UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

CARLOS ANDRÉ CARIOCA DA SILVA JÚNIOR

DESENVOLVIMENTO DE UM SISTEMA DE RECONHECIMENTO DE FALA PARA O CONTROLE DE UM ROBÔ

SÃO LUÍS
2018

CARLOS ANDRÉ CARIOCA DA SILVA JÚNIOR

DESENVOLVIMENTO DE UM SISTEMA DE RECONHECIMENTO DE FALA PARA O CONTROLE DE UM ROBÔ

Monografia apresentada ao Curso de Engenharia Elétrica da Universidade Federal do Maranhão como requisito para obtenção do grau de Bacharel em Engenharia Elétrica.

Orientador: Prof. Dr. Manuel Leonel da Costa Neto
Coorientador: Prof. Dr. Luciano Buonocore

SÃO LUÍS
2018

Silva Junior, Carlos André Carioca da.
Desenvolvimento de um sistema de reconhecimento de fala para o controle de um robô / Carlos André Carioca da Silva Junior. - 2018.
75 f.

Coorientador(a): Luciano Buonocore.
Orientador(a): Manuel Leonel da Costa Neto.
Monografia (Graduação) - Curso de Engenharia Elétrica, Universidade Federal do Maranhão, UFMA, 2018.

1. Comunicação Wireless. 2. Reconhecimento de Fala. 3. Redes Neurais. 4. Robótica Móvel. I. Buonocore, Luciano. II. Costa Neto, Manuel Leonel da. III. Título.

CARLOS ANDRÉ CARIOCA DA SILVA JÚNIOR

DESENVOLVIMENTO DE UM SISTEMA DE RECONHECIMENTO DE FALA PARA O CONTROLE DE UM ROBÔ

Aprovada em 18 / 12 / 2018

BANCA EXAMINADORA

Prof. Dr. Manuel Leonel da Costa Neto (Orientador)
Departamento de Engenharia Elétrica - UFMA

Prof. Dr. Luciano Buonocore (Co Orientador)
Departamento de Engenharia Elétrica - UFMA

Prof. Dr. Andre Borges Cavalcante (Membro)
Departamento de Engenharia Elétrica – UFMA

“Não temas, porque eu sou contigo; não te assombres, porque eu sou teu Deus; eu te fortaleço, e te ajudo, e te sustento com a destra da minha justiça.”

Isaías 41:10

AGRADECIMENTOS

Agradeço a Deus, que em sua graça derramada diariamente me trouxe até aqui, renovando as minhas forças e me ensinando através de cada desafio que aparece.

Agradeço aos meus familiares, cada um que individualmente e de sua forma colaborou para que eu conquistasse este sonho. Me sinto honrado em ter sido abençoado com tanta generosidade e apoio, em especial minha avó, Eunice Costa Moura, minhas tias Yangra, Sandra Maria, Yanna, Marizilda, Iara e Solange Mourão; meus tios Osvaldo e Marcos; Meus irmãos Ana Carla e João António Carioca e Minha mãe, Taciane Costa Moura, que sem dúvidas alguma faz parte de cada uma das páginas desta apresentação.

Agradeço também à Daniel, Monik, Moyzeonne, Luís Felipe, Lucas, Arthur, Gabriel. Amigos da turma 2015.1 de Engenharia Elétrica da UFMA, imprescindíveis para minha formação quanto profissional e cidadão.

Por fim, agradeço aos meus orientadores Dr. Manuel Leonel da Costa Neto e Dr. Luciano Buonocore, pelo inestimável apoio no desenvolvimento deste trabalho.

RESUMO

A voz é um dos meios mais antigos e primordiais que a humanidade encontrou para se comunicar. Graças ao sistema de produção e reconhecimento sonoro humano, podemos emitir uma variedade de sons que possibilitam uma comunicação clara e eficiente entre seres humanos. No decorrer dos anos foram desenvolvidos meios de interação do homem com objetos inanimados, que necessitam de comandos por algum meio, como impulsos elétricos, toque, sinais, efeitos luminosos, entre outros. Dentro desse contexto, para gerar comodidade e eficiência na comunicação entre homem e máquina, propõe-se o desenvolvimento de algoritmos com a função de realizar uma interface entre a fala humana e uma codificação compreensível para acionar uma máquina. Com os avanços tecnológicos apresentados nos últimos anos, tornou-se possível a implementação prática do processamento e reconhecimento da fala. Através desses avanços são desenvolvidos *hardware* e *software*, com um poder de processamento relativamente elevado, essenciais na análise e parametrização de sinais. Partindo desta base de informação construiu-se, através de linguagem Python, um algoritmo capaz de realizar uma interface e comandar um robô de forma simples (deslocamentos ou giros) ou em movimentos combinados, além de sua parada. Para a validação da teoria e prática o sistema desenvolvido foi submetido à testes, com a intenção de torná-lo robusto e pouco suscetível a interferências externas. Foram obtidos resultados satisfatórios, conforme os objetivos propostos, com taxa de acerto no reconhecimento de fala de mais de 90% e com o robô realizando movimentos e paradas conforme solicitação vocal.

Palavras-chave: Reconhecimento de fala, Redes Neurais, Robótica móvel, Robótica móvel, Comunicação *wireless*.

ABSTRACT

The voice is one of the most important and primordial means that humanity found to communicate. Thanks to our sound production and recognition system, we can provide a variety of sounds that enable clear and efficient communication between humans. In the course of the years, they were the means of man's interaction with inanimate objects, which are sensitive to sound, such as electrical impulses, touch, signs, luminous effects, among others. In the context of, which is to generate and communication in between machine and, proposed the development of algorithm with a function of pre-interface between a comprehensible speech and codification, to acceding a machine. With the technological advances in recent years, one can participate in the practical implementation of speech processing and recognition. These initiatives are developed with hardware and software, with a relatively high processing power, with analysis and parameterization of signals. Partition based information of construct-by-the-text, Python, one software may be used in the interface and command the robot of simple gems. The application of theory and practice has been tested, with the intention of becoming robust and less susceptible to external interference. Satisfactory results were obtained, with defined objectives, with 90% speech recognition access rate and with the objective of performing sessions and series according to vocal request.

Keywords: Speech recognition, Neural networks, Mobile robotics, Wireless Communication.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

HMM	*Hidden Markov* Mode
GMM	*Gaussian Misture Models*
IIR	*Infinite Impulse Response*
FFT	*Fast Fourier Transform*
RoSeLi	Robô Seguidor de Linha
LRC	Laboratório de Robótica Móvel e Comunicação Sem Fio
MLP	*Multi Layer Perceptron*
LED	*Light Emitter Diode*
RNN	*Recurrent Neural Network*
I2C	*Inter Integrated Circuit*
API	*Application Programming Interface*
SPI	*Serial Peripheral Interface*
A/D	Analógico/Digital
GPIO	*General Purpose Input Output*
PWM	*Pulse Width Modulation*
DC	*Direct Current*

SUMÁRIO

1. INTRODUÇÃO

A teoria relativa ao processamento de sinais sonoros, assim como a extração de suas características com potencial de informação, é conhecida há várias décadas e presente em diversas literaturas, (RABINER e SCHAFER, 2007), (ALCAIM e OLIVEIRA, 2011) e (CAMPBELL,1997). A voz humana, correspondente a ondas sonoras, que carrega traços inerentes ao ser humano. Nela pode-se extrair informações a respeito do que está sendo dito, quem está falando, qual o contexto em que se expressa e até mesmo o estado emocional do locutor (ROSA, 2017). O surgimento de *hardware* mais potentes, em termos de processamento e armazenagem da informação, trouxe uma diversidade de *software* capaz de extrair e processar essas características, tais como: Cortana, Siri e Google Now[1]. Esses programas usam diversificadas técnicas e estruturas para análise de dados, possibilitando lidar com um número grande de informações e extrair delas uma diversidade de características.

Tabela 1- Histórico do reconhecimento de fala.

Final da década de 1950	Primeiras pesquisas tecnológicas para o reconhecimento de voz.
1964	IBM apresenta um sintetizador de voz para a fala de dígitos.
1978	A Texas Instruments lançou o primeiro chip dedicado à síntese de voz.
1993	IBM lança o primeiro software comercial para reconhecimento de voz, o IBM Personal Dictation System, para OS/2.
1993	Apple apresenta um conjunto de rotinas para Mac, para reconhecimento e síntese de voz.
1993	Universidade Federal do Rio de Janeiro desenvolve Dosvox, com síntese de voz em português, para deficientes visuais usarem PC's com DOS.
1994	Dragon Systems apresenta o Dragon Dictate para ditados.
1996	IBM apresenta o MedSpeak/Radiology, primeiro produto para reconhecimento da fala contínua em tempo real.
1996	OS/2 Warp é o primeiro sistema a embutir comandos de voz.
1997	Dragon Systems lança o primeiro programa de uso geral para reconhecimento da fala contínua em inglês.
1997	IBM lança o ViaVoice, para fala contínua.
1998	IBM lança ViaVoice em português.
1998	MicroPower lança DeltaTalk, sintetizador de voz em português.
1999	Philips lança FreeSpeech 2000, com reconhecimento da fala em português.
1999	Lotus e Corel acrescentam recursos de voz a seus pacotes de aplicativos.
2000	L&H adquire Dragon Systems e lança L&H Dragon NaturallySpeaking 5.0.
2001	Telemar lança Vocall, primeiro serviço de voz aberto ao público, com síntese e reconhecimento da fala, para e-mails e agenda.
2001	L&H é colocada à venda, por se encontrar em grave crise financeira.
2001	Microsoft acrescenta recursos de voz (para ditados e comandos) ao Office XP. Na versão em português, essa facilidade está ausente.

Fonte: http://web.ist.utl.pt/ist178719/

A maior dificuldade inerente ao processo de reconhecimento de fala é comparar parâmetros semelhantes que, por vezes, são confundidos do ponto de vista humano, fazendo-se necessário o uso de técnicas avançadas para distinguir uma locução de outra. Uma das técnicas mais avançadas e eficientes

[1] Site: https://www.digitaltrends.com/computing/cortana-vs-siri-vs-google-now/

que existe hoje em dia para análise de dados, é a rede neural (KOLEN, 1994), a qual permite taxas de acertos superiores a 95%, em troca de maior custo computacional em sua fase de treinamento. No sistema desenvolvido neste trabalho é usada esta técnica, pois gera resultados relevantes e possui um suporte razoável em linguagem Python[2].

Portanto, nesta pesquisa é desenvolvido um sistema de reconhecimento de fala para controle dos movimentos de um robô. São usados comandos de voz via *wireless*, de modo que o robô possa realizar movimentos simples ou combinados, em direções pré-definidas. Assim, estes comandos são provenientes da fala humana e exequíveis para o dispositivo. O desenvolvimento do sistema é realizado com base nos conhecimentos adquiridos na disciplina TEEE – Processamento Digital de Voz e Aprendizagem em Linguagem de Máquina, no curso de Engenharia Elétrica da UFMA. O robô foi desenvolvido pelo aluno Daniel Ribeiro junto ao professor Dr. Luciano Buonocore, em um trabalho de monografia na UFMA, já tendo sido testado em diversas aplicações, tais como mapeamento de ambiente e localização por imagem.

1.1 Objetivos

O objetivo geral deste trabalho é implementar um sistema de controle de movimentos para um robô por intermédio de um sistema de reconhecimento da fala.

Os objetivos específicos são:

a) Realizar o Controle dos movimentos do robô em sete possibilidades de comando, sendo eles (Frente, Atrás, Esquerda, Direita, Acelerar, Parar), além de reconhecer o próprio nome e referências diretas a ele, através da fala;
b) Fazer a comunicação entre o sistema de controle, *software* no *smartphone*, com o robô móvel através de comunicação *wireless*; e
c) Mostrar a eficiência de comandos de voz se comparados a outros tipos de comandos usuais, considerando a velocidade e efetividade da ação.

1.2 Motivação

A motivação para desenvolvimento deste tema deve-se inicialmente a necessidade de criação de tecnologia, a nível nacional, que acompanhe o desenvolvimento mundial, a fim de aprimorar sistemas, técnicas e propostas no ramo da inteligência artificial. O desenvolvimento de tecnologias que atendem a

[2] Site: https://pytorch.org/

necessidade nacional e que permitem a aplicação no mercado, aliada à competitividade comercial e qualidade no desempenho, é um desafio a ser vencido.

No mesmo sentido é explorada a teoria conhecida sobre processamento digital de voz, validando o que a muito tempo já foi descoberto, mas que por limitações tecnológicas não pode ser empregado.

Por fim, tem-se a contribuição da implementação e validação desta proposta de trabalho, na formação acadêmica, aprimorando os conhecimentos obtidos na disciplina TEEE – Processamento digital de voz, e os *software* anteriormente desenvolvidos de reconhecimento de locutor.

Espera-se, também, que este material possa ser utilizado como uma das referências para estudo sobre o reconhecimento de fala e que seja futuramente aprimorado, realizando o controle do robô em funções mais diversas.

1.3 Organização do trabalho

O trabalho está organizado em oito capítulos, além desta introdução ao tema.

No Capítulo 2 é realizada uma abordagem da teoria do reconhecimento de fala, sua organização e divisão, levando-se em consideração a problemática, possíveis soluções com o uso de técnicas de inteligência artificial. Também são descritas as características do robô que será objeto de controle neste trabalho.

No Capítulo 3 é apresentado, de modo mais aprofundado, toda a fase de pré-processamento a qual deve ser submetido um sinal, para que este tenha características válidas e coerentes com a realidade, de modo que influencie positivamente na assertividade do sistema.

No Capítulo 4 é descrita a predição linear e sua fundamentação matemática, ou seja, a técnica base neste trabalho, pois influencia diretamente no rendimento do sistema. Graças ao emprego da predição pode-se criar redes neurais mais simples que economizam processamento e memória.

No Capítulo 5 é apresentada a rede neural, um dos modelos mais promissores de inteligência artificial já desenvolvido. Tratando-se de sua base histórica, cientifica e do modelo proposto. Sua complexidade e alto custo computacional são compensadas pela alta assertividade em comparação de dados.

No Capítulo 6 são apresentadas as informações abordadas nos Capítulos 2, 3 e 4, em termos de *software* e programação na linguagem Python. Neste capítulo tem-se de fato a apresentação do *software* do sistema.

No Capítulo 7 é demonstrada a efetividade do sistema proposto, detalhando de forma estatística, após os testes no robô físico com o funcionamento completo do *software*. São apresentados os resultados obtidos e os fatores que influenciam na assertividade, seja de modo positivo ou negativo.

Finalmente, no Capítulo 8 apresentam-se as conclusões e propostas para trabalhos futuros que podem ser realizados a partir das informações tratadas, com a possibilidade de uma margem de expansão para esta proposta.

2. FUNDAMENTAÇÃO TEÓRICA

Atualmente existem várias técnicas de acionamento e controles de robôs, usados em operações específicas. Uma das técnicas que permite a interação de forma simples entre o ser humano e o robô é através de comandos de voz e ,para tal, é necessário o desenvolvimento de um sistema de reconhecimento de locutor. O desenvolvimento de uma proposta desse tipo não é uma tarefa trivial, devido as características e geração da fala humana. O procedimento é extremamente complexo, não segue um padrão fixo, um comportamento lógico e é altamente variável com o contexto[3]. Dessa forma são usadas técnicas probabilísticas, como: *Hidden Markov Models* (HMMs) ou Inteligência Artificial, através das Redes Neurais. Neste capítulo é apresentada uma fundamentação teórica sobre reconhecimento de fala, o robô controlado através de comandos de voz e a tecnologia usada no desenvolvimento do sistema.

2.1 Estrutura geral de um sistema de reconhecimento de fala

A teoria de processamento da fala humana começou a ser desenvolvida há várias décadas passadas e pode ser encontrada em diversas literaturas, tais como: (RABINER e SCHAFER, 2007), (ALCAIM e OLIVEIRA, 2011), (CAMPBELL, 1997). No processamento, o sinal de voz é modelado com formato periódico, e carrega com ele características importantes da fala e do próprio locutor (ALCAIM, 2011).

As limitações computacionais no passado não permitiam o desenvolvimento de algoritmos capazes para tratar com precisão as informações contidas nas locuções da fala, limitando aos dispositivos que aplicavam esta teoria a funções básicas. Os primeiros protótipos criados com intuito de reconhecer fala foram desenvolvidos na década de 60, do século passado, pela Bell Labs e IBM, chamados de ALDREY e SHOEBOX respectivamente, sendo altamente limitados se comparados com os sistemas atuais[4]

Com o desenvolvimento dos HMMs (NORRIS, 1997) e o avanço na capacidade de *hardware* ocorreu um crescimento exponencial na capacidade dos *software* para reconhecimento da fala, tornando os sistemas mais funcionais. Atualmente, técnicas como HMMs, Mistura gaussiana e Redes Neurais, são bastante empregadas nos algoritmos existentes para reconhecimentos de fala e suas teorias se encontram desenvolvidas e consolidadas (CARDOSO, 2009), (FAUSET, 1994), (PICONE, 1993) e (ROCHA, 2018).

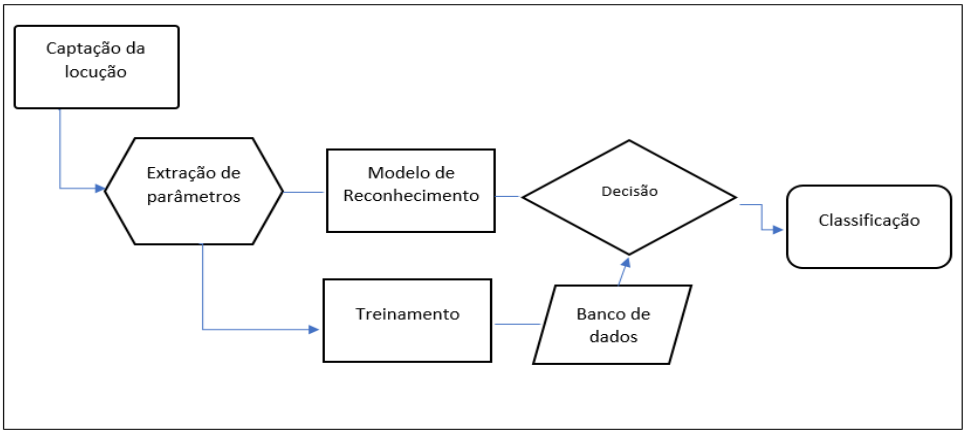
Atualmente, com avanço tecnológico de *software* e de *hardware*, é possível criar sistemas mais robustos e precisos no reconhecimento de locutor (CAMPBELL,1997)

Esses sistemas, em geral, possuem uma estrutura conforme apresentado no diagrama de blocos da Figura 1. Inicialmente é realizada a captação da locução (aquisição do sinal da fala) através de um dispositivo adequado, como por exemplo um microfone. A seguir, são extraídos os parâmetros do sinal da fala e realizado um treinamento no sistema, com os dados obtidos, cujo resultado e guardado em um

[3] Site: https://www.youtube.com/watch?v=W95YlM5-iPk&t=2460s
[4] Site: http://web.ist.utl.pt/ist178719/

banco de dados. Posteriormente, ocorre o reconhecimento da fala quando o sistema captura uma nova locução, similar dentre as que foram previamente determinadas. É realizada novamente a extração dos parâmetros, que são aplicados a um modelo de reconhecimento, previamente definido e desenvolvido. Em seguida, é feito um teste de decisão através do reconhecimento de padrões entre as amostras que constam no banco de dados e a amostra que consta na saída do modelo de reconhecimento. Por último, tem-se o estágio de classificação que é o principal diferencial entre os diversos sistemas de reconhecimento da fala.

Figura 1- Estrutura do reconhecimento de fala



Fonte: Autor

Verifica-se, que nesses sistemas o banco de dados é fundamental para que seja possível, por comparação, classificar uma locução como mais semelhante a um determinado grupo (HAYKIN, 1998). Esse banco é gerado através de um treinamento do sistema com um conjunto de amostras das locuções. O algoritmo deve ser capaz de decidir com base em um modelo à que classe a locução mais se assemelha, assim classificando-a em classes pré-definidas.

Neste trabalho é desenvolvido um sistema que foi capaz de obter uma assertividade maior que 90% no reconhecimento da fala, que é utilizado para comandar movimentos de um robô.

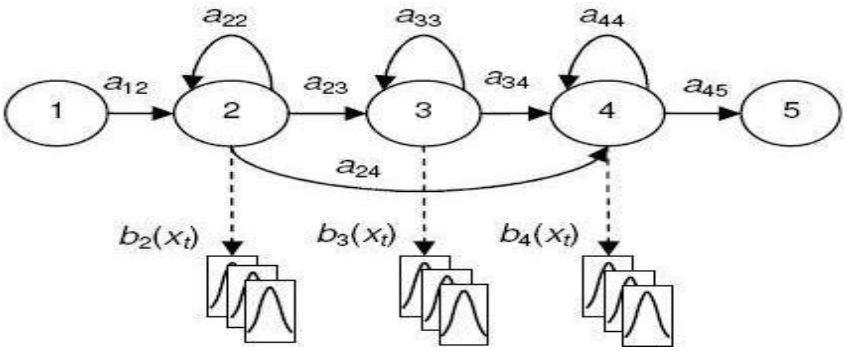
2.2 Técnicas para reconhecimento da fala

Atualmente, existem várias técnicas de classificação de amostras do sinal que podem ser empregadas no reconhecimento de fala, porém as técnicas de Mistura Gaussiana e Redes Neurais se destacam pela capacidade de se obter taxas de acertos superiores a 90%.

A Mistura Gaussiana, de acordo com CARDOSO (2009), é uma técnica que consiste em modelar os locutores como um conjunto de funções de densidade probabilística gaussiana. Cada componente da mistura é representada por uma função de densidade de probabilidade e são atribuídos pesos a esses componentes que modelam o sistema gaussiano (ANDERSON e WILLIAM, 1991). Esses sistemas baseiam-se em uma máquina de estado e trata a locução como um sistema de passos, onde ocorre uma

dependência entre o passo atual e o passo anterior, como visto no modelo com cinco estados, conforme mostrado na Figura 2.

Figura 2- Modelo de GMM



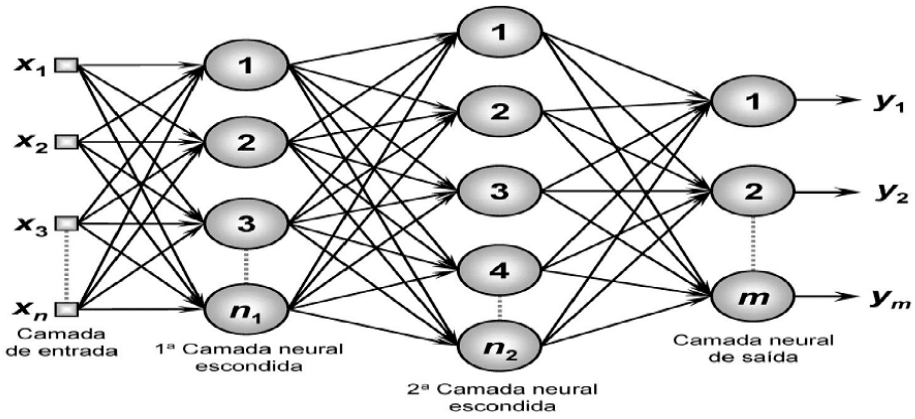
Fonte:https://www.gta.ufrj.br/grad/09_1/versao-final/impvocal/hmms.html

Neste modelo, os estados 1,2,3,4 e 5 podem representar uma palavra, fonema ou silaba, havendo troca de estado a cada unidade de tempo pré-definida. As saídas geradas por cada estado de a_{12} ate a_{45} são vetores acústicos que carregam uma função de densidade de probabilidade associada (b2(x1),..., b4(x1)), sendo caracterizadas por um modelo de (Gaussian Misture Models - GMM). Essa capacidade de tomar decisões baseada em estados anteriores é muito importante em qualquer sistema de reconhecimento de fala, pois em muitos casos as locuções serão muito semelhantes precisando de um sistema robusto que considere o máximo de informação útil para a tomada de decisão.

Os sistemas mais recentes de classificação da fala utilizam Redes Neurais. Nestes sistemas é possível se obter taxas de acertos superiores a 95%, a custo de alta capacidade computacional, podendo ser combinado a HMMs formando uma Rede Neural recorrente, (KOLEN, 1994). Neste trabalho é utilizado um modelo de rede neural especialistas que visam minimizar erro médio quadrático reproduzido na saída.

Geralmente, o modelo elementar de uma Rede Neural baseia-se em um conjunto de neurônios que podem ser ajustados e treinados para tomar decisões elementares, mas que combinados podem gerar modelos capazes de distinguir um número muito grande de informações. A arquitetura de uma rede é apresentada na Figura 3, onde X1, X2,..X2 correspondem aos dados de entrada, Y1,Y2, ...Ym, correspondem aos dados de saída e entre entradas e saídas tem-se as camadas neurais escondidas. Este modelo é descrito de forma mais detalhada no capitulo 5, pois é empregado neste trabalho.

Figura 3- Modelo de Rede Neural elementar.



Fonte:https://www.gta.ufrj.br/grad/09_1/versao-final/impvocal/hmms.html

2.3 Problemáticas em um sistema de reconhecimento da fala

Com base em estudos realizados e desenvolvimento de sistemas de reconhecimento de fala (RABINER; SHAFER, 2007) foram identificados vários tipos problemas, dentre os quais se destacam:

2.3.1 Variabilidade do Sinal da Fala

A variabilidade do sinal da fala está relacionada a aspectos internos e externos do fenômeno de produção da mesma. Deve-se levar em consideração a alterabilidade do locutor, que vai depender do estado emocional, do contexto da conversação, do sotaque, da linguística e qualquer fenômeno cultural ou físico que influencie na fala. No meio externo deve-se levar em consideração os dispositivos de aquisição da fala, transmissão da informação, estrutura de conversão e até mesmo a presença de ruídos (sinais indesejáveis), pois esses fatores influenciaram na taxa de acerto do sistema.

Figura 4- Variabilidade do sinal da fala

Fonte: https://korntraducoes.com.br/sotaque-mesmo-idioma-pronuncias-diferentes/

2.3.2 Efeito de coarticulação no sinal da fala

No processo natural de produção da fala não existe pausa nem separação entre fonemas, pois devido ao efeito de coarticulação os elementos são influenciados mutuamente. O que permite separar fonemas, silabas e palavras é o conhecimento que os indivíduos têm sobre a sua linguagem, algo que não é simples do ponto de vista do reconhecimento da fala.

2.3.3 Redundância no sinal da fala

Em um sistema de reconhecimento deve-se dar uma atenção especial na extração dos parâmetros que caracterizem o tipo de informação útil, mesmo que não sejam definidas as regras que descrevam quais são os parâmetros úteis ou não. Esses parâmetros dependem da aplicação que se deseja obter, pois a fala carrega diversas características, muitas delas redundâncias que permitem identificar o locutor, seu estado emocional, seu sotaque, etc. Neste trabalho estas informações não são consideradas pois fogem do escopo do trabalho.

2.3.4 Quantidade de dados a processar

Os sistemas de reconhecimento da fala precisam armazenar e processar, de forma relativa, uma grande quantidade de dados para estabelecer as características que fazem a diferença entre classes diferentes no reconhecimento. Isso exige *software* e *hardware* mais robustos, com alta capacidade, que felizmente são mais comuns atualmente do que foram outrora. O uso de Redes Neurais amplia este problema, pois durante a fase de treinamento milhares de operações matemáticas de gradiente e ajuste são realizadas, o que pode fazer com que a fase de treinamento requeira horas de processamento para ser concluída.

2.4 Possíveis áreas de aplicação do reconhecimento de fala

Existe uma grande margem de empregabilidade para sistemas de reconhecimento de fala. Atualmente, eles são utilizados em *smartphones*, *notebooks*, como senha para acesso a determinados sistemas, etc. Com a evolução da tecnologia espera-se tornar o comando por voz o meio mais comum de se ordenar tarefas, pois é mais rápido e eficiente, se comparado ao método de digitar em um teclado ou a um controle remoto. Um exemplo disso é o emprego do sistema de reconhecimento da fala na automação residencial ou em atividades em que o usuário esteja com as duas mãos ocupadas.

Uma área de interesse para aplicação do reconhecimento de fala é a robótica móvel. Em tentativas de criar dispositivos capazes de realizar tarefas de maneira autônoma, necessita-se de uma interface de comando simples e eficiente. Para tal, a fala é o meio mais simples e eficiente que o ser humano possui para realizar esse interfaceamento. Porém, esse processo tido como básico para o cérebro humano é altamente complexo para a serem implementados usando técnicas de inteligência artificial. Por isso, ao

longo de pesquisas desenvolvidas nos últimos anos, buscou-se métodos de controle que mesclassem eficiência e inteligibilidade por parte dos dispositivos.

2.5 Métodos de controle utilizados na robótica

Desde de criação dos primeiros robôs, muitas problemáticas surgiram em torno dessa plataforma. Os primeiros pesquisadores preocuparam-se em sanar dúvidas como: Como faço para ele se deslocar de um ponto a outro? como planejo o movimento de um robô para que ele esquive de obstáculos? Como fazer para ele se localizar?

Com a evolução das tecnologias eletrônicas e computacionais, foram realizadas várias descobertas e propostas para resolver os problemas apresentados. Este trabalho, como muitos outros, visa propor uma metodologia de comando via *wireless*, assim como em ZHAI, (1992), onde, além do meio de comunicação deve-se focar na metodologia de aquisição de dados e na sua interpretação.

Como proposto em RIBEIRO(2017), tem-se o robô denominado RoSeLi, com capacidade de guiar-se, tendo como referência linhas no chão e a leitura de TAG´s que marcam poses em relação a um sistema de coordenadas globais que servem de referência para a máquina. Outra possibilidade é o controle táctil demonstrado em LIMA(2017) que explora a capacidade de gerar comandos de máquina através do movimento da mão humano.

Desta forma, a evolução no comando do robô pode ser utilizada para agregar a dinâmica de comunicação e eficiência de resposta da máquina. Espera-se, portanto, que ao utilizar a fala, como modo mais rápido e eficiente de gerar comandos, tenha-se dispositivos capazes de permitir interfaces homem-máquina mais interativas e eficientes.

2.6 Robô RoSeLi

A capacidade de executar as ordens dadas via qualquer tipo de comando depende diretamente da capacidade do dispositivo de executar adequadamente estes comandos. Neste trabalho o módulo reconhecimento da fala é utilizado para controle de determinados deslocamentos junto ao robô RoSeLi. Este robô é composto de uma estrutural em metal servido de dois motores de passo com rodas que permitem movimentos em qualquer sentido. O comando é acionado através uma placa Raspiberry PI 3B, programada em linguagem Python. Assim, essa estrutura permite ao robô executar comando de movimentação em qualquer direção, conforme apresentada em seção seguinte.

2.7 Linguagem PYTHON

A escolha da Linguagem Python, usada neste trabalho, decorre em função da facilidade de interação desta com a placa Raspberry PI 3B. Além disso, com base na ideia de construção de um *software* capaz

de reconhecer fala por Rede Neural, a utilização da linguagem Python[5] torna-se efetiva, pois a mesma conta com vastas bibliotecas que auxiliam no tratamento e processamento de voz. Acrescenta-se também os *frameworks* livres, altamente eficazes e de fácil utilização e interpretação como o KIVY[6] e PYTORCH[7], ideais para confecção de interfaces gráficas avançadas e Redes Neurais artificiais respectivamente.

[5] Site: https://www.python.org/doc/
[6] Site: https://kivy.org/doc/stable/
[7] Site: https://pytorch.org/

3. PRÉ-PROCESSAMENTO DO SINAL DA FALA

Para otimizar o desempenho das técnicas usadas no reconhecimento de fala deve-se submeter as locuções a um conjunto de pré-processamentos que sejam capazes de tratar o sinal de voz, ressaltando as informações relevantes contidas na fala e atenuando a quantidade de informação indesejada. Os pré-processamentos englobam desde a captação da fala útil, passando por algumas filtragens, até a segmentação e sobreposição da fala em blocos, que serão, posteriormente, convertidos em coeficientes de predição linear, ou coeficientes LPC. A Figura 5 ilustra essa etapa.

Figura 5- Etapas do pré-processamento

Fonte: Autor

3.1 Detecção de sinal sonoro

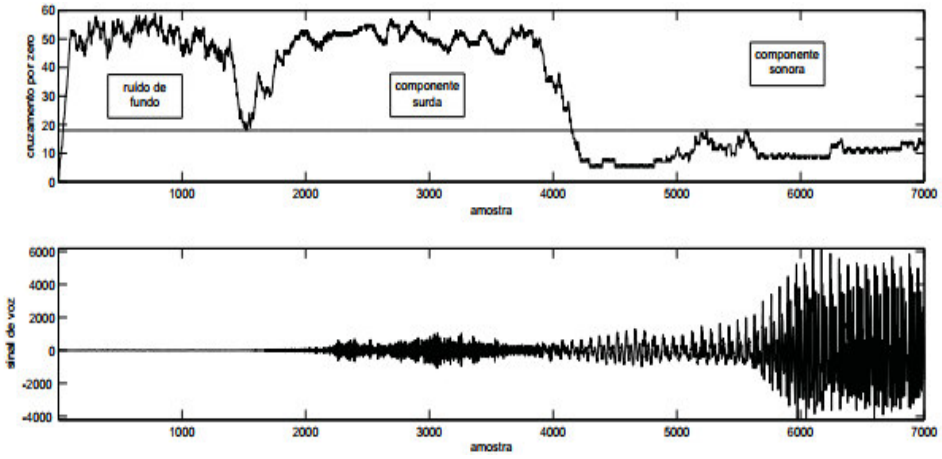
Para que se possa iniciar o processamento da fala, a primeira etapa é fazer a aquisição da locução. Este procedimento deve ser feito do modo mais eficiente possível, para que não haja perda de informação útil e ao mesmo tempo a atenuação de informações irrelevantes, como ruído ou sinal não vozeados. As técnicas mais usuais, neste caso, são: taxa de cruzamento por zero e método de energia mínima (CARDOSO, 2009), (RABINER e SHAFER, 2007).

3.1.1 Taxa de cruzamento por zero

A técnica de cruzamento por zero consiste em verificar o número de vezes em que o sinal da fala alterna entre níveis positivos e negativos ao longo de uma janela vista no domínio do tempo. Como o ruído e a componente surda comportam-se como um sinal aleatório, espera-se que as taxas de cruzamento por zero nesses segmentos sejam maiores do que as taxas nos segmentos sonoros. A Figura 6 ilustra um segmento da fala com segmentos sonoros e não sonoros.

Deve-se ressaltar que sinais consonantais tem taxa de cruzamento por zero com valores mais próximos as taxas de componentes surdas do que as taxas de componentes vozeadas. Desta forma é difícil definir o limiar de classificação entre trecho vozeado e não vozeado.

Figura 6- Taxa de cruzamento por zero



Fonte: (Cardoso,2009)

3.1.2 Método da energia mínima

O método energia mínima é definido como a razão entre a energia local de determinado quadro (ou segmento) pela energia mínima do ruído presente na locução. Como as locuções serão tratadas como sinais de tempo discreto pode-se determinar a energia de um sinal de voz conforme dado na Equação 3.1.

$$E = \sum_{m=-\infty}^{\infty} x^2[m] \qquad (3.1)$$

Assim, considerando um valor mínimo para energia do ruído presente no meio, pode-se definir um valor, que caso superado, pode ser definido como um trecho de fala vozeado. Nesse caso, o cálculo para ter validade deve ser realizado em locuções de curta duração, aplicadas em n janelas, e determinado pela Equação 3.2.

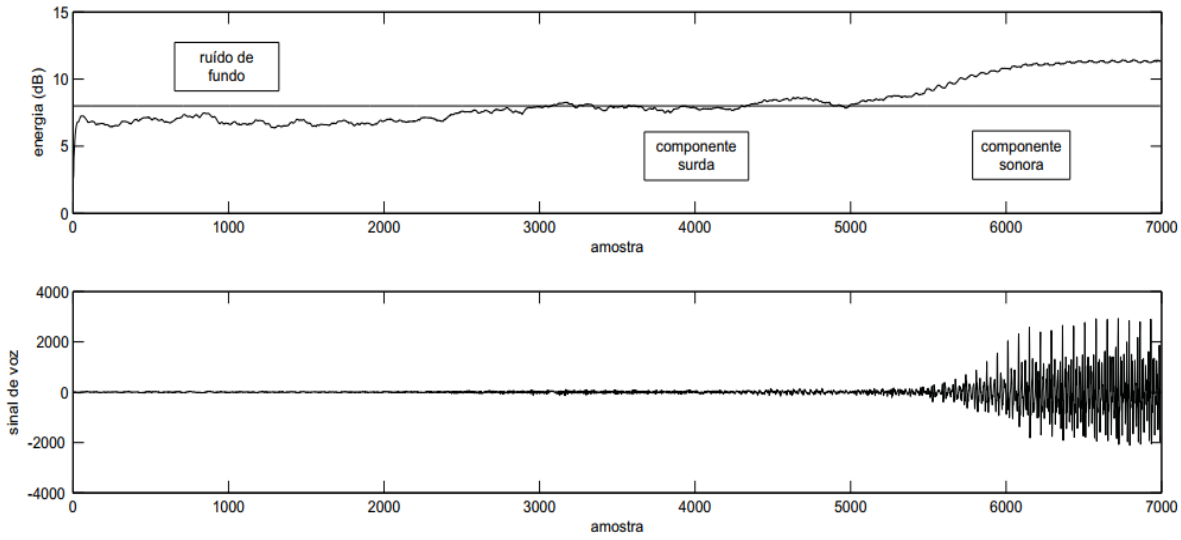
$$E_n = \sum_{m=-\infty}^{\infty} [x(m) * w(n-m)]^2 \qquad (3.2)$$

Normalmente, as janelas devem ter em torno de 20ms, pois se forem muito longas e constantes em amplitude, a energia varia pouco em relação ao tempo comportando-se como um filtro de passa-baixa de banda estreita, produzindo uma função que não condiz com a realidade energética do sinal. Por outro lado, se a janela for muito estreita não se obtém condições suficientes para produzir um resultado comparável ao sinal de ruído. Assim, para o intervalo de 20 ms o sinal torna-se ideal para análise. Este assunto será abordado posteriormente, na Seção 3.4.

Neste trabalho foi utilizado o método de energia mínima, pois oferece maior confiabilidade para iniciar uma gravação de locução e definir um critério de parada. Deve-se salientar que o método da taxa

de cruzamento por zero apresenta dificuldades para se distinguir sinais não vozeados de sinais não sonoros. Isso ocorre porque determinados segmentos, em especiais os de consoante assemelham-se a sinais não vozeados, como visto em alguns segmentos do sinal mostrado na Figura 5. Desta forma, quando se utiliza cruzamento por zero é necessária uma avaliação mais profunda do que realmente é fala útil. Este problema é atenuado no método de energia mínima, representado na Figura 7, já que ao elevar-se a parcela da equação 3.2 ao quadrado ressalta-se a diferença de energia contida em uma locução vazia de um trecho consonantal, facilitando a distinção da fala.

Figura 7- Energia mínima no sinal de fala



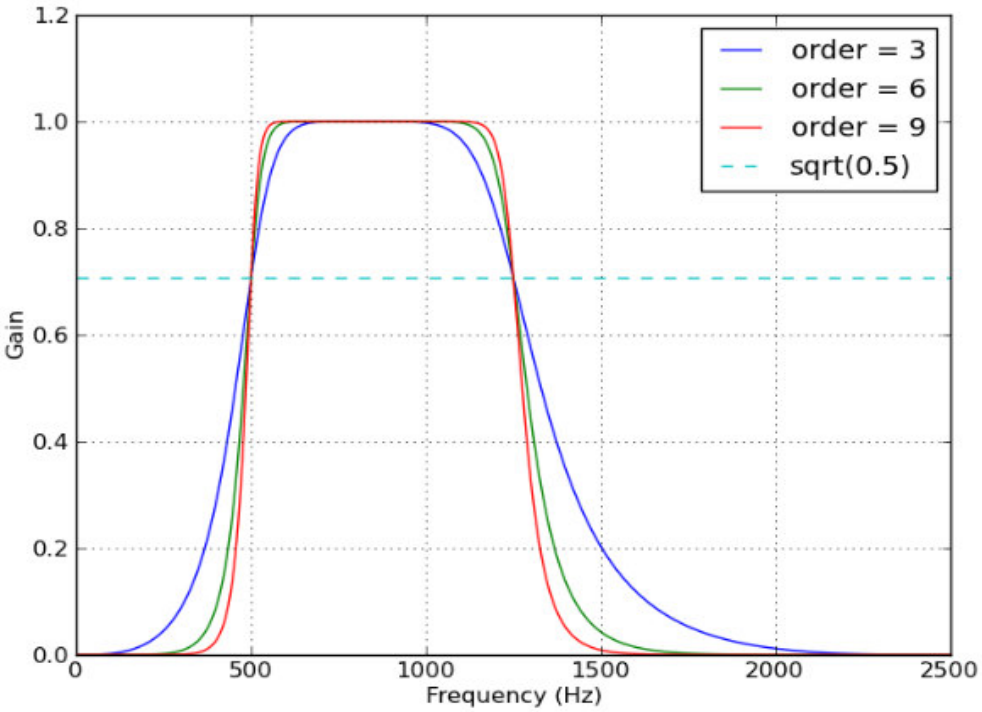
Fonte: (Cardoso, 2009)

3.2 Eliminação de ruídos no sinal da fala

Antes de aplicar as locuções obtidas para a detecção dos segmentos sonoros ou não sonoros deve-se submetê-las a uma filtragem, com intenção de se eliminar ruídos ocasionados na aquisição. Para realização desse processo pode ser usado um filtro digital IIR, passa faixa, *Butterworth*, de 5ª ordem (QUESADA, 1991). Utiliza-se este tipo de filtro, pois sua resposta, como pode ser visto em Figura 7, não possui oscilação na zona de transição, mas possui uma faixa larga nesta zona. Observa-se que o problema decorrente da faixa larga é corrigido aumentando a ordem do filtro, por isso é utilizado um filtro de 5° ordem. Além disso, um filtro digital pode ser implementado via *software*, reduzindo a complexidade para a obtenção de ordens mais elevadas se comparados a filtros analógicos. Desta forma, é possível implementar um filtro deste tipo com o auxílio de linguagem PYTHON, de forma mais simples que um analógico, em função do poder computacional.

Os filtros *Butterworth* foram desenvolvidos inicialmente por Stephen Butterworth em 1930, e tratado posteriormente, (QUESADA, 1991). Um filtro deste tipo, tem a resposta plana em determinada faixa de frequência (banda passante), com determinado ganho, sendo esse aproximadamente zero na banda de rejeição, conforme Figura 8.

Figura 8- Filtro *Butterworth*



Fonte: fonte:https://stackoverflow.com/questions/12093594/how-to-implement-band-pass-butterworth-filter-with-scipy-signal-butter

Verifica-se, então, que este tipo de filtro mantém uma resposta em frequência similar na banda passante para ordens mais elevadas, porém uma inclinação mais íngreme na faixa de transição. O custo disso é que o filtro *Butterworth* necessita de ordens maiores que seus semelhantes (Bessel, Chebyshev e Elíptico) para obter o mesmo resultado de transição conforme literatura pesquisada (QUESADA, 1991).

De modo geral, é possível se obter um filtro passa-faixa como resultado da associação em serie de um filtro passa-alta com o filtro passa-baixa. A Equação 3.3 define de forma aproximada a função de transferência desse tipo de filtro (QUESADA, 1991).

$$|H(w)| = \frac{1}{\sqrt{1+\varepsilon^2\psi_n^2(w)}} \quad (3.3)$$

Em que ε ≤ 1 é um número real positivo e ψ n(w) é um polinômio de ordem n contendo apenas potências pares (ou potências ímpares) do argumento w.

Filtros pertencentes a classe *Butterworth* são obtidos fazendo-se $\varepsilon = 1$ e $\psi_n(w) = w^n$. resultando na função de transferência dada pela Equação 3.4.

$$|H(w)| = \frac{1}{\sqrt{1+w^{2n}}} \quad (3.4)$$

Independentemente do valor da ordem n terem-se $|H(0)| = 1$ e uma resposta semelhante à Figura 7, onde a ordem do polinômio, define, inclinação da curva em relação ao ponto de inflexão, que pode resultar em uma otimização da resposta do sistema.

Para o filtro *Butterworth* as n primeiras derivadas de $|H(w)|$ são nulas na origem, w=0, e determinadas pela Equação 3.5:

$$\left.\frac{d^k|H(w)|}{dw^k}\right|_{w=0} = 0 \leq k \leq n \quad (3.5)$$

Por outro lado, no domínio s podem ser determinados os polos da função de transferência de H(s) de um filtro *Butterworth* de ordem n, no caso geral, onde σ_k *e* w_k são dados na Equação 3.6, obedecendo a relação: $\sigma_k^2 + w_k^2 = 1$

$$\begin{cases} \sigma_k = -sen(2k-1)\frac{\pi}{2n} \\ w_k = \cos(2k-1)\frac{\pi}{2n} \end{cases}, \qquad k = 1,2,\ldots..,n. \quad (3.6)$$

Obtém-se, então, a função de transferência H(s) determinando-se o "Polinômio de *Butterworth*" para o denominador do sistema.

Tomando como exemplo um filtro de 3° ordem, tem-se a função de transferência dada pela Equação 3.7[8].

$$H(s) \cdot H(-s) = \frac{1}{1-s^6} \quad (3.7)$$

Neste caso, as raízes do denominador são dadas por:

$$\pm 1, -\frac{1}{2} \pm j\frac{\sqrt{3}}{2} \; e \; \frac{1}{2} \pm j\frac{\sqrt{3}}{2}$$

Logo a função de transferência para este exemplo, no domínio da frequência é dada pela Equação 3.8.

$$H(w) = \frac{1}{1-2w^2+j(2w-w^3)} \quad (3.8)$$

No domínio s a função de transferência é dada pela Equação 3.9.

$$H(s) = \frac{1}{s^3+2s^2+2s+1} \quad (3.9)$$

[8] Site: http://www2.ee.ufpe.br/codec/aula%20pcom%20filtros.pdf

O polinômio no denominador da Equação 3.9 é o "Polinômio de *Butterworth*" de interesse que, devido a sua dificuldade de ser calculado, principalmente para ordens mais elevadas, é tabelado. A Tabela 2 mostra os polinómios com seus coeficientes até a 5º ordem, facilitando o projeto do filtro.

Tabela 2 - Polinômios de *Butterworth* até 5ª ordem

Ordem	**Polinômio de *Butterworth***
1ª	s+1
2ª	$s^2 + 1{,}4142s + 1$
3ª	$s^3 + 2s^2 + 2s + 1$
4ª	$s^4 + 2{,}6131s^3 + 3{,}4142s^2 + 2{,}6131s + 1$
5ª	$s^5 + 3{,}2361s^4 + 5{,}2361s^3 + 5{,}2361s^2 + 2{,}6131s + 1$

Fonte: Autor

3.3 Filtro Pré-ênfase

O filtro pré-ênfase utilizado no sistema de reconhecimento de fala tem a função de enfatizar informações que estão presentes em frequências mais elevadas no sinal da fala. Desta forma, este procedimento faz com que as informações em frequências mais elevadas da componente útil do sinal, que apresentam menor energia, sejam ressaltadas ao mesmo patamar das frequências baixas. No domínio da transformada Z este filtro é representado pela Equação 3.10 e no domínio do tempo discreto pela Equação 3.11.

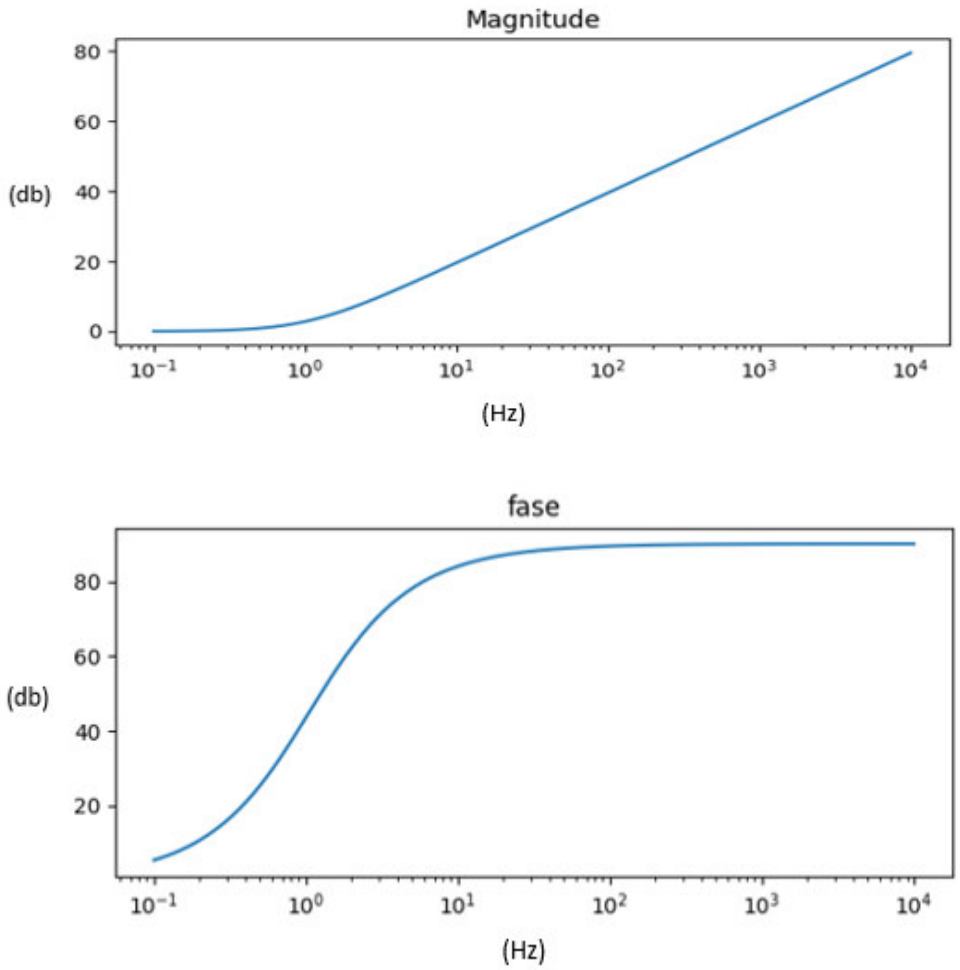
$$H(z) = 1 - a * z^{-1} \qquad (3.10)$$

$$s'(n) = s(n) - a * s(n-1) \qquad (3.11)$$

O coeficiente a pode assumir qualquer valor no intervalo de 0,9 a 0,98[9]. No trabalho aqui desenvolvido é usado a = 0,95. O ganho desse filtro é mostrado na Figura 9 onde pode-se observar que os valores de amplitude podem dobrar com o aumento de cada década na frequência. Ao mesmo tempo em altas frequências ocorre uma defasagem no sistema.

[9] Site: https://prezi.com/wqedronjakk7/d-filtro-de-preenfasis/

Figura 9- Filtro pré-ênfase ganho e fase no Python



Fonte: Autor

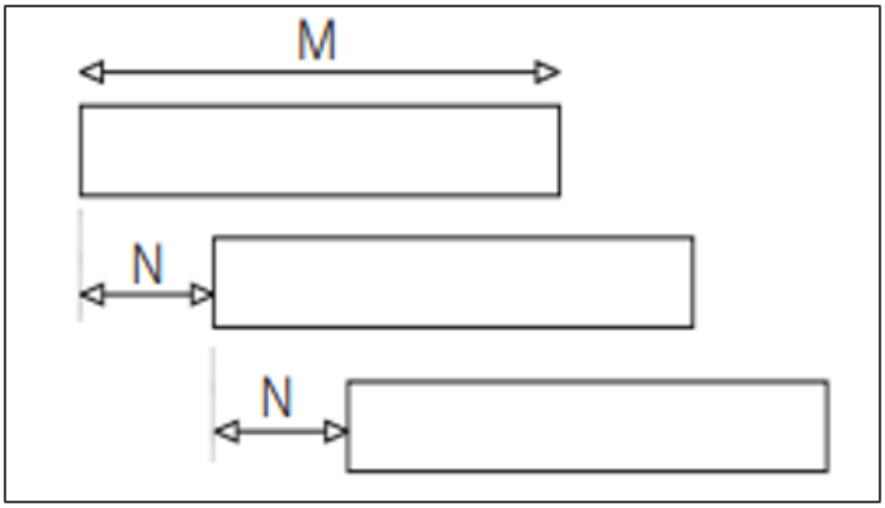
3.4 Janelamento e segmentação

Campbell (1997) descreve que os parâmetros do sinal de voz podem ser considerados periódicos e invariantes quando tratados em curtos intervalos de tempo na ordem de 10 a 30 ms. Desta forma, considera-se o sinal como periódico e estacionário, para o desenvolvimento deste trabalho.

Na prática, o sinal de voz é dividido em frames de tamanho fixo, obtidos nos intervalos de tempo dentro das locuções, contendo características presentes na fala em curtos espaços de tempo. Durante a obtenção destes frames surgem descontinuidades bruscas na fala resultando em perda de informação útil contida nestes intervalos. Com o intuito de suavizar este problema deve-se sempre fazer com que os quadros adjacentes compartilhem um determinado número de amostras. Assim, um quadro de comprimento M terá (M – N) amostras sobrepostas ao quadro posterior (CARDOSO, 2009), como ilustrado na Figura 10. No trabalho aqui desenvolvido foi arbitrada uma sobreposição de 50% que atende

as necessidades do projeto sem aumentar demais o número de informações que podem ser irrelevantes , havendo o compartilhamento da metade das amostras de quadros adjacentes..

Figura 10- Segmentação do sinal de voz



Fonte: (Cardoso, 2009)

Com os quadros devidamente segmentados, deve-se tratar a suavização das bordas causadas pela variação brusca de amplitude presentes no início e termino de cada quadro. Este processo é obtido multiplicando o segmento de fala por uma janela, determinado pela Equação 3.12, atenuando as bordas do segmento e conservando a parte central do segmento da voz.

Na Tabela 3 tem-se os diversos tipos de janelas existentes e aplicáveis. No algoritmo desenvolvido neste trabalho é utilizada a janela de *Hamming*, conforme a curva mostrada na Figura 11, que produz uma suavização considerável nas bordas de cada frame.

$$y(n) = x(n) * w(n) \qquad (3.12)$$

Como exemplo de janelamento tem-se na Figura 12 um segmento do sinal de fala, correspondente a palavra acelerar, sem janelamento e na Figura 13 tem-se o segmento após o janelamento de *Hamming*, resultando da convolução de um frame com a janela proposta. Observa-se na Figura 13 um sinal no domínio do tempo mais suave com relação ao segmento básico da Figura 12.

Conclui-se que as informações apresentadas nesta seção constituem uma base para o desenvolvimento proposto neste trabalho, concernente ao processamento do sinal da fala.

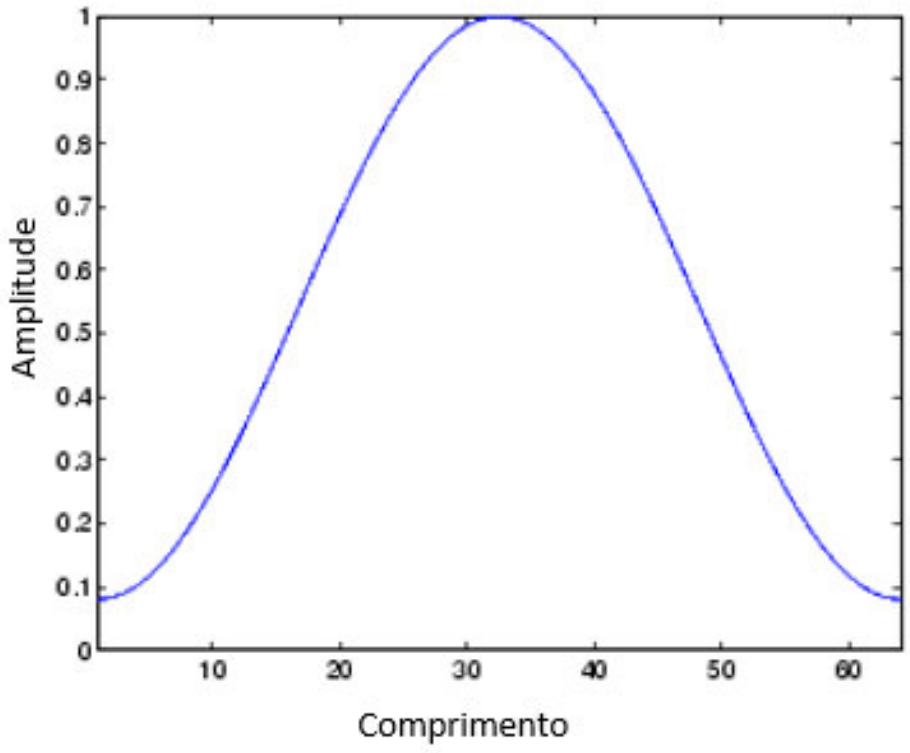
Através da ferramenta ***signal.hamming*** pode-se obter uma janela semelhante à da Figura 10.

Tabela 3- Equações de janelas existentes

JANELA	***EQUAÇÃO***
Retangular	$w[n] = \begin{cases} 1, & 0 \leq n \leq M \\ 0, & outros \end{cases}$
Bartlett	$w[n] = 1 - \frac{2\left\lvert n - \frac{M}{2} \right\rvert}{M}$
Blackman	$w[n] = 0{,}42 - 0{,}5\cos\left(\frac{2\pi n}{M}\right) + 0.08\cos(\frac{2\pi n}{M})$
Hamming	$w[n] = 0{,}54 - 0{,}46\cos\left(\frac{2\pi n}{M}\right)$
Hanning	$w[n] = 0{,}5 - 0{,}5\cos\left(\frac{2\pi n}{M}\right)$
Kaiser	$\mathrm{w[n]} = \frac{I_0\left\{\beta\sqrt{1-\left(\frac{n-\frac{M}{2}}{\frac{M^2}{2}}\right)}\right\}}{I_0\{\beta)}$

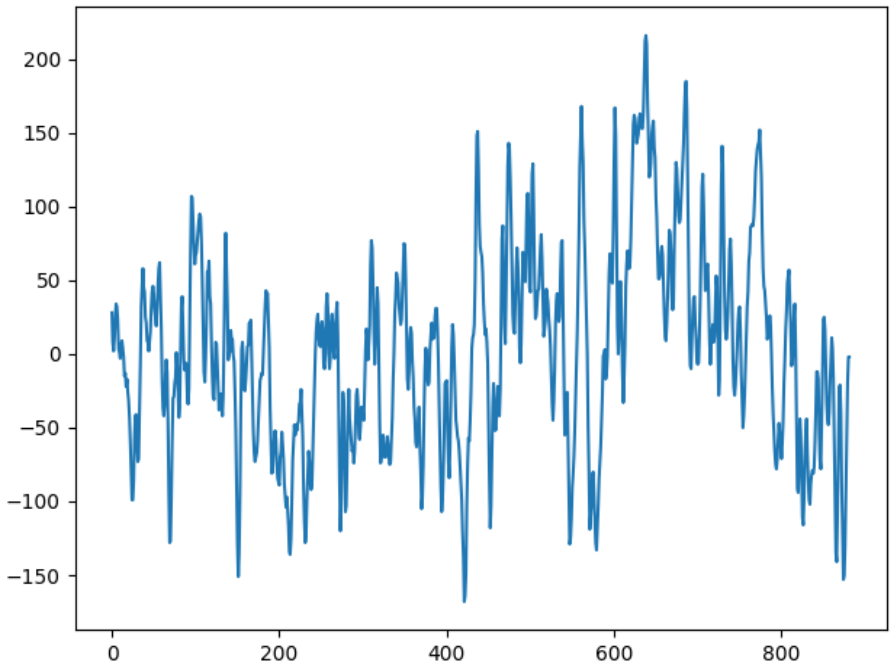
Fonte: Autor

Figura 11- Janela de *Hamming*



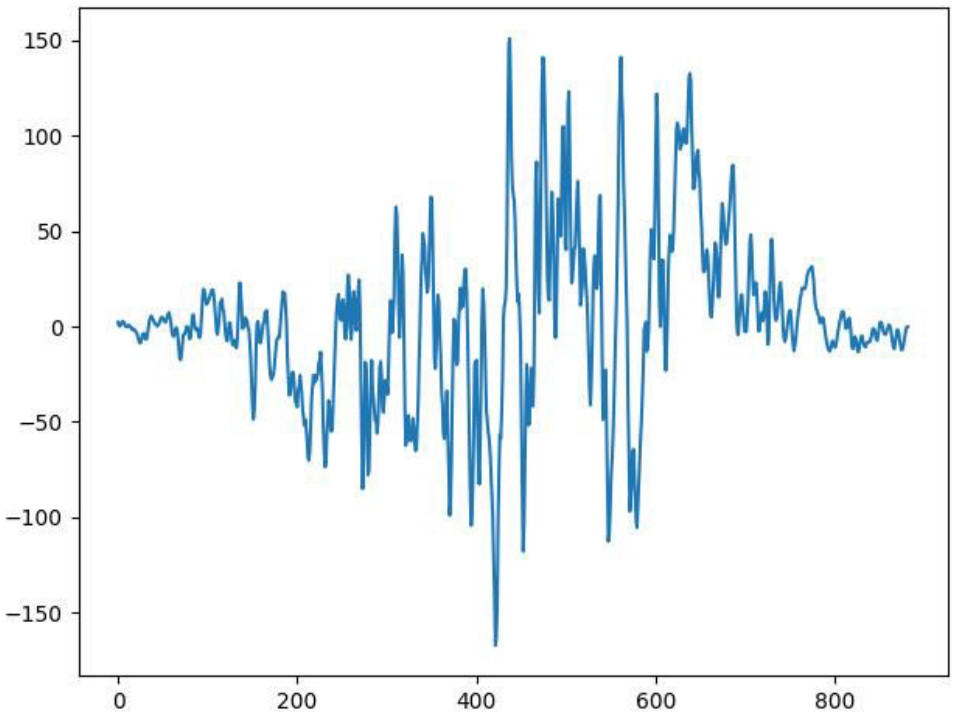
Fonte: Autor

Figura 12- 1° frame do segmento de voz “acelerar” sem janela



Fonte: Autor

Figura 13- 2° frame do segmento de voz “acelerar” com janela

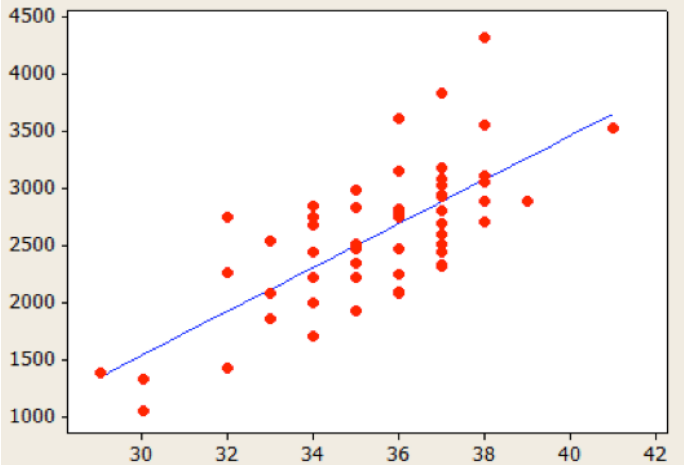


Fonte: Autor

4. PREDIÇÃO LINEAR

A predição linear (Figura 14) consiste em determinar coeficientes de valores futuros de um sinal a partir de amostras recentes e passadas deste sinal combinados na mesma entrada. Existe diversos métodos de obtenção de coeficientes LPC que podem ser usados em várias aplicações, como por exemplo no processamento de áudio (CARDOSO, 2009). Neste trabalho é utilizado o algoritmo de LAVISON (SEIFFERT e JIN YUN, 2014), para obtenção destes coeficientes, o qual pode ser implementado com relativa facilidade e eficiência quando associado a biblioteca scikit.talkbox[10] do Python.

Figura 14- Representação gráfica da Predição linear



Fonte:https://pt.wikibooks.org/wiki/Inicia%C3%A7%C3%A3o_%C3%A0_Pesquisa_Cient%C3%ADfica_em_Sa%C3%BAde/_Teste_de_correla%C3%A7%C3%A3o_linear_simples

4.1 Algoritmo de Levinson-Durbin

Neste trabalho é utilizado o algoritmo de Levinson-Durbin para obter os coeficientes de LPC, minimizando o erro médio quadrático (CLARKE e DISNEY, 1979). O algoritmo de Levinson, desenvolvido em 1947 (SEIFFERT e JIN YUN,2014) resolve o problema da inversão de uma matriz de Toeplitz (CLARKE e DISNEY, 1979), que soluciona o problema da matriz autocorrelação no caso da predição linear. Posteriormente, foi aplicado na estimação de coeficientes do modelo AR por Durbin (FERNANDES e FERREIRA, 2012). A Equação 4.1 pode ser usada para determinar os coeficientes de predição linear, com intuito de se obter o erro mínimo quadrático.

$$e[n] = x[n] - \sum_{k=1}^{p} a_k x[n-k] \qquad (4.1)$$

[10] Site: https://pypi.org/project/scikits.talkbox/

Por outro lado, erro quadrático médio para uma janela de tamanho N é determinado pela Equação 4.2.

$$e = \sum_{n=0}^{N-1} e^2[n] = \sum_{n=0}^{N-1} (x[n] - \sum_{k=1}^{p} a_k x[n-k])^2 \qquad (4.2)$$

O erro mínimo pode ser obtido derivando-se a Equação 4.5 e igualando o resultado a zero, conforme apresentado na Equação 4.3.

$$\frac{dE}{da_i} = \frac{\partial E}{\partial e} \frac{\partial e}{\partial a_k} = 2 \sum_{n=0}^{N-1} (x[n] - \sum_{k=1}^{p} a_k x[n-k]) x[n-i] = 0 \qquad (4.3)$$

Nessa igualdade, resulta a Equação 4.4.

$$\sum_{n=0}^{N-1} x[n]x[n-i] = \sum_{k=1}^{p} a_k \sum_{n=0}^{N-1} x[n-k]x[n-i] \quad para\ i = 1,2 \ldots . p \quad (4.4)$$

Além disso, a covariância de curto prazo $\phi(\mathrm{i}, k)$ de s_n é dada pela Equação 4.5.

$$\phi(i,k) = \sum_{m} x[m-i]x[m-k] \qquad 1 \leq i,k \leq p \quad (4.5)$$

Assim, defini-se o mínimo erro quadrático que permite definir uma matriz covariância, conforme apresentado na Equação 4.6.

$$\hat{E}_n = \sum_{m} x^2[m] - \sum_{k=1}^{p} \hat{a}_k \sum_{m} x[m]\, x[m-k] \quad (4.6)$$

A equação pode ser resumida quando se leva em consideração uma janela de tamanho N, resultando na Equação 4.7.

$$\hat{E}_n = \sum_{m=0}^{N-1+p} e_n^2\,(m) \quad (4.7)$$

Se ainda desconsidera-se pesos associados a cada segmento tem-se como resultado a Equação 4.8.

$$E_n = \sum_{m=0}^{N-1} e_n^2\,(m) \quad (4.8)$$

Finalmente define-se a matriz covariância que deve ser aplicada de forma a gerar os coeficientes de predição, conforme a Equação 4.9.

$$\phi(i,k) = \sum_{m=-i}^{N-i-1} x[m]x[m+i-k] \qquad 1 \le i \le p,\ \ 0 \le k \le p \quad (4.9)$$

4.1.1 Aplicação do Algoritmo Lavinson-Durbin

Uma análise mais aprofundada do desenvolvimento da Equação 4.9 nos permite dizer que a matriz $\phi(\mathrm{i}, k)$, resulta na matriz de autocorrelação R_{ss}, como mostrada na Equação 4.10.

$$\Phi = \begin{bmatrix} R(0) & R(1) & \dots & R(p-1) \\ R(1) & R(0) & \dots & R(p-2) \\ \vdots & \vdots & \dots & \vdots \\ R(p-1) & R(p-2) & \dots & R(0) \end{bmatrix} \quad (4.10)$$

Esta matriz apresenta os elementos da diagonal principal constantes caracterizando uma simetria de Toeplitz. Além disso, apresenta simetria conforme 4.10. Neste tipo de matriz obtém-se sua resolução recursivamente conforme algoritmo de Lavinson-Durbin descrito nas Equações 4.11 e 4.12.

$$\begin{cases} E^0 = R(0) \\ L_i = \dfrac{\left\{R(i) - \sum_{j=1}^{i-1} a_j^{i-1} R(|i-j|)\right\}}{E^{(i-1)}} \ para\ 1 \le i \le p \end{cases} \quad (4.11)$$

$$\begin{cases} a_i^i = L_i \\ a_j^i = a_j^{i-1} - L_i a_{i-j}^{i-1} \quad para\ j = 1, 2, \dots, i-1 \\ E^{(i)} = (1 - L_i^2) E^{(i-1)} \end{cases} \quad (4.12)$$

Analisando os valores obtidos verifica-se que eles correspondem aos coeficientes de predição linear de um segmento de fala, ou seja, um coeficiente c_k, definido na Equação 4.13.

$$c_k = a_k^p \qquad 1 \le k \le p \quad (4.13)$$

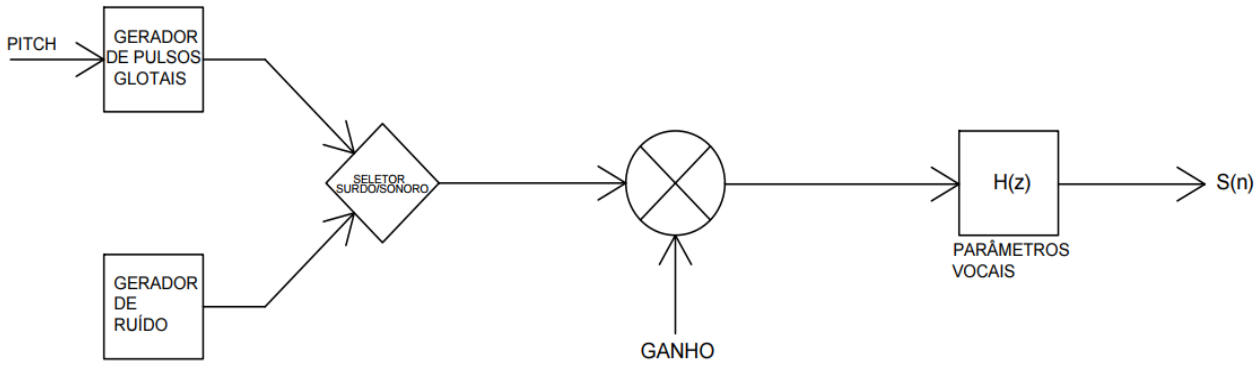
4.2 Modelo do Aparelho fonador e técnicas de predição

Pode-se modelar o aparelho fonador conforme apresentado na Figura 15. Inicialmente são gerados pulsos correspondentes aos estímulos glotais, com determinado período de *pitch*, relacionados a produção das vogais, como também um sinal de um ruído, relacionado a produção das consoantes. Os dois sinais são somados e realizado um ajuste de ganho na intensidade da parte do sinal sonoro (G). O sinal resultante é aplicado a um filtro de síntese para ajuste dos parâmetros vocais. Este modelo da Figura 15 representa matematicamente o aparelho fonador humano onde o *pitch* pode ser obtido fazendo a autocorrelação do sinal, e o filtro de síntese modelado através dos coeficientes LPC (CARDOSO, 2009).

Após a obtenção dos coeficientes de predição linear é possível determinar a função de transferência do modelo, dada pela Equação 4.14.

$$H(z) = \frac{S(z)}{U(z)} \qquad (4.14)$$

Figura 15- Modelo do aparelho fonador



Fonte: Autor

O desenvolvimento matemático da Equação 4.1 resulta na Equação 4.15.

$$H(z) = \frac{G}{1 - \sum_{i=1}^{n} a(i) z^{-1}} \qquad (4.15)$$

Que no domínio do tempo resulta na Equação 4.16.

$$x[n] = \sum_{k=1}^{p} a_k x[n-k] + e[n] \qquad (4.16)$$

onde n é o número de coeficientes empregados e G considerado 1.

Na prática, quando se aplica o *pitch* de uma pessoa ao modelo representado tem-se como resultado o contorno do espectro no domínio da frequência da locução. Neste caso o ganho G deve ser previamente definido para cada locutor.

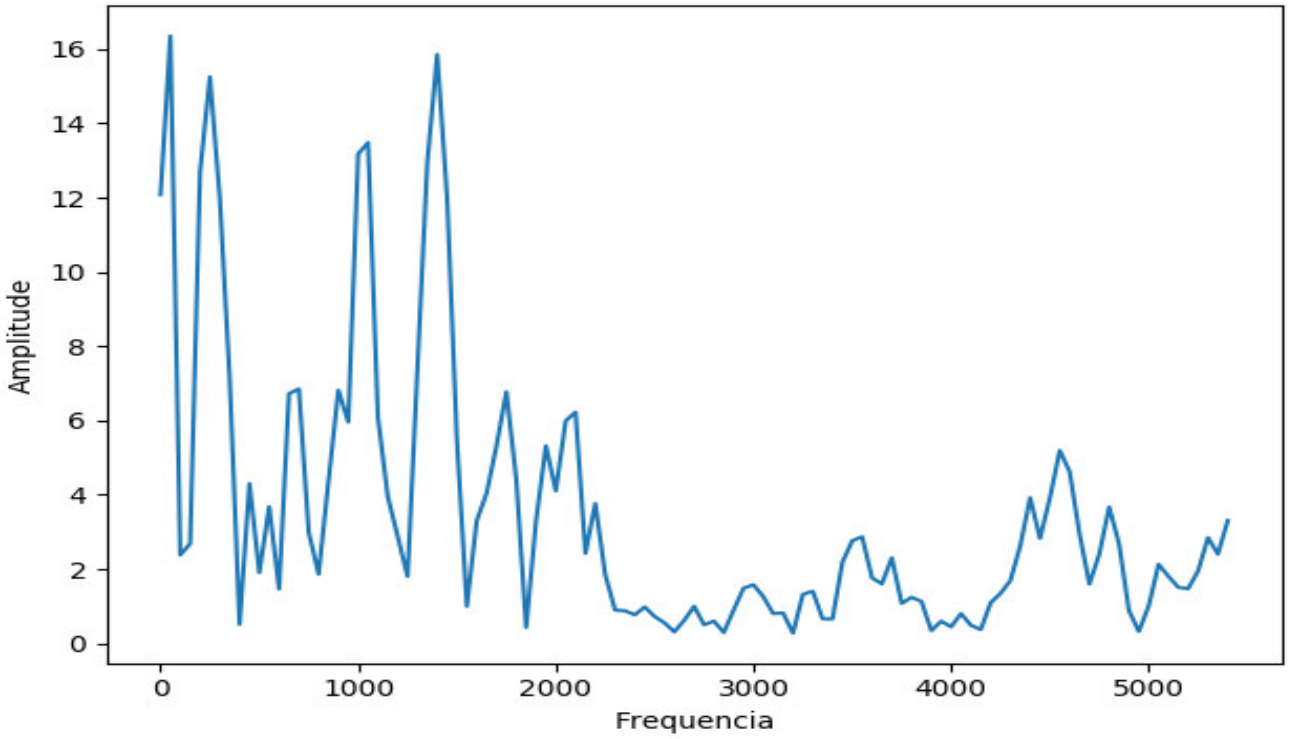
4.3 Codificação da fala por predição linear

O processo de codificação da fala é bastante utilizado no ramo das telecomunicações para fazer a redução dos dados de voz, já que é impraticável transmitir na integra dados de voz em seu tamanho original. Para isso, são utilizadas técnicas de processamento de sinais e de codificação da fala (ALCAIM e OLIVEIRA,2011). No processo de transmissão da fala deve-se transmitir um som inteligível, bem como as características presente na voz do locutor. Outro fator importante é a velocidade de se processar e tratar um sinal de voz, já que na maioria dos casos a comunicação deve ser feita em tempo real.

Neste trabalho é apresentada a técnica de codificação de fala por predição linear, com intuito de reduzir a quantidade de dados no sinal de voz, preservando ao máximo as características relevantes para se compreender a locução.

O uso da técnica LPC para codificação da fala é possível conforme demonstrado na Equação 4.2, podendo-se utilizar os coeficientes de predição linear como parâmetros de uma função de transferência que ao aplicar-se um conjunto de pulsos glotais retorna o contorno espectral do sinal de voz, conforme mostrado na Figura 16. Analisando as Figuras 17, 18 e 19 observa-se que quando aumenta-se a quantidade de coeficientes melhora-se a resolução do espectro da fala, sendo possível obter o sinal, no domínio da frequência.

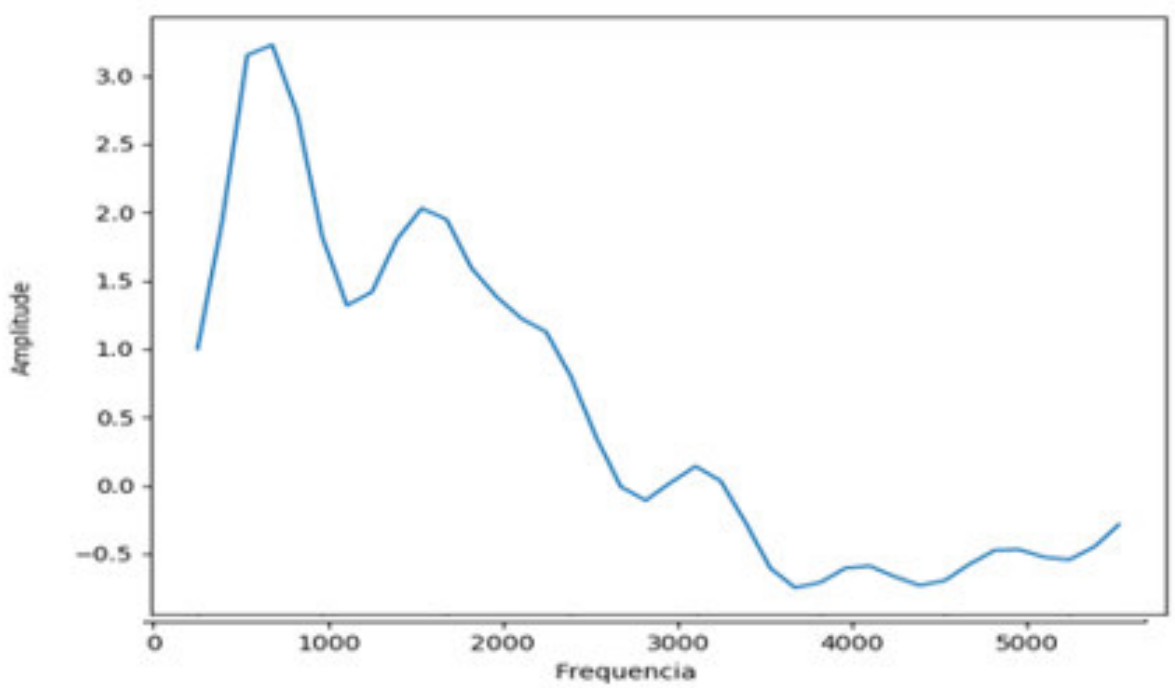
Figura 16- Espectro do sinal de voz no domínio da frequência



Fonte: Autor

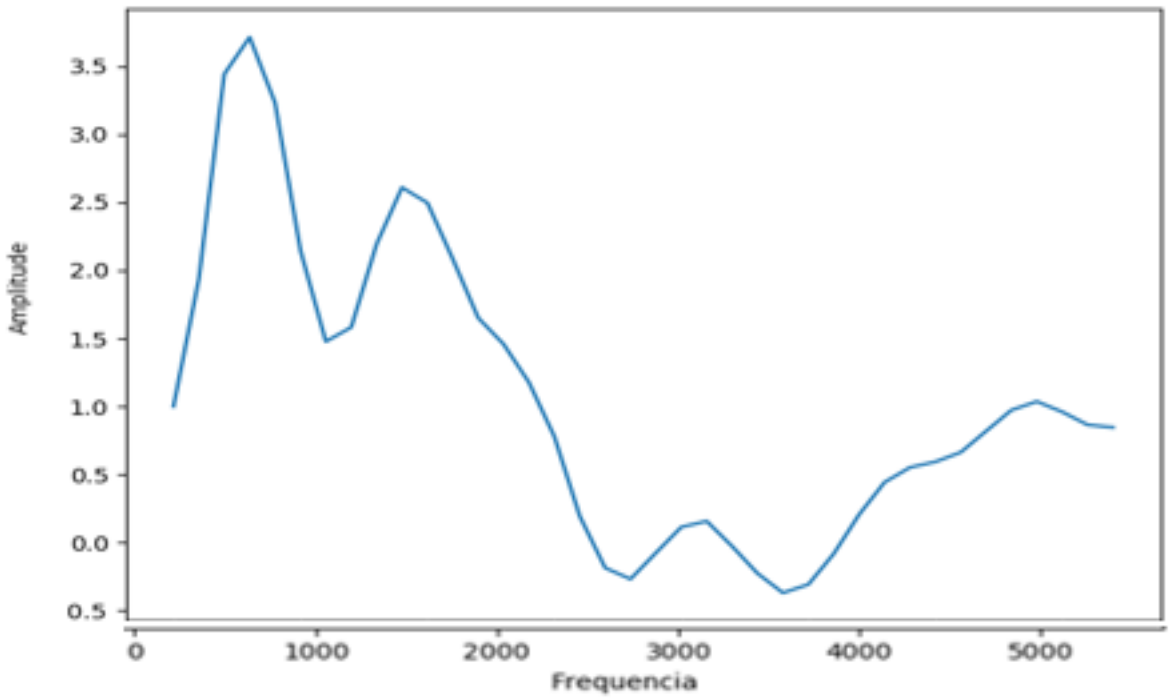
Aplicando-se FFT inversa, pode-se restituir o sinal de voz no domínio do tempo. É importante lembra que se comparar o sinal restituído como o sinal original, as amplitudes divergem. Isso ocorre porque o ganho G, da função de transferência H(z) é arbitrado como 1. Este ganho é uma propriedade inerente a cada locutor, ou seja, deve ser definido individualmente para cada locutor (CARDOSO, 2009).

Figura 17- Espectro do sinal de voz e do filtro H(z) com 10 coeficientes de predição linear



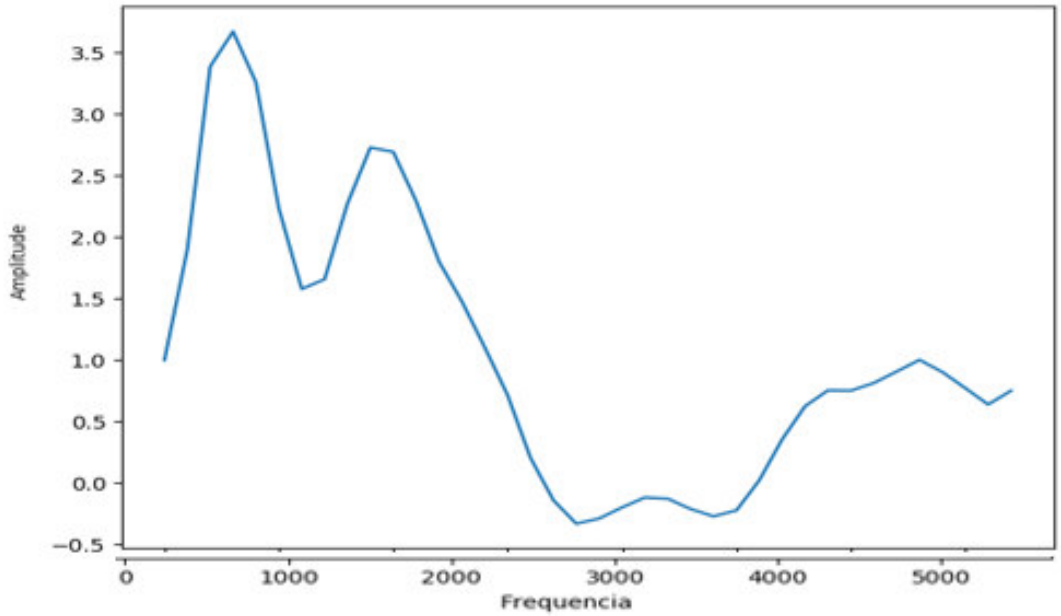
Fonte: Autor

Figura 18- Espectro do sinal de voz e do filtro H(z) com 20 coeficientes de predição linear



Fonte: Autor

Figura 19- Espectro do sinal de voz e do filtro H(z) com 40 coeficientes de predição linear



Fonte: Autor

Como o reconhecimento de fala desenvolvido neste trabalho é independente de locutor todos locutores são tratados com ganho unitário, (G = 1).

Evidencia-se assim a efetividade do uso de LPC na codificação e restituição do sinal de voz.

Verifica-se que a grande vantagem de se aplicar este procedimento às locuções treinadas ocorre na economia abrupta de processamento. Tanto na fase de treinamento quanto na fase utilização da rede neural. Este fato se deve porque as locuções são salvas em arquivo ".wav" que possuem uma frequência de amostragem de 44100 Hz. Logo para cada segundo de gravação existem 44100 pontos contendo dados do sinal. Quando se aplica a predição linear reduz-se cada segundo para 500 coeficientes. Deste modo haverá redução de 98,9% a quantidade de dados a ser processado pelo sistema sem que haja perda de informação útil ao reconhecimento de fala.

4.4 Determinação do número de coeficientes por segmento da fala

O sistema desenvolvido neste trabalho contém uma rede neural simples, de três camadas, descrita no Capítulo 5. Diferentemente de uma rede complexa como o caso das convulacionais (HAYKIN, 1998), este tipo de rede necessita receber dados que possuam sempre os mesmos tamanhos na camada de entrada, durante a fase de treinamento. Como a predição linear garante sempre a mesma quantidade de coeficientes em função do tempo, locuções com durações diferentes, terão número de coeficientes diferentes.

Em uma situação hipotética em que um frame tenha seus parâmetros fixados em 20 coeficientes LPC, com duração de 20ms e sobreposição aos frames adjacentes em 50%. As locuções terão características como mostradas na Tabela 4. Pode-se observa através dessa tabela que locuções de tamanho diferente geram uma discrepância muito grande no número de coeficientes. O que ocasiona um problema já que estes coeficientes alimentaram as redes neurais, que por sua vez devem possuir o número de entradas sempre fixas.

Tabela 4- Número de coeficientes LPC

Locução	**Duração média (s)**	**Número aproximado de coeficientes**
RÔBO	0.410	800
ROSELI	0.615	1200
FRENTE	0.483	960
ATRAS	0.374	720
ESQUERDA	0.592	1180
DIREITA	0.737	1460
ACELERAR	0.708	1400
PARAR	0.421	840

Fonte: Autor

Uma solução para contornar este problema, inerente as redes neurais simples, é construir a rede considerando a menor locução. Desta forma deveria-se ter 720 entradas na 1° camada da rede, já que está se adequaria ao tamanho da menor locução. Porém, ao fazer isso pode-se perder uma quantidade considerável de informação das demais locuções, como no caso da locução “acelerar” que perderia em média 50,7% da quantidade de informação útil.

Sendo assim, é adotada uma nova metodologia onde é fixada a sobreposição entre quadros, que permanecem em 50%, o tamanho das janelas em 20ms e o número de coeficientes que se deseja obter, que será escolhido como 800. Aplicando o seguinte processo:

1. Determinar o número de quadros N_q, usando a Equação 4.17.

$$N_q = \frac{t_l - t_j}{s * t_j} + 1 \qquad (4.17)$$

onde: t_l é o tamanho da locução, t_j=*20ms*-é o tamanho da janela e s=0,5 a sobreposição

Substituindo o valor de s, resulta na Equação 4.18.

$$N_q = \frac{2 * t_l}{t_j} - 1 \qquad (4.18)$$

Como só é possível ter um número inteiro de quadros, o valor de N_q deve ser arredondado para baixo.

2. Determinar o número de coeficientes por quadro N_c usando a Equação 4.19.

$$N_c = \frac{800}{N_q} \qquad (4.19)$$

Novamente deve-se ter arredonda N_c para baixo pois só se pode ter número inteiros de coeficientes.

O arredondamento para baixo pode causar, no máximo a perda de um quadro em uma determinada locução. A Tabela 5 resume o resultado encontrado.

Tabela 5- Número de coeficientes gerado dinamicamente

Locução	**Duração média (s)**	**Número de quadros Nc**	**Número de coeficiente por quadro Nq**	**Número de coeficiente Aproximado a 800**
RÔBO	0.410	40	20	800
ROSELI	0.615	60	13	780
FRENTE	0.483	47	17	799
ATRAS	0.374	36	22	792
ESQUERDA	0.592	58	13	754
DIREITA	0.737	72	11	792
ACELERAR	0.708	69	11	759
PARAR	0.421	41	19	779

Fonte: Autor

Nota-se na Tabela 5 que a variação da quantidade de coeficientes é muito baixa, já que em locuções muito grandes o sistema compensa reduzindo o número de coeficientes por quadro; em locuções pequenas aumenta-se o número de coeficientes por quadro de forma que a quantidade de coeficientes não caia muito a ponto de perder características da fala.

Deste modo, pode-se escolher uma locução de menor tamanho, em termos de coeficientes, e perder menos que 6% de informação.

5. ANÁLISE COMPARATIVA POR REDES NEURAIS

Uma rede neural artificial é composta por núcleos de processamento semelhantes aos neurônios do cérebro humano, que são conectadas por canais de comunicação associados a um peso pré-definido e ajustável (MCCULLOCH e PITTS,1943). As unidades de neurônios processam informações locais, com seus dados locais recebidos em suas conexões, e lança a outras unidades de processamento. Cada núcleo é capaz de tomar decisões simplistas, mas que quando combinado a interações de toda a rede formam um sistema complexo e inteligente. Os pesos das conexões entre elementos são ajustados seguindo alguma regra definida pelo usuário, onde deve-se também realizar um treinamento com grande quantidade de dados da condição que se deseja fazer o sistema aprender (KOLEN, 1994). Em outras palavras o sistema aprende vendo exemplos de situações verídicas.

5.1 Processos de Aprendizado

Uma rede neural artificial tem como fundamental característica a capacidade de aprender em um processo iterativo de ajustes de pesos entre as suas conexões, denominado treinamento, este processo muitas vezes possui alto custo computacional além de necessitar de um ajuste fino de parâmetros que permitam condições de convergência para rede. Esses ajustes muitas vezes são imprevisíveis e difíceis de se definir em algum processo matemático, sendo encontrado em muitos casos por tentativa e erro. O aprendizado completo ocorre quando a rede neural artificial consegue generalizada para todas as classes de um determinado problema um modelo baseado nas amostras que ela analisou e tomou como exemplo (HAYKIN, 1994).

Em outras palavras quando a rede consegue compreender um dado com qual ela nunca interagiu, mas já analisou algo semelhante, ela tornou-se capaz de generalizar uma classe. Este processo de aprendizagem deve seguir um algoritmo denominado algoritmo de aprendizagem composto por um conjunto de regras bem definidas para a generalização do sistema (HAYKIN, 1994).

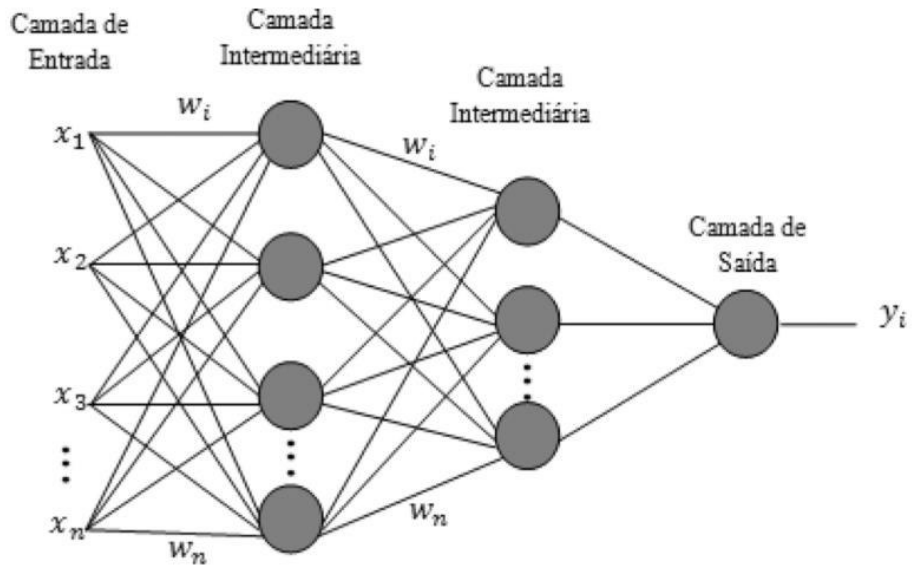
Neste trabalho é desenvolvida uma estrutura de rede simples de três camadas, treinada dentro de um aprendizado supervisionado e utilizando um meio externo que indica à mesma o objetivo desejado para um conjunto de dados aplicado. Além disso os dados serão apresentados em ciclos *batch* onde todos os exemplos de um conjunto de treinamento são apresentados à rede, porem apenas uma correção nos pesos é feita em cada ciclo.

5.2 Redes Perceptron de múltiplas camadas

Uma Rede Perceptron de múltiplas camadas tem uma arquitetura que consiste de um conjunto de unidades de neurônios formando uma camada de entrada, outro formando as camadas intermediárias

(denominadas escondidas) e uma camada de saída onde o dado de entrada é propagado sequencalmente, da entrada para a saída, conforme mostrado na Figura 20. O aprendizado é supervisionado e utiliza o algoritmo de *backpropagation*, para minimizar o erro de saída (ROSENBLATT, 1957). Este algoritmo é baseado numa regra de aprendizagem que reduz o erro na saída à medida que realiza seu treinamento.

Figura 20- MLP de múltiplas camadas



Fonte: http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0102-77862016000100024

O processo de retro-propagação do erro, e consequente aprendizagem, é dividido em duas fases denominadas, *feedforward* e *backpropagation*. Na fase *feedforward*, fixa-se os pesos dos neurônios e os vetores de dados são aplicados às unidades de entrada, onde seu efeito se propaga pela rede gerando uma saída. Na *backpropagation* para corrigir o erro, os pesos são ajustados de acordo com um critério de correção, onde o sinal de erro é propagado da saída para a entrada ajustando os pesos de cada camada de modo que a distância entre a resposta do sistema e a desejada seja reduzida.

5.3 Lotes de treinamento (*BATCH*)

Tipicamente, em algoritmos que utilizam retro-propagação o aprendizado é resultado de inúmeras aplicações de amostras referente ao conjunto de dados, à rede perceptron de múltiplas camadas ou MLP, (MINSKY e PAPERT, 1969). A aplicação de todo o conjunto de treinamento durante o processo de aprendizagem é denominada época. Durante o processo de aprendizagem repete-se épocas de modo a alcançar-se o mínimo erro quadrático médio do conjunto de treinamento. Durante a aplicação das épocas é comum e recomendável fazer com que a ordem de apresentação das amostras seja feita aleatória. Pois à

variabilidade tende a fazer com que a atualização espacial de pesos se aproxime de um sistema estocástico ao longo dos ciclos de treinamento. Sendo assim as redes ficam menos tendenciosas a uma determinada classe de amostras. Como dito anteriormente foi utilizado aprendizado em lotes. Sendo assim a atualização dos pesos só é feita após a aplicação de todas as amostras que constituem uma época.

A aprendizagem em lote fornece uma estimativa mais precisa do vetor gradiente ao custo de maior espaço de armazenamento. Para o sistema proposto entende-se que este método seja mais viável.

5.4 Critérios de Parada

O processo de minimização em uma rede neural não tem convergência garantida ou qualquer tipo de critério de parada. Logo, utiliza-se como critério de parada interromper o treinamento após um número fixo de épocas. Este processo não é muito recomendável pois não há garantias de convergência ou qualquer resultado minimamente satisfatório (MINSKY, 1952).

Um critério mais aceito e definível, é considerar a possibilidade de existência de mínimos locais. Onde o vetor de parâmetros θ^* que denota um ponto de mínimo, seja ele local ou global. Uma condição necessária para que θ^* seja um mínimo, é que o vetor gradiente $\nabla J(\theta)$ derivado em primeira ordem em relação ao vetor de pesos θ seja igual a zero. Desta forma pode-se dizer que o algoritmo de retro propagação convergiu quando a norma euclidiana da estimativa do vetor gradiente ($\|\nabla J(\theta)\|$) atingiu um valor próximo a zero (MINSKY, 1952).

Neste trabalho será utilizado como critério de parada um número fixo de iterações visto que este pode ser definido através de um processo de tentativa e erro. Enquanto que derivar o gradiente igualando a zero pode ser tido, em muitos casos, com tempo de treinamento muito longo, além de ser computacionalmente custoso já que requer cálculo de norma e derivada.

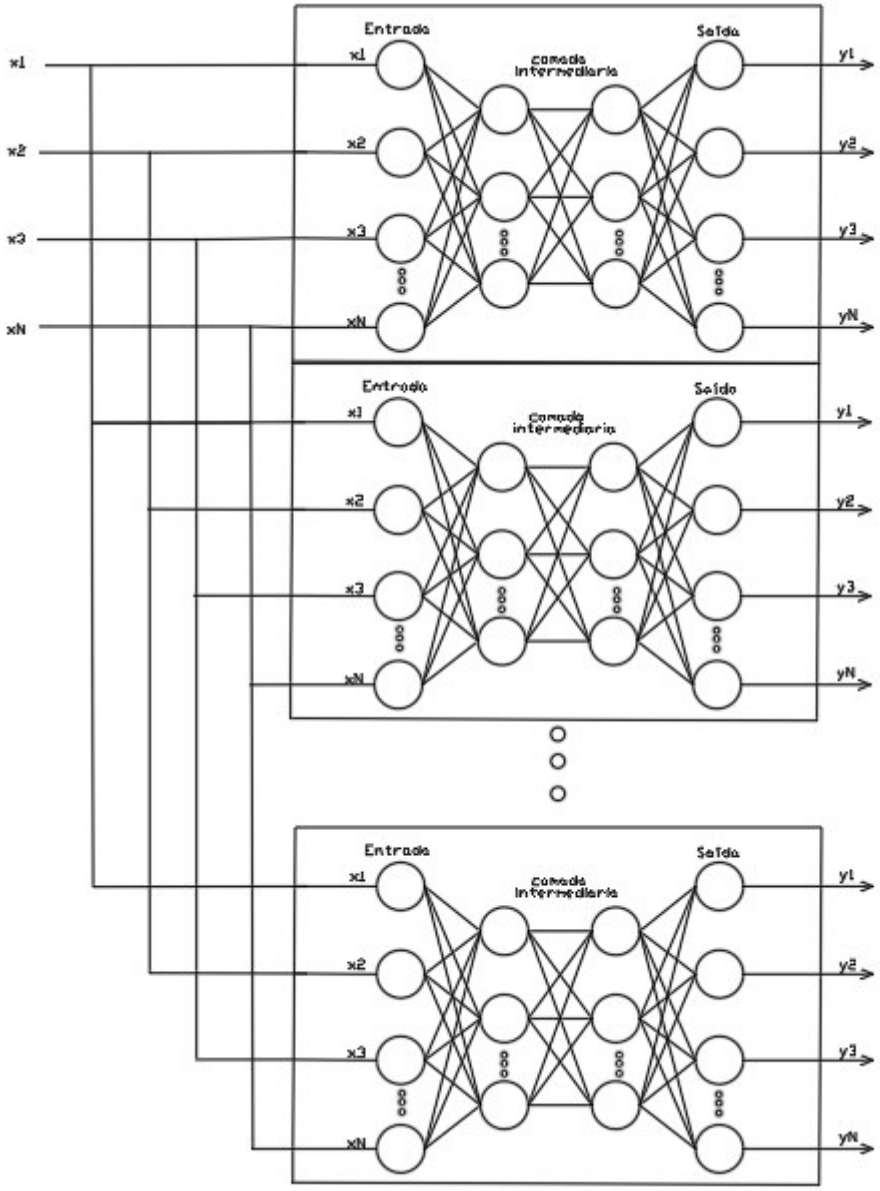
5.5 Modelo Desenvolvido

Dentro do esquema proposto é utilizado um sistema de rede neural como mostrado na Figura 21.

Neste modelo treina-se as redes de modo que cada uma se torne especialista em reconhecer um determinado tipo de locução. Em sistemas convencionais, geralmente com estruturas semelhantes a Figura 20, utiliza-se uma rede neural que seja capaz de generalizar um modelo que reconheça um grupo de locuções de interesse. Nestes sistemas, mesmo que se faça o uso de RNN ou MLP, é extremamente difícil conseguir gerar um modelo com taxas de acerto superior a 90%. Na proposta deste trabalho, baseada em (SHIGUIAKE, 2015), ao se treinar as redes individualmente para reconhecer um tipo especifico de locução, os modelos de cada rede serão altamente especializados em reconhecer um tipo especifico de fala ao mesmo tempo esses modelos ignoram as demais locuções. Deve-se iniciar a criação do sistema

treinando um modelo que seja capaz de reconhecer a fala com base em um banco de dados previamente definido.

Figura 21 - Modelo de rede neural proposto



Fonte: Autor

Desta forma é treinado individualmente o modelo da Figura 21, repetindo o processo sete vezes, uma para cada classe.

O reconhecimento de fala se dá de forma que as redes são estruturadas para reproduzir a própria entrada na sua saída e que os pesos da rede sejam atualizados, baseado no erro mínimo quadrático entre entrada e saída. Desta forma, as redes tentaram reproduzir as entradas nas saídas. A rede que errar menos neste processo será classificada como aquela do provável locutor.

A equação que define o critério de atualização dos pesos na rede se dará pela Equação 5.1 tendo como base na estrutura da Figura 20

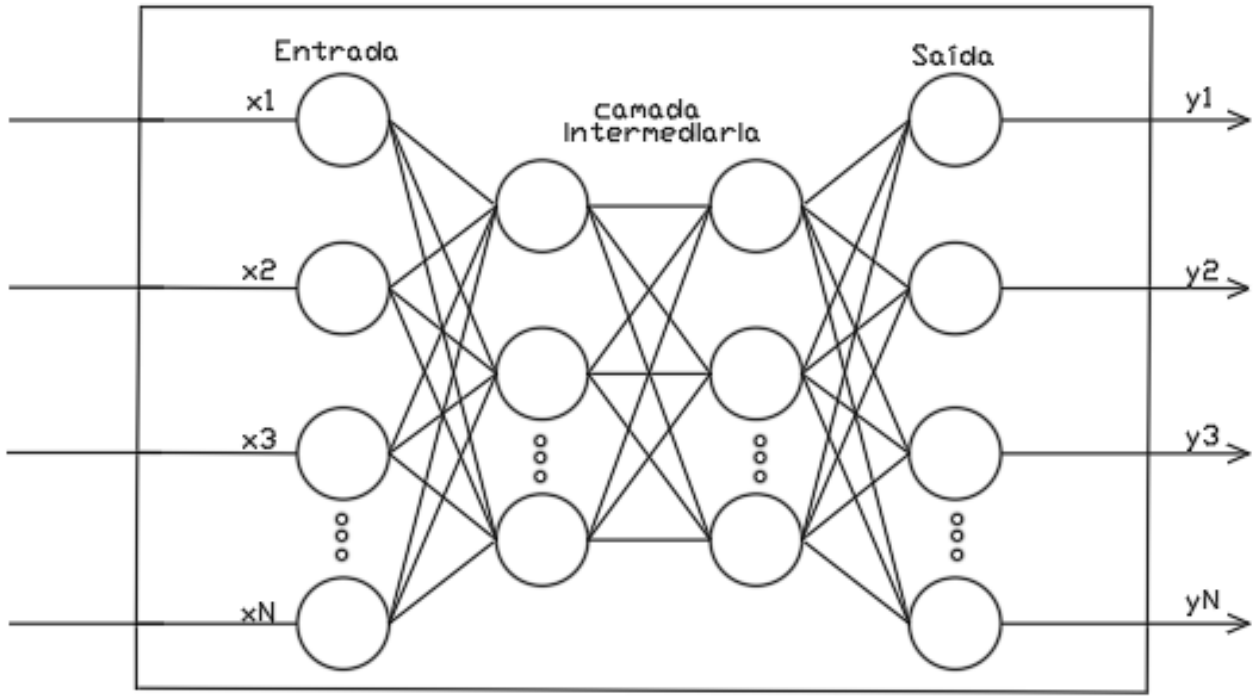
$$e = \frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2 \qquad (5.1)$$

onde n é o número de saídas.

Nesta condição é importante lembra que a rede não busca minimizar a *loss* (perda) com base em uma locução trajada, e sim minimizar o erro de saída após um determinado número de épocas.

Um ciclo de treinamento, para este caso, onde encontrou-se o ponto ideal, corresponde a 300000 épocas de treino de uma única rede usando um banco de 100 locuções onde 90 geram um banco de treino e 10 um banco de teste. Ao fim do ciclo gera-se um arquivo do tipo ".ckpt" onde contém os pesos que serão usados posteriormente para recriar as redes e empregá-las de fato. Este processo é mostrado na Figura 22.

Figura 22-Sub rede do modelo

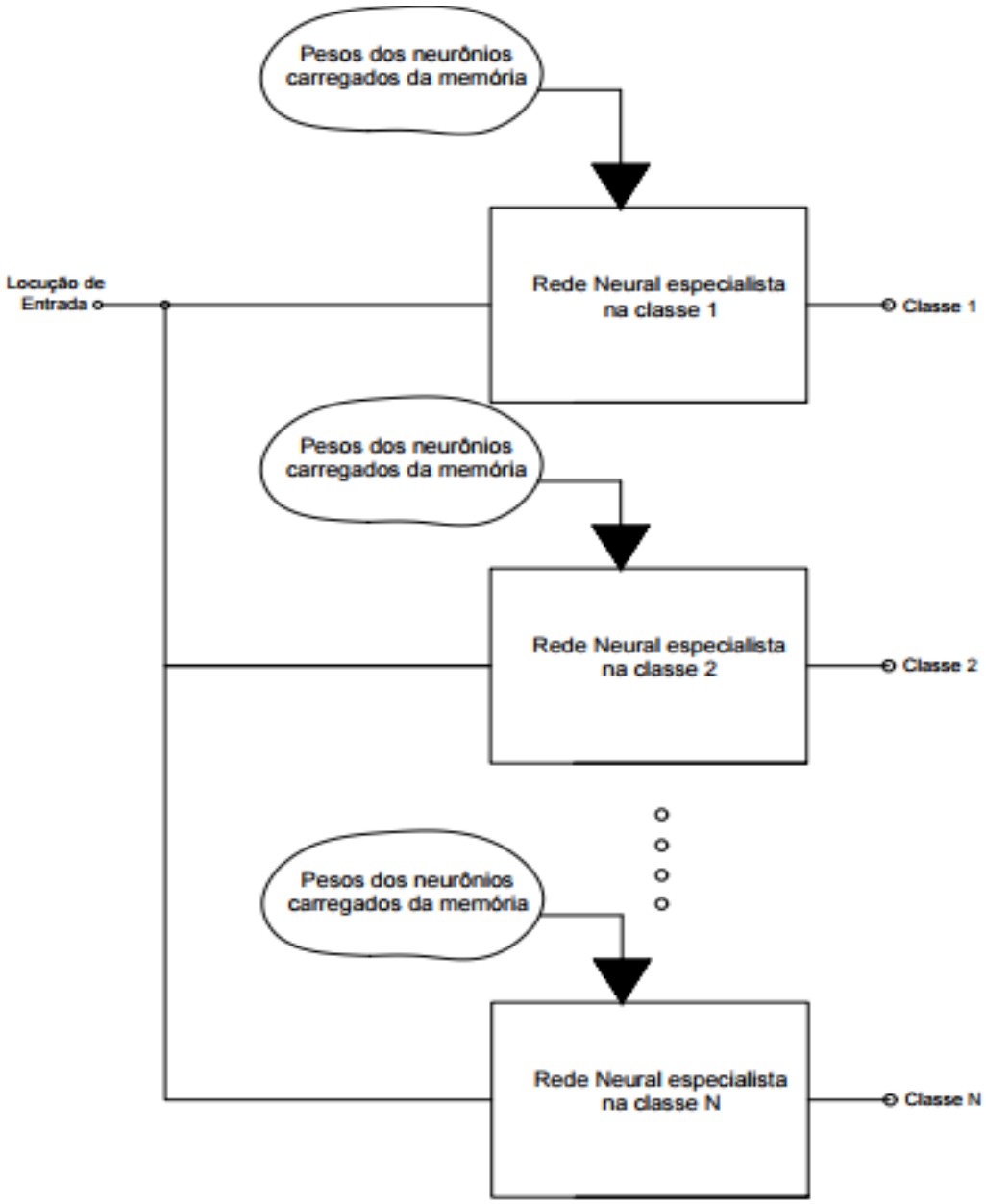


Fonte: Autor

Finalmente, para colocar o sistema em funcionamento é necessário recriar a rede de treinamento, com o diferencial que ao invés de treinar o modelo, inicia-se os valores de pesos gerados na fase de treinamento, salvos em memória. Em tese haverá a mesma rede criada ao término do treinamento em condições de uso. Deste modo ao se aplicar um dado de entrada no sistema, uma das redes reproduzirá

um erro mínimo, menor que as demais, de saída classificando a amostra. A Figura 23 apresenta o diagrama em blocos com o esquema completo utilizado neste trabalho.

Figura 23- Esquema completo de utilização da rede e carregamento dos dados



Fonte: Autor

6. DESENVOLVIMENTO DO *SOFTWARE* DE RECONHECIMENTO DE FALA

O reconhecimento de fala baseia-se em características inerentes a fala que são extraídas diretamente das locuções. A vantagem de se utilizar informações biométricas extraídas de uma locução deve-se a facilidade com que esta é capturada do ambiente e processada, não exigindo equipamentos ou custos elevados. A identificação se dá pela comparação de parâmetros extraídos da fala e armazenados em memória, com uma nova locução realizada em tempo real. O índice de acertos no sistema é influenciado por diversos fatores, sendo eles a qualidade do sinal, diferenças no canal de comunicação, qualidade do enunciado, entre outros. Neste trabalho é utilizada a técnica de comparação por rede neural, através da qual é obtida uma rede para cada classe de locução, permitindo que ao ser fornecida uma nova locução de teste ao sistema, seja possível verificar o modelo de fala mais correlacionado com as propriedades da locução inserida. Este modelo é então eleito pelo sistema de identificação como o mais correlacionado com as propriedades da fala, sendo assim a mais provável palavra dita por um locutor aleatório.

Todo o sistema é criado exclusivamente a partir de uma linguagem de programação PYTHON, mais precisamente em sua versão de interpretador 2.7, onde é utilizado o ambiente de trabalho PyCharm 2017.2.3[11] , em virtude da maior comodidade e facilidade na aquisição de bibliotecas especificas.

6.1 Descrição robô ROSELI

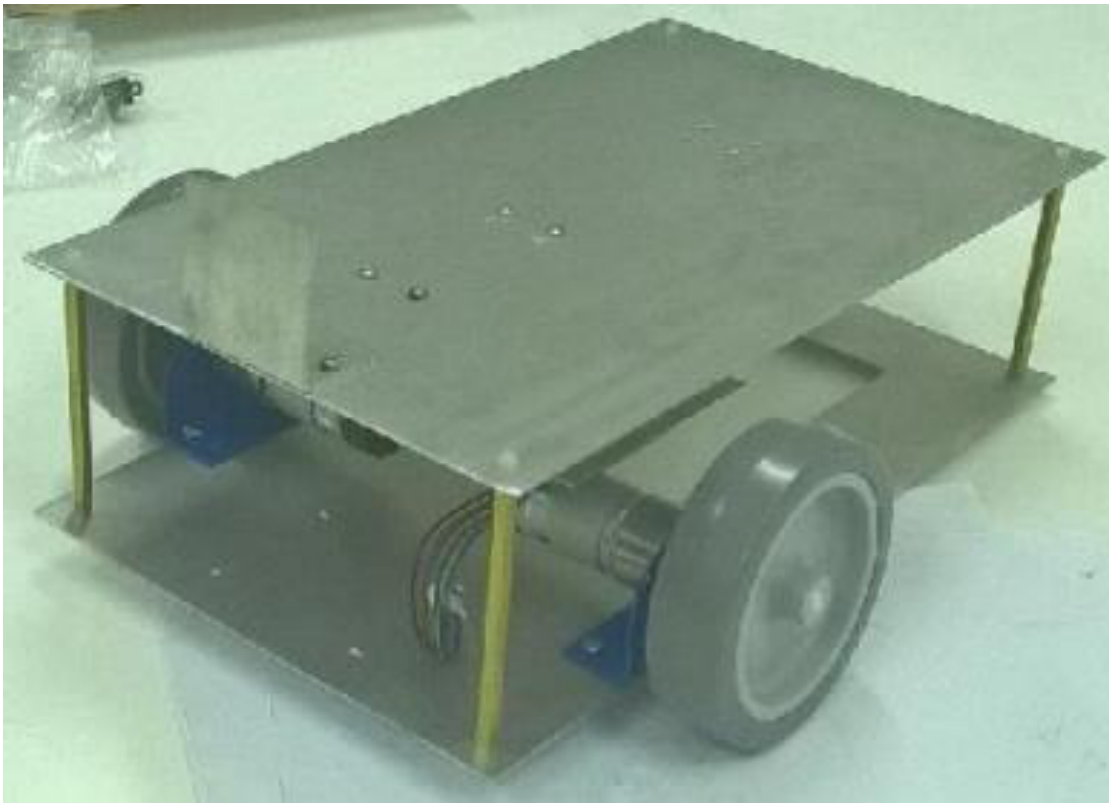
Quanto a parte física do robô, a estrutura mecânica foi construída em um estabelecimento especializado na cidade de São Luís e a parte eletrônica e computacional no laboratório (Laboratório de Robótica Móvel e Comunicação Sem Fio). Quanto ao *software*, foi desenvolvido e descrito em (RIBEIRO, 2017).

6.1.1 Estrutura mecânica

O robô foi construído com chapas de alumínio, totalizando três lâminas de 1/16”, duas para a estrutura principal (base e superior) e uma para a sustentação dos sensores e placas (fixados na parte de baixo da lâmina superior). Com a estrutura confeccionadas, montou-se um sistema de propulsão fixando-se motores DC por parafusos. As duas placas de alumínio foram separadas através de quatro parafuso espaçador de cobre de 20 cm. A estrutura mecânica completa é mostrada na Figura 24.

[11] Site: https://www.jetbrains.com/pycharm/

Figura 24- Estrutura mecânica completa



Fonte: (RIBEIRO, 2017)

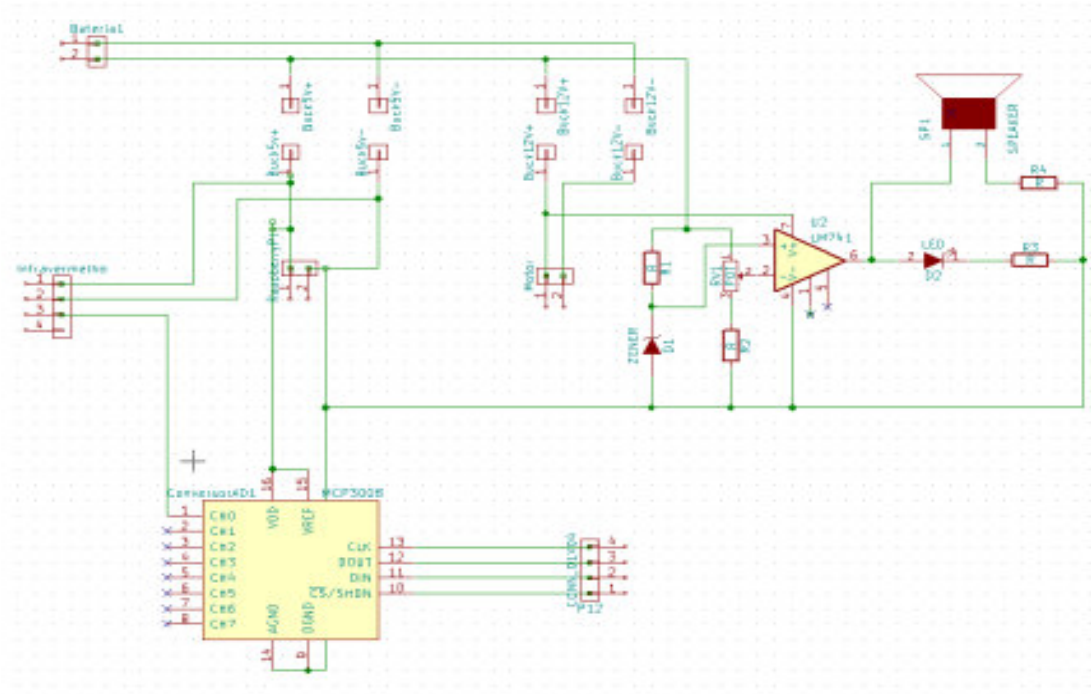
6.1.2. Placas e componentes de interfaceamento

A Figura 25 mostra o detalhamento do interfaceamento proposto ao robô com o intuito de garantir que a placa de interface seja capaz de cumprir sua função de forma satisfatória e os testes em protoboard.

Para testar o circuito de interface I2C para comunicação com a placa MD25, foram feitos diversos experimentos enviando comandos de leitura e escrita para movimentação dos motores e leitura dos *encoders*. Os resultados obtidos nestes testes isolados foram sempre satisfatórios (RIBEIRO, 2017). Na parte superior esquerda da placa (Figura 25), são localizados três conectores: o primeiro é usado para a alimentação geral do circuito, podendo variar entre 12 e 18 VDC. O segundo e o terceiro conectores são para fornecer saída regulada em 12 V e 5 V, para a alimentação dos motores e Raspberry Pi 3B, respectivamente. Na região localizada à direita dos conectores são colocados dois conversores *step-down*, que convertem o valor de entrada para os valores desejados de saída mencionados acima.

Abaixo dos conversores, existe um circuito comparador de tensão, para verificar o nível de tensão da bateria, o qual utiliza um amplificador operacional LM741, uma malha resistiva, um diodo *zenner* e um LED em conjunto com um buzzer que servem como forma de sinalização.

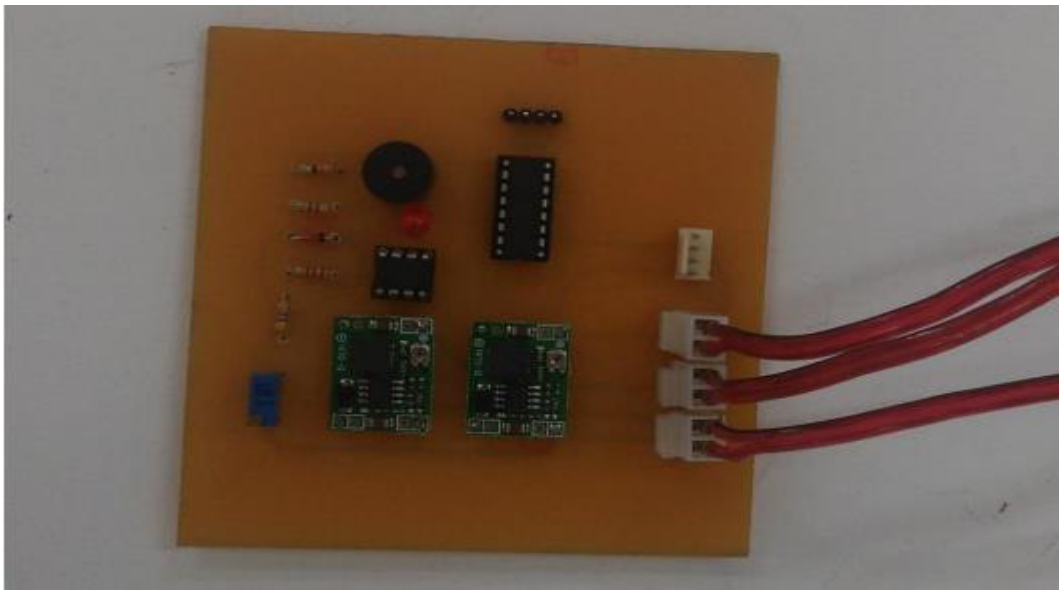
Figura 25- Diagrama elétrico



Fonte: (RIBEIRO ,2017)

À esquerda do circuito comparador de tensão, existe um conversor A/D, que tem a finalidade de fazer a comunicação entre o sensor infravermelho e o Raspberry Pi 3 (RIBEIRO, 2017). Como descrito em Hart (2010), o nível de tensão de saída para o circuito proposto é definido pela frequência de chaveamento ajustada através de um *trimpot* ligado à um oscilador.

Figura 26 - Placa de circuito impressa com componentes soldados



Fonte: (RIBEIRO, 2018)

6.1.3. Visão geral dos módulos de *software*

Para melhor organização dos *software* desenvolvidos para o controle do robô, houve a sua divisão em módulos, separados por funções que executam em arquivos diferentes. O primeiro módulo contém as funções de movimento do sistema de propulsão do robô. A comunicação pelo protocolo I2C é transparente neste código para o usuário, já que são disponibilizadas funções primitivas que contém todo este padrão embutido. Foram criadas funções para mover o robô em translação e rotação (em torno do próprio eixo), além de realizar sua parada, a leitura dos encoders nos dois tipos de movimentos e realizar o *reset* da contagem dos encoders. Essa última função é necessária para garantir que os motores atuassem na distância pretendida e, para implementar essa funcionalidade, sempre antes do início de um movimento (rotação ou translação), eram feitos os resets dos encoders (RIBEIRO, 2017).

O segundo módulo contém as funções para tratar os dados do sensor infravermelho. Devido a placa Raspberry Pi 3 não possuir pinos para leitura de valores analógicos, a leitura do sensor infravermelho é feita através do barramento SPI no acesso ao conversor A/D. Também foram desenvolvidas funções primitivas para esta finalidade, facilitando a programação e o entendimento por parte do usuário em futuras fases de manutenção. Nesse módulo existem funções para abertura e fechamento do protocolo SPI, além da função para medição de distância com o sensor infravermelho, que retorna o valor medido em centímetros (RIBEIRO, 2017).

Os demais componentes de *software* desenvolvidos por (RIBEIRO, 2017) tratam do processamento de imagem e localização no espaço, fugindo do escopo deste trabalho. Será tratado aqui dos módulos de movimentação necessários para gerar respostas ao comando de voz.

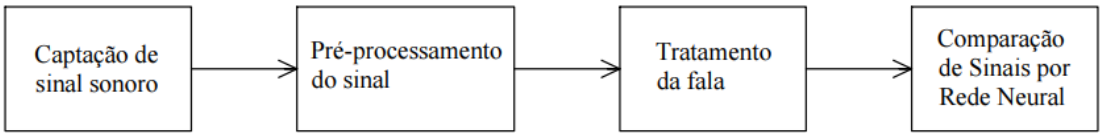
6.2 Por que utilizar Python?

Dentre as linguagens atuais de programação a Python é a que mais cresceu nos últimos anos, pois permite que trabalhos complexos possam ser realizados com poucas linhas de comandos e de modo muito simples para compreensão futura. Além disso, possui uma vasta comunidade mundial que permite a criação de fóruns e comunidades via *internet* que interagem com o intuito de facilitar o desenvolvimento de diversas aplicações e bibliotecas especificas para a linguagem. Praticamente todo tipo de operação matemática, método analítico, operadores de sistema, interfaceamento e até mesmo modelos para processamento de voz já existem em bibliotecas do Python, o que aumenta exponencialmente a eficiência de qualquer programador, o rendimento do algoritmo e a facilidade de identificar problemas em algoritmos complexos (*Debuggin*).

6.3 Módulos implementado no Python

O algoritmo de reconhecimento de fala é tratado em módulos, que é um artifício de programação, onde o programa pode ser desenvolvido e compilado em partes, aumentando a capacidade da programação. Os módulos desenvolvidos seguem a estrutura mostrada na Figura 27 e se dividem em: Captação de sinal sonoro, Pré-processamento do sinal, Tratamento da fala e Comparação de Sinais por Rede Neural.

Figura 27- Módulos desenvolvidos



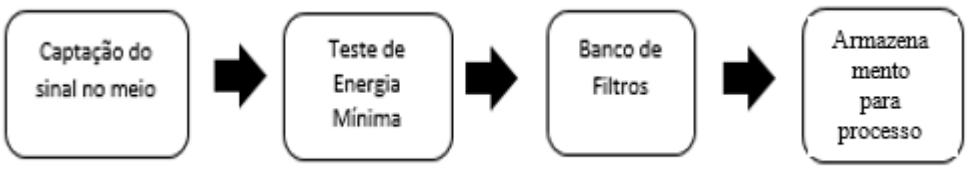
Fonte: Autor

É importante salientar que os módulos e seus componentes divergem em alguns aspectos da estrutura mostrada na Figura 4 do Capitulo 3. Isso ocorre, pois em alguns casos é mais viável empregar alguns tratamentos dentro de algum processo, como no caso da captação de sinal sonoro que emprega algumas rotinas de pré-processamento, economizando memória e tempo.

6.4 Captação de sinal sonoro

A estrutura do algoritmo de captação do sinal sonoro é mostrada em Figura 28. Esta estrutura constitui o primeiro ciclo do programa onde o sistema encontra-se em stand-by até que seja perturbado por algum tipo de som que atinja o critério de energia mínima. Caso o critério seja atingido o programa inicia a gravação dos próximos 0.75 segundos e os armazena em arquivos “.wav”. O intervalo de tempo de gravação é definido de modo a gravar a maior locução possível, sem que ocorra excessos.

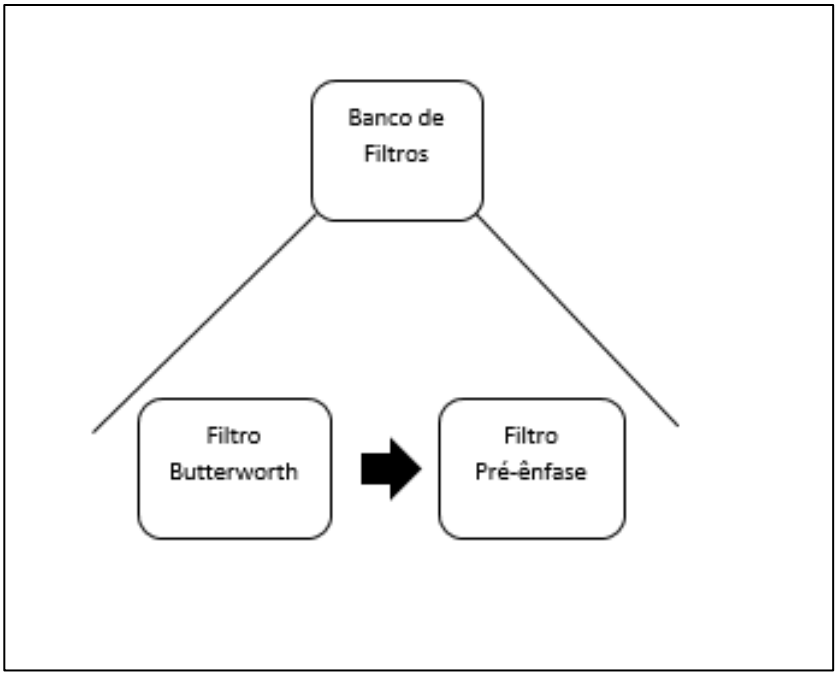
Figura 28- Algoritmo em blocos de captação sonora



Fonte: Autor

Após obtenção de arquivo “.wav” o sistema atravessa um conjunto de filtros, conforme mostrado na Figura 29 e descritos anteriormente no Capitulo 3.

Figura 29- Filtro *Butterworth* e pré-ênfase



Fonte: Autor

Em termos de programação a energia mínima é definida como o somatório de múltiplos pontos de uma janela sonora. Caso esse valor ultrapasse um limiar inicia-se a gravação. Caso contrário mante-se a condição de stand-by. O algoritmo apresentado no Quadro 1 descreve este processo.

Quadro 1 – Algoritmo de energia mínima

```
soma=0
for n in range(0,len(janela),1):
    if janela[n]>0:
        soma= soma+janela[n]
    else: soma=soma+(-janela[n])
if soma > limiar_sonoro:
    return 1
else:
    return 0
```

Fonte: Autor

Quanto ao processo de filtragem, é possível implementar qualquer tipo de filtro com uso de bibliotecas como scipy.signal[12] onde primeiramente define-se as frequências de corte do filtro, caso seja necessário, para se obter os parâmetros da função de transferência do filtro. Utiliza-se o comando:

b, a **= butter_bandpass (**data**,** lowcut, highcut, fs, order=ordem**)**

onde 'a' são os coeficientes do numerador, 'b' são os coeficientes do denominador, 'fs' é a frequência de amostragem e 'data' é a saída dos dados filtrados;

A função butter_bandpass foi criada através de outras funções definidas em scipy.signal.

Com os parâmetros definidos, aplica-se a função .lfilter capaz de gerar filtros com a seguinte estrutura:

Y = **lfilter**(b, a, data)

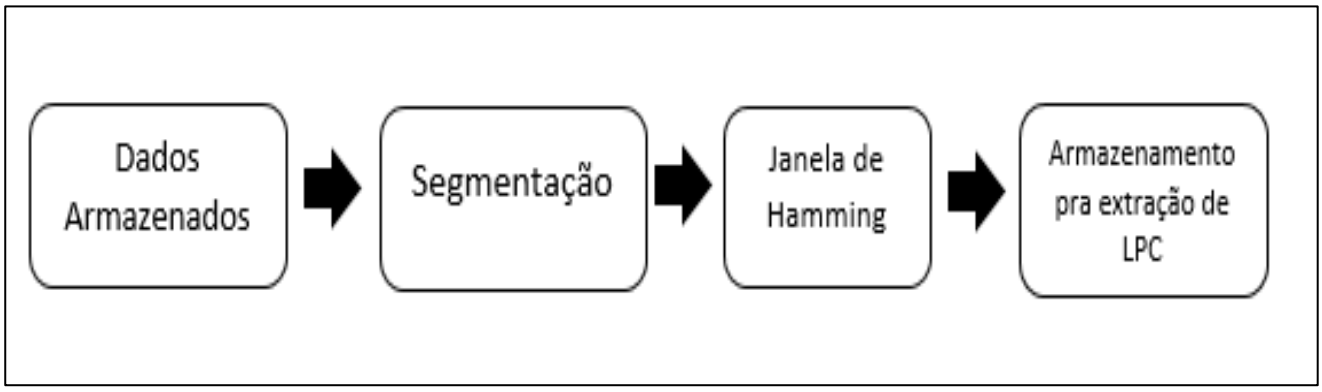
onde 'Y' é a saída com os dados filtrados.

Finalmente gravam-se os dados em uma memória para que os mesmos passem por um pré-processamento.

6.5 Pré-processamento do sinal

Dentre as atividades cabíveis ao pré-processamento, conforme mostrado na Figura 30, no que diz respeito a filtragem, o mesmo foi empregado durante a captação sonora. Fato decorrente da economia de recursos e aumento da eficiência. Acrescenta-se, assim, o processo de segmentação para este modulo.

Figura 30- Estrutura do pré-processamento



Fonte: Autor

O algoritmo apresentado no Quadro 2 é relacionado ao pré-processamento, onde os dados de uma captação linear no tempo serão convertidos em matrizes de segmentos sobrepostos em 50% e expostos a uma janela de *Hamming*.

[12] Site: https://docs.scipy.org/doc/scipy/reference/signal.html

Quadro 2 – Algoritmo de sobreposição entre frames

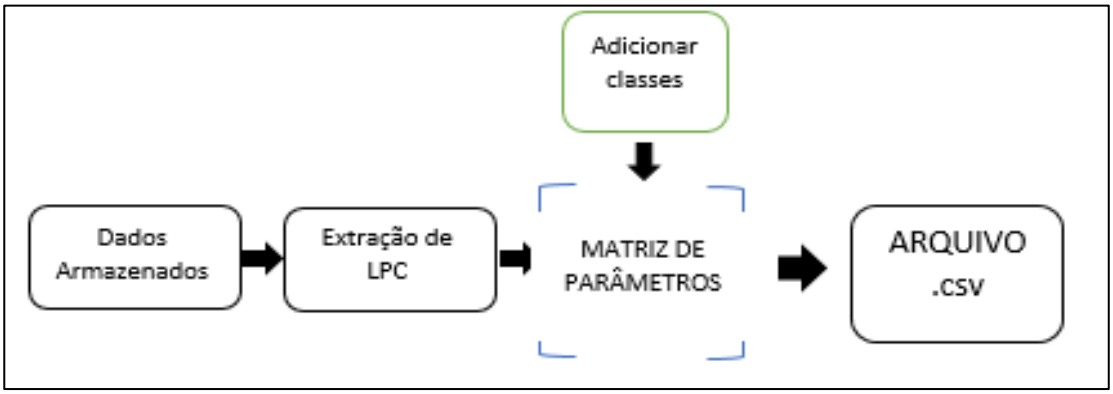
```
tamanho_da_amostra = valor_definido;
numero_frames= valor_definido;
janela = np.hamming(tamanho_da_amostra)

#deslocamento de 50%
deslocamento = int(0.5 * tamanho_da_amostra);
lista_frames_ham = []
inicio = 0
for i in range(0, numero_frames, 1):

lista_frames_ham.append(vet_espectro[inicio:tamanho_da_amostra +
inicio:1] * janela[0:tamanho_da_amostra:1])
inicio = deslocamento * (i + 1);
break;
```

Fonte: Autor

6.6 Estrutura do processo de organização da fala

A estrutura do processo de organização da fala usada neste trabalho é apresentada na Figura 31. Neste caso, é necessário que os dados de fala sejam organizados de modo que possam ser interpretados e tratados pela rede neural e reduzidos através de predição linear, armazenados em arquivos com extensão ".csv". A estrutura consiste em criar um banco de dados para que se possa extrair deles parâmetros LPC que serão utilizados na fase de treinamento quando se definir as locuções desse banco tarjas que definam suas classes. Ao fim desses processos pode-se gerar um arquivo ".csv" que será utilizado para treinar a rede neural.

Figura 31- Estrutura do processo de organização da fala



Fonte: Autor

Durante o processo de organização da fala deve-se associar os coeficientes LPC de uma locução a uma classe numérica, isto se faz necessário para que a rede entenda de modo numérico a situação na qual ela está exposta. Esta associação é feita como mostrado na Tabela 6.

Tabela 6 – Classes associadas a locuções de comando ao robô

Classe	**Locução**
1	“Robô”
2	“Roseli”
3	“Atrás”
4	“Parar”
5	“Esquerda”
6	“Direita”
7	“Acelerar”

Fonte: Autor

O algoritmo apresentado no Quadro 6.3 demonstra a aplicação da predição linear e sua associação as locuções.

Ao fim do processo de organização da fala terá-se um arquivo do tipo “.csv” que pode ser convertido posteriormente em uma classe dados interpretáveis por redes neurais. Este arquivo é estruturado em tabelas conforme modelo mostrado na Tabela 7.

Tabela 7 – Modelo de arquivo “.csv”

P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	….	PX
classe	coef1	coef2	coef3	coef4	coef5	coef6	coef7	coef8	coef9	coef10	coef11	coef12	…..	coefX
1	1	0.345	0,234	0.154	0.656	0.865	0.864	0.378	0.231	0.713	0.145	0.021	…..	0.637
2	1	0.642	0,233	0.189	0.972	0.872	0.812	0.638	0.532	0.216	0.049	0.121	…..	0.652
3	1	0785	0,635	0.523	0.152	0.164	0.252	0.532	0.142	0.014	0.623	0.763	…..	0.156
4	1	0.034	0,521	0.981	0.421	0.619	0.732	0.476	0.345	0.129	0.155	0.921	…..	0.523
5	1	0.962	0,378	0.664	0.942	0.006	0.829	0.468	0.341	0.621	0.237	0.324	…..	0.832
6	1	0.264	0,721	0.231	0.627	0.951	0.076	0.642	0.823	0.745	0.867	0.543	…..	0.378
7	1	0.194	0,089	0.374	0.056	0.632	0.619	0.843	0.654	0.945	0.459	0.253	…..	0.912

Fonte: Autor

6.7 Comparação de dados por rede neural

Através do uso de *frameworks* e bibliotecas do Python pode-se criar, com facilidade, redes neurais artificiais. Dentre os *frameworks* disponíveis destaca-se o PYTORCH[13]. Esta poderosa ferramenta nos permite criar redes neurais com maior precisão, bem como realizar todo processo de treinamento e uso da estrutura.

O processo comparação de dados, conforme o sistema definido neste trabalho, ocorre da seguinte forma:

- **Treinamento** – Neste estágio a rede é treinada utilizando o banco de dados previamente gerado nas etapas anteriores, conforme mostrado no diagrama de blocos da Figura 32. Nesta fase é desenvolvida uma rede neural para cada classe. Cada rede será responsável por uma classe de forma que ao empregar-se um dado de entrada ela minimize de forma e única e precisa a perda (*loss*) para um determinado dado. Por fim, um arquivo contendo todos os pesos dos neurônios da rede é armazenado em uma memória.
- **Utilização** – Neste estágio são recriadas todas as redes de uma só vez, assentando os pesos dos neurônios com os dados salvos previamente em arquivo, conforme diagrama de blocos apresentado na Figura 33. Para se utilizar a rede basta que o dado que se deseja reconhecer seja empregado na entrada do sistema. Neste caso, a rede que está associada a uma classe e que reproduzir menor erro (*loss*) tem a sua saída classificada como provável locução. Os resultados obtidos no treinamento e a taxa de acerto serão mostradas no Capitulo 7.

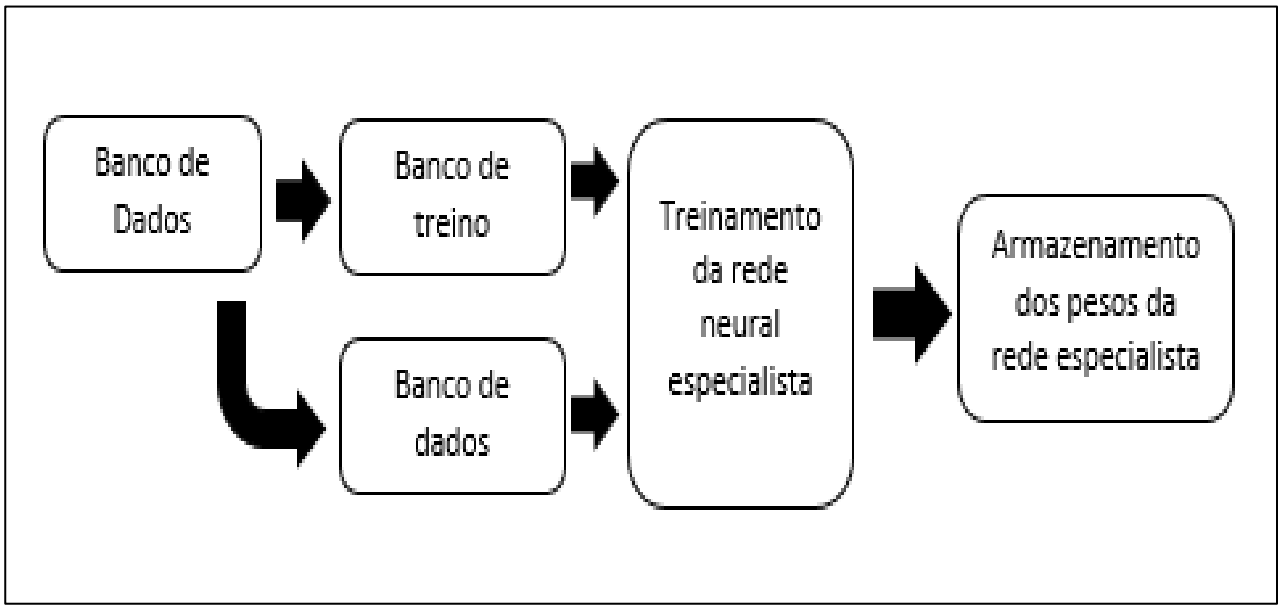
6.8 Associação do comando de voz com comando de robô

Ao se classificar um dado de entrada como pertencente a uma determinada classe, deve-se de alguma forma passar essa informação ao robô, objeto de controle de forma que o mesmo possa compreendê-lo.

Este processo pode ser feito de forma simples associando comando de voz a comandos já existentes nas bibliotecas padrões do sistema do robô Roseli, descrito em (RIBEIRO, 2017). Esta associação em termos práticos é feita associando os protocolos de comunicação do dispositivo a locuções especificas. A informação é transmitida via Wi-fi, onde o robô funciona simultaneamente como um servidor aguardando ordens via fala. Como os comandos de voz estão associados a movimentação da máquina, deve-se também estabelecer parâmetros para as funções responsáveis por realizar esta movimentação, indicando direção, velocidade e sentido de giro. A extração de parâmetros LPC mostradas na Tabela 3 será comentados na Seção 7.2, quando do uso do sistema implementado nos testes com o robô.

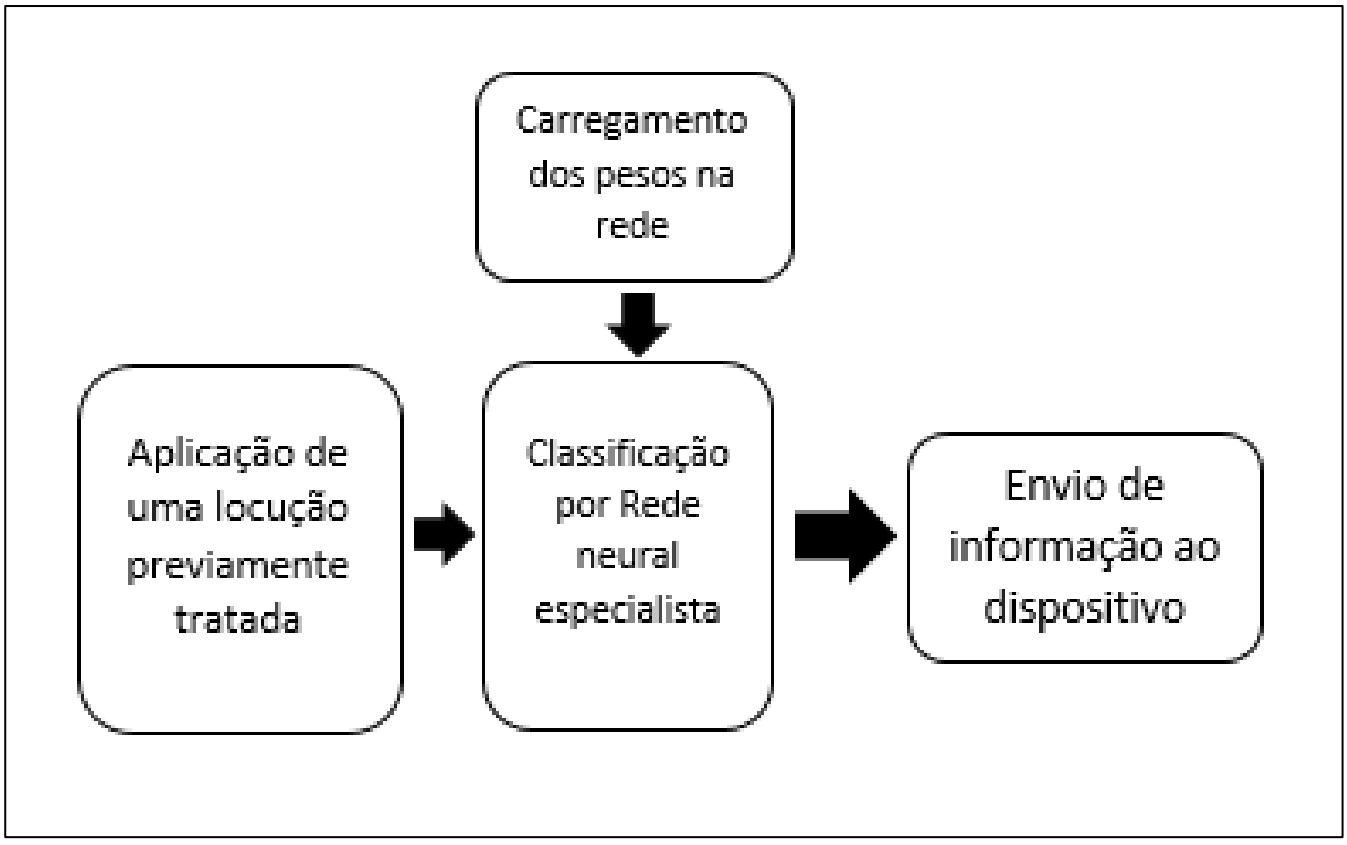
[13] Site: https://pytorch.org/

Figura 32- Modelo de treinamento da fala



Fonte: Autor

Figura 33- Modelo de utilização do reconhecimento da fala



Fonte: Autor

Quadro 3 – Extração de parâmetro LPC junto a classe

```
#associação das classes a locução
if (classe == 'robô'):
    num_classe = 1.
elif (classe == 'Roseli'):
    num_classe = 2.
elif (classe == 'Atrás'):
    num_classe = 3.
elif (classe == 'parar'):
    num_classe = 4.
elif (classe == 'esquerda'):
    num_classe = 5.
elif (classe == 'direita'):
    num_classe = 6.
elif (classe == 'acelerar'):
    num_classe = 7.
#inicia a gravação do arquivo csv
f = open('tabela.csv', 'a')
writer = csv.writer(f)

#extração de parâmetros LPC
for i in range(0,len(arquivo),1):
        lpc=get_lpc(arquivo[i])
    vet2=[]
    for i in range(0, len(lpc),1):
        for j in range(0,len(lpc[0]),1):
            vet2.append(lpc[i][j])
     #adiciona o numero na primeira posição correspondente a classe
     vet2.inserte(0,num_classe)
     writer.writerow(vet2)
  f.close()
```

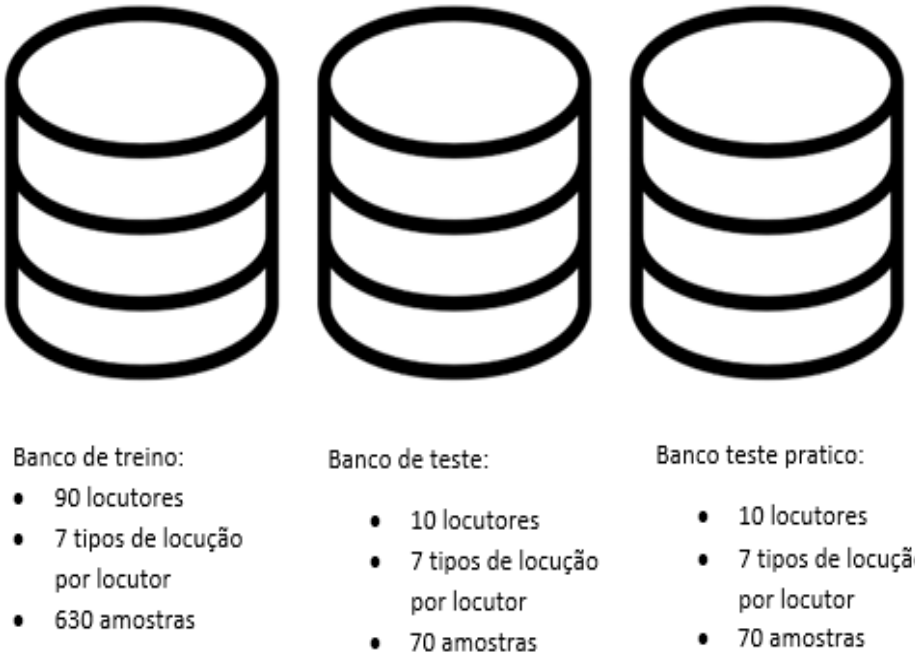
Fonte: Autor

7. EXPERIMENTO E RESULTADOS

Neste capitulo são apresentados os resultados obtidos usando-se o modelo de rede neural desenvolvido no reconhecimento de fala, e comparando-o como outros modelos de redes neurais. É destacada a RNN, como modelo comumente utilizado em trabalhos de reconhecimento de fala, pois atinge resultados em tornos de 90%, no que se trata de sua assertividade.

Será considerado um banco de dados inicial como mostrado na Figura 34, para treinamento e teste da rede.

Figura 34- Banco de dados utilizado



Fonte: Autor

7.1 Resultados obtidos com a rede neural

Na Tabela 8 são apresentados os parâmetros mais relevantes do processo de criação da rede neural, seus rendimentos e consequentemente seus maiores valores de assertividade. Em ambos os casos os experimentos foram realizados considerando-se o melhor ajuste de parâmetros que se obteve ao longo de 6 meses.

Analisando individualmente cada tipo de rede pode-se analisar, minuciosamente seus parâmetros , sendo que a sua alteração implica na assertividade do sistema. Ao mesmo tempo isso demonstra a robustez do sistema.

Tabela 8-Parâmetros de desempenho

Parâmetros	***RNN***	**Rede Especialista**
Entrada	784	784
Número de Camadas	2	3
Batch	20	8
Taxa de aprendizagem	0,01	0,02
Saída	7	784 por Rede
Tempo de treinamento	36 horas	100 minutos por rede (11:36 horas)
Taxa máxima de acerto	**68%**	**93%**

Fonte: Autor

7.1.1 Taxa de aprendizagem

O primeiro parâmetro analisado é a taxa de aprendizagem. O ajuste desse parâmetro está associado diretamente a capacidade de minimização de erro da rede e o tempo para que o sistema possa generalizar as classes.

Como observado na Tabela 9, para a RNN taxas de aprendizagem muito elevadas, (próximas a 1), podem produzir uma perda (*loss*) divergente e o sistema pode se tornar incapaz de reconhecer qualquer tipo de classe. Em contrapartida valores muito baixos, (próximo de 10^{-7}), tem o mesmo efeito, ou geram modelos de baixa assertividade.

Tabela 9- Taxa de acerto em função da taxa de aprendizado para RNN

Taxa de aprendizado	***Loss***	**Taxa de acerto**
0,5	Não converge	0%
0,1	2,3979	11%
0,01	0,0042	68%
0,001	0,0203	52%
0,0001	0,7415	44%
0,00001	1,8564	17%
0,000001	2,3529	21%
0,0000001	2,3983	10%

Fonte: Autor.

Como a rede especialista busca minimizar o erro entre entrada e saída, a taxa de aprendizado influencia no tempo em que o sistema demora para alcançar alta assertividade. Neste modelo a *loss* não é um parâmetro relevante pois em muitos casos ela não será minimizada. Sendo assim variar a taxa de aprendizado para este caso só agrega em termos de rendimento do sistema, diminuindo o tempo necessário para se alcançar um resultado satisfatório (Tabela 10\).

Tabela 10- Taxa de acerto em função da taxa de aprendizado para Rede Especialista

Taxa de aprendizado	**Taxa de acerto**	**Épocas**
0,05	90%	200000
0,03	90%	200000
0,02	93%	300000
0,01	90%	500000
0,001	80%	600000
0,0001	72%	1000000
0,00001	65%	1000000
0,000001	60%	1000000
0,0000001	Não converge	Não converge

Fonte: Autor

Vale ressaltar que em ambos os casos os demais parâmetros estão escolhidos de modo a otimizar a rede.

7.1.2 Número de camadas

A teoria de redes neurais nos mostra que não é necessário mais que 3 camadas intermediárias para que se crie modelos generalistas. Neste contexto é feita uma análise nos dados obtidos ao varia-se o número de camadas da rede e apresentados na Tabela 11 e 12.

Tabela 11- Taxa de acerto para RNN em função dos neurônios da camada em seu máximo rendimento

Camada 1	**Camada 2**	**Taxa de Acerto**
300	150	68%
200	100	68%
100	50	64%
80	40	62%
50	25	58%

Fonte: Autor

Tabela 12- Taxa de acerto para rede especialista em função dos neurônios da camada em seu máximo rendimento

Camada 1	Camada 2	Camada 3	Taxa de Acerto
100	100	784	93%
100	50	784	85%
60	30	784	65%
30	10	784	50%
60	20	784	40%
30	10	784	30%

Fonte: Autor

Verifica-se, então, que aumentando-se o número de neurônios na camada intermediária tem-se uma elevação, até certo ponto, da capacidade de aprendizado na rede, com o custo de exigir mais poder computacional e tempo. No caso da rede RNN, este aumento de neurônios incrementa muito tempo na fase de treinamento, sem agregar assertividade maior que a mostrada na Tabela 10. Esse custo computacional maior se deve ao fato da existência de memória na rede, agregando mais cálculos complexos ao sistema.

7.1.3 Banco de dados

O tamanho do banco de dados, que serve de base para a rede criar seus modelos também influência no sistema. Tomando como base o modelo de rede especialista percebe-se que a redução do banco de dados implica no aumento de épocas na fase de treino. Em outras palavras, quanto menos informações o sistema possui mais ele deve considerar as que possui no aprendizado. Evidentemente que se o banco for reduzido demasiadamente as informações restantes não darão condições ao sistema de criar modelos capazes de generalizar as classes. Consequentemente o modelo, por mais que seja treinado se torna incapaz de aprender todas as características da classe. Este fato pode ser observado nos dados obtidos nos testes e apresentados na Tabela 13.

7.1.4 *Batch*

Para um sistema robusto pode-se perceber que o *batch* influencia no tempo de convergência e na assertividade da rede. A Tabela 14 mostra o desempenho da rede especialista ao variar-se o *batch* buscando 93% de assertividade, que foi nosso máximo alcançado.

Tabela 13- Taxa de acerto com base no banco de dados para rede especialista

Número de locuções por classe	Taxa máxima de acerto	Número de épocas para alcançar o acerto máximo
90	93%	300000
80	93%	400000
70	85%	400000
60	80%	500000
50	72%	500000
40	65%	600000
30	53%	800000
20	40%	1000000
10	28%	1000000

Fonte: Autor

Tabela 14- Numero de épocas por *batch*

Batch	**Épocas**	**Taxa máxima de acerto**
8	300000	93%
16	300000	85%
24	400000	85%
40	500000	80%

Fonte: Autor

7.1.5 Erro Médio quadrático

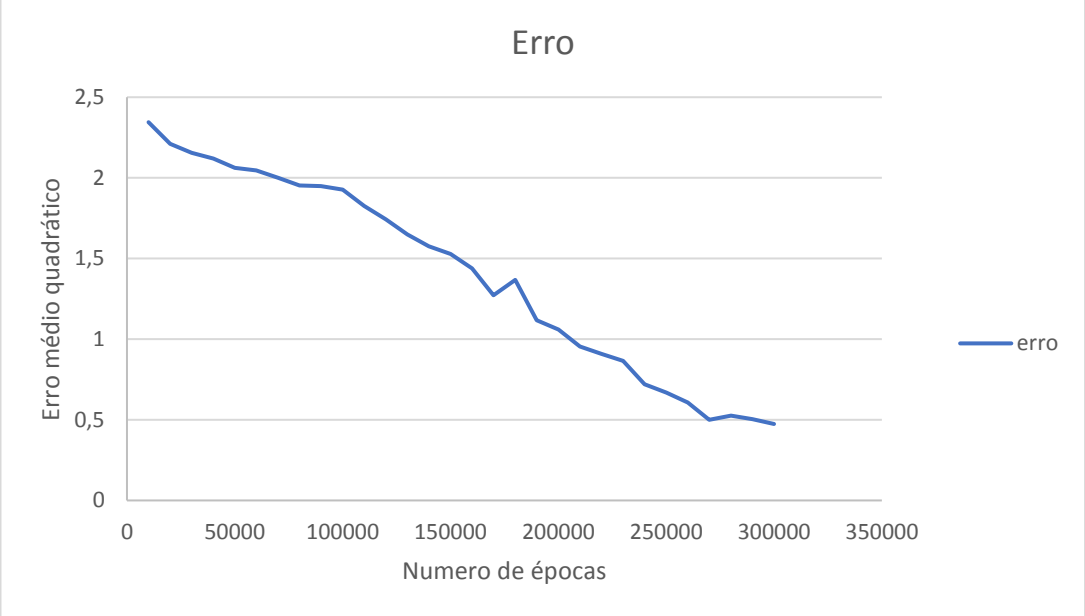
A aprendizagem da rede está diretamente ligada a sua capacidade de minimizar o seu erro de saída, conhecido como *loss*, isso no caso de redes tradicionais e RNN. No modelo de rede especialista proposto ao invés de se minimizar a *loss* se buscara minimizar o erro entre entrada e saída da rede. É importante ressaltar que o erro médio quadrático entre entrada e saída diverge numericamente da *loss*, já que, quando se busca minimizar o erro em muitos casos pode-se até aumentar ou manter o valor da loss. Quanto menor for o erro médio quadrático obtido no sistema, mas provável é que a rede tenha aprendido a reconhecê-lo. Isto é, e mais provável que a rede tenha aprendido a reproduzir uma determinada classe de informação em sua saída. Nas Figuras 35 e 36 ver-se os desempenhos das redes com os erros sendo minimizados ao longo do tempo. Nota-se que a rede RNN alcança seu melhor resultado após 97 épocas, obtendo uma *loss* de

0,0203. Por outro lado, a rede especialista precisa de 300000 época para obter erro médio de 0.474. Em ambos os casos as redes encontraram seus máximos de assertividade nesses pontos.

Figura 35- *Loss* em função do tempo RNN

Fonte: Autor

Figura 36- Erro médio quadrático em função do tempo em rede especialista



Fonte: Autor

Mesmo que a rede especialista demore 300 mil épocas para alcançar seu auge em acerto e a rede RNN demore 97 épocas. É importante relembrar que a rede RNN demora 22 minutos para concluir uma época de treino, enquanto a rede especialista demora 0.3ms para concluir uma época.

7.2 Resultados obtidos no controle do robô

Após realizar-se uma série de 100 testes por locuções verificou-se que, como esperado, não houvesse falha na execução dos movimentos por parte do robô.

Este resultado já era previsível devido a estrutura de comunicação proposta, uma vez que não é da competência do robô realizar o reconhecimento de fala ou qualquer atividade deste tipo, apenas a execução do movimento. Como seu sistema encontra-se em perfeito funcionamento, o mesmo não falhou em executar os comandos que recebeu.

O *software* de reconhecimento de fala fica hospedado em um *hardware* de controle separado, que pode ser um *smartphone*, *tablet*, *notebook* ou mesmo um desktop. Para todos esses dispositivos têm-se as mesmas assertividades, que ficou em torno de 93%.

Este índice deve-se ao fato da filtragem antirruído, realizada na captação e na rede neural especialista implementada e treinada, que minimizou os erros da entrada e da saída mesmo que o sinal possua um nível alto de interferência. Este tipo de rede possui alta robustez à ruídos externos já que, mesmo que o erro entre a entrada e a saída para uma dada classe seja alto, ainda assim será, na maioria dos casos, menor que os produzidos nas demais classes da rede.

Durante o processo de reconhecimento de fala até a execução do movimento por parte do robô pode-se ver nos Quadros 4 e 5 o comportamento dos dispositivos durante a ação. Nessas duas figuras verifica-se um exemplo de arquivo *log* gerado por parte do sistema de reconhecimento de fala e por parte do robô quando se aplica o comando “DIREITA”. Lembrando-se que de acordo com o Quadro 3, o sistema reconhece essa classe numericamente com “6”. Neste modo, como não ocorre falhas por parte do robô, os resultados entre os *logs*, quando o reconhecimento de fala é bem-sucedido, serão idênticos.

Nos casos em que o reconhecimento de fala não for bem-sucedido, o sistema de reconhecimento de fala resultara no Quadro 6, onde vê-se que a comunicação não é estabelecida, retornado o sistema a *stand-by.* Logo, o robô permanece inerte sem gerar comandos de operação.

Analisando o sistema em termos de robustez vê-se que o sistema possui a capacidade de reconhecer derivados das palavras treinadas ou locuções dentro de frases. Na Tabela 15 mostra-se o resultado gerado ao aplicar-se a locuções com variação ao sistema de reconhecimento de fala.

Nota-se que o sistema é bem-sucedido em reconhecer a variante da locução a custo de maior erro médio quadrático. Vê-se assim que o sistema é capaz de classificar corretamente locuções derivadas das treinadas por ele a custo de maior erro.

Quadro 4 - *Log* do sistema ao receber comando “ACELERAR”

```
inicio
gravando...
fim da gravação...
testando
modelo 1:
        erro médio quadrático: 0.9675
modelo 2:
        erro médio quadrático: 0.7891
modelo 3:
        erro médio quadrático: 2.4395
modelo 4:
        erro médio quadrático: 1.3697
modelo 5:
        erro médio quadrático: 0.6275
modelo 6:
        erro médio quadrático: 0.1795
provável locutor: modelo 6
ERRO INFERIOR A 0.6-- OK
enviando...
mensagem "6"
recebido: "6" recebido!!!
```

Fonte: Autor.

Quadro 5 – *Log* do robô ao receber comando “ACELERAR”

```
inicio...

recebido: "6"
enviando: "6"
executando comando 6
fim...

Aguardando novo comando....
inicio...
```

Fonte: Autor.

Quadro 6 – *Log* sistema quando o reconhecimento de fala falha

```
inicio...
gravando...
fim da gravação...

testando
modelo 1:
erro médio quadrático: 0.8723
modelo 2:
erro médio quadrático: 0.9179
modelo 3:
erro médio quadrático: 2.082
modelo 4:
erro médio quadrático: 1.476
modelo 5:
erro médio quadrático: 0.9985
modelo 6:
erro médio quadrático: 0.7452

provável locutor: MODELO X
ERRO SUPERIOR A 0.6-- FALHOU!!!
inicio...
```

Fonte: Autor.

Tabela 15- Resumo de erro ao aplicar-se a variação de uma locução

Locução	Erro médio quadrático	Locução variante	Erro médio quadrático da locução variante
“Direita”	0.1795	“Andar para direita”	0.3723
“Esquerda”	0.4157	“Andar para esquerda”	0.8264
“Acelerar”	0.2941	“Aceleração”	0.4529
“Átras”	0.3491	“Atrasar”	0.5617
“Frente”	0.2376	“Em frente”	0.4983
“Parar”	0.3007	“Aparar”	0.3954

Fonte: Autor.

Pode-se extrair também da Tabela 15 o melhor desempenho do sistema se comparado às locuções na qual o algoritmo foi treinado para reconhecer. Percebe-se que as locuções que geram menor erro médio quadrático são também as que possuem mais robustez às variâncias da locução.

8. CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho foi proposto e desenvolvido um sistema de reconhecimento de fala aplicado ao controle de dispositivos móveis via *wireless*. Na implementação da proposta, em termos de dispositivo mecatrônico, foi empregado o robô RoSeLi disponível no LRC, o qual foi desenvolvido por (RIBEIRO, 2017) e também utilizado no comando por gestos com *feedback* táctil (LIMA, 2017), tendo o robô sido adaptado. As alterações feitas no robô ocorrem em termo de *software* com intuito de modificar a forma de controle e os parâmetros de desempenho.

Foram realizados inúmeros testes em componentes individuais, parciais e, por fim, foi envolvendo todo o sistema completo. Os principais testes se concentraram no desempenho do sistema com relação ao reconhecimento de fala visto que, caso o mesmo funcione corretamente, dificilmente deve ocorrer erros na execução dos movimentos por parte do robô. Sendo assim, o objetivo dos testes foi verificar a robustez do sistema, a sua eficiência e os parâmetros que influenciavam tanto de forma positiva quanto negativa no desempenho final.

Quando ao que tange a movimentação do robô, era esperado obter, em termos de precisão e variação resultados semelhantes aos apresentados em (RIBEIRO, 2017). Quanto a eficiência no reconhecimento de fala, o sistema obteve assertividade em torno de 93%, sendo considerado excelente se comparado a outros trabalhos semelhantes.

Apesar da complexidade da pesquisa, evolvendo três áreas distintas, no caso, reconhecimento de fala, redes neurais e robótica, o sistema desenvolvido demonstrou grande robustez e desempenho elevado, dando margem para futuramente expandir consideravelmente o banco de dados, além de permitir locuções mais complexas e interruptas.

Vários trabalhos futuros que podem ser conduzidos imediatamente a partir do atual estágio implementado neste trabalho, recomendando-se:

a) Combinar diversos movimentos do robô ao longo de fala;

b) Expandir o sistema para comando de outros tipos de dispositivos inteligentes, como na Domótica;

c) Aumentar o banco de dados de voz e o número de classe possíveis de modo a testar os limites das redes especialistas; e

d) Mesclar sistema de reconhecimento de fala com reconhecimento de locutor.

REFERÊNCIAS BIBLIOGRÁFICAS

ALCAIM, A.; OLIVEIRA, C. A. S. Fundamentos do Processamento de sinais de voz e imagem. primeira Edição. Rio de Janeiro; PUC RIO, 2011.

ANDERSON, W. J., **Continuous-time Markov chains**. An applications-oriented approach, Springer series in Statistics, NY, 1991.

CAO, Y., FUKUNAGA, A.; KAHN, A. **Cooperative mobile robotics: Antecedents and directions**. 4, Autonomous Robots, 1997, pp. 1-23

CAMPBELL, J. P. **Speaker recognition: a tutorial**. Proceedings of the IEEE, v. 85, n. 9, p. 1437–1462, Set 1997.

CARDOSO, D. P. **Identificação de locutor usando modelos de mistura de Gaussianas**, São Paulo, USP, 2009.

CLARKE, A. B.; DISNEY, R. L. Probabilidade e Processos Estocásticos, Rio de Janeiro: Livros Técnicos e Científicos Editora, 1979.

DINIZ P. S. R. *et al*. **Processamento Digital de Sinais.** Projeto e Análise de Sistemas", Editora Bookman, 2001.

FAUSET, L. **Fundamentals of Neural Networks: Architectures, Algorithms, and Applications**. Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1994

FERNANDES, T. J.; FERREIRA, E. B.; FERREIRA, D. F. Avaliação Monte Carlo do teste de normalidade de qui-quadrado sob diferentes critérios do número de classes. Rev. Bras. Biom., São Paulo, v.30, n.2, p.185-198, 2012.

GOLUB, G.H; VAN LOAN,C.F. Matrix Computations. 4. ed. Baltimore: The Johns Hopkins University Press, 2013.

HAYKIN, S. **Neural Networks: A Comprehensive Foundation**. 2nd Prentice Hall PTR Upper Saddle River, NJ, USA, 1998

HEBB, D. O. The organization of behavior: A neuropsychological theory. John Wiley And Sons, Inc., New York, 1949

KOLEN, J.F. **Exploring the Computational Capabilities of Recurrent Neural Networks**. Ph.D. Thesis, The Ohio State University, 1994.

KOVÁCS, Z. L. **Redes Neurais Artificiais: Fundamentos e Aplicações**. Edição: 4ª, Livraria da Física, São Paulo, 2006.

LEVINSON, N. The Wiener RMS (Root Mean Square) Error Criterion in Filter Design and Prediction. Journal of Mathematics and Physics, v. 25, no. 4, p. 261-278, jan. 1947.

LIMA, W. P. J. **Robô controlado por gestos com *feedback* táctil**, São Luís, UFMA, 2017.

MANNING, C. D.. Computational Linguistics and Deep Learning Computational Linguistics. 41(4), 2015, 701–707.

McCULLOCH, W. S.;PITTS, W. A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5(4):115–133, 1943.

MINSKY, M; PAPERT, S. Perceptrons. An Introduction to Computational Geometry. M.I.T. Press, Cambridge, Mass., 1969.

MINSKY, M. A neural-analogue calculator based upon a probability model of reinforcement. Harvard University Pychological Laboratories internal report, 1952

QUESADA, V. M. Analisis del filtro de tercer orden de butterworth y aplicacion em um crossover electronico de 3 vias em dos canales, con frecuencia de corte variable, ESCUELA POLITECNICA NACIONAL, 1991, 142P.

RABINER, L. **Fundamentals of speech recognition**. United States, New Jersey: Prentice Hall PTR, 1993.

RABINER, L. **Fundamentals of speech recognition**. United States, New Jersey: Prentice Hall PTR, 2007.

PICONE, J. W. **Signal modeling techniques in speech recognition**. Proceedings of the IEEE, v. 81, n. 9, Sep 1993.

RIBEIRO, D. V. **ROSELI: Robô Seguidor de Linha para Mapeamento de Ambientes Internos**. Monografia do Curso de Graduação em Engenharia Elétrica, Universidade Federal do Maranhão, 2017, 94p.

ROCHA, L. P. Reconhecimento de Voz utilizando Seleção Dinâmica de Redes Neurais, São Luís, UFMA, 2018.

ROSA, J. J. **Reconhecimento automático de emoções através da voz**, Santa Catarina, UFSC, 2017.

ROSENBLETT, F. The perceptron, a perceiving and recognizing automaton Project Para. Cornell Aeronautical Laboratory, 1957.

SEIFFERT, L.; YUN, Y. J. O Algoritmo de Levinson para Resolução de Sistemas Lineares, Universidade Federal do Paraná, 2015.

S. L. Lima, O. R. Saavedra and V. Miranda, "A Two-Level Framework to Fault Diagnosis and Decision Making for Power Transformers," in *IEEE Transactions on Power Delivery*, vol. 30, no. 1, pp. 497-504, Feb. 2015.

WIDROW, B. et al. Adaptive ”Adaline” neuron using chemical ”memistors”. Number Technical Report 1553-2. Stanford Electron. Labs., Stanford, CA, October 1960.

ZHAI, S.; MILGRAM, P. Human Robot Synergism and Virtual Telerobotic Control. Proc. HFAC. Canadá. 1992.