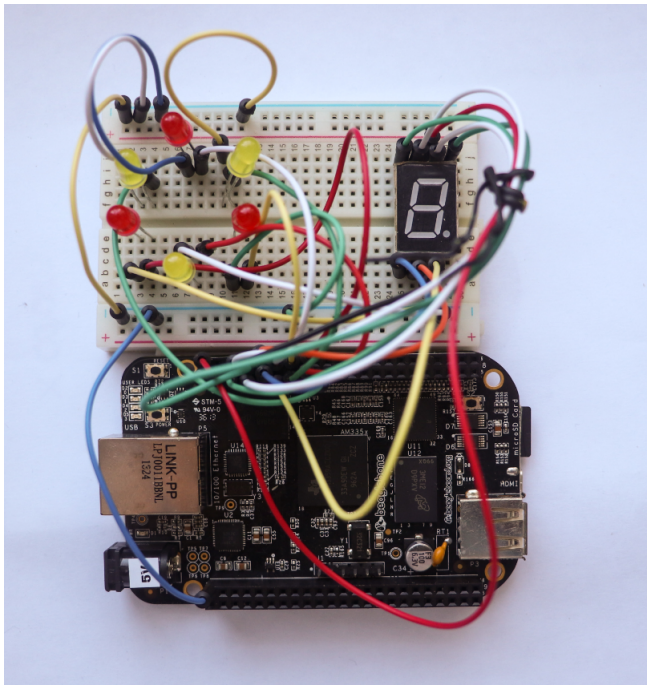
Porém, mesmo com os problemas citados, a precisão da localização da fonte sonora do módulo auditivo artificial é condizente com o desejado na aplicação prática, principalmente sabendo-se do fato que este módulo não é o único a realizar localização no robô assistivo, sendo, portanto, um resultado suficiente.

6.4 CONSTRUÇÃO DO MÓDULO AUDITIVO ARTIFICIAL

Nessa última etapa do trabalho, reúne-se todo o conhecimento adquirido previamente e os integram de maneira única ao modulo. O projeto das etapas levou a construção de cada etapa separadamente e, por fim, a construção final, em que todas as tarefas estão integradas.

Conforme visto anteriormente na Figura 55, o módulo Kinect será utilizado como forma de captura de áudio. Esse áudio passará pela extração de características, identificação do locutor e de emoção. Com a diferença de tempo de quando o evento sonoro atingiu cada um dos quatro microfones, calcula-se a posição do locutor.

Figura 58 – BeagleBone Black integrado a uma protoboard para dar resultados de forma mais dinâmica.



Fonte: Autor

Para que se pudesse fazer um teste com pelo menos dois locutores diferentes, o autor do trabalho gravou dois dos cinco locutores, sendo um com sua voz normal e outro

com uma voz extremamente aguda, para se diferenciar de sua voz normal.

Para aumentar a interatividade do módulo, além dos resultados aparecerem no MATLAB, uma pequena integração com uma protoboard foi feita, de maneira que a placa diz qual o número do locutor que está falando e a direção de sua voz, conforme pode ser visto na Figura 58.

Para esse teste, o locutor se posicionou nos mesmos lugares marcados previamente para os testes da localização de um locutor vistos anteriormente na seção 6.3. A sua voz foi gravada por um determinado tempo para que pudesse fazer 3 testes de quem é o locutor e sua emoção, para que, assim, se pudesse aumentar a confiança do resultado. Utilizou-se a rede com 5 locutores da construção do módulo, vista anteriormente na seção 6.1, com seus 98% de acerto para a identificação de um locutor e as mesmas redes vistas previamente na seção 6.2, com sua taxa de acerto dentro do esperado para a identificação de uma emoção estereotipada.

6.4.1 Testes e Resultado para o Módulo Auditivo Artificial

Conforme foi visto na proposta, existe um fluxograma do módulo (Figura 26 do Capítulo 4). Após a captura do áudio, geraram-se os coeficientes cepstrais e TDOA para identificar o locutor, sua emoção e sua localização, respectivamente.

6.4.1.1 Primeiro Teste

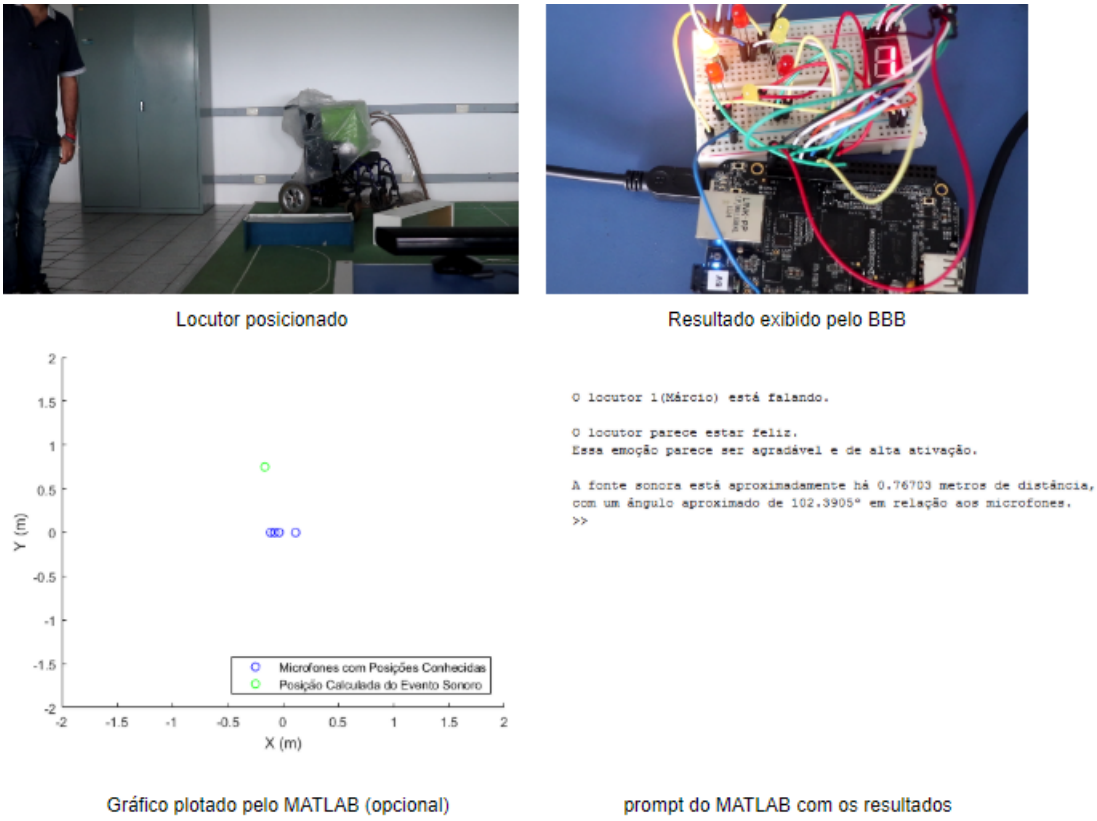
Primeiramente o locutor se posiciona e fala uma curta frase como seu nome e sua idade, por exemplo. Após a captura do áudio, ele é cortado em 3 trechos para que hajam as identificações de locutor e emoção de uma forma mais precisa.

Os resultados do primeiro teste podem ser vistos na Figura 59. As redes identificaram corretamente quem era o locutor, sendo esse resultado apresentado na protoboard e no *prompt* do MATLAB. As redes identificaram que o locutor estava feliz, quando na verdade ele estava calmo. Se consideramos o *core affect*, a resposta deveria estar no quarto quadrante, quando foi colocada no primeiro quadrante. Porém, as redes acertam que é uma emoção agradável e erram na ativação. É uma emoção de baixa ativação ao invés de alta. Na localização da fonte sonora, há um erro de 1,42 metros (64,5% de erro) e de 34,45º, mas a localização a esquerda dos microfones é correta tanto no gráfico como na protoboard (o resultado é único e não poderia ser diferente entre o MATLAB e a protoboard).

6.4.1.2 Segundo Teste

Para o segundo teste, o locutor se posiciona na frente do Kinect e fala seu nome e idade com uma voz mais aguda para se diferenciar de sua voz normal e ser um locutor

Figura 59 – Resultados do primeiro teste do módulo auditivo artificial.



Fonte: Autor

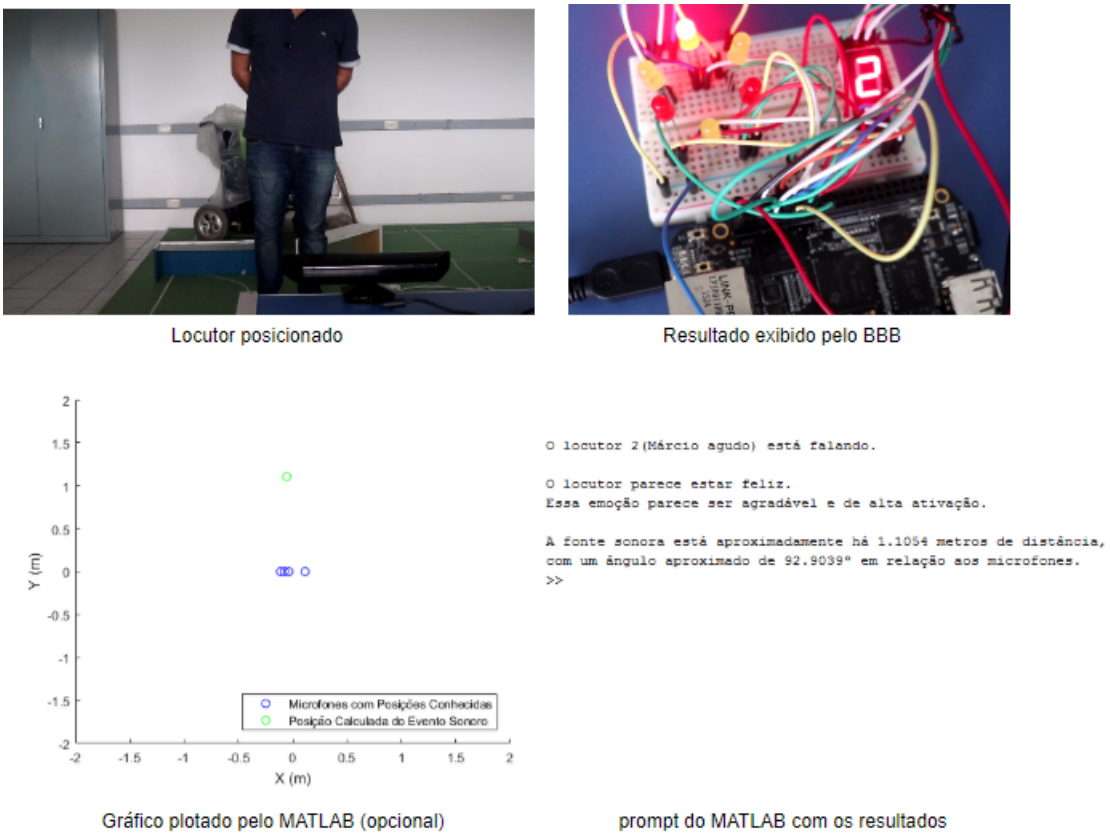
diferente para as redes. Como no teste anterior, o áudio após capturado é cortado em 3 trechos.

Na Figura 60 estão os resultados do segundo teste. Mais uma vez, a identificação do locutor foi precisa, identificando que era o locutor 2 (com uma voz mais aguda) que estava falando, sendo esse resultado visível tanto no *prompt* do MATLAB quanto na protoboard. Para a identificação da emoção estereotipada, aconteceu a mesma coisa do primeiro teste. Era uma emoção calma e a rede achou que era feliz. Igualmente como no primeiro caso, houve uma acerto por ser uma emoção positiva e um erro por ser de baixa ativação e não de alta. Já na localização da origem sonora, houve um erro de 0,7 metros (38,8% de erro) e de 2,9º. Houve um acerto tanto no gráfico quanto na placa da posição da fonte sonora, sendo localizada em frente ao Kinect.

6.4.1.3 Terceiro Teste

No terceiro teste, o locutor volta a falar com sua voz normal e se posiciona a direita dos microfones. O mesmo tipo de tratamento e captura é feito, seguindo a sequência dos testes anteriores.

Figura 60 – Resultados do segundo teste do módulo auditivo artificial.



Fonte: Autor

Conforme pode ser visto na Figura 61, onde estão os resultados do terceiro teste, a identificação de um locutor acerta mais uma vez e o locutor 1 está falando novamente, sendo que esse resultado é visto tanto na placa quanto no MATLAB. Pela terceira vez, a emoção identificada é felicidade, quando deveria ser calma, fazendo que, outra vez, haja um acerto que é uma emoção agradável e um erro no tipo de ativação. Na localização da fonte sonora, houve um erro de 1,1 metros (47,8% de erro) e 31,4°. Porém, novamente houve um acerto na identificação que a fonte sonora estava a direita dos microfones.

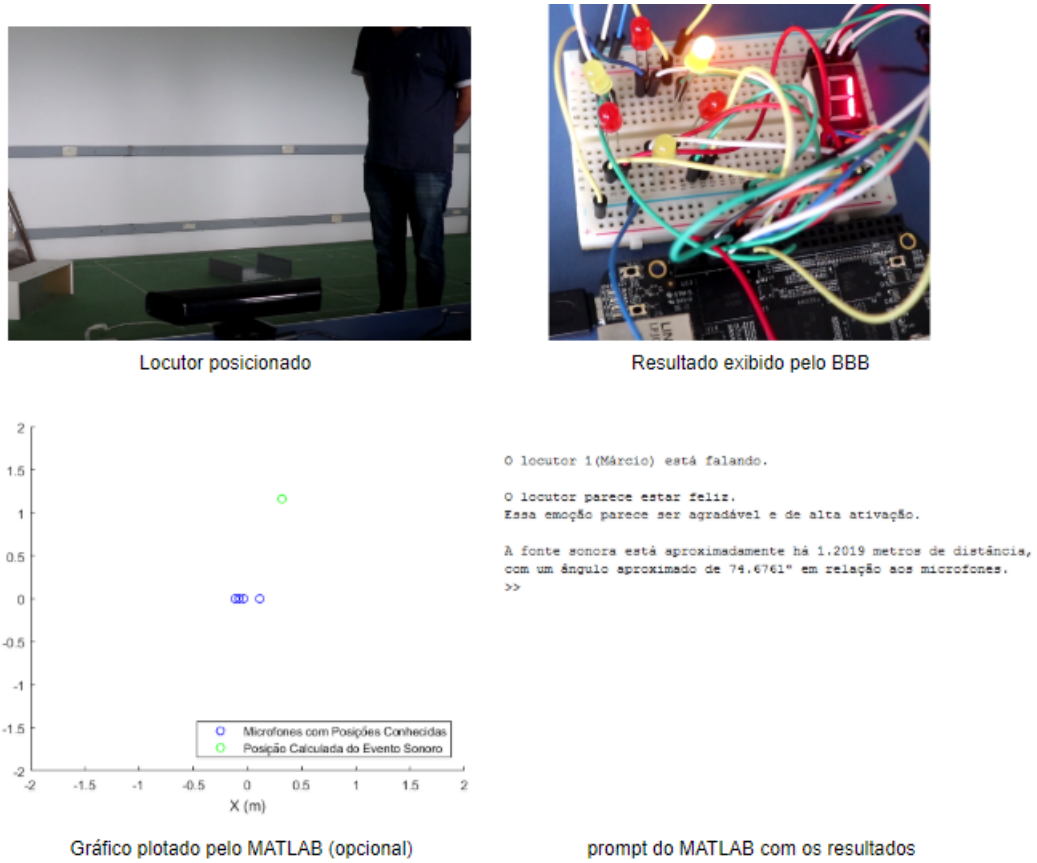
6.4.1.4 Tempos de execução

O tempo de execução é importante para aplicações reais que saem do mundo das simulações, pois uma resposta que demore 1 minuto para ser processada é fora dos padrões esperados para um sistema dessa natureza.

A conexão inicial entre o MATLAB e o BeagleBone Black pode demorar entre 10 e 50 segundos. Após a conexão inicial ser estabelecida, rodam-se as rotinas.

Após a captura do áudio, as rotinas demoraram em torno de 4 segundos (valor típico) para serem executadas, gerarem gráficos e saídas no BBB, no *prompt* e gráfico. Desse tempo,

Figura 61 – Resultados do terceiro teste do módulo auditivo artificial.



Fonte: Autor

aproximadamente 2 segundos são gastos com comunicação entre o BBB e o MATLAB. A execução da identificação de um locutor e de emoções e feita em aproximadamente 1 segundo. O cálculo do MLAT é feito em aproximadamente 0,8 segundos. O resto do tempo é de execução de outras tarefas. Na Figura 62 podem-se ver gráficos com os tempos de execução de cada uma das tarefas descritas anteriormente.

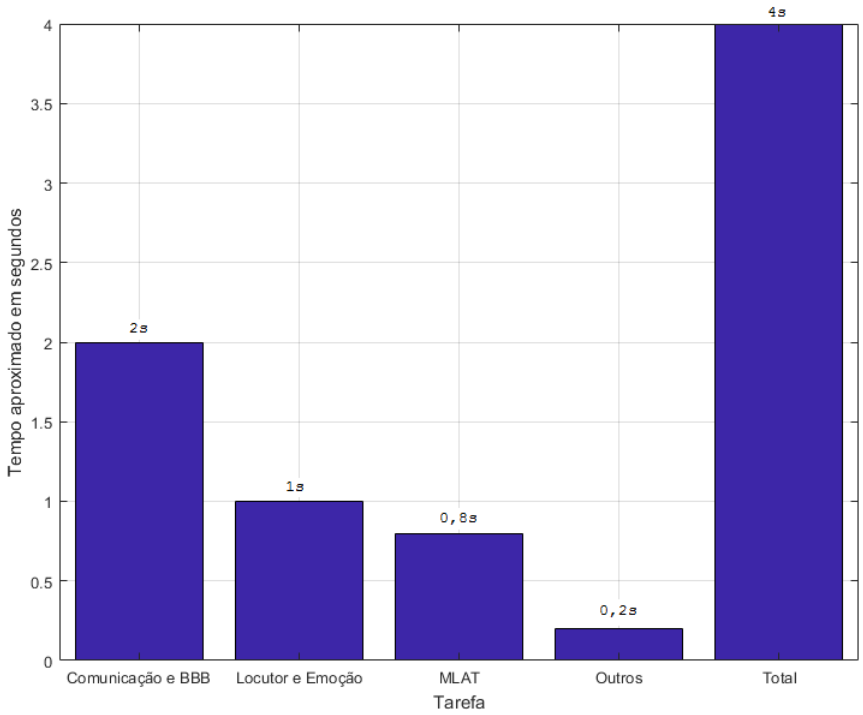
Num futuro trabalho, para diminuir o tempo de execução, precisa-se embarcar completamente o sistema, fazendo com que ele não precise mais do intermédio do computador, pois uma comunicação direta com o periférico seria mais veloz.

6.4.1.5 Resultados da Construção do Módulo Auditivo Artificial

O módulo auditivo artificial proposto foi elaborado e obtiveram-se resultados que foram julgados ser compatíveis com o esperado à partir do que foi visto na literatura previamente.

A etapa de identificação do locutor mostrou-se bastante eficaz em suas tarefas, mesmo considerando que o ambiente do laboratório possuía ruídos externos e eco. É um ótimo resultado e pode ser aplicado ao robô assistivo. A identificação de emoções

Figura 62 – Tempo típico de execução do módulo auditivo artificial após a captura do áudio.



Fonte: Autor

estereotipadas obteve um resultado mais baixo do que o da etapa de projeto, o que já era esperado por se saber da relação entre locutor e emoção que é feita por classificadores ao tentar identificar a emoção, porém os resultados obtidos estão dentre os parâmetros esperados e são adequados para o robô assistivo. A localização da fonte sonora obteve erros esperados. Com melhores equipamentos, esses erros poderiam ser reduzidos. Porém, os resultados obtidos foram adequados para as aplicações desejadas.

Considerando-se que foi construído um módulo auditivo artificial para um robô assistivo de baixo custo, os resultados obtidos na elaboração do mesmo foram coerentes e condizentes. Dessa maneira, conhecendo suas limitações, esse módulo construído é viável para um robô assistivo.

7 CONCLUSÃO E TRABALHOS FUTUROS

Esse é o capítulo final desse trabalho, apresentando suas conclusões à partir do desenvolvimento e resultados obtidos e expondo futuras perspectivas para a continuação da pesquisa.

7.1 CONCLUSÃO

As tarefas de identificação de locutores, identificação de emoções estereotipadas e localização de fontes sonoras vem avançando já há muitos anos. O estudo do áudio está sempre presente e suas aplicações são diversas, desde interações humanas, como simular uma conversa, até o de automatização de serviços (como teleatendimento, por exemplo). Fazer com que robôs consigam realizar tarefas humanas é extremamente complexo e ainda continuamos longe da perfeição. Porém, com o passar dos anos, cada vez mais nos aproximamos dessa realidade. A "audição" robótica é importante para que essa área continue sempre a avançar.

Nesse trabalho, se projetou e construiu um módulo auditivo artificial, que num futuro, possa ser integrado e fazer parte de um robô social assistivo.

Os principais objetivos do trabalho foram alcançados, pois estudou-se, elaborou-se projetou-se sistemas compatíveis com a literatura para os três tópicos propostos e construiu-se um módulo auditivo artificial para um robô social de baixo custo que realiza as tarefas de: (i) identificação de locutor, (ii) identificação de emoção estereotipada e (iii) localização de uma fonte sonora. O estudo aprofundado de cada uma dos tópicos fez com que se chegasse aos atuais resultados dentre vários testados e simulados.

Para esse trabalho, elaborou-se um banco de falas em português com 5 locutores distintos e com 500 palavras, 100 de cada locutor. Esse banco de falas construído está disponível no link: <https://bit.ly/2LgfoZu>, de maneira que ele pode ser utilizados para outros trabalhos futuros.

Para a identificação do locutor, resultados bem favoráveis foram encontrados, sendo esse o mais preciso de todas as tarefas realizadas por esse módulo, com respostas condizentes com tudo que foi estudado. Na fase de projeto, um acerto de 97% foi encontrado. Na fase de construção do módulo, um acerto de 98% foi obtido.

A identificação da emoção estereotipada tem uma boa taxa de acerto na etapa de projeto, porém que cai drasticamente em uma situação real. Para resultados melhores teria que haver um estudo mais profundo, considerando-se a relação entre locutor e emoção.

Para a fase de projeto, houve um certo de 75% para a identificação da emoção e de 80,5% para a identificação utilizando *core affect*, ambos utilizando o banco SAVEE. Para a fase de construção do módulo, o acerto da emoção "calma" por 5 diferentes locutores foi de 56%. Já o acerto das 6 emoções interpretadas por um locutor foi de 21,6% e de 37,7% com o *core affect*.

Na localização da fonte sonora, a simulação obteve 100% de acerto, resultado que foi longe de ser real nos testes realizados, nos quais houve um erro médio de 1,1 metros (47,8%) e 31,4º. Porém, os resultados obtidos na etapa de construção são adequados para o uso do Kinect. Esses resultados também se mostraram condizentes com o que foi encontrado na literatura.

Há um desejo de que esse projeto seja continuado e que ele continue a ser desenvolvido até, finalmente ser integrado a um robô assistivo operacional. Para tal, todas as rotinas da fase de projeto foram disponibilizadas no link: <https://github.com/marciooluizz/Modulo-Auditivo-Artificial>; assim, outras pesquisas envolvendo MFCC, áudio, redes neurais poderiam se beneficiar do que já foi desenvolvido até o momento.

Por fim, esse trabalho é único no sentido de que produziu todas suas rotinas utilizadas e no sentido que uniu três tópicos distintos, cada um exigindo técnicas e análises diferentes, em um único trabalho, em um único módulo de processamento.

7.2 TRABALHOS FUTUROS

Para futuros trabalhos, propõe-se um estudo mais aprofundado das relações entre emoção e locutor e como superar esse problema. Três soluções são propostas. Produzir um banco de emoções em português e tentar aumentar a robustez do sistema, trocar a técnica de classificação ou utilizar locutores tentando atuar suas próprias emoções no processo de treinamento das redes para identificação de emoção.

Outra proposta para trabalhos futuros é a utilização de microfones mais precisos, com uma melhor resposta na frequência, para a localização de uma fonte sonora. Ou talvez, a criação de uma placa específica para a captura dos áudios, com alta precisão e com uma geometria favorável para se localizar a origem sonora.

Há também a necessidade de fazer o acoplamento do BeagleBone Black (ou outra placa) com os microfones diretamente para que o mesmo não dependesse mais de um computador com Windows e MATLAB para realizar as tarefas do módulo e, dessa maneira, se torne de baixo custo para a sua aplicação no robô. Sugere-se apenas a utilização de softwares "*open source*" e programação em ROS e/ou Python.

Por fim, é necessário fazer a integração desse módulo com outros módulos para a construção final de um robô assistivo de baixo custo para fins socais.

REFERÊNCIAS

BANSAL, P.; BHARTI, R. Speaker Recognition using MFCC , shifted MFCC with Vector Quantization and Fuzzy. *International Conference on Soft Computing Techniques and Implementations (ICSCTI)*, p. 41–44, 2015. Citado 4 vezes nas páginas 30, 43, 62 e 66.

BURKHARDT, F. et al. A Database of German Emotional Speech. In: *Proceedings Interspeech*. [S.l.: s.n.], 2005. Citado 2 vezes nas páginas 22 e 45.

BURRUS, C. S. *Fast Fourier Transforms*. [S.l.: s.n.], 2012. 244 p. Citado na página 106.

BUSSO, C. et al. *IEMOCAP : Interactive emotional dyadic motion capture database*. Los Angeles,, 2008. 1–30 p. Citado na página 5.

CAMPBELL JR, J. P. Speaker Recognition : A Tutorial. *Proceedings of the IEEE*, v. 85, n. 9, p. 1437–1462, 1997. Citado 2 vezes nas páginas 11 e 12.

CHAKRABORTY, K. Voice Recognition Using MFCC Algorithm. *International Journal of Innovative Research in Advanced Engineering (IJIRAE)*, v. 1, n. 10, p. 158–161, 2014. Citado 3 vezes nas páginas 14, 15 e 42.

CHAN, Y. T.; HO, K. C. A Simple and Efficient Estimator for Hyperbolic Location. *IEEE Transactions on Signal Processing*, v. 42, n. 8, p. 1905–1915, 1994. Citado na página 34.

CHAUHAN, N.; CHANDRA, M. Speaker Recognition and Verification Using Artificial Neural Network. *IEEE WiSPNET*, p. 1147–1149, 2017. Citado na página 66.

CORTELLESSA, G. et al. A Cross-Cultural Evaluation of Domestic Assistive Robots Evaluating Social Assistive Robots. *Association for the Advancement of Artificial Intelligence*, p. 1–8, 2008. Citado na página 7.

DAHAKE, P. P.; SHAW, K. Speaker Dependent Speech Emotion Recognition using MFCC and Support Vector Machine. *International Conference on Automatic Control and Dynamic Optimization Techniques*, p. 1080–1084, 2016. Citado 7 vezes nas páginas 1, 2, 5, 30, 31, 43 e 75.

DEMIRCAN, S.; KAHRAMANLı, H. Feature Extraction from Speech Data for Emotion Recognition. *Journal of Advances in Computer Networks*, v. 2, n. 1, p. 28–30, 2014. Citado 2 vezes nas páginas 31 e 75.

DENG, L.; LI, X. Machine Learning Paradigms for Speech Recognition : An Overview. *IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING*, v. 21, n. 5, p. 1060–1089, 2013. Citado 2 vezes nas páginas 23 e 24.

DREISEITL, S.; OHNO-MACHADO, L. Logistic Regression And Artificial Neural Network Classification Models : A Methodology Review. *Journal of Biomedical Informatics*, v. 35, n. 2002, p. 352–359, 2003. Citado na página 29.

GONCALVES, S. C. L. *Banco de dados Iboruna: amostras eletrônicas do português falado no interior paulista*. 2003. Citado na página 22.

HAQ, S.; JACKSON, P. *Surrey Audio-Visual Expressed Emotion (SAVEE) Database*. 2011. Disponível em: <http://kahlan.eps.surrey.ac.uk/savee/Download.html>. Citado 2 vezes nas páginas 22 e 45.

HAQ, S.; JACKSON, P. J. B.; EDGE, J. Audio-visual feature selection and reduction for emotion classification. *Centre for Vision, Speech and Signal Processing (CVSSP), University of Surrey, Guildford, UK*, 2008. Citado na página 22.

HASAN, R. et al. Speaker Identification Using Mel Frequency Cepstral Coefficients. In: *3rd International Conference on Electrical & Computer Engineering (ICECE)*. [S.l.: s.n.], 2004. p. 28–30. Citado 6 vezes nas páginas 1, 2, 14, 15, 17 e 42.

HE, N.; HUANG, H.-W.; WOLTMAN, B. D. The Use of BeagleBone Black Board in Engineering Design and Development. In: *ASEE North Midwest Section Conference*. [S.l.: s.n.], 2014. p. 1–8. Citado na página 37.

HEATH, S. *Embedded Systems Design*. 2. ed. [S.l.]: Newnes, 2002. 430 p. ISBN 0750655461. Citado na página 37.

HUANG, J.; OHNISHI, N.; SUGIE, N. Building Ears for Robots: Sound Localization and Separation. *Artif Life Robotics*, p. 157–163, 1997. Citado na página 5.

JANG, J. R. ANFIS: Adaptive Network-Based Fuzzy Inference System. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS*, v. 23, n. 3, p. 665–685, 1993. Citado na página 30.

KAELBLING, L. P. *Learning in Embedded Systems*. 2. ed. [S.l.]: MIT Press, 2008. 193 p. ISBN 9780262512787. Citado na página 37.

KIM, J. C. et al. Audio-based Emotion Estimation for Interactive Robotic Therapy for Children with Autism Spectrum Disorder. *14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, p. 39–44, 2017. Citado 3 vezes nas páginas 5, 68 e 85.

KINNUNEN, T.; KARPOV, E.; FR, P. Real-Time Speaker Identification and Verification. *IEEE Transactions on Audio, Speech, and Language Processing*, v. 14, n. 1, p. 277 – 288, 2005. Citado 2 vezes nas páginas 5 e 81.

KINNUNEN, T.; LI, H. An overview of text-independent speaker recognition: From features to supervectors. *Speech Communication*, Elsevier B.V., v. 52, n. 1, p. 12–40, 2010. ISSN 0167-6393. Citado 2 vezes nas páginas 8 e 9.

KRISHNAVENI, V.; KESAVAMURTHY, T.; APARNA, B. Beamforming for Direction-of-Arrival (DOA) Estimation-A Survey. *International Journal of Computer Applications*, v. 61, n. 11, p. 4–11, 2013. Citado na página 88.

LALITHA, S. et al. Emotion Detection using MFCC and Cepstrum Features. *Procedia Computer Science*, Elsevier Masson SAS, v. 70, p. 29–35, 2015. Citado 4 vezes nas páginas 17, 22, 31 e 75.

LUGGER, M.; YANG, B. CASCADED EMOTION CLASSIFICATION VIA PSYCHOLOGICAL EMOTION. *IEEE International Conference on Acoustics, Speech and Signal Processing*, p. 3–6, 2008. Citado 5 vezes nas páginas 22, 32, 49, 74 e 75.

MAESA, A. et al. Text Independent Automatic Speaker Recognition System Using Mel-Frequency Cepstrum Coefficient and Gaussian Mixture Models. *Journal of Information Security*, v. 2012, n. October, p. 335–340, 2012. Citado 3 vezes nas páginas 5, 43 e 66.

MAHENDRU, H. C. Quick Review of Human Speech Production Mechanism. *International Journal of Engineering Research and Development*, v. 9, n. 10, p. 48–54, 2014. Citado na página 8.

MARTINEZ, J.; PEREZ, H.; ESCAMILLA, E. Speaker recognition using Mel Frequency Cepstral Coefficients (MFCC) and Vector Quantization (VQ) Techniques. *22nd International Conference on Electrical Communications and Computers (CONIELECOMP)*, p. 248–251, 2012. Citado 6 vezes nas páginas 14, 30, 42, 43, 62 e 66.

MICHALSKI, R. S.; CARBONELL, J. G.; MITCHELL, T. *Machine Learning: An Artificial Intelligence Approach*. 1. ed. [S.l.: s.n.], 1983. 3–38 p. Citado na página 23.

MOURABIT, I.; BADRI, A. Modeling Approach of LTE Mobile Positioning Simulator for OTDOA Measurements. *International Journal of Advanced Research in Computer Science and Software Engineering*, v. 5, n. 11, p. 32–36, 2015. Citado na página 35.

MUDA, L.; BEGAM, M.; ELAMVAZUTHI, I. Voice Recognition Algorithms using Mel Frequency Cepstral Coefficient (MFCC) and Dynamic Time Warping (DTW) Techniques. *Journal of Computing*, v. 2, n. 3, p. 138–143, 2010. Citado 3 vezes nas páginas 18, 19 e 31.

NILSSON, N. J. *Principles of Artificial Intelligence*. [S.l.: s.n.], 1980. 1–16 p. Citado na página 22.

NILSSON, N. J. *INTRODUCTION TO MACHINE LEARNING*. [S.l.: s.n.], 2005. 1–14 p. Citado 2 vezes nas páginas 23 e 26.

NORRDINE, A. An Algebraic Solution to the Multilateration Problem. *2012 International Conference on Indoor Positioning and Indoor Navigation*, p. 1–4, 2015. Citado 4 vezes nas páginas 1, 2, 34 e 36.

PEI, L. et al. Sound Positioning Using a Small-scale Linear Microphone Array. *IEEE International Conference on Indoor Positioning and Indoor Navigation*, p. 1–7, 2013. Citado 5 vezes nas páginas 6, 39, 52, 86 e 89.

PEREIRA, F.; MITCHELL, T.; BOTVINICK, M. Machine learning classifiers and fMRI : A tutorial overview. *NeuroImage*, Elsevier Inc., v. 45, n. 1, p. S199–S209, 2009. ISSN 1053-8119. Citado na página 25.

PIERCE, J. A. AN INTRODUCTION TO HYPERBOLIC NAVIGATION , WITH PARTICULAR REFERENCE TO LORAN. *Convention of the INSTITUTE OF RADIO ENGINE*, p. 243–245, 1946. Citado na página 33.

RASO, T.; MELLO, H. *Spoken Corpora and Linguistic Studies*. [S.l.]: John Benjamins Publishing Company, 2014. 1–24 p. ISBN 978 90 272 03694. Citado na página 21.

RIEDMILLER, M. U. o. K. *Rprop - Description and Implementation Details*. Karlsruhe, 1994. 1–2 p. Citado 2 vezes nas páginas 29 e 108.

RUSSELL, J. A. Core Affect and the Psychological Construction of Emotion. *Psychological Review*, v. 110, n. 1, p. 145–172, 2003. Citado 2 vezes nas páginas 31 e 32.

SILVA, M. F. D. *APLICAÇÃO DO MÉTODO DE FUSÃO PARA VERIFICAÇÃO DE LOCUTOR INDEPENDENTE DE TEXTO*. 105 p. Tese (Doutorado) — PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL, 2015. Citado 2 vezes nas páginas 9 e 13.

SóRIA, R. A. B.; JR, E. F. C. Speaker recognition with artificial neural networks and MEL-frequency cepstral coefficients correlations. *8th European Signal Processing Conference (EUSIPCO 1996)*, p. 1–4, 1996. Citado na página 43.

SVOZIL, D.; KVASNIEKA, V.; POSPICHAL, J. Introduction to multi-layer feed-forward neural networks. *Chemometrics and Intelligent Laboratory Systems*, v. 39, p. 43–62, 1997. Citado 3 vezes nas páginas 26, 27 e 28.

TAKAYAMA, L.; DOOLEY, D.; JU., W. Expressing thought: improving robot readability with animation principles. *6th ACM/IEEE International Conference on Human-Robot Interaction*, p. 69–76, 2011. Citado na página 1.

TOGNERI, R.; PULLELLA, D. An Overview of Speaker Identification: Accuracy and Robustness Issues. *IEEE Circuits and Systems Magazine*, v. 11, n. 2, p. 23 – 61, 2011. Citado 7 vezes nas páginas 14, 15, 16, 17, 19, 20 e 21.

TRUAX, B.; SCHAFER, R. M. *Handbook For Acoustic Ecology*. [S.l.: s.n.], 1999. Citado na página 9.

VALIN, J.-M. *Auditory System for a Mobile Robot*. 92 p. Tese (Doutorado) — University of Sherbrooke, 2005. Citado na página 2.

WAHYUNI, E. S. Arabic Speech Recognition Using MFCC Feature Extraction and ANN Classification. *2nd International Conferences on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, p. 22–25, 2017. Citado 2 vezes nas páginas 16 e 18.

WATILE, A.; ALAGDEVE, V.; JAIN, S. Emotion Recognition in Speech by MFCC and SVM. *International Journal of Science, Engineering and Technology Research*, v. 6, n. 3, p. 404–407, 2017. Citado na página 30.

WEBB, J.; ASHLEY, J. *Beginning Kinect Programming with the Microsoft Kinect SDK*. [S.l.]: Apress, 2012. 1–22 p. Citado 2 vezes nas páginas 6 e 50.

XAVIER, F. P.; BARBACENA, I. L. *BeagleBone Black Material de Apoio*. [S.l.]: Instituto Federal de Educação, Ciência e Tecnologia da Paraíba - IFPB, 2015. 1–48 p. Citado 2 vezes nas páginas 37 e 38.

YU, D.; HU, Q.; BAO, W. Combining Multiple Neural Networks for Classification Based on Rough Set Reduction. *IEEE Int. Conf. Neural Networks d Signal Processing*, p. 543–548, 2003. Citado na página 29.

Apêndices

APÊNDICE A – PALAVRAS GRAVADAS PARA O BANCO DE DADOS DA SIMULAÇÃO DE IDENTIFICAÇÃO DE UM LOCUTOR

As seguintes palavras foram gravadas para gerar o banco de vozes na identificação de locutores, conforme mencionado na subseção 4.2.2:

Olho, maçã, fantasma, guitarra, assobio, torneira, bigode, coruja, giz, câmera, televisor, cavalo, colher, café, amarelo, barco, sol, arroz, grande, carne, abelha, perna, leite, pato, gelo, sete, dinheiro, vaca, crocodilo, banheira, lavatório, ovo, espaguete, peruca, boca, meias, arco-íris, tambor, baixo, óculos, estrela, tulipa, lobo, sujo, verde, pêra, caramelo, sanduíche, pimenta, selo, barriga, transporte, rosa, tesoura, iogurte, escada, ventilador, calçada, faca, carta, cabeça, caminhão, nove, livro, leão, frango, cubo, futebol, águia, furacão, zebra, pasta, caracol, estudante, lanterna, rato, orelhas, coelho, feijão, pescador, motocicleta, terapia, grama, sorveteria, saia, edifício, paralelepípedo, montanha, corpo, avião, veleiro, rins, isca, balões, berço, sinusite, surdo, bicicleta, empoeirado, mecânico.

A escolha das palavras foi aleatória, porém, rica em diferentes vocábulos da língua portuguesa para tentar capturar melhor o estilo de fala de diferentes locutores.

Esse banco de falas construído está disponível no link: <https://bit.ly/2LgfoZu>.

APÊNDICE B – FRASES GRAVADAS POR CADA LOCUTOR NA IDENTIFICAÇÃO DE UM LOCUTOR DO MÓDULO.

As seguintes frases, retiradas de músicas do Djavan, foram gravadas por cada um dos locutores para que se pudesse fazer a identificação de locutores pelo módulo, conforme mencionado na subseção 4.2.4.

Açaí: Solidão de manhã. Poeira tomando assento. Rajada de vento. Som de assombração. Coração. Sangrando toda palavra sã. A paixão puro afã. Místico clã de sereia. Castelo de areia. Ira de tubarão, ilusão. O sol brilha por si. Açaí, guardiã. Zum de besouro um ímã. Branca é a tez da manhã.

Sina: Pai e mãe, ouro de mina. Coração, desejo e sina. Tudo mais, pura rotina, jazz. Tocarei seu nome pra poder falar de amor. Minha princesa, art-nouveau. Da natureza, tudo o mais. Pura beleza, jazz. A luz de um grande prazer. É irremediável neon. Quando o grito do prazer. Açoitar o ar, réveillon. O luar, estrela do mar, o sol e o dom. Quiçá, um dia, a fúria desse front. Virá lapidar o sonho até gerar o som. Como querer Caetanear o que há de bom.');

Linha do Equador: Luz das estrelas. Laço pro infinito. Gosto tanto dela assim. Rosa amarela. Voz de todo grito. Gosto tanto dela assim. Esse imenso, desmedido amor. Vai além de seja o que for. Vai além de onde eu vou. Do que sou, minha dor. Minha Linha do Equador. Esse imenso, desmedido amor. Vai além que seja o que for. Passa mais além do céu de Brasília. Traço do arquiteto. Gosto tanto dela assim. Gosto de filha, música de preto. Gosto tanto dela assim. Essa desmesura de paixão é loucura de coração. Minha Foz do Iguaçu. Pólo Sul, meu azul. Luz do sentimento nu. Esse imenso, desmedido amor. Vai além que seja o que for. Vai além de onde eu vou. Do que sou, minha dor. Minha Linha do Equador. Mas é doce morrer nesse mar de lembrar. E nunca esquecer. Se eu tivesse mais alma pra dar, eu daria. Isso pra mim é viver.

Anexos

ANEXO A – CÓDIGOS E PSEUDO-CÓDIGOS

O pseudo-código da Figura 63 é um algoritmo de Cooley–Tukey para executar a FFT, citado na subseção 3.4.4.

O algoritmo da Figura 64 mostra a implementação de uma RNA simples do tipo *feedfoward* com 3 camadas e função sigmoidal como ativação, citado na subseção 3.7.2.

O pseudo-código da Figura 65 é utilizado para implementação do Rprop, citado na subseção 3.7.3.

Todos os códigos que foram escritos para esse trabalho estão disponíveis em: <https://github.com/marciooluizz/Modulo-Auditivo-Artificial>.

Figura 63 – Pseudo-código para a execução da FFT.

$\mathbf{Y}[0,...,n-1] \leftarrow$ recfft $2(n, \mathbf{X}, \iota)$:
IF n=1 THEN
    $Y[0] \leftarrow X[0]$
ELSE
    $\mathbf{Y}[0,...,n/2-1] \leftarrow$ recfft2 $(n/2, \mathbf{X}, 2\iota)$
    $\mathbf{Y}[n/2,...,n-1] \leftarrow$ recfft2 $(n/2, \mathbf{X}+\iota, 2\iota)$
    FOR $k_1 = 0$ TO $(n/2)-1$ DO
        $t \leftarrow \mathbf{Y}[k_1]$
        $\mathbf{Y}[k_1] \leftarrow t + \omega_n^{k_1} \mathbf{Y}[k_1 + n/2]$
        $\mathbf{Y}[k_1 + n/2] \leftarrow t - \omega_n^{k_1} \mathbf{Y}[k_1 + n/2]$
    END FOR
END IF

Fonte: Burrus (2012)

Figura 64 – Algoritmo para implementar uma simples rede *feedfoward*.

```
%Rede Neural Feed Foward Simples

for i=1:NumInput                                    %Layer Input
Input(i) = 1.0/(1.0 + exp(In(i)/100)) ;             %Sigmoide
end


for j = 1:NumHidden                                 %Layer Hidden
  SumH(j) = WeightIH(1,j) ;
    for i = 1:NumInput
      SumH(j) = SumH(j) + (Input(i) * WeightIH(i,j)) ; %Soma e pesos
    end
  Hidden(j) = 1.0/(1.0 + exp(-SumH(j))) ;           %Sigmoide
end


for  k = 1:NumOutput                                %Layer Output
  SumO(k) = WeightHO(1,k) ;
    for j = 1:NumHidden
      SumO(k) = SumO(k) + (Hidden(j) * WeightHO(j,k)) ; %Soma e pesos
    end
  Output(k) = 1.0/(1.0 + exp(-SumO(k))) ;           %Sigmoide
end
```

Fonte: Autor

Figura 65 – Pseudo-código para implementar Rprop.

$\forall i,j : \triangle_{ij}(t) = \triangle_0$
$\forall i,j : \frac{\partial E}{\partial w_{ij}}(t-1) = 0$
Repeat
  Compute Gradient $\frac{\partial E}{\partial w}(t)$
  For all weights and biases{
    if $(\frac{\partial E}{\partial w_{ij}}(t-1) * \frac{\partial E}{\partial w_{ij}}(t) > 0)$ **then** {
      $\triangle_{ij}(t) =$ **minimum** $(\triangle_{ij}(t-1) * \eta^+, \triangle_{max})$
      $\triangle w_{ij}(t) = -$ **sign** $(\frac{\partial E}{\partial w_{ij}}(t)) * \triangle_{ij}(t)$
      $w_{ij}(t+1) = w_{ij}(t) + \triangle w_{ij}(t)$
      $\frac{\partial E}{\partial w_{ij}}(t-1) = \frac{\partial E}{\partial w_{ij}}(t)$
    }
    else if $(\frac{\partial E}{\partial w_{ij}}(t-1) * \frac{\partial E}{\partial w_{ij}}(t) < 0)$ **then** {
      $\triangle_{ij}(t) =$ **maximum** $(\triangle_{ij}(t-1) * \eta^-, \triangle_{min})$
      $\frac{\partial E}{\partial w_{ij}}(t-1) = 0$
    }
    else if $(\frac{\partial E}{\partial w_{ij}}(t-1) * \frac{\partial E}{\partial w_{ij}}(t) = 0)$ **then** {
      $\triangle w_{ij}(t) = -$ **sign** $(\frac{\partial E}{\partial w_{ij}}(t)) * \triangle_{ij}(t)$
      $w_{ij}(t+1) = w_{ij}(t) + \triangle w_{ij}(t)$
      $\frac{\partial E}{\partial w_{ij}}(t-1) = \frac{\partial E}{\partial w_{ij}}(t)$
    }
  }

Fonte: Riedmiller (1994)