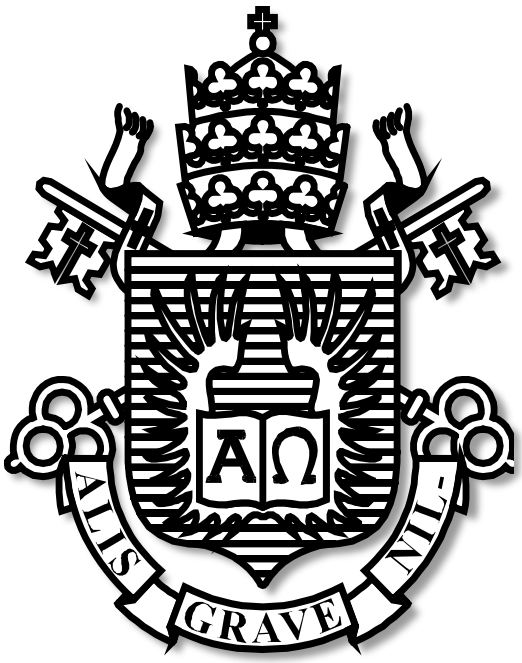

PONTIFÍCIA
UNIVERSIDADE
CATÓLICA
DO RIO DE JANEIRO

Projeto de Graduação em engenharia de Controle e Automação

ELE1009

Reconhecimento de fala por algoritmo de rede neural embarcado em DSP.

Guilherme Rodrigues Sampaio de Paula

Departamento de engenharia mecânica

PUC-Rio

Orientador: Marco Antônio Meggiolaro

Co-Orientador: Alexandre Galvão

Sumário

Índice de figuras 4
Introdução: 5
Métodos numéricos para reconhecimento de fala: 8
Placa de desenvolvimento com DSP: 10
Rede Neural Artificial: 12
Processamento de sinais: 18
Programação do DSP pelo MatLab e Simulink: 26
Fluxo de dados no sistema: 28
Conclusões: 30
Referências bibliográficas: 31

Índice de figuras

Figura 1 – Diagrama de um Hidden Markov Model 8

Figura 2 - Placa de desenvolvimento eZdspTM F28335 12

Figura 3 - Performance da rede neural 15

Figura 4 - Saída da rede neural para R=0.82905 16

Figura 5 - Saída da rede neural para R=0.7622 16

Figura 6 - Saída da rede neural para R=0.7698 17

Figura 7 - Performance de treinamento da rede neural. 17

Figura 8 – Pré - amplificador para microfone 19

Figura 9 - Espectro da voz “direita” em função da amostra 20

Figura 10 - Espectro da voz “esquerda” em função da amostra 21

Figura 11 - Power Spectrum do sinal ”direita” em função da freqüência 23

Figura 12 - Spectrum do sinal ”esquerda” em função da freqüência 23

Figura 13 - Spctrograma do sinal “direita” 24

Figura 14 - Spctrograma do sinal “esquerda” 25

Figura 15 - Bloco do DSPF28335 no Simulink 27

Figura 16 - Fluxo de dados no sistema 29

Introdução:

Um sistema de reconhecimento de fala é definido como um sistema capaz de decodificar e identificar sinais sonoros produzidos pela fala humana. O objetivo principal deste sistema é auxiliar na interface homem-máquina. Tais sistemas tiveram grande desenvolvimento nas últimas décadas devido aos avanços na tecnologia nas áreas de computação, processamento de sinais e eletrônica embarcada. Algumas das aplicações para reconhecimento de fala de maior uso atualmente são: Discagem por voz (em celulares), comandos para aplicações domesticas (automação residencial), identificação de conteúdo em *broadcast* (encontrar palavras específicas em uma transmissão). Há diversos métodos de se implementar um sistema de reconhecimento de fala, existem soluções que oferecem um sistema de reconhecimento de tamanho pequeno contudo têm pouca flexibilidade e demandam um longo tempo de processamento. Por outro lado, existem soluções baseadas em software. Estas soluções, por sua vez, dependem de um computador para serem rodadas e são difíceis de ser embarcadas em sistemas autônomos com uso de microprocessadores.

Tais sistemas podem ser classificados por requererem, ou não, que o usuário treine o sistema para reconhecer seus padrões particulares de fala, por ter a habilidade de reconhecer fala contínua ou por requerer que o usuário fale pausadamente, e pelo tamanho do vocabulário que é capaz de reconhecer (pequeno, da ordem de dezenas a centenas de palavras, ou grande, com milhares de palavras).

Sistemas que requerem pouco treinamento podem capturar continuamente a fala com um amplo vocabulário, em ritmo normal, com precisão de acerto de cerca de

98% enquanto sistemas que não requerem treinamento podem reconhecer um número pequeno de palavras como, por exemplo, os dez dígitos do sistema decimal. Tais sistemas são populares por direcionar chamadas telefônicas recebidas em grandes organizações, aos seus destinos.

Sistemas comerciais para reconhecimento da fala têm estado disponíveis desde os anos 90, porém é interessante notar que, apesar do aparente sucesso dessa tecnologia, poucas pessoas os usam. Ao que parece a maioria dos usuários de computador pode criar e editar documentos mais rapidamente com um teclado convencional, apesar do fato de que muitas pessoas são capazes de falar consideravelmente mais rápido do que podem digitar. Além disso, o uso intenso dos órgãos da fala pode resultar em sobrecarga vocal e conseqüente fadiga física demasiada.

Alguns dos problemas técnicos chaves do reconhecimento da fala são:

- Diferenças entre os interlocutores são freqüentemente grandes.
- Não está claro quais características da fala são independentes do falante.
- A interpretação de vários fonemas, palavras e frases são sensíveis ao contexto. Por exemplo: os fonemas são geralmente mais curtos em palavras longas do que em palavras pequenas.
- As palavras têm significados diferentes em frases diferentes. Por exemplo: "Philip lies" [1] pode ser interpretado como Philip sendo um mentiroso ou como Philip deitando-se na cama.
- A entonação e o timbre da fala podem mudar completamente a interpretação de uma palavra ou frase. Por exemplo: "Vai!", "Vai?" e "Vai." podem ser claramente reconhecidos por um humano, mas não tão facilmente por um computador.

- Palavras e frases podem ter várias interpretações válidas de modo que o falante deixe a escolha da correta para o ouvinte.
- A linguagem escrita precisa de pontuação de acordo com regras estritas que não estão fortemente presentes na fala e são difíceis de inferir sem conhecer o significado (vírgulas, fim de frase, citações).

O entendimento do significado das palavras ditas é pensado como um campo separado do entendimento natural da linguagem. Há vários exemplos de frases que soam iguais e só podem ser desambiguadas pela aparição do contexto: uma famosa camisa vestida por pesquisadores da Apple dizia "I helped Apple wreck a nice beach" (Eu ajudei a Apple a destruir uma bela praia], o que, quando pronunciado, soa como "I helped Apple recognize speech" [Eu ajudei a Apple a reconhecer a fala).

Uma solução geral para muitos dos problemas acima requer efetivamente conhecimento humano, experiência e uma avançada tecnologia em inteligência artificial. Especificamente, modelos estatísticos de linguagem são freqüentemente empregados para desambiguação e melhoramento da precisão do reconhecimento.

Há basicamente duas vertentes no reconhecimento de fala. Uma primeira busca identificar comandos, o que se fala. Esta será o objeto principal deste trabalho. Outra aplicabilidade desta tecnologia é o reconhecimento de quem fala. Tais sistemas são muito utilizados com propósitos de segurança de acesso e patrimonial, em conjunto ou não com outros tipos de reconhecimento de características biométricas do ser humano.

Métodos numéricos para reconhecimento de fala:

O método mais utilizado para reconhecimento de voz em geral é o *Hidden Markov Model (HMM)* que é um método estatístico que dá como saída um vetor de quantidades (pesos). Em um modelo HMM o estado não é diretamente visível para o observador, mas as saídas, que dependem do estado, são. A característica de Escondido (hidden) se refere somente ao estado do modelo, os parâmetros são visíveis (Figura 1). O método HMM é um método voltado para a classificação de padrões temporais, por esta razão foi descartado neste estudo, como será discutido em detalhes na seção de processamento de sinais deste trabalho.

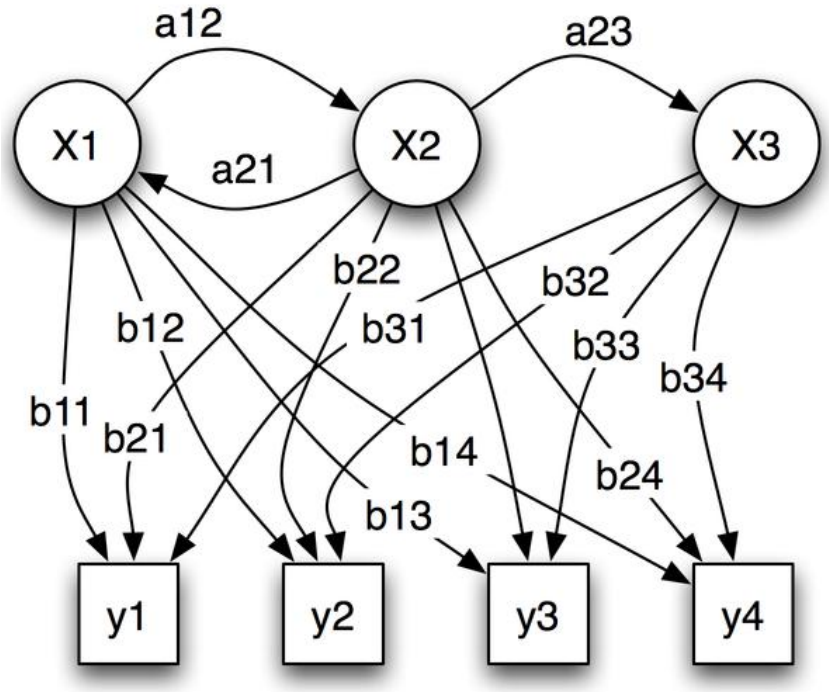

Figura 1 – Diagrama de um Hidden Markov Model

Outro método muito utilizado para este fim é o Linear Predictive Coding (LPC) aonde é feita uma análise inversa da geração dos sinais vocálicos humanos estimando as estruturas bio-mecânicas que combinadas produzem a voz audível e retirando do sinal as “interferências” causadas por estas estruturas. Os números obtidos através desta análise que caracterizam uma voz específica são então armazenados. Estes números dizem respeito às intensidades e freqüências das distorções aplicadas à voz pelas estruturas do sistema vocálico humano. O LPC então sintetiza o sinal em sentido inverso, isto é, usa os parâmetros de vibração e os resíduos para criar um sinal base. Então, usa as distorções para criar um filtro, e aplica então este filtro ao sinal base, obtendo a voz novamente. Devido às características singulares deste método, ele é mais indicado para a identificação do interlocutor e não a identificação de ordens, como se deseja neste trabalho.

Além desses, outro método é o *Artificial Neural Network (ANN)* que foi o escolhido para ser aplicado neste trabalho por ser mais facilmente implementado. As redes neurais lidam melhor com sinais com ruído, conseguindo classificações muito mais precisas do que os métodos anteriores nestas condições. Contudo, as redes neurais dependem de treinamento específico o que inviabiliza o seu uso isolado para identificação de muitos padrões diferentes como, por exemplo, quando deseja-se identificar um padrão em meio a um vocabulário muito estendido.

Há ainda sistemas descritos que utilizam combinações de redes neurais tanto com modelos HMM como com LPC. Com estas combinações é possível utilizar as vantagens de cada método, como por exemplo, lidar com sinais ruidosos em meio à classificação em vocabulários grandes. A desvantagem destes métodos “compostos” é que muitas vezes a interação deles gera problemas durante o processamento dos sinais além da maior dificuldade de programação, que se torna mais extensa e em muitos casos demandando um esforço computacional maior, que se traduz em maior

tempo de processamento ou maior custo dos microcontroladores no caso de sistemas embarcados.

Placa de desenvolvimento com DSP:

Nas últimas décadas os DSPs tiveram um grande avanço. Há aplicações nos mais diversos campos para os DSPs como, por exemplo: Processamento de voz, comunicações, aparelhos portáteis de entretenimento, indústria aeroespacial, sistemas de auxílio automotivo como acionamento de air-bags, sistemas de frenagem de emergência, suspensão ativa, etc.

Os DSPs de maior capacidade hoje já são capazes de realizar operações com ponto flutuante, e diversas outras funções “periféricas” como endereçamento “bit-reversed” para transformadas rápidas de fourrier (FFT), portas de comunicação serial, timers, endereçamento direto de memória, sistemas avançados de interrupção, sistemas A/D e D/A, PWMs, etc.

Neste trabalho foi utilizada a placa modelo eZdspTM F28335 da Spectrum digital que tem como processador principal um *Digital signal controler* TMS320F28335 fabricado pela Texas Istruments.

O eZdsp F28335 é uma placa stand alone que permite ao desenvolvedor testar e utilizar o DSP TMS320F28335. Diversas portas de expansão estão presentes na placa permitindo assim a sua integração ao resto do sistema. A placa conta ainda com interface USB para conexão com computador e conector JTAG para interface com emuladores para *Debug* rápido. Abaixo está uma vista geral da placa de

desenvolvimento utilizada (Figura 2), no centro, em maior tamanho vê-se o socket do processador TMS320F28335.

A placa de desenvolvimento eZdsp F28335 foi testada para uso por dois métodos diferentes. No primeiro, a placa atua no processamento dos sinais e na execução das tarefas relativas à classificação. Neste método a placa é utilizada em conjunto com o computador (IBM-PC).

O segundo, e mais complexo, método de uso desta placa é independentemente do computador (stand-alone code). Neste método o código deve ser gravado na memória flash do chip DSP e o método de boot deve ser alterado para acessá-lo. O método de boot é alterado por meio de chaves que estão presentes na placa, o datasheet da placa de desenvolvimento contém informações precisas sobre esta modificação.

Além da modificação física na placa há a necessidade da instalação no computador que irá programar o DSP do driver Flash 2833x API que é um arquivo .lib que deve ser inserido no MatLab e contém as informações necessárias para que o compilador possa executar o código no DSP.

Este arquivo pode ser encontrado no site da Texas Instruments (www.ti.com) para download.

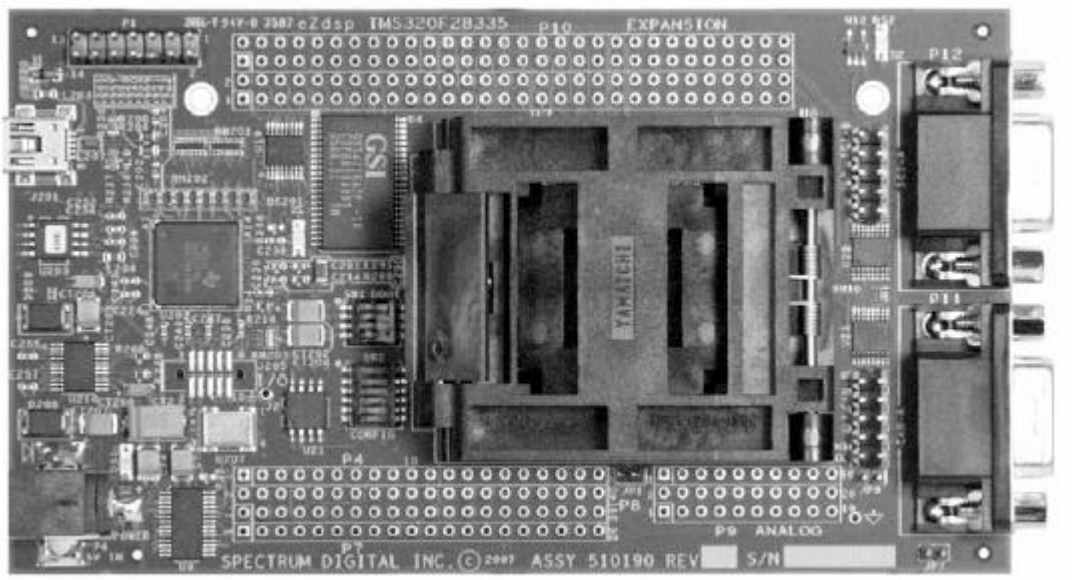

Figura 2 - Placa de desenvolvimento eZdspTM F28335

Rede Neural Artificial:

As redes neurais artificiais são um método para solucionar problemas através da simulação do cérebro humano, inclusive em seu comportamento, ou seja, aprendendo, errando e fazendo descobertas. São técnicas computacionais que apresentam um modelo inspirado na estrutura neural de organismos inteligentes e que adquirem conhecimento através da experiência.

As redes neurais possuem nós ou unidades de processamento. Cada unidade possui ligações para outras unidades, as quais recebem e enviam sinais. Cada unidade pode possuir memória local. Essas unidades são a simulação dos neurônios, recebendo e retransmitindo informações. Somam-se as entradas e se retorna uma saída, caso esta seja maior que o valor da soma.

Uma rede neural pode possuir uma ou múltiplas camadas. Exemplificando com três camadas, poderíamos ter a camada de entrada, em que as unidades recebem os padrões; a camada intermediária, onde é feito processamento e a extração de características; e a camada de saída, que conclui e apresenta o resultado final.

O número de camadas define a capacidade de representação das relações entre o espaço de entrada e o de saída. A inexistência da camada intermediária, característica do modelo perceptron, condiciona-o a representar bem somente relações linearmente independentes. A existência de camadas intermediárias, característica do modelo perceptron de múltipla camada (PMC), retira tal limitação. Se houver apenas uma camada intermediária, o PMC pode representar (com qualquer grau de aproximação, por menor que seja) qualquer função contínua. Duas ou mais camadas ampliam o universo de representação a qualquer função, contínua ou não.

As redes neurais artificiais estão muito associadas à adaptação de conexões (sinapses) entre neurônios, o conexionismo. Cabe registrar, entretanto, a existência de modelos nos quais as conexões não são adaptadas, mas apenas transmitem estimulação entre neurônios. Tais modelos são chamados redes neurais sem pesos (weightless neural networks). Para completar, há modelos em que as sinapses não são adaptadas, mas calculadas previamente, servindo a tarefas de otimização, geralmente.

Neste trabalho a rede neural foi implementada no software MatLab. Este software foi escolhido para a implementação da rede pois já possui diversos *toolboxes* que facilitam a sua criação, sem os quais a programação da rede ficaria demasiadamente extensa, consumindo muito tempo de programação. O *toolbox* utilizado foi o *neural network toolbox*. Foi escolhido uma rede do tipo *feed-forward backpropagation network* com função de tranferência *tansig para* a camada escondida devido às suas características que se aplicam melhor à solução do problema de reconhecimento de fala.

A rede neural artificial utilizada neste trabalho tem como objetivo classificar um sinal sonoro de entrada, após este ter passado por um pré processamento e tratamento de sinais, em dois padrões possíveis: Direita e Esquerda. Assumiu-se que

não seriam dadas ordens diferentes a estas duas ao sistema, pois isso implicaria na criação de mais uma saída possível para a rede: Erro.

Para o treinamento da rede foram feitas experiências a fim de determinar empiricamente o tamanho adequado do conjunto de treinamento que resultasse numa rede confiável, com alto índice de acertos, mas que não fosse grande demais para evitar o problema conhecido com *overtraining*. Primeiramente o treinamento da rede foi feito com 25 padrões de cada comando (direita e esquerda). Este número mostrou-se insuficiente para a classificação dos dois padrões de objetivo. Contudo, curiosamente, este treinamento com um conjunto muito pequeno, mostrou-se razoavelmente capaz de identificar tons diferentes de voz, ele diferia com aproximadamente 80% de precisão uma voz masculina de uma voz feminina aguda.

Após este resultado não satisfatório para o objetivo do trabalho que é identificar os comandos “direita” e “esquerda” dados por uma pessoa, foram criados mais 25 padrões de treinamento para cada comando (direita e esquerda), totalizando 50 padrões de cada um dos dois comandos. Com este conjunto a rede neural mostrou um comportamento na classificação próximo do ideal, atingindo mais de 90% de acerto.

Na Figura 3, abaixo está representada a performance da rede neural quando treinada com 50 padrões de cada comando.

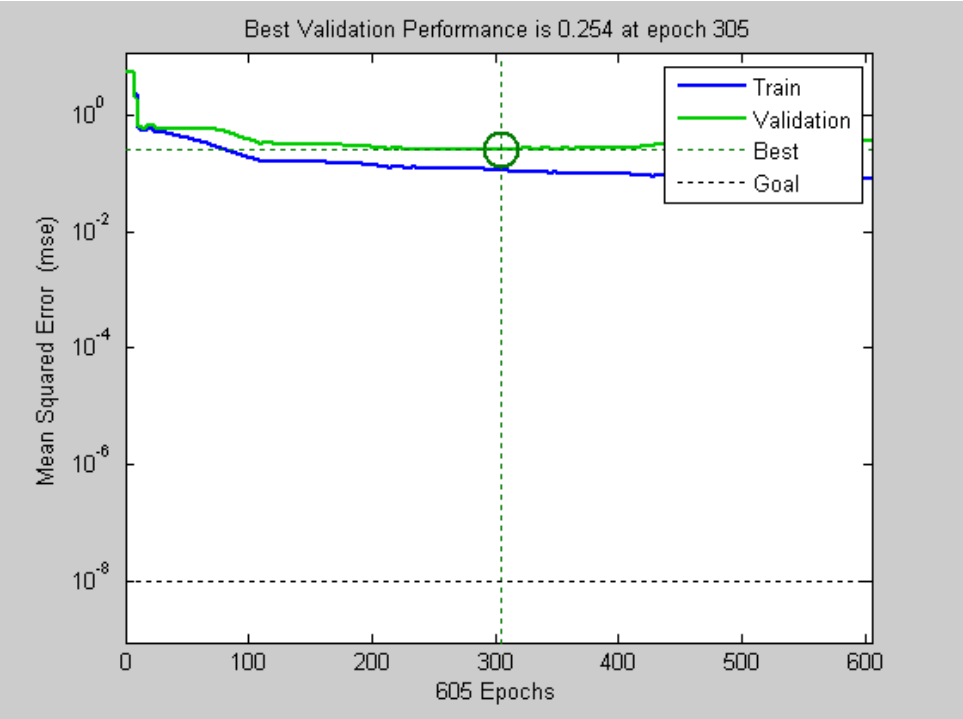

Figura 3 - Performance da rede neural

Nas três figuras (Figura 4,Figura 5 ,Figura 6) abaixo pode-se observar a separação correta dos padrões, 0 é a saída correspondente à “direita” e 1 é a saída para “esquerda”

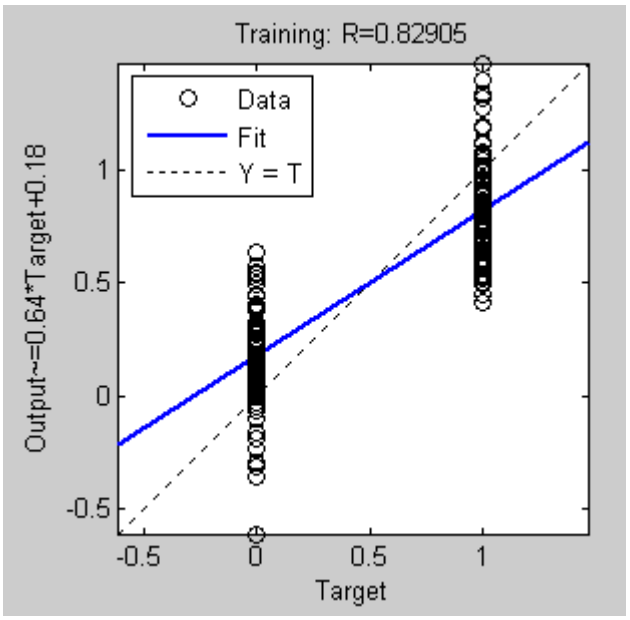

Figura 4 - Saída da rede neural para R=0.82905

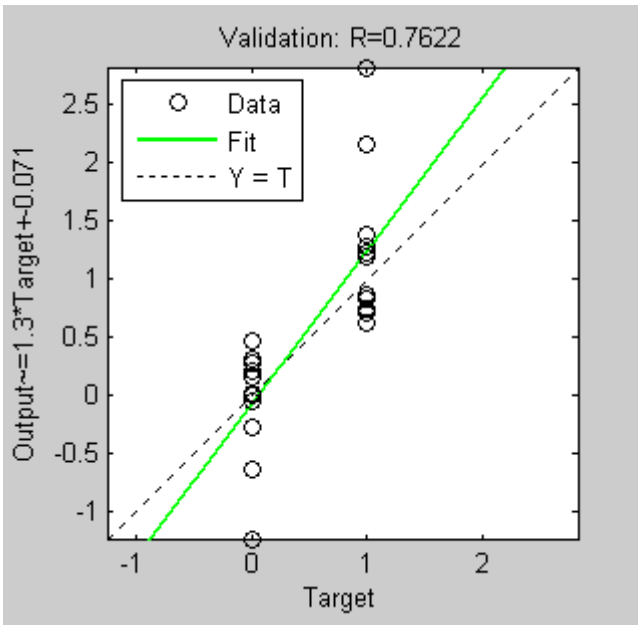

Figura 5 - Saída da rede neural para R=0.7622

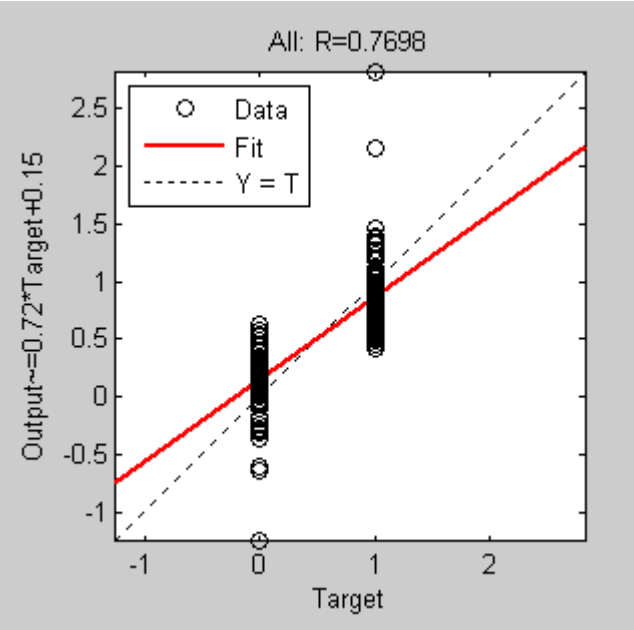

Figura 6 - Saída da rede neural para R=0.7698

Na Figura 7 abaixo observa-se o treinamento da rede neural. Percebe-se que aproximadamente por volta da epoch 300 a validação começa a falhar muito.

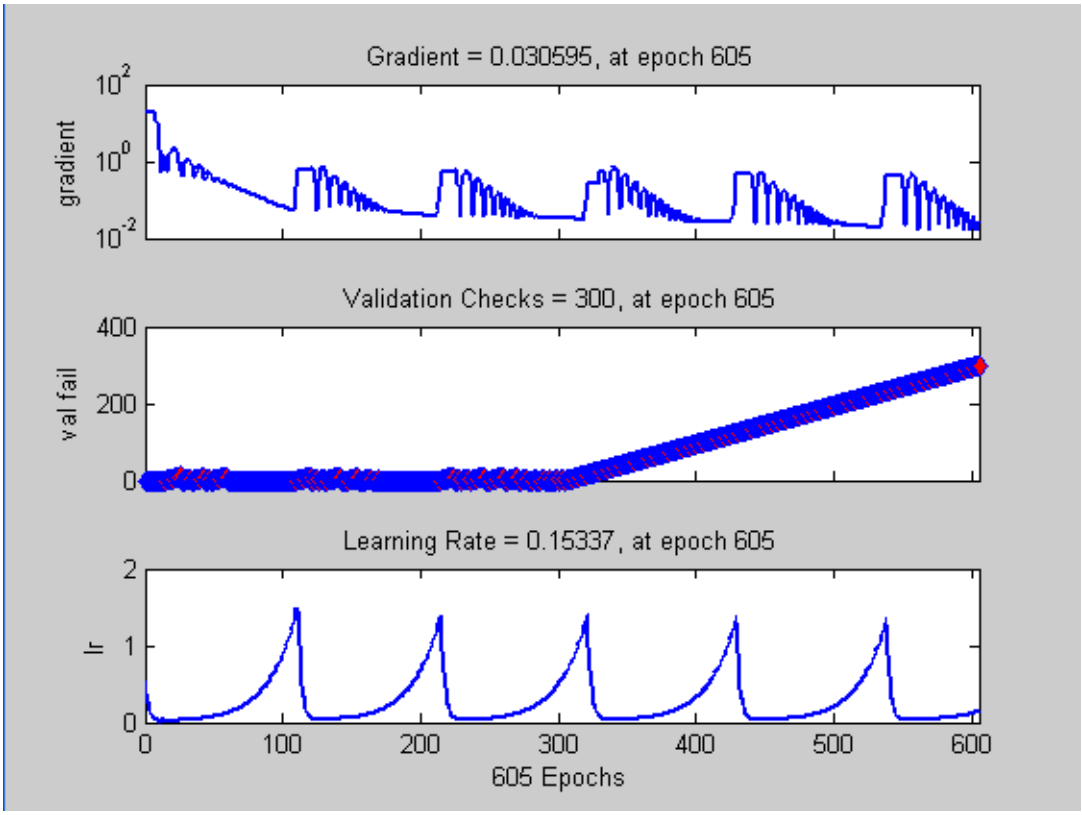

Figura 7 - Performance de treinamento da rede neural.

Foi ainda testado um treinamento com 80 padrões, houve melhora na taxa de acerto obtida com a rede, contudo, foi pequena, já que a rede anterior, treinada com 50 amostras de cada padrão mostrou-se aplicável e satisfatória.

Não foi observado durante este estudo a existência de *overtraining* com os treinamentos dos tamanhos descritos. Também não houve a motivação para a criação de mais padrões de treinamento pois a rede já havia atingido um resultado satisfatório na classificação de um sinal sonoro como um comando “direita” ou “esquerda”.

Processamento de sinais:

O Processamento de Sinais consiste na análise e/ou modificação de sinais de forma a extrair informações dos mesmos e/ou torná-los mais apropriados para alguma aplicação específica. O processamento de sinais pode ser feito de forma analógica ou digital.

Diversos dispositivos que podem ser usados no processamento digital de sinais, como DSPs, microcontroladores e FPGAs. O Processamento Digital de Sinais (Digital Signal Processing) possui diversas técnicas computacionais que podem ser utilizadas diretamente em um sistema computacional baseado no IBM-PC padrão, sem necessitar do uso de equipamentos de hardware específicos como FPGAs ou microcontroladores.

Os sinais provenientes da voz humana são captados através de um microfone que transforma as ondas sonoras em impulsos elétricos de baixa voltagem. Estes sinais elétricos de baixa voltagem por si só não seriam suficientes para serem aplicados como entrada em um conversor analógico-digital (A/D). Para resolver este problema é necessário o uso de um pré-amplificador (Figura 8) de microfones. Este

circuito é um simples amplificador não inversor baseado em um amplificador operacional (OpAmp) cujo ganho é controlado por um trimpot no circuito de realimentação.

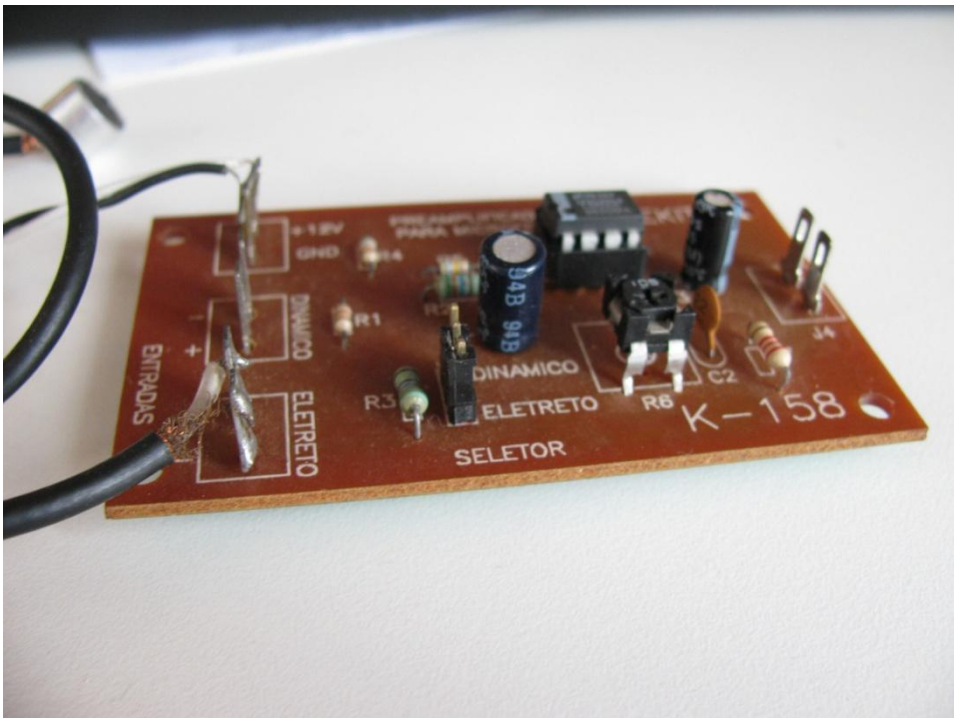

Figura 8 – Pré - amplificador para microfone

Para a gravação dos sinais para treinamento e teste da rede neural no computador não é necessário um hardware exclusivo para esse fim uma vez que a placa de som presente na maioria dos atuais computadores já possui um circuito de pré-amplificação para microfones embutido (onboard).

Para os ensaios com a placa de desenvolvimento eZdspTM F28335 foi necessária a confecção de um pré-amplificador exclusivo, mostrado na Figura 8.

Após a pré-amplificação do sinal, ele será convertido para um sinal digital pelo conversor D/A que também está presente na placa de som dos computadores ou no

DSP TMS320F28335 usado na placa de desenvolvimento. Neste trabalho foi utilizada uma taxa de aquisição de sinais de 8000Hz. Quanto maior a taxa de aquisição mais qualidade teríamos no sinal digital gerado, no entanto, este sinal ficará maior, o que dificultará os processamentos posteriores, necessitando-se de um maior poder de processamento e de armazenamento da eletrônica que irá atuar. O valor de 8kHz foi encontrado experimentalmente, já havendo sido relatado em trabalhos publicados anteriormente *M. Austin(2003)* como satisfatório para a identificação de fala. O DSP TMS320F28335 possui um conversor A/D de 12 bits com 16 canais de entrada. Nas figuras abaixo (Figura 9, Figura 10) vê-se o sinal direita e esquerda em função do tempo (amostras):

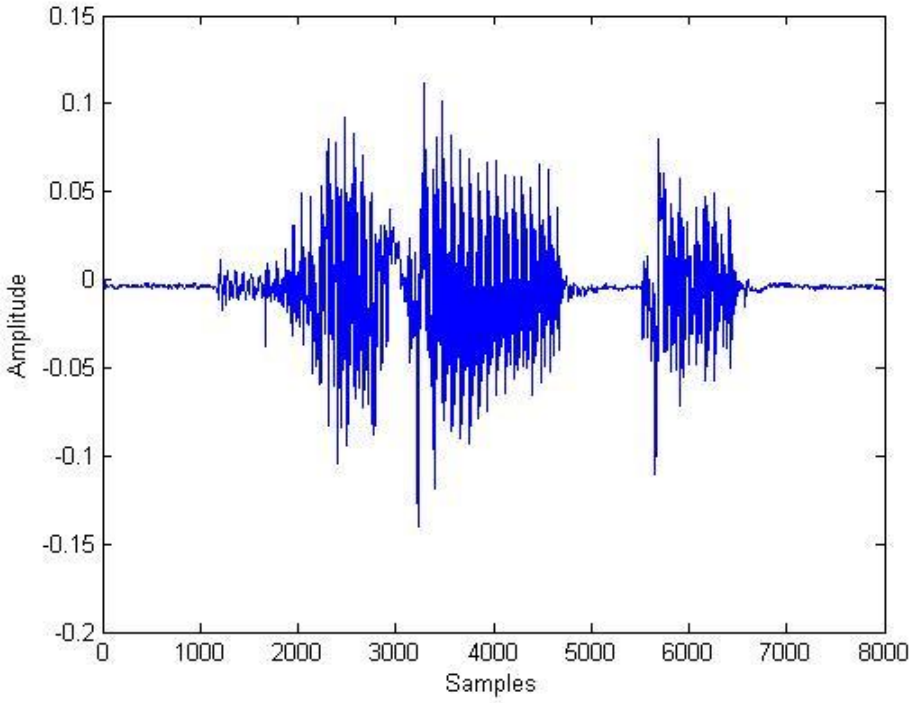

Figura 9 - Espectro da voz “direita” em função da amostra

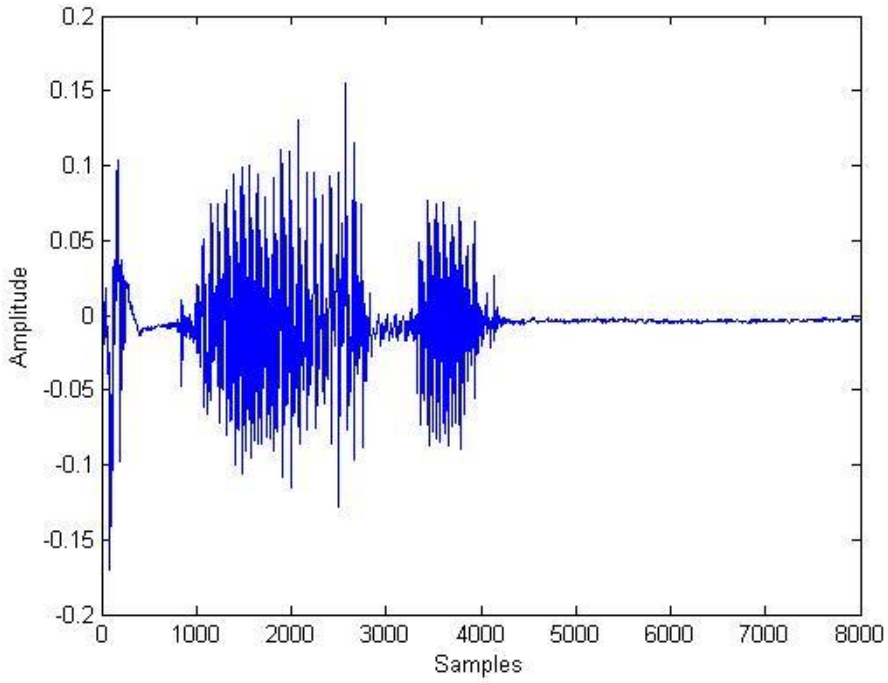

Figura 10 - Espectro da voz “esquerda” em função da amostra

Definiu-se ainda, empiricamente o tempo de gravação de 1 segundo, pois é o suficiente para se falar um comando simples como, por exemplo: Direita, Esquerda, Frente, etc.

O sinal digital armazenado durante 1 segundo e aquisitado a uma freqüência de 8kHz torna-se um vetor de elementos (valores). Contudo, este vetor do sinal da voz digital tem os valores da amplitude do sinal em função da amostra. As amostras por sua vez, estão diretamente ligadas ao tempo, já que são aquisitadas a uma taxa constante no tempo.

O sinal em função do tempo pode gerar problemas se for colocado diretamente como entrada da rede neural. Isto se deve ao fato de que é praticamente impossível começar e terminar a palavra no exato mesmo instante da gravação. Além de que, o locutor pode pronunciar a ordem mais rápida ou mais pausadamente. Devido a isso,

foi escolhido um método de processamento de sinais sonoros baseado em freqüência porque assim, não haveria mais a dependência do tempo.

O método mais conhecido para a conversão de sinais relativos ao tempo para sinais relativos à freqüência é a transformada de Fourier ou sua variante, a transformada rápida de Fourier (FFT) cuja complexidade é O(n log n) contra O(n2) necessários para o mesmo cálculo.

Neste trabalho foi utilizada uma rotina que calcula a energia do sinal em função da sua freqüência (Power Spectrum), esta operação resulta em um vetor de 4097 elementos de energia do sinal. Abaixo está o segmento do código em MatLab que implementa esta transformação:

```
DE = {D;E};
for j = 1:2,
    C = DE{j};
    for i = 1:length(C),
        t = 0:1/Fs:1;          % Time vector
        nfft = 2^(nextpow2(length(C{i})));    % Find next power of 2
        fftx = fft(C{i},nfft);
        NumUniquePts = ceil((nfft+1)/2);
        fftx = fftx(1:NumUniquePts);
        mx(:,i,j) = abs(fftx);
        mx(:,i,j) = mx(:,i,j)/length(C{i});
        mx(:,i,j) = mx(:,i,j).^2;
        if rem(nfft, 2)    % Odd nfft excludes Nyquist
            mx(2:end,i,j) = mx(2:end,i,j)*2;
        else
            mx(2:end -1,i,j) = mx(2:end -1,i,j)*2;
        end

        f = (0:NumUniquePts-1)*Fs/nfft;
    end
end
```

Após esta transformação, o sinal “direita” e “esquerda” pode ser visualizado nas figuras abaixo (Figura 11, Figura 12):

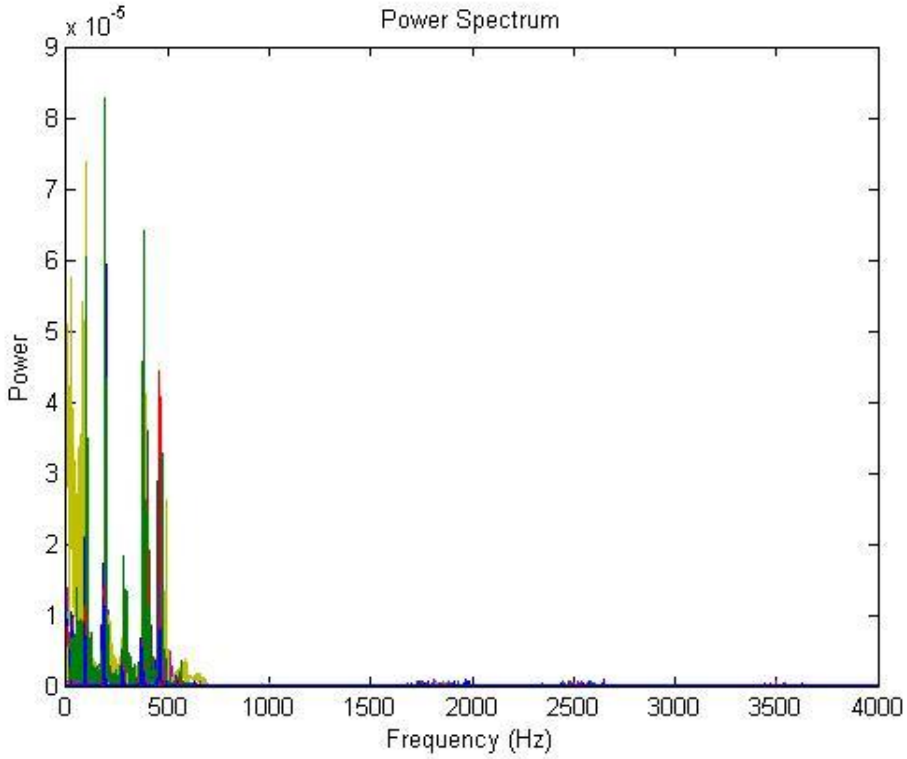

Figura 11 - Power Spectrum do sinal ”direita” em função da freqüência

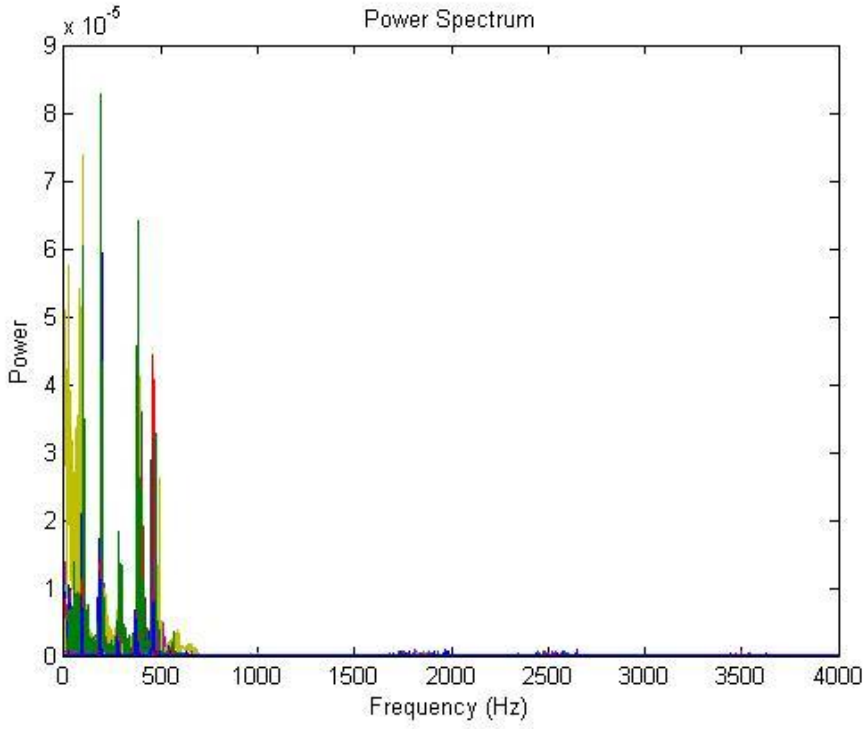

Figura 12 - Spectrum do sinal ”esquerda” em função da freqüência

Uma alternativa à abordagem do problema descrita acima é o uso do spectrograma do sinal, há uma função do MatLab que já calcula diretamente o spectrograma de um sinal no tempo, aonde Y é o vetor de valores do sinal em função do tempo (amostra):

```
spectrogram(Y)
```

Este método, apesar de já ter sido descrito como válido para o processamento de sinais de voz, não foi utilizado porque apesar de fornecer uma boa representação visual do sinal, varia muito entre as amostras. Além disso, ele gera um vetor de 3 dimensões, o que dificultaria a programação e o entendimento fácil por outra pessoa que analisasse o código. Nas figuras abaixo (Figura 13 e Figura 14) estão representados os spectrogramas dos sinais “direita” e “esquerda” aonde a cor indica a intensidade do sinal naquele ponto:

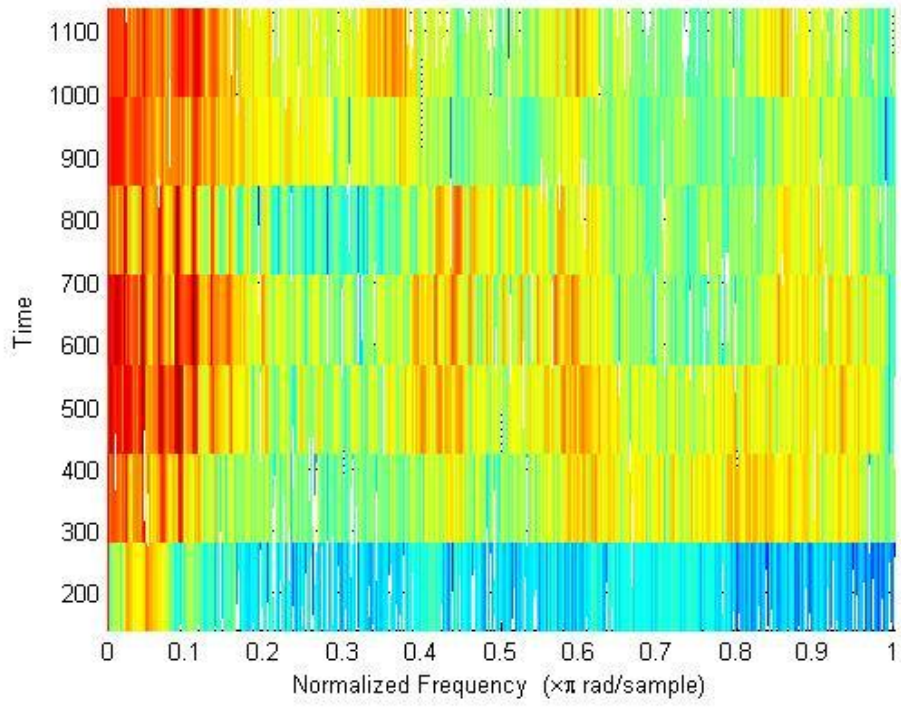

Figura 13 - Spctrograma do sinal “direita”

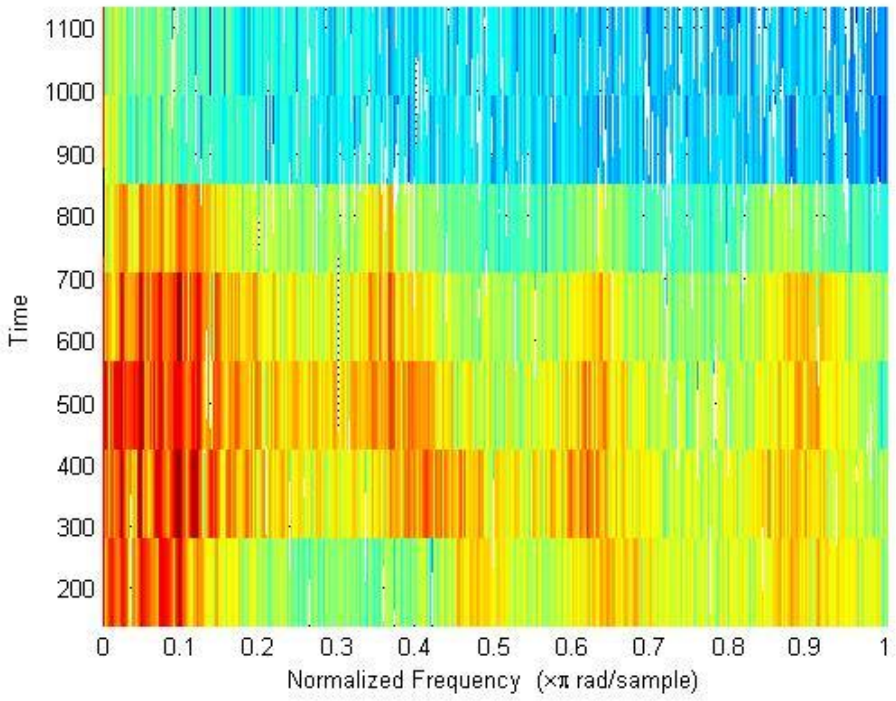

Figura 14 - Spctrograma do sinal “esquerda”

O vetor de 4097 elementos é muito grande para ser processado, pois isso implicaria em um custo computacional elevado. Devido a isto, uma rotina de loop reduz este vetor para um vetor de 50 elementos transferindo para um novo vetor alguns elementos do vetor original a um intervalo constante. São ainda excluídos do vetor os valores das componentes acima de 1KHz, pois como pode ser observado nos gráficos do Power Spectrum acima, estes valores são em sua grande maioria nulos. O segmento do código em MatLab abaixo implementa este procedimento, aonde mxd e mxe são respectivamente os padrões de treinamento para o comando direita e esquerda.

```
padroes=[mxd mxe];

for n = [1:100];  % numero de colunas da matriz
    input = padroes(:,n);
    for k = [1:50];  % numero de linhas após a redução
        for l = [(k*20)-19:k*20];
            vec(k,n)=vec(k,n)+input(l);
        end

    end

end
```

Programação do DSP pelo MatLab e Simulink:

A programação do DSP pode ser realizada por várias linguagens de programação. Podem ser utilizadas desde linguagens básicas como Assembler até o simulink que é uma interface de programação gráfica (em blocos) da empresa Math Works que em conjunto com o MatLab proporciona uma poderosa ferramenta de simulação e programação.

Através do toolbox Real Time Workshop do simulink podem ser gerados códigos em C ou até mesmo ser feita uma interação direta com o compilador da Spectrum Digital para programar o DSP que irá ser utilizado.

A utilização deste método de programação do DSP exige a instalação no computador que será utilizado dos Drivers do DSP fornecidos pelo fabricante, do Code Composer Studio que é o compilador utilizado na placa de desenvolvimento da Spectrum digital que está sendo utilizada, do MatLab com simulink e dos toolboxes do MatLab: Real Time Workshop e Neural Network Toolbox.

Para o Real Time Workshop funcionar o programa escrito no simulink não deve possuir qualquer interação com o ambiente MatLab, isto é, não podem ser utilizadas funções definidas no Matlab, não podem ser acessadas variáveis, etc...

Por este motivo, a rede neural é gerada e treinada no MatLab seguindo os procedimentos relatados acima. Após o treinamento e teste da rede, é gerado um bloco no simulink que corresponde à rede. Este bloco torna-se então, independente do MatLab, rodando apenas no Simulink, para que possa ser compilado pelo Real Time Workshop. O comando utilizado para este fim está descrito abaixo:

```
gensim(net,st)
```

O real time workshop cria alguns blocos no Simulink que contém os ajustes e geram os códigos de programação para o DSP. Neste caso foi utilizado o bloco mostrado na Figura 15 abaixo, ele não recebe qualquer entrada ou saída diretamente, mas deve estar presente no programa criado no simluink para que a partir deste possa ser gerado um código compatível com o Code Composer Studio.

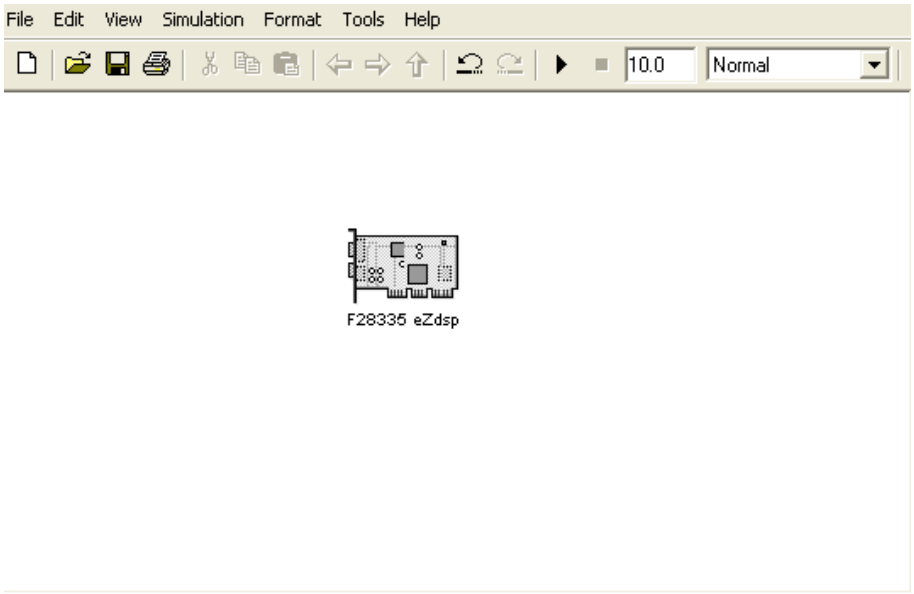

Figura 15 - Bloco do DSPF28335 no Simulink

Fluxo de dados no sistema:

Na Figura 16 abaixo há um diagrama esquemático que exemplifica o fluxo de dados no sistema completo. O sinal de voz durante a fase de treinamento da rede neural é direcionado ao treinamento e ao modelo da rede.

Após a rede estar devidamente treinada e pré testada por uma rotina de testes aleatórios implementada no MatLab, os sinais de voz são direcionados à comparação de padrões que ocorre na rede neural para que a rede dê uma classificação para o padrão. Neste caso “direita” ou “esquerda.

O lado esquerdo (Recognition Phase) pode ocorrer no ambiente MatLab ou no sistema embarcado na placa de desenvolvimento ezDSP.

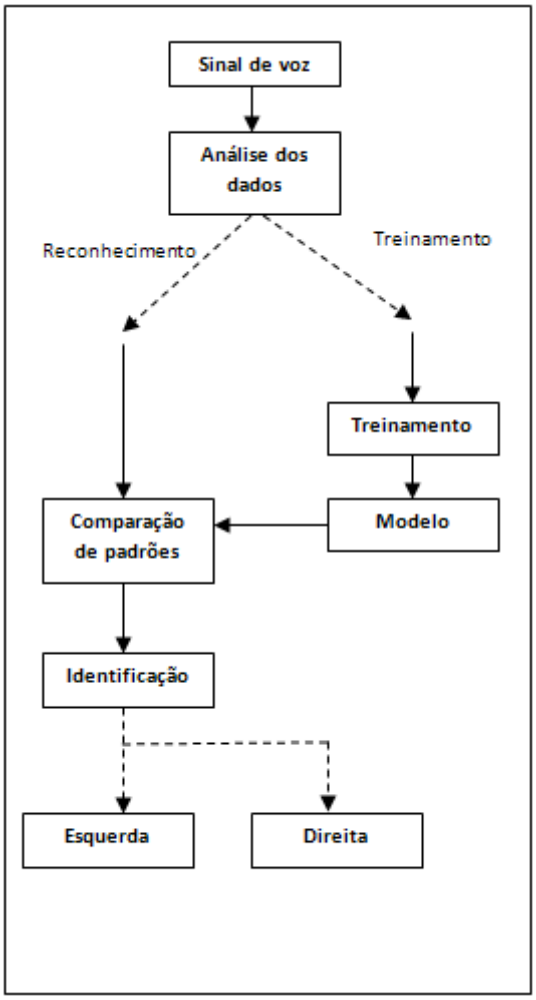

Figura 16 - Fluxo de dados no sistema

Conclusões:

Durante a execução deste trabalho foram testados e desenvolvidos hardwares como a placa de pré-amplificação aonde foram aplicados conceitos apresentados nas aulas de circuitos elétricos e eletrônica analógica. Além disto, houve extensa pesquisa acerca das possibilidades e funcionalidades do software MatLab e do DSP TMS320F28335 assim como testes e estudos sobre sua programação.

A rede neural demonstrou-se eficaz e eficiente na classificação dos padrões “direita” e “esquerda” que foram objetivo deste trabalho. Após devidamente treinada foram obtidos resultados similares aos descritos na literatura científica nos últimos anos, o que encoraja a continuar o desenvolvimento do presente trabalho.

Devido aos bons resultados obtidos com o uso da rede neural para a classificação dos padrões, não houve a necessidade de testar a implementação dos outros métodos de classificação descritos neste trabalho.

Apesar de todo o desenvolvimento e estudo, até o presente não houve a implementação da rede neural no DSP. Contudo, foi verificada a real possibilidade de fazê-lo através dos estudos realizados e do levantamento de papers aonde foram descritas implementações similares executadas com êxito. Foram ainda feitos diversos testes de programação bem sucedidos do DSP pelo MatLab seguindo o método apresentado acima.

Referências bibliográficas:

Austin, M., ***Artificial Neural Network for Speech Recognition***, 2nd Annual Student Research Showcase, (2003)

S. Gannot, V.Avrin, ***A Simulink and Texas Instruments C6713 Based Digital Signal Processing Laboratory***, Bar-Ilan University, Israel

Demuth , H., ***Neural Network Toolbox for use with MATLAB***, The MathWorks.

PUC-MG, ***Curso de Redes Neurais Artificiais***, Engenharia de Controle e Automação

C. Duran, ***Ultimate Trends in Integrated Systems to Enchance Automatic Speech Recogniton Performance***, University of Pamplona, Colombia

Moutinho, A ,***Introdução ao uso de redes neurais com Matlab,*** (2004)

Symons, ***Signals and Systems***

Pavani , A., ***Notas de aula do curso de sinais e sistemas***, PUC-Rio

Velasco, M., ***Notas de aula do curso de inteligência computacional aplicada***, PUC-Rio

Ding, P.; He, L.; Yan, X.; Zhao, R.; Hao, J., ***Robust Technologies towards Automatic Speech Recognition in Car Noise Environments,*** Signal Processing, 8th International Conference on, pp. 16-20, ISBN: 0-7803-9737-1, Beijing, (2006).

Gómez, A.J. , ***Diseño e Implementación de un Sistema de Reconocimiento de Patrones de Voz Basado en un DSP Blackfin***, XII simposio de tratamiento de señales, imágenes y visión artificial. STSIVA, Barranquilla (Colombia), (2007).

Jiankun, H.Z.X.; Jennings, A.; Lee, H.Y.J.; Wahyudi, Kota, R.; Abdelhamied, K.A.; Goshorn, E.L. (1993). ***Isolated word recognition of deaf speech using artificial neural networks***, Biomedical Engineering Conference, New Orleans, (1993)

Lim, C.P.; Woo, S.C.; Loh, A.S.; Osman, R. (2000). ***Speech Recognition Using Artificial Neural Networks***, First International Conference on Web Information Systems Engineering (WISE'00), Volume 1, pp. 419, ISBN: 0-7695-0577-5-1, IEEE Computer Society, (2000)

Texas Instruments, ***TMS320F28335 datasheet***

http://www.mathworks.com/matlabcentral , acessado em 07/2010

http://e2e.ti.com/support/microcontrollers/tms320c2000_32-bit_real-time_mcus/f/171/tags/28335/default.aspx , acessado em 05/2010