VITOR HIRAYAMA

**Classificador de Qualidade de Álcool
Combustível e Poder Calorífico de Gás
GLP**

Dissertação apresentada à
Escola Politécnica da
Universidade de São Paulo
para obtenção do Título de
Mestre em Engenharia

SÃO PAULO
2004

VITOR HIRAYAMA

**Classificador de Qualidade de Álcool
Combustível e Poder Calorífico de Gás
GLP**

Dissertação apresentada à
Escola Politécnica da
Universidade de São Paulo
para obtenção do Título de
Mestre em Engenharia

Área de Concentração:
Microeletrônica

Orientador:
Prof. Dr.
Walter Jaimes Salcedo

SÃO PAULO
2004

FICHA CATALOGRÁFICA

Hirayama, Vitor

**Classificador de qualidade de álcool combustível e poder
Calorífico de gás GLP / Vitor Hirayama-- São Paulo, 2004.
82 p.**

**Dissertação (Mestrado) – Escola Politécnica da Universidade
de São Paulo. Departamento de Microeletrônica**

**1. Redes neurais 2. Reconhecimento de padrões 3. Fuzzy
4. Componentes principais (análise) 5. Combustíveis gasosos
(classificação) I. Universidade de São Paulo. Escola Politécnica.
Departamento de Microeletrônica II.t**

Aos meus pais e familiares, que tanto me
apoiam na realização deste trabalho.

AGRADECIMENTOS

Gostaria de agradecer o apoio do meu orientador e amigo prof. Walter Jaimes Salcedo, e também ao meu amigo prof. Francisco Javier Ramirez-Fernandes por dividir o seu imenso intelecto para que fosse possível realizar esta obra.

Agradeço aos meus familiares, pelo suporte e incentivo. À empresa Alstom Brasil LTDA. Unidade Casa Verde, por me incentivar e permitir a realização deste trabalho.

Também agradeço a todos os meus amigos do grupo SIM: Germán, Daniel, Sílvia, Wanderson, Maurício, Gérson, Henrique, Thiago, Christian e Paulo; além de outras pessoas que diretamente ou indiretamente me ajudaram a realizar esta Dissertação.

Resumo

Este trabalho apresenta os resultados obtidos com o desenvolvimento de um sistema robusto como uma alternativa de reconhecimento da qualidade de vapor de álcool combustível e do poder calorífico do gás combustível GLP em um nariz eletrônico. Foram implementadas duas metodologias experimentais para a extração de atributos dos padrões de vapor de álcool combustível e de gás GLP. Na primeira abordagem de tratamento dos dados, foram usados um Sistema de Inferência Fuzzy (FIS), e dois algoritmos de treinamento de Redes Neurais Artificiais (RNA) para reconhecer padrões de vapor de álcool combustível: a Backpropagation e Learning Vector Quantization. A segunda abordagem para o tratamento dos dados foi desenvolver um sistema reconhecedor do poder calorífico do gás GLP robusto à perda aleatória de um dos sensores. Foram usados três sistemas. No primeiro foi implementada uma RNA para reconhecer todos os dados que simulavam a falha de um sensor aleatório. O resultado desse sistema foi de 97% de acertos. O segundo implementou sete RNA's treinadas com subconjuntos dos dados de entrada, tais que seis RNA's foram treinadas com um sensor diferente com falha; e a sétima RNA foi treinada com dados dos sensores sem falhas. O resultado desse sistema foi de 99% de acertos. O terceiro implementou uma Máquina de Comitê Estática Ensemble constituída de dez RNA's em paralelo para resolver o problema. O resultado foi de 97% de acertos. As RNA's tiveram melhores respostas que os FIS. Foram sugeridas algumas formas de implementação em hardware do sistema reconhecedor em sistemas pré-fabricados com DSP's e micro-controladores.

Abstract

This work shows the results of a robust system development as an alternative to recognize the quality of an alcohol fuel vapor sample and Liquid Petrol Gas (LPG) heat power in an electric nose. Two experimental methodologies were implemented to extract the features of alcohol fuel vapor and LPG gas patterns. The first approach to process the data used an Fuzzy Inference System (FIS) and two training algorithms of Artificial Neural Networks (ANN) to recognize alcohol fuel vapor patterns: Backpropagation and Learning Vector Quantization. The second approach consists of process data to develop an LPG heat power recognizing system robust to one-random-sensor-loss. Three systems were used. The first implemented an ANN to recognize all data that simulated the failure of a random sensor. This system had 97% of right responses. The second implemented seven ANN's trained with input data subsets, such that six ANN's were trained with a different failure sensor, and the seventh ANN was trained with data of all sensors without failure. This system had 99% of right responses. The third implemented an Ensemble Static Learning Machine containing ten parallel RNA's to solve the problem. The result were 97% of right responses. RNA's had better results than FIS. Some ways of hardware implementation of the recognizing system were suggested in DSP and micro-controllers pre-built systems.

SUMÁRIO:

LISTA DE TABELAS

LISTA DE FIGURAS

1	Introdução.....	1
1.1	Objetivos	1
1.2	Justificativa	2
1.3	Estrutura do Texto	3
2	Fundamentos Teóricos	6
2.1	Princípio de Funcionamento do Sensor de Gás Taguchi	6
2.2	Sensores que utilizam tecnologias comuns e não-comuns	8
2.3	Sensores inteligentes	9
2.3.1	Sensores que utilizam Redes Neurais Artificiais	9
2.3.2	Sensores que utilizam Lógica Difusa	11
2.4	Sensores inteligentes para reconhecer a qualidade do álcool	13
3	As Redes Neurais Artificiais	15
3.1	As Máquinas de Comitê	21
3.2	A Lógica Fuzzy.....	23
4	Metodologia Experimental.....	27
4.1	Procedimento para coleta da amostra de álcool combustível.....	29
4.2	Primeiro método de extração de características do gás GLP	31
4.3	Procedimento para extração e tratamento dos dados do GLP.....	33
4.3.1	Procedimento para coleta da amostra de gás GLP	33
4.3.2	Procedimento para efetuar as medições de gás GLP	35
5	Pré-Processamento dos Dados	37
5.1	Reconhecimento da qualidade de álcool com Redes Neurais	43
5.2	Implementação da RNA em DSP (Digital Signal Processor)	45
5.2.1	O programa implementado no DSP	47

5.2.2	Compilação e Linkagem	48
5.2.3	Implementação da Rede Neural em Hardware	48
5.2.4	Resultados do programa implementado no DSP	49
5.3	Implementação do Sistema de Inferência Fuzzy	50
5.3.1	FIS com os dados sem pré-processamento e PCA.....	51
5.3.1.1	Modelamento das Variáveis Lingüísticas	53
5.3.1.2	Base de Conhecimento do FIS	55
5.3.1.3	Metodologia utilizada para realizar os tratamentos dos dados	56
5.3.1.4	Resultados Obtidos com o Sistema de Inferência Fuzzy	57
5.3.2	FIS utilizando dados com pré-processamento e PCA.....	59
5.3.2.1	Modelamento das Variáveis Lingüísticas	61
5.3.2.2	Metodologia utilizada para realizar os tratamentos dos dados	63
5.3.2.3	Resultados Obtidos com o Sistema de Inferência Fuzzy	64
5.4	Reconhecimento do poder calorífico do GLP em Redes Neurais ...	65
5.4.1	Tratamento dos dados experimentais do GLP.....	68
5.4.2	Primeiro Experimento: Rede Neural Simples	70
5.4.3	Tratamento dos dados que simulam um sensor com falha	73
5.4.4	Segundo Experimento: RNA de sistema com um sensor em falha	73
5.4.5	Terceiro Experimento: Implementação das Máquinas de Comitê ...	74
5.4.6	Quarto Experimento: Redes Ensemble	75
6	Conclusões e Perspectivas Futuras	77
7	Bibliografia:	79

LISTA DE FIGURAS:

Fig. 2.1a: Características do Sensor Taguchi TGS822.....	7
Fig. 2.1b: Polarizações dos sensores Taguchi por divisão de tensão e a que evita a resistência por contatos.....	7
Fig. 2.2a: Assinaturas de cores a concentrações baixas de gases.	8
Fig. 2.3.2a: Reconhecimento de sinais de olfato com lógica fuzzy.	12
Fig. 2.3.2b: Análises feitas do sinal de olfato pelo sistema fuzzy.	12
Fig. 2.4a: Sistema para reconhecimento de qualidade de álcoois	13
Fig. 2.4b: Sistema para reconhecer a qualidade de álcool combustível.	14
Fig. 3a: Modelo de um neurônio artificial.....	16
Fig. 4.1a: Câmara de medidas.	29
Fig. 5.4a: Comportamento típico dos sensores na injeção da amostra.	32
Fig. 5.4b: Dados Sintéticos usados para a criação da Rede Neural e o FIS	32
Fig. 4.3.1a: Câmara de medidas.....	33
Fig. 4.3.1b: Sistema de coleta do gás	34
Fig. 4.3.1c: VI que controla a mistura de gás com nitrogênio.....	35
Fig. 5a: O aplicativo Data Sculptor	37
Fig. 5b: Gráfico obtido com o Data Sculptor antes do pré-processamento...	38
Fig. 5c: Gráfico obtido com o Data Sculptor após o pré-processamento.....	41
Fig. 5.1a: Variação da resistência versus resistência inicial (sensor 6)	45
Fig. 5.3a: Clusterização Subtrativa utilizada no FIS	51
Fig. 5.3.1a: Dados sintéticos de cada sensor usados no FIS.....	52
Fig. 5.3.1b: Sistema Fuzzy com os dados sem pré-processamento e PCA.	53
Fig. 5.3.1.1a: Função de Pertinência da Variável Resistência Inicial	54
Fig. 5.3.1.1b: Funções de Pertinência ajustadas para cada sensor.....	55
Fig. 5.3.1.1c: Funções de pertinência da saída do FIS.	55
Fig. 5.3.2a: Gráfico dos dois Componentes Principais com os Clusters dos dados de treino	60
Fig. 5.3.2b: Dados sintéticos de teste usados no FIS.	60
Fig. 5.3.2c: Sistema FIS utilizado com os dados de treino submetidos ao pré-processados e análise de componentes principais	61

Fig. 5.3.2.1a: Função de Pertinência da Variável Lingüística PrinComp1	61
Fig. 5.3.2.1b: Função de Pertinência da Variável Lingüística PrinComp2.....	62
Fig. 5.3.2.1c: Funções de pertinência da saída do FIS.	62
Fig. 5.3.2.3a: Resultado do FIS para os dados com pré-processamento e análise de componentes principais.	65
Fig. 5.4b: Amostras do Gás GLP e da mistura GLP com Nitrogênio.....	69
Fig. 5.4.2a: RNA usada no Reconhecimento dos dados experimentais	71
Fig. 5.4.2b: Exemplo de curva de erro obtido dos dados experimentais	71
Fig. 5.4.2c: Rede Neural com PCA treinada com os dados experimentais ...	72
Fig. 5.4.2d: Exemplo de curva de erro da RNA com dados experimentais ...	72
Fig. 5.4.5a: Primeira proposta de máquina de comitê	74
Fig. 5.4.6a: Máquina de Comitê Ensemble.....	76

LISTA DE TABELAS:

Tabela 4a: Detalhamento dos experimentos realizados	27
Tabela 4a: Compostos Detectados para cada sensor	28
Tabela 5a: Valores Desejados adotados para as saídas	38
Tabela 5b: Separação dos dados feita pelo aplicativo Data Sculptor	41
Tabela 5.1a: Resultados obtidos das RNA's no reconhecimento de álcool ..	45
Tabela 5.2a: Coeficientes utilizados no aplicativo Neural Works	46
Tabela 5.2b: Iterações necessárias para o treinamento da rede neural	46
Tabela 5.2.4a: Resultados obtidos nas simulações das amostras de álcool	50
Tabela 5.4a: Treinamentos das RNA's com Matlab	66
Tabela 5.4b: Treinamento das RNA's com Neural Works	67
Tabela 5.4.2a: Propriedades da RNA sem PCA	71
Tabela 5.4.2b: Parâmetros utilizados na RNA com PCA.....	72
Tabela 5.4.6a: Resultados de RNA simples ou máquinas de comitê, que simulavam a falha de um dos sensores.	76

Acrônimos, Abreviaturas e Notações Utilizadas:

ID – identification number – número de identificação do equipamento.

η – Taxa de aprendizado;

α – Momentum. Os pesos são atualizados na proporção do erro (e) e as entradas na conexão (x). O termo momentum é usado para ajudar a suavizar as mudanças dos pesos.

θ – Limiar ou threshold. O aprendizado pode ser limitado a um limiar de erro, que corresponde ao valor de θ . Caso θ for positivo, qualquer valor absoluto de erro menores que θ serão ignorados.

I – Soma ponderada para os neurônios da camada corrente. Os componentes desse vetor são $I = (I_1, I_2, \dots, I_h)$, onde I_i é a soma ponderada do i -ésimo neurônio da camada corrente.

Y – Saída da camada corrente. É o resultado da computação da soma ponderada, função de transferência, e então a saída. As componentes de Y são $Y=(y_1, y_2, \dots, y_n)$, onde y_i é a saída do i -ésimo neurônio da camada.

D – Saída desejada para a camada corrente. A estratégia de controle usualmente assume que d_i está contido no campo erro do i -ésimo neurônio.

E – Vetor de erro. Caso seja a camada de saída, o erro será o corrente ou o transformado pela derivada da função de transferência. Caso seja de outras camadas, o erro será acumulado, ou o erro transformado retropropagado.

X_i – Conexões de entrada para o i -ésimo neurônio da camada. São as saídas dos neurônios nos quais o neurônio corrente é conectado.

W_i – Vetor inicial dos pesos na camada. Somente pesos variáveis participam do processo de aprendizado. Os outros tipos (fixed, set & mod) não aprendem.

W_i' – Vetor de erro depois que ele foi atualizado pela regra de aprendizado.

M_i – Memória da última mudança nos pesos. Muitas regras de aprendizado chamam esse termo de momentum.

A_i – Campo auxiliar do peso, que é usado para o momentum em algumas regras de aprendizado.

n – Número de entradas do neurônio corrente.

RNA – Abreviação de Redes Neurais Artificiais.

$[x]^1$ - Função especial definida como $[x]^1 = 1$ se $x > \alpha$; 0 caso contrário.

$[x]^p$ – Função definida como $[x]^p = 1$ se $x > 0$; 0 caso contrário.

e-nose – Abreviação de Eletronic Noses. São os chamados narizes eletrônicos.

FIS – Abreviação de Fuzzy Inference System, ou Sistema de Inferência Fuzzy.

GLP – Abreviação para Gás Liquefeito de Petróleo

MLP – Abreviação para Multilayer Perceptron ou Perceptron de Múltiplas Camadas.

PCA – Abreviação de Principal Component Analysis ou Análise de Componentes Principais.

SOM – Abreviação para Self Organizing Maps, ou Mapas Auto Organizáveis.

1 Introdução

A possibilidade de realizar medidas diretas com poucos refinamentos e facilidade de implementação [1], levou ao crescimento do interesse dos chamados narizes eletrônicos, tanto no meio acadêmico como na indústria. Um nariz eletrônico tem diversas aplicações, tais como reconhecimento de cheiros de alimentos sólidos e líquidos [2, 3, 4], perfumes e reagentes químicos [4], detecção de câncer de pulmão através do ar expirado do paciente [5], medida do teor alcoólico do hálito de um motorista [1], monitoração da qualidade de água potável [6], reconhecimento de padrões de combustíveis [7], entre outros.

Um nariz eletrônico é em geral implementado com o auxílio de Redes Neurais Artificiais, por causa da sua robustez a ruídos que podem estar presentes nas amostras analisadas [8], além de sua grande capacidade de generalização, o qual promove a inferência de reconhecimento de amostras novas, muitas vezes correta, fora do universo de amostras usadas na fase do seu treinamento. A aplicação discutida neste trabalho será o reconhecimento da qualidade do vapor de álcool e do poder calorífico do gás GLP.

Quando há muitos atributos a serem analisados, em geral é usada a análise de componentes principais para reduzir a dimensionalidade do sistema, de modo a manter o máximo de informação presente nos dados [1, 3, 4]. As medidas obtidas pelos seis sensores foram analisadas em três abordagens: dados crus, dados submetidos à análise de componentes principais, e dados que simulam um sensor com falha.

Os métodos tradicionais de obtenção do poder calorífico do gás combustível podem ser divididos em três categorias [9]: a combustão de uma amostra gasosa em uma bomba calorimétrica, combustão do gás em um queimador de chama aberta, e a combustão catalítica sem chama. Em geral esses métodos requerem um maquinário caro. Um sistema embarcado de reconhecimento que utiliza sensores não seletivos e de baixo custo pode ser uma alternativa com uma boa relação custo-benefício para classificar o poder calorífico de um dado gás combustível.

1.1 Objetivos

O primeiro objetivo deste trabalho é reconhecer o poder calorífico de uma amostra de gás combustível. As amostras utilizadas serão de vapor de álcool e de gás combustível GLP, a partir de padrões diferentes. As redes neurais serão estudadas como

uma alternativa de solução ao problema de reconhecimento. As resistências dos sensores de gases Taguchi serão os atributos utilizados neste trabalho.

O segundo objetivo é implementar um sistema robusto capaz de reconhecer um dado padrão mesmo com a perda de um dos sensores de gás, isto é, quando algum dos sensores parar de responder a estímulos da entrada do sistema. Com isso, poderá ser simulada uma possível falha de um dos sensores de gás. Para tentar contornar este problema serão utilizadas as redes neurais em comitê [8,10], pelo método de Ensemble e será usado um sistema de sete redes neurais treinadas com subconjuntos dos dados cada um simulando nenhum ou um sensor com falha.

Será discutida também a lógica Fuzzy como substituto da rede neural para solucionar o problema. A abordagem utilizada neste trabalho será a implementação da clusterização subtrativa nos dados de entrada do sistema.

Também serão discutidos o pré-processamento e a análise de componentes principais dos dados de treinamento da rede, e qual o impacto desses métodos na etapa de treinamento da rede neural e também na generalização da mesma.

A implementação em hardware da solução obtida também será discutida. O reconhecimento dos padrões de álcool combustível com uma rede neural será implementado em uma placa DSP da Analog Devices. Será analisado o uso de um dispositivo dedicado baseado em um microcontrolador.

1.2 Justificativa

Atualmente a análise dos compostos presentes em um determinado gás combustível é feita em geral com o auxílio dos cromatógrafos. Os resultados da análise de um cromatógrafo são as proporções dos diversos componentes do combustível em análise. Estas proporções são convertidas em quantidades, que são aplicadas em fórmulas matemáticas para obter o poder calorífico do combustível em questão.

Apesar do aparelho cromatógrafo ser muito preciso, ele é muito caro. O problema proposto neste trabalho é de reconhecer o poder calorífico de um dado gás combustível. Deste modo, um conjunto de sensores de concentrações de gás de baixo custo e pouco seletivo poderá ser uma alternativa muito mais barata. Como o sistema reconhecedor não precisa ser muito preciso, a relação custo – benefício desta alternativa poderá ser viável.

Este trabalho terá como ênfase o uso das redes neurais artificiais e a lógica fuzzy para a etapa de reconhecimento e condicionamento de sinais do sistema embarcado reconhecedor de padrões de vapor de álcool e de GLP. Essas novas tecnologias estão ganhando espaço no mercado de consumo, pelos seus resultados satisfatórios [22].

1.3 Estrutura do Texto

Esta Dissertação está dividida em 6 capítulos organizados com a estrutura detalhada a seguir.

Capítulo 1: Introdução

- Descrição do problema a ser resolvido, detalhamento dos objetivos, e justificativas do trabalho proposto.

Capítulo 2: Fundamentos Teóricos

- Estudo do funcionamento dos sensores de gases utilizados no experimento.
- Pesquisa de sensores tradicionais e não tradicionais para o reconhecimento de padrões de combustíveis.

Capítulo 3: As Redes Neurais Artificiais

- Breve introdução das Redes Neurais Perceptron de Múltiplas Camadas e SOM, e suas principais características.
- Introdução ao pré-processamento dos dados e à análise de componentes principais.
- Breve explicação das abordagens de sistemas de Redes Neurais Artificiais e sistemas de Lógica Difusa para resolver o problema de reconhecimento de padrões.

3.1 As Máquinas de Comitê

- Introdução breve das máquinas de comitê estáticas: Média de Ensemble e Reforço (Boosting)

3.2 A Lógica Fuzzy

- Breve introdução das definições e principais características da lógica Fuzzy. Serão descritos os funcionamentos de um controlador lógico fuzzy e de um sistema de inferência fuzzy.

Capítulo 4: Metodologia Experimental

- Descrição dos distintos procedimentos utilizados para coletar as amostras de álcool e de gás combustível.

Capítulo 5: Pré-Processamento de Dados

- Descrição do uso do aplicativo Data Sculptor para realizar o pré-processamento dos dados obtidos do álcool combustível.
- Implementação do pré-processamento de dados e análise de componentes principais através do aplicativo Matlab para os dados obtidos do gás GLP.

5.1 Reconhecimento da qualidade de álcool com Redes Neurais

- Descrição da implementação do sistema reconhecedor da qualidade do vapor de álcool combustível utilizando um Kit de DSP da Analog Devices.
- Análise dos resultados obtidos e limitações da implementação do sistema no hardware.

5.2 Implementação do Sistema de Inferência Fuzzy

- Implementação do FIS para resolver o problema de reconhecimento do poder calorífico do gás GLP.
- Análise dos resultados obtidos com o sistema FIS utilizando os dados com ou sem pré-processamento e análise de componentes principais.

5.3 Reconhecimento do poder calorífico do GLP em Redes Neurais

- Descrição das transformações dos dados de entrada do sistema para que eles possam ser usados nas Redes Neurais Artificiais.

- Implementação da primeira abordagem de RNA em que foram usados somente os dados experimentais originais de treinamento. Foi feita a análise das redes com e sem pré-processamento e análise de componentes principais.
- As três outras abordagens de RNA's utilizaram dados sintéticos que simulavam o comportamento de falha de um dos sensores. A primeira delas utilizou uma rede neural para realizar todo o reconhecimento. Já as outras duas abordagens implementaram dois tipos de máquinas de comitê estáticas.

Capítulo 6 Conclusões e Perspectivas Futuras

- Análise global dos resultados obtidos experimentalmente e balanço do que foi desenvolvido neste trabalho.
- Listagem de algumas idéias a serem desenvolvidas para enriquecer ou melhorar as atividades desenvolvidas.

Bibliografia:

Lista de referências bibliográficas utilizadas.

2 Fundamentos Teóricos

2.1 Princípio de Funcionamento do Sensor de Gás Taguchi

O sensor de gás é feito de um óxido de metal (SnO_2 , por exemplo). Ao se aquecer um óxido de metal, moléculas de gás oxigênio são adsorvidos na superfície do mesmo com uma carga negativa. Há então a formação de uma camada de depleção na superfície do metal, fazendo com que sua resistência aumente. Isso porque é criada uma barreira de potencial, impedindo o fluxo de elétrons pelo metal [12].

Na presença de um gás (substância redutora), a densidade de moléculas de gás oxigênio carregadas negativamente diminui, fazendo com que a barreira de potencial seja reduzida. E, conseqüentemente, a resistência do sensor também diminui.

A relação entre a resistência do sensor e a concentração do gás é dada pela equação (2.1a) sobre uma determinada faixa de concentração de gás:

$$R_s = A [C]^{-\alpha} \quad (2.1a)$$

Onde:

R_s = Resistência elétrica do sensor.

A = Constante.

$[C]$ = Concentração do gás.

α = Inclinação da curva R_s .

Características dos sensores de gás

Os sensores de gás Taguchi têm as seguintes características [12]:

- A sua resistência diminui com concentrações reduzidas de oxigênio;
- Sua sensibilidade varia de acordo com o gás usado;
- Ao ser exposto a um gás, sua resistência varia rapidamente. Quando o gás é retirado, a variação para o estado inicial é mais lenta.

A taxa das reações químicas feitas na superfície do sensor depende da temperatura. A umidade também afeta o funcionamento do sensor, por causa da possível adsorção de vapor de água na superfície do óxido de metal.

As curvas características relativas ao sensor TGS 822 demonstram essas características. O sensor tem sensibilidade diferente para cada substância que é aplicada ao sensor, como mostrado na Fig. 2.1a, que mostra R_s/R_o em relação à concentração de gás e R_s/R_o em relação à temperatura do ambiente. R_s é a resistência do sensor após ele ter sido submetido ao gás e estar em regime permanente; e R_o é a resistência inicial do sensor em regime permanente.

Fig. 2.1a: Características do Sensor Taguchi TGS822

Fig. 2.1b: Polarizações dos sensores Taguchi por divisão de tensão e a que evita a resistência por contatos

Há duas formas de polarizar o sensor Taguchi [13], conforme mostrado na Fig. 2.1b: da esquerda para a direita são apresentadas a polarização por divisão de tensão, e a polarização que evita resistência por contatos. A polarização por divisão de tensão é a mais usual. Ela faz a ligação em série do sensor com uma resistência de carga. A tensão a ser medida será obtida pela queda de tensão no resistor de carga. A outra forma de polarização do sensor faz uso da tensão medida, obtida diretamente da queda de tensão no sensor. Essa outra configuração de polarização evita o sério problema de resistência de contato nas medidas de tensão do sensor de gás. O fabricante fornece as seguintes especificações do sensor Taguchi:

- Tensão de polarização do sensor (V_s): 24V_{máx.} AC ou DC
- Tensão para o aquecimento do sensor (V_a): $5,0 \pm 0,2V$ AC ou DC

2.2 Sensores que utilizam tecnologias comuns e não-comuns

O uso de sensores comuns é a mais barata e intuitiva solução do problema de reconhecimento de gases. Porém, como os sensores em geral não são lineares com relação à concentração de gases detectados, há a necessidade de um condicionamento dos sinais medidos. Ou seja, quando é necessário maior precisão, a linearização dos dados medidos é essencial.

Com a incorporação de circuitos de condicionamento de sinais, o projeto do sistema tradicional começa a ficar cada vez mais complicado e caro. Há vários fatores a serem considerados, como o ambiente de operação do sistema, as especificações técnicas do sensor utilizado, a precisão requerida do sistema, etc.

Em contraste, há sensores em estado da arte que utilizam tecnologias não tradicionais. São os sensores que reconhecem odores a partir da visualização de cores em uma matriz de amostras de metais específicos [14]. A resposta química seletiva de um conjunto de vapores imobilizados em um grande conjunto de metais especiais permite a visualização de um grande conjunto de gases (álcoois, aminas, éteres, dentre outros). Ou seja, é um sensor de análises qualitativas [15], que reconhece a identidade do gás utilizado, e não reconhece quantidades individuais. A Fig. 2.2a abaixo mostra a resposta de uma matriz minimizada de 4 metais especiais (Sn(TPP)(Cl₂), Co(TPP)(Cl), Zn(TPP) e Fe(TFPP)(Cl), vistos no sentido horário, de cima para baixo) quando eles foram submetidos aos gases n-octilamina, dodecanetiol, e tri-n-butilfosfina a 1.8 p.p.m.

Fig. 2.2a: Assinaturas de cores a concentrações baixas de gases.

Vapores de gases não afetam a performance do dispositivo, o qual mostra boa resposta linear a amostras. Assinaturas de cor únicas podem ser obtidas desde concentrações de amostras abaixo de duas partes por milhão até respostas de 100 partes por milhão.

A única desvantagem destes sensores – e a que os impossibilita de serem utilizados em sistemas de controle tradicionais – é que eles oferecem somente respostas visuais. Não é possível captar sinais elétricos para odores aplicados à matriz de sensores. Uma forma de utilizar esses sensores seria monitorá-los através de câmeras de vídeo. O que não compensaria, pois o sistema ficaria mais complexo, e seria necessário um sistema de reconhecimento de imagens para automatizar o sistema.

2.3 Sensores inteligentes

Os sensores inteligentes têm várias funcionalidades adicionais [16] aos sensores comuns, e são capazes de tomarem decisões por si próprios. Os sistemas de sensores inteligentes devem realizar ao menos uma das seguintes atividades listadas a seguir:

- **Calibração automática:** realizar sozinhos a sua calibração.
- **Processamento do sinal:** deve ser capaz de tratar o sinal de medição de forma a fornecer seu valor final.
- **Tomada de decisões:** podem tomar decisões para realizar ações que fazem com que o sensor cumpra a sua tarefa, independentemente do comando principal do sistema.
- **Fusão com outros sensores:** podem ser combinados ou adicionados sinais de outros sensores a fim de melhorar a robustez do resultado.
- **Capacidade de aprendizado:** pode aprender com experiências passadas, a fim de aumentar sua performance e robustez.

Um sensor inteligente pode ser descrito como um sistema composto de uma combinação de sensores convencionais, processamento de sinais, métodos de extração de características, algoritmos de aprendizado ou métodos de representação de conhecimento integrados. A inteligência do sensor inteligente pode ser implementada com técnicas de redes neurais, lógica difusa, algoritmos genéticos, sistemas especialistas ou árvores de decisão.

2.3.1 Sensores que utilizam Redes Neurais Artificiais

Sistemas embarcados de reconhecimento de padrões baseados em redes neurais artificiais em geral utilizam sensores comerciais comuns. Portanto, as condições de operação do sistema reconhecedor obedecem às condições de operação do sensor

utilizado. As redes neurais servem basicamente para substituir as etapas de modificação e condicionamento de sinais.

Houve vários estudos de aplicações de sensores que utilizam tecnologia de redes neurais para o reconhecimento do gás [13, 17, 18, 19 e 20]. Em [21] é enfatizado que muitos modelos de redes neurais usuais, tais como o MLP, não podem ser usados para análises quantitativas de gases. Porém eles podem ser usados para reconhecimento de padrões que se aplica na área de qualificação de análise química. Ou seja, para reconhecer qual gás foi submetido à matriz de sensores, com uma determinada quantidade de gás aplicada.

Portanto, o modelo da rede neural artificial MLP deve ser modificado para tratar o problema real de quantificação da concentração de gases. Vários estudos foram feitos para modificar a rede neural, dentre elas [21] recomenda o uso do conceito do ‘ChemNet’, em que teorias químicas são incorporados na estrutura da rede neural, fazendo com que a rede guarde um conhecimento a priori quando um modelo é construído. Também foi apresentado o conceito químico de erro relativo em análises quantitativas, e discutido a sua aplicação na análise da mistura gasosa real.

Porém, o problema objeto de estudo deste trabalho não envolve análise quantitativa. Não serão criadas várias saídas da rede neural para medir com exatidão a concentração das amostras de gases que são submetidas à rede neural. Para o caso do reconhecimento de vapor de álcool combustível, por exemplo, será criada somente uma saída da rede neural indicando nível lógico “1” caso a amostra de álcool tenha concentrações entre 92.6% e 94.3% (álcool de boa qualidade), e nível lógico “0” caso contrário (álcool de má qualidade ou desperdício). A rede neural artificial MLP, portanto, reconhecerá dois padrões: álcool bom ou álcool ruim.

O treinamento de uma rede neural MLP é um processo randômico. Primeiramente, escolhe-se aleatoriamente os pesos iniciais para as sinapses. As amostras são apresentadas à rede uma a uma em ordem aleatória, ou em lotes. Portanto, a especificação de um sistema que utiliza redes neurais é difícil de ser obtida. É necessária uma avaliação estatística dos experimentos de redes neurais [22]. Deve-se treinar várias vezes uma rede neural, separando aleatoriamente o conjunto de dados disponíveis em amostras de treino e de teste, e somente após vários experimentos, pode-se estimar uma distribuição gaussiana de características da rede neural. Mas isso

ocorrerá, somente se os valores randômicos utilizados no treinamento da rede neural seguirem uma distribuição gaussiana, ou qualquer outra distribuição definida.

Porém, uma vez treinados os parâmetros do sensor inteligente por Redes Neurais Artificiais, o seu uso é trivial. Isso porque a arquitetura da rede e os pesos sinápticos já estão determinados. Somente é necessário inserir os dados medidos nas entradas da rede neural, e as saídas são obtidas por meio de simples contas matemáticas de adição e multiplicação com números reais.

2.3.2 Sensores que utilizam Lógica Difusa

Um exemplo de sistema que utiliza tecnologia de lógica difusa é apresentado em [23]. Trata-se de um reconhecedor de sinais de olfato baseado em lógica fuzzy, que usa sensores de gases poliméricos. Neste sistema, os sinais de olfato são amostrados com uma frequência de amostragem de 2Hz, e o valor inicial de cada sinal é subtraído de cada amostra do sinal. Esse pré-processamento simples elimina os efeitos de deriva de corrente produzidos pela dopagem e temperatura dos sensores poliméricos condutores. E os sinais são analisados pelo reconhecedor fuzzy.

O sistema está mostrado na Fig. 2.3.2a. É composto por dois reconhecedores: o **reconhecedor baseado na forma do sinal**, e o **reconhecedor baseado na faixa dinâmica**. O reconhecedor baseado na forma é formado por um **pré-processador lingüístico**, um **repositório de modelos lingüísticos** e um **comparador**. O pré-processador lingüístico converte a resposta da matriz de sensores em uma representação lingüística do odorante a ser testado. A transformação é baseada na partição lingüística do espaço ocupado pelo sinal. O comparador compara essa representação lingüística com a representação contida no repositório de modelos lingüísticos. Todos os modelos contidos no repositório de modelos lingüísticos são obtidos na fase de treinamento. Detalhes do reconhecedor baseado na forma (Shape-Based Recogniser) estão mostrados na figura da direita.

As variações em amplitude do sinal produzidas pelos odorantes podem ser exploradas para determinar a identidade do odorante utilizado. O reconhecedor baseado na faixa dinâmica constrói um modelo fuzzy para tentar detectar o odorante de acordo com sua resposta em amplitude do sinal.

Fig. 2.3.2a: Reconhecimento de sinais de olfato com lógica fuzzy.

Pela Fig. 2.3.2b pode-se notar as definições de funções de pertinência para a amplitude do sinal e também para o eixo do tempo. Mostrou-se que esse sistema obteve uma taxa de acerto entre 85 e 93%.

Fig. 2.3.2b: Análises feitas do sinal de olfato pelo sistema fuzzy.

Uma outra abordagem que utiliza um sistema FIS, e que foi usado neste trabalho para resolver o problema de classificação do poder calorífico do gás combustível, foi feita de modo que a base de conhecimento do FIS fosse extraída a partir dos próprios dados sintéticos.

A especificação de um sistema que utiliza lógica fuzzy depende de diversos fatores, tais como: funções de pertinência utilizadas, conjunto de regras fuzzy implementadas, o método de defuzzificação do sistema, método da intersecção dos conjuntos fuzzy, entre outros. São escolhidos os parâmetros que melhor resolvem o problema proposto. Para um sistema de controle, a especificação de um sistema que

utiliza lógica fuzzy é feita por características externas do mesmo, como por exemplo: tempos de resposta, amortecimento do sinal, ativação, entre outros.

Após as definições dos parâmetros do sistema fuzzy utilizado, o uso do sistema também é trivial. Porém, dependendo da estratégia de defuzzificação utilizada, pode-se exigir maior poder computacional, pois as contas matemáticas empregadas poderão não ser triviais. Operacionalmente, somente será necessário apresentar os dados de entrada; e os dados de saída são produzidos pelo sistema.

2.4 Sensores inteligentes para reconhecer a qualidade do álcool

Em [13] foi proposto um sistema para reconhecimento de substâncias aromáticas aplicadas a uma matriz de sensores de gás. O sistema foi baseado em um microcontrolador 80C552 e um conjunto de sensores de gás de óxido de estanho. As respostas dos sensores foram analisadas com o auxílio da transformação rápida de Fourier (FFT), e foram efetuados ensaios com substâncias químicas em dois grupos diferentes. O primeiro grupo de substâncias utilizadas foram álcool de arroz, álcool de erva doce, mistura de álcool com erva doce e mistura de álcool de arroz com anis. O segundo grupo foi composto por aguardentes brasileiras e pisco chileno.

O aparato experimental utilizado foi composto de uma câmara de ensaio, um conjunto de rotâmetros para controlar o fluxo de gases Nitrogênio e Oxigênio e um sistema de aquisição de dados, como mostra a Fig. 2.4a. Dentro da câmara de ensaio foi colocada uma placa de circuito impresso com o conjunto de sensores de gás, umidade e temperatura. A tensão sobre os sensores foi medida com uma placa de conversão A/D e os dados gravados em um computador pessoal. O reconhecimento foi realizado com redes neurais artificiais treinadas com diferentes taxas de aprendizado.

Fig. 2.4a: Sistema para reconhecimento de qualidade de álcoois

O sistema adotado neste trabalho para o reconhecimento da qualidade de álcool combustível é similar ao descrito anteriormente. A proposta foi reconhecer vapores de álcoois com concentrações classificadas como bom (de concentrações entre 92,6% e 94,3% de álcool) ou ruim (concentrações fora desta faixa). O sistema é mostrado na Fig. 2.4b.

Fig. 2.4b: Sistema para reconhecer a qualidade de álcool combustível.

Amostras de vapor de gás são injetados na câmara por meio de uma seringa. A câmara contém uma matriz de seis sensores de gases. Por meio de um sistema de aquisição de sinais, são medidas as resistências dos sensores de gases antes e depois da injeção das amostras de gases. São então obtidos valores iniciais e finais de resistência dos sensores, e são obtidos também os atributos que serão usados como parâmetros de entrada do sistema de reconhecimento. Os dados obtidos passaram por uma etapa de pré-processamento de dados, onde houve uma normalização dos dados para valores no intervalo entre zero e um.

O sistema de reconhecimento escolhido foi a rede neural, por sua simplicidade de implementação, e robustez à perda de sensores (propriedade de generalização da rede neural). Os resultados obtidos tiveram acertos entre 80 e 95% das amostras utilizadas como teste.

3 As Redes Neurais Artificiais

Segundo [8], uma rede neural é um sistema processador paralelamente distribuído constituído de unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível para o uso. O conhecimento é adquirido pela rede através de um processo de aprendizado. O armazenamento do conhecimento é feito nas conexões entre neurônios, conhecidos como pesos sinápticos.

Em [24] é exposto que além do modelo do neurônio artificial, os paradigmas da rede incluem também a topologia da rede neural, ou seja, o modo como os neurônios estão interligados entre si; e os processos de aprendizado, que obedecem aos **algoritmos de aprendizado** descritas por expressões matemáticas, as **equações de aprendizado**. As equações de aprendizado descrevem as regras de aprendizado, que por sua vez dita como é feito o processo de auto-ajustar seus pesos sinápticos. O tipo de aprendizado normalmente utilizado é o **aprendizado supervisionado**, em que a resposta ao estímulo de uma entrada é comparada com seu respectivo valor desejado. Caso sejam diferentes, a rede gera um **signal de erro**, o qual é usado para calcular os ajustes que deverão ser feitos nos pesos sinápticos da rede. E esse processo necessita de um professor ou supervisor.

Antes de uma rede neural ser utilizada como reconhecedor de padrões, a mesma deverá ser treinada. E para isso, deve-se escolher uma regra de aprendizado. A regra normalmente utilizada é a chamada backpropagation. Trata-se de um aprendizado supervisionado, em que o erro obtido na saída da rede neural é utilizado para atualizar os pesos das entradas de todos os neurônios da rede neural.

Os dados de treinamento utilizados pela rede neural devem ser os mais representativos do sistema a ser projetado. Se forem escolhidos dados ruins na etapa de treinamento da rede, ela não realizará uma generalização adequada. Segundo [8], o termo generalização se refere à capacidade de uma rede neural conseguir acertar o reconhecimento de um dado padrão, utilizando amostras que não foram apresentadas à rede neural em sua fase de treinamento.

Dependendo da função de ativação utilizada pela rede neural, os dados de treinamento podem ser normalizados em um intervalo real entre 0 e 1 (para a função de

ativação de sigmóide, por exemplo) ou entre -1 e 1 (para a função de ativação de tangente hiperbólico, por exemplo). As saídas resultantes da rede neural deverão ser transformadas de modo a compensar a normalização imposta nos dados de entrada.

Após o treinamento da rede neural, ela se assemelha a um circuito lógico. Isso porque o uso da rede torna-se direto, necessitando somente apresentar as amostras nas entradas para que a rede neural produza as saídas, de acordo com o seu treinamento.

Modelo de um Neurônio

O neurônio é o processador básico de uma rede neural. Cada neurônio tem uma saída, o qual pode alimentar muitos outros neurônios. Cada neurônio recebe muitas entradas por essas conexões, chamadas sinapses. As entradas são multiplicadas pelos pesos das sinapses, e somadas entre si. A ativação do neurônio é computada aplicando uma função de limiar a essa soma. Um modelo abstrato é mostrado na Fig. 3a.

Fig. 3a: Modelo de um neurônio artificial

Funções de Ativação

A função de ativação é geralmente uma função não-linear. Uma função simples não linear e que é apropriada para redes neurais discretas é a função degrau. Uma variante da função degrau está descrita abaixo:

$$f(x) = \begin{cases} 1 & , \text{ se } x \geq 0 \\ -1 & , \text{ se } x < 0 \end{cases} \quad (3a)$$

onde x é a soma dos produtos das ativações dos neurônios de entrada com seus respectivos pesos sinápticos.

$$x = \sum_{i=0}^n A_i * w_i \quad (3b)$$

onde n é o número de neurônios de entrada mais o valor do bias de entrada, A é o vetor de neurônios de entrada, e w é o vetor dos pesos sinápticos que conectam os neurônios de entrada que estão sendo examinados.

Outra classe popular de função de ativação mais apropriada para redes analógicas com valores de saída positivos, é a função sigmóide. Um exemplo é a função logística mostrada abaixo:

$$f(x) = \frac{1}{1 + e^{-x}} \quad (3c)$$

Uma outra opção, que considera valores de saída positivos e negativos é a tangente hiperbólica:

$$f(x) = \tanh(x) \quad (3d)$$

A característica mais importante da função de ativação é a sua não-linearidade. Caso a função de ativação seja linear, o poder computacional da rede será equivalente a uma rede de uma camada [25].

Aprendizado

Todo o “conhecimento” de uma rede neural é guardado nas “sinapses”, os pesos das conexões entre os neurônios. Uma vez que o conhecimento está presente nos correspondentes pesos sinápticos da rede, a apresentação de um padrão na entrada de uma rede terá maior probabilidade de produzir a saída correta. A rede adquire esse conhecimento através do treinamento. Associações de padrões são apresentados à rede em sequência e os pesos são ajustados para capturar esse conhecimento. O esquema de ajuste dos pesos é conhecido como *algoritmo de aprendizado*.

Um dos primeiros métodos de aprendizado foi formulado por Donald Hebb. O aprendizado hebbiano é descrito pelo ajuste dos pesos das conexões baseadas nos valores de ativação dos neurônios conectados ao neurônio:

$$\Delta w_{ij} = \alpha * a_i * a_j \quad (3a)$$

onde α é a taxa de aprendizado, a_i é a ativação do i -ésimo neurônio em uma camada de neurônios, a_j é a ativação do j -ésimo neurônio da outra camada, e w_{ij} é o peso da conexão entre os dois neurônios. Uma variante dessa regra de aprendizado é a lei do sinal Hebbiano:

$$\Delta w_{ij} = -w_{ij} + f(a_i) * f(a_j) \quad (3b)$$

onde f é uma função sigmóide.

A Rede Perceptron de Múltiplas Camadas

As primeiras redes neurais utilizavam o modelo do neurônio perceptron. Como classificadores de padrões, esses neurônios conseguiam somente resolver problemas de separações lineares. Surgiram, portanto, as redes perceptron multicamada. Segundo [8] essas redes neurais utilizavam um algoritmo de aprendizado muito popular – o algoritmo backpropagation. Trata-se de um algoritmo de aprendizado supervisionado por correção de erro, com eficiência computacional satisfatória. São feitos dois passos para este tipo de aprendizado: um passo para frente, em que as saídas da rede neural são geradas dadas amostras de entrada; e um passo para trás, onde os pesos sinápticos de todos os neurônios da rede neural são atualizados de acordo com a regra de correção de erro. A atualização surge da comparação das saídas geradas com as respostas desejadas da rede dada a amostra de entrada apresentada à rede no passo anterior. O uso da rede neural necessita somente do passo para frente.

Um perceptron de múltiplas camadas tem três características principais:

- O modelo de cada neurônio inclui uma função de ativação não linear;
- A rede contém uma ou mais camadas de neurônios ocultos, que não são parte da camada de entrada ou saída da rede neural;
- A rede tem um alto grau de conectividade, determinado pelas sinapses da rede.

Essas características são responsáveis pela capacidade do perceptron de múltiplas camadas de aprender com a experiência através de treinamento. Porém, essas mesmas características são responsáveis também pela deficiência do conhecimento do projetista sobre o comportamento da rede. Isto porque o fato da função de ativação do neurônio ser não linear e a rede ter alta conectividade, faz com que a análise teórica da rede neural seja dificultada. Como a rede tem camadas de neurônios ocultas, acaba por dificultar a visualização do processo de aprendizado. Em [8], o algoritmo de retropropagação é mostrado matematicamente em detalhes. Neste trabalho iremos nos concentrar nas propriedades de generalização da rede neural a ser usada.

A generalização é a capacidade da rede neural produzir respostas adequadas para entradas fora do universo das amostras de treino da rede neural. A capacidade de generalização depende da etapa de treinamento da rede neural. A falta de treinamento

causa a má generalização da rede neural. Porém o excesso de treinamento também poderá prejudicar a generalização da rede neural. Isto porque as amostras utilizadas para o treino da rede neural podem não ser representativas do problema em questão a ser resolvido. A rede neural pode “memorizar” os dados de treino, detectando características intrínsecas nos mesmos (devido a ruídos, por exemplo) que os detalham bem, porém não detalham corretamente o comportamento do sistema a ser modelado. Deste modo, com o treinamento em excesso da rede neural, apesar de se obter erros pequenos da rede com relação às amostras de treino, a mesma poderá ter erros significativos no processamento das amostras de teste e validação.

Por isso é importante a escolha cuidadosa das amostras que serão usadas para o treinamento da rede neural. A escolha de amostras “ruins” acarretará o aprendizado incorreto da rede neural, fazendo com que a generalização da mesma seja pobre.

A rede LVQ (learning vector quantization):

De acordo com [8;26], Redes Neurais do tipo Mapa Auto-Organizáveis (SOM - self organizing maps) são redes que podem aprender a detectar regularidades e correlações nas suas entradas e adaptar as suas futuras respostas para aquela determinada entrada. Os neurônios das redes competitivas aprendem a reconhecer grupos de vetores de entradas similares. Mapas auto-organizáveis aprendem a reconhecer grupos de vetores de entrada similares de um modo tal que os neurônios fisicamente próximos entre si respondem a vetores de entrada similares.

Quantização Vetorial com Aprendizado (LVQ - learning vector quantization) é um método para o treinamento de camadas competitivas de uma maneira supervisionada. A camada competitiva automaticamente agrupa os vetores de entrada. Porém, as classes que essa camada competitiva encontra são dependentes somente das distâncias entre os vetores de entrada. Não há nenhum mecanismo em um projeto de uma camada estritamente competitiva que diz se dois vetores de entrada são ou não da mesma classe ou de diferentes classes.

Redes LVQ, por outro lado, aprendem a classificar vetores de entrada em classes-alvo escolhidos pelo usuário. A regra de aprendizado utilizado foi a Kohonen.

Em [36], o LVQ básico do aplicativo Matlab tem alguns atalhos que são endereçados pelas extensões LVQ1 e LVQ2. O LVQ1 é a primeira etapa de

aprendizado, que utiliza um fator de "consciência" que encoraja todos os neurônios a tomar uma parte ativa do aprendizado. O LVQ2 é a segunda etapa de aprendizado, que se trata de um mecanismo de sintonia fina que refina os limites da classe.

Os neurônios da camada escondida de uma rede LVQ competem entre si para aprender. Somente o vencedor (e opcionalmente os seus vizinhos) aprende a partir de um dado vetor de treinamento.

De acordo com [35], no aplicativo NeuralWorks o número de neurônios da camada escondida pode ser definido como uma porcentagem do conjunto de dados de treino. Porém a rede LVQ tem o paradigma do número de neurônios da camada Kohonen ser um múltiplo do número de neurônios de saída.

Pré-processamento de dados e Análise de Componentes Principais

A parte mais complicada no desenvolvimento de uma rede neural é o pré-processamento dos dados. Quando os dados são processados corretamente, muitos problemas podem ser resolvidos por somente uma rede neural. Por outro lado, um pré-processamento indevido pode fazer com que um problema fique muitas vezes insolúvel.

A Análise de Componentes Principais faz com que o espaço dos dados de entrada (dados de treino) seja transformado em um espaço reduzido de características efetivas [8], e ainda reter o máximo das características “intrínsecas” dos dados de entrada. Ou seja, os dados de entrada serão submetidos a uma redução de dimensionalidade.

Buscam-se três efeitos com a análise de componentes principais [8, 25, 26]:

- Ortogonalizar os componentes dos vetores de entrada, para que eles não estejam correlacionados entre si;
- Ordenar os componentes ortogonais resultantes – os componentes principais – para que os componentes com maior variação sejam inseridos nas primeiras posições da ordenação;
- Eliminar aqueles componentes que contribuem com valores menores que um mínimo de variação no conjunto de dados.

3.1 As Máquinas de Comitê

Para resolver problemas complexos de reconhecimento de padrões, podem ser utilizadas técnicas mais eficazes de reconhecimento, que levam em conta não somente um sistema reconhecedor, mas um conjunto deles. Essas técnicas se baseiam, segundo [8], no princípio de dividir e conquistar muito usado em engenharia. Este princípio descreve a resolução de problemas complexos através da combinação de soluções de um determinado número de tarefas computacionais simples.

As máquinas de comitê podem ser definidas, segundo [8], como uma combinação de sistemas especialistas, que no caso de aprendizado supervisionado dividem entre si a tarefa de aprendizagem computacional. Essas máquinas podem ser divididas em duas grandes categorias: estruturas estáticas ou estruturas dinâmicas.

Nas estruturas estáticas, as respostas dos vários especialistas são combinadas por meio de um mecanismo que não depende dos sinais de entrada. Nesta categoria são incluídos os métodos de Média de Ensemble, onde as saídas dos diferentes especialistas são combinadas linearmente para produzir a saída global; e o método de Boosting, em que um algoritmo fraco de aprendizagem é convertido em outro com precisão arbitrária alta.

As estruturas dinâmicas envolvem as entradas do sistema no mecanismo de combinação dos vários sistemas especialistas. Em [8], são citadas duas estruturas: a Mistura de Especialistas, em que as respostas individuais de cada especialista são combinadas não-linearmente por meio de uma única rede de passagem; e a Mistura Hierárquica de Especialistas, em que a combinação dos especialistas é feita com várias redes de passagem arranjadas hierarquicamente.

O método de Ensemble foi usado em [27] para aproximar uma função ruidosa senoidal. Em [28] foram reconhecidos 14 tipos de óleos usando um nariz eletrônico comercial e duas abordagens de máquinas de comitê: a primeira abordagem utilizava um Perceptron Multi-Camadas e SIMCA (modelamento de software independente de analogia de classe) para uma classificação hierárquica dos óleos. A segunda abordagem utilizou uma máquina de aprendizado chamada PND (dicotomizadores paralelos não-lineares), baseada na decomposição de um problema de classificação de K classes em

um conjunto de reconhecedores (chamados dicotomizadores) de duas classes. Neste trabalho foram implementadas somente as máquinas de comitê estáticas.

Média de Ensemble

O método da média de ensemble combina um número de especialistas (redes neurais) treinadas diferentemente (por exemplo, iniciadas com pesos diferentes), que compartilham um conjunto de dados de entrada comum, e são combinadas linearmente para produzir uma saída global. Segundo [8,10], a motivação do seu uso tem vários aspectos:

- Para resolver um problema complexo de reconhecimento de padrões utilizando somente uma rede neural, esta teria de ter uma complexidade compatível (muitos neurônios em uma ou mais camadas escondidas) para resolver o problema, resultando em um número grande de parâmetros a serem ajustados. O tempo de treinamento é provavelmente maior do que o caso em que sistemas especialistas mais simples fossem treinados em paralelo.
- O risco de ajuste em excesso (problema de “overfitting”) aumenta quando o número de parâmetros a serem ajustados é grande comparado com o tamanho do conjunto das amostras de treino.
- Os dados de treinamento podem não prover informações suficientes para escolher um melhor classificador. Muitos algoritmos de classificação consideram um grande espaço de hipóteses. Mesmo eliminando hipóteses errôneas, há ainda muitas hipóteses restantes. A máquina ensemble formada a partir desta coleção de hipóteses restantes será provavelmente mais robusta.
- Para encontrar os pesos corretos da rede neural a partir de um conjunto de treinamento, os algoritmos de redes neurais usam inicialização randômica e métodos de procura locais. Inicializações diferentes levam a diferentes conjuntos ótimos de pesos. O método Ensemble pode ser visto como uma maneira de compensar esses algoritmos de procura imperfeitos.

Método de Boosting

É um método geral que tenta reforçar o acerto de um dado algoritmo de aprendizagem fraca [8,10]. Um algoritmo de aprendizagem fraca é aquele que encontra

uma hipótese de reconhecimento com uma taxa de erro um pouco menor que $1/2$. Deste modo, o algoritmo de aprendizagem fraca terá quase a mesma taxa de acertos do que uma estimativa aleatória.

Ao contrário do método de Ensemble em que os especialistas utilizam um mesmo conjunto de dados de entrada, no método de boosting os especialistas são treinados com conjuntos de dados com distribuições totalmente diferentes. O método de boosting pode ser implementado de três modos diferentes [8]:

- Reforço por filtragem, usada com conjuntos grandes de treinamento. Os dados são descartados ou mantidos durante a fase de treinamento.
- Reforço por subamostragem, usado com um conjunto de treinamento de tamanho fixo.
- Reforço por ponderação, similar ao anterior, porém o algoritmo de aprendizagem fraca pode receber exemplos “ponderados”.

3.2 A Lógica Fuzzy

A Lógica Fuzzy é uma ferramenta que formaliza o pensamento humano e a linguagem natural, capturando a natureza inexata do mundo real [13]. A lógica Fuzzy é usada principalmente em Controladores Lógicos Fuzzy (FLC – Fuzzy Logic Controller) para modelar processos que seriam complexos demais para serem equacionados com métodos tradicionais.

De acordo com [29], um conjunto ou classe fuzzy é caracterizado por uma função de pertinência que associa cada ponto de entrada a um intervalo fechado de números reais $[0;1]$. Os conjuntos crisp (exatos ou precisos), associam cada ponto de entrada a dois possíveis valores numéricos discretos: 0 ou 1. A operação de união entre conjuntos Fuzzy é definida como o máximo das funções de pertinências dos conjuntos fuzzy de entrada, e a intersecção é definida como o mínimo.

São estipuladas regras fuzzy para o FLC, além das implementações das etapas de fuzzificação das entradas e defuzzificação das saídas. E esta caixa preta é aplicada diretamente ao sistema o qual se quer controlar. As regras são divididas em regras precedentes, e regras conseqüentes. Essas regras são combinações de atributos de linguagem natural (adjetivos tais como alto, baixos, médios, pequenos, grandes, rápidos, lentos, etc.). Em [25,26] há mais detalhes da lógica fuzzy aplicada em um FLC.

Assim como os sensores que utilizam redes neurais artificiais, a lógica fuzzy se aplicará nos sensores inteligentes nas etapas de modificação e condicionamento de sinais.

Segundo [30], um FLC é um conjunto de regras que relaciona os conceitos duais de implicação fuzzy e a regra composicional de inferência. O FLC promove um algoritmo que converte a estratégia de controle lingüístico baseado no conhecimento especialista em uma estratégia de controle automático. O FLC é dividido em quatro partes principais:

- **Fuzzificação**: etapa inicial de um FLC em que são feitas as transformações necessárias nos dados para o seu uso no FLC;
- **Base de Dados**: é dividido em dois componentes: uma base de dados contendo as definições para o uso de regras de controle lingüístico e manipulação de dados fuzzy; e uma base de regras que representa o conhecimento do especialista transformado em regras canônicas.
- **Lógica de tomadas de decisões**: simula a tomada de decisões feita pelos seres humanos, de acordo com os vários resultados possíveis e seus diferentes graus de pertinências.
- **Defuzzificação**: possibilita a atuação “crisp” (precisas) feita pelo FLC, transformando a resposta do FLC no domínio fuzzy em uma saída de resposta “crisp”, tratável pelo mundo real.

A estratégia de fuzzificação para entradas de valores precisos é representada como um fuzzy singleton, ou seja, um pulso com amplitude unitária em torno do valor da entrada. Para dados com ruídos randômicos no operador fuzzy (um triângulo isósceles, por exemplo) os dados probabilísticos podem ser convertidos em um número fuzzy. Nos casos em que há dados precisos e imprecisos simultaneamente, é usado o conceito de números híbridos.

A base de conhecimento de um FLC é dividida em duas partes: a **base de dados** e a **base de regras** do controlador fuzzy.

A base de dados é baseada na opinião subjetiva do especialista. Para a sua construção, devem ser considerados os aspectos de discretização/normalização dos universos de discurso, em que o mapeamento linear ou não linear dos dados de entrada é

feito de acordo com um conhecimento a priori do especialista dentro do intervalo real fechado entre 0 e 1; além do aspecto de partição fuzzy dos espaços de entrada e saída, em que as variáveis fuzzy são divididas em partições, que representam os seus possíveis valores com significados característicos relacionados ao sistema, tais como: muito pouco, pouco, muito, exagerado, etc.

A base de regras é composta por regras canônicas que relacionam os atributos da entrada e os seus respectivos valores fuzzy em relações do tipo: SE <condição satisfeita> ENTÃO <conjunto de conseqüências>.

O algoritmo fuzzy deverá prever uma ação resultante para todos os estados possíveis do sistema (completeza). E para assegurar esta propriedade, são definidas duas estratégias: A **estratégia da base de dados**, em que a união dos suportes nos quais os conjuntos fuzzy estão definidos deverão cobrir o universo de discurso em relação a algum nível setado ϵ . Escolhendo-se um nível no ponto de crossover, sempre existirá uma regra dominante. O caso extremo será a escolha de duas regras dominantes com níveis iguais a 0,5. A **estratégia da base de regras** é baseada na experiência do especialista do sistema. Novas regras são incluídas no sistema quando uma condição fuzzy não está incluída na base de regras, ou quando as condições pré-definidas fuzzy têm crenças menores que um determinado valor, por exemplo, 0,5.

Há duas formas de definir as funções de pertinência de um conjunto fuzzy: A **definição numérica**, em que o grau de pertinência é representado como um vetor de dimensão dependente do grau de discretização, é usada quando o universo de discurso é discreto. Na **definição funcional**, as funções de pertinência são definidas como funções, e é usado quando o universo de discurso é contínuo.

Para a definição da base de regras, deve-se escolher as variáveis de estado do processo, e as variáveis de controle das regras do controlador fuzzy. As regras podem ser vindas do conhecimento de engenharia de controle e do especialista, baseados em ações de controle dos operadores, no modelo fuzzy de um processo, ou baseados no aprendizado.

Segundo [31], uma regra de controle fuzzy é uma relação fuzzy expressa como uma implicação fuzzy. Os critérios básicos de uma função de implicação são: propriedade fundamental, suavidade, interferência irrestrita, simetria de modus ponens/tollens generalizado (GMP/GMT), e medida de propagação de vaguidade.

O mecanismo de inferência empregado em um FLC é bem mais simples que o usado em sistemas especialistas, já que o conseqüente de uma regra não se torna o antecedente de outra. Pode-se mostrar que o operador sup-mínimo é comutativo. Isso faz com que a ação inferida pelo sistema completo é equivalente ao resultado agregado de cada regra individualmente.

Há quatro tipos de raciocínio fuzzy empregados em aplicações FLC:

- **Operação de Mínimo (Mandani)**, é o método mais simples, pois utiliza o operador mínimo para o cálculo do valor resultante de cada regra. Necessita do uso da etapa de defuzzificação. **Produto de Larsen**: em que o resultado da regra é o produto de dois fatores: o valor escalar máximo da intersecção entre o valor medido e os predicados lingüísticos da regra antecedente; e o predicado lingüístico da regra conseqüente. **Uso de funções de pertinência monotônicas (Tsukamoto)**: simplifica os cálculos, pois as funções de pertinência são monotônicas. Com isso, não é necessária a etapa de defuzzificação, e a atuação será calculada como a média ponderada das regras envolvidas. **A conseqüência de uma regra é função das variáveis lingüísticas de entrada (Sugeno)**: as regras são expressas na forma: SE $\langle x \text{ é } A_i, \dots, y \text{ é } B_i \rangle$ ENTÃO $\langle z = f_i(x \dots y) \rangle$. Assim, o valor crisp de atuação é obtido com uma média ponderada das entradas relacionadas com seus graus de pertinência α .

A defuzzificação é a etapa em que as ações de controle fuzzy são transformadas em um universo de discurso “crisp” para as atuações de controle. As estratégias possíveis são o critério de máximo, média do máximo ou centro de área.

Um dos problemas do FLC é de que o processo de transferência do conhecimento do especialista para o FLC demanda tempo e não é trivial. Serão necessários procedimentos com fundamentos de projeto de sistemas de controle clássico.

4 Metodologia Experimental

O texto deste trabalho está organizado segundo mostra a Tabela 4a abaixo. Foram estudados dois problemas de reconhecimento de padrões: amostras de vapor de álcool combustível e amostras de gás GLP. Para resolver esses problemas foram aplicadas três abordagens: Redes Neurais Artificiais, Lógica Fuzzy e as Máquinas de Comitê Estáticas.

Tabela 4a: Detalhamento dos experimentos realizados

	Redes Neurais	Lógica Fuzzy	Máquinas de Comitê
Com Pré-Processamento e Análise de Componentes Principais	<ul style="list-style-type: none"> • Álcool e GLP • Dados Sintéticos e Experimentais 	<ul style="list-style-type: none"> • GLP • Dados Sintéticos 	<ul style="list-style-type: none"> • GLP • Dados Sintéticos e Experimentais
Sem Pré-Processamento e Análise de Componentes Principais	<ul style="list-style-type: none"> • Álcool e GLP • Dados Sintéticos e Experimentais • Implementação em Hardware (DSP) 	<ul style="list-style-type: none"> • GLP • Dados Sintéticos 	<ul style="list-style-type: none"> • GLP • Dados Sintéticos e Experimentais

Foram observadas as influências das ferramentas de pré-processamento dos dados e análise de componentes principais na velocidade de treinamento das redes neurais e na capacidade de generalização das mesmas.

Neste item são apresentadas duas metodologias adotadas para a extração dos dados do vapor de álcool combustível, e do gás GLP. Foram feitas várias medidas das resistências dos sensores antes e depois as injeções das amostras.

As medidas dos vapores de álcool foram feitas pontualmente, isto é, somente os valores iniciais e finais das tensões dos sensores, com os sinais em regime permanente (variação da resistência menor do que 5%) foram extraídos. As medidas do GLP foram feitas no domínio do tempo. Mas neste trabalho, os valores das medidas usadas do GLP foram os valores iniciais e finais das tensões dos sensores em regime permanente.

Os sensores utilizados nas câmeras de sensores nos experimentos descritos nos itens seguintes foram os de óxido de estanho do fabricante Taguchi. Os sensores e os compostos que eles detectam estão listados na Tabela 4a [12]:

Tabela 4a: Compostos Detectados para cada sensor

Sensores Taguchi	Compostos Detectados
TGS-2442	Monóxido de Carbono
TGS-2600	Etanol, Metanol, Butano e Propano
TGS-2602	Etanol, Tolueno e Amônia
TGS-2610	Propano e Butano
TGS-2611	Gás Natural / Metano
TGS-2622	Etanol, Tolueno, Xileno e Solventes Orgânicos

Estes seis sensores foram escolhidos de modo que eles não ficassem “sintonizados” para os compostos detectados. Para tanto, eles deverão responder a um grande número de compostos que compõe a substância a ser medida [32], e com isso, a seletividade dos compostos será feita pela etapa de reconhecimento de padrões. Ou seja, os critérios da escolha do número de sensores e do tipo de cada um foram baseados na diversidade dos componentes a serem analisados e na diversidade nas respostas dos sensores.

A obtenção dos dados experimentais está descrita nos capítulos seguintes. A extração dos atributos obtidos dos dados foi feita de acordo com a Variação Fracional da Resistência (adaptada a partir da Variação Fracional da Condutância em [32]). A Variação Fracional da Resistência foi definida pela equação (4a).

$$R_{\text{NORM}} = (R_{\text{INICIAL}} - R_{\text{FINAL}}) / R_{\text{INICIAL}} \quad (4a)$$

A razão da escolha da equação da Variação Fracional da Condutância como equação base na extração dos atributos dos dados experimentais foi a de que o tempo de aprendizado da rede neural é menor com relação a outras equações de extração de atributos, tais como a Condutância Relativa, Logaritmo da Condutância Relativa, etc. E além disso, o desempenho (e portanto, a capacidade de generalização) do sistema reconhecedor é aproximadamente o mesmo se tomados os algoritmos de extração de atributos relacionados em [32].

4.1 Procedimento para coleta da amostra de álcool combustível

As amostras de vapor de álcool combustível foram extraídas com o auxílio da câmara ilustrada na Fig. 4.1a:

Fig 4.1a: Câmara de medidas.

O Ventilador serviu para efetuar a purga ou limpeza da câmara com os sensores. A câmara com o composto Sílica-Gel foi usada para absorver a umidade e os gases da câmara com os sensores. As válvulas 1 e 2 controlavam o fluxo de gás no interior do sistema. Na figura, as válvulas estão fechadas, pois suas “alavancas” estão posicionadas na vertical. A abertura das válvulas é feita girando as “alavancas” até elas ficarem na horizontal. As direções de abertura/fechamento estão descritas no corpo das válvulas. O círculo central escuro no centro da Câmara com os Sensores representa um material de borracha, que serviu como uma válvula para que a amostra de vapor de álcool pudesse ser injetada na câmara por meio de uma seringa.

Os sensores foram alimentados com 12Vcc. A tensão dos aquecedores dos sensores foi ajustada com 5Vcc. Os fios de alimentação da placa com o sensor de umidade foram alimentados com 5Vcc. O conector DB9 foi ligado a uma placa de aquisição de sinais, que por sua vez foi ligado ao equipamento PXI.

O sistema de coleta do vapor de álcool foi feito por meio de uma seringa. O procedimento operacional empregado para efetuar as medidas foi:

- 1) Ligar as fontes de alimentação dos sensores e dos aquecedores dos mesmos, além da fonte dos sensores de umidade e temperatura.
- 2) Verificar a correta interligação do sistema com o equipamento PXI e o conversor analógico/digital.
- 3) Executar o programa cliente de extração de medidas instalado no equipamento PXI.
- 4) Verificar a Câmara com Sílica Gel, e observar se o mesmo está em condições de uso. Se a Sílica Gel estiver vermelha, ela deverá ser aquecida com o auxílio de um “hot-plate” para eliminar a umidade do mesmo.
- 5) Abrir as válvulas 1 e 2, e ligar o ventilador para efetuar a purga ou limpeza da câmara com os sensores. Observar a resposta dos sensores no PXI, e esperar a estabilização de todos os sensores.
- 6) Desligar o ventilador e fechar as válvulas 1 e 2. Espere novamente a estabilização dos sensores.
- 7) Inicie uma medida com o equipamento PXI clicando em “Gravar” (grava todas os pontos da amostragem de todos os sensores em um período de amostragem definida no programa em um arquivo texto) ou “Snapshot Inicial” (grava somente os pontos iniciais e finais da medida). Neste experimento quase todas as medidas foram realizadas com o uso do “Snapshot”.
- 8) Preparar a seringa sugando o vapor de uma amostra de álcool combustível. Espetar a seringa na Câmara com os Sensores, e injetar a amostra lentamente.
- 9) Ao estabilizar as medidas dos sensores, finalizar a gravação do arquivo texto iniciado no passo 7 no equipamento PXI.
- 10) Para realizar uma nova medida, repetir os passos 5 a 9.
- 11) Para finalizar as medidas, faça a limpeza da câmara segundo o passo 5, feche as válvulas 1 e 2, e desligue a alimentação do ventilador. Não desligue a alimentação dos sensores.

Os dados obtidos por este método foram usados no capítulo 5.1 deste trabalho.

4.2 Primeiro método de extração de características do gás GLP

A primeira abordagem para a extração de características do poder calorífico do gás GLP foi baseado em [7]. O sistema consiste de uma câmara com seis sensores Taguchi, que detectam concentrações de diferentes tipos de gases (monóxido de carbono, etanol, metano, butano, propano, amônia, etc.). O combustível gasoso foi injetado nesta câmara junto com o gás nitrogênio. As proporções das concentrações de cada componente desta mistura foram reguladas através de rotâmetros.

A primeira metodologia de extração de amostras dos gases foi a seguinte: Fixada uma vazão do gás combustível GLP (5mm do rotâmetro, que corresponde a uma vazão de 3ml/min), fixou-se a vazão do gás nitrogênio ajustada através de outros dois rotâmetros. Essa mistura dos gases foi aplicada na câmara dos sensores através do ajuste do Rotâmetro R7 em 8mm. A variação dos rotâmetros que controlam a vazão do gás nitrogênio simulou a presença dos dois gases combustíveis de poderes caloríficos diferentes, através da maior ou menor concentração de GLP na mistura.

Três amostras do padrão de GLP puro com alto poder calorífico foram obtidas com os valores em regime permanente das resistências dos sensores antes e após cada injeção de combustível. E com esses dados, foi montado um banco de dados com seis atributos, correspondentes aos valores dos seis sensores. Esse banco de dados foi subdividido em um conjunto de dados de treino e outro conjunto de dados de teste.

Para aumentar o banco de dados de amostras dos diversos padrões de gases GLP foram gerados dados sintéticos baseados em alguns dados medidos disponíveis de um padrão de gás GLP. Foram criados 100 pontos para cada padrão de combustível. Como foram simulados três padrões, o total de pontos foi de 300. Os comportamentos típicos dos valores de resistências dos sensores estão mostrados na Fig. 5.4a abaixo. Os gráficos dos dados sintéticos dos seis sensores para um dado valor de umidade e temperatura estão mostrados na Fig. 5.4b. Os valores do desvio padrão e média de variação da resistência normalizada para a primeira classe foram obtidos a partir das medidas existentes. Como uma primeira abordagem de comparação entre as soluções de etapa de reconhecimento por RNA ou FIS, os diferentes padrões foram obtidos com os mesmos desvios padrões da primeira classe, porém a média do segundo padrão corresponde a uma razão de 1/3 do primeiro padrão, e a média do terceiro padrão

corresponde a uma razão de 0,5. A implementação deste item foi feita através de um programa escrito em Matlab [36] denominado “AmSint1.m”.

Fig. 5.4a: Comportamento típico dos sensores na injeção da amostra.

Fig. 5.4b: Dados Sintéticos usados para a criação da Rede Neural e o FIS

Os dados de cor vermelha foram relativos ao padrão de combustível com maior poder calorífico. O padrão2 em azul representa o padrão de combustível com o menor

poder calorífico. O padrão³ em verde representa o padrão de combustível com um poder calorífico intermediário com relação aos anteriores.

Os dados obtidos por este método foram usados no capítulo 5.3 deste trabalho.

Como a obtenção das amostras experimentais deste método era dificultada pelo manuseio dos rotâmetros, optou-se por adotar uma nova metodologia experimental para extrair e tratar os dados do gás GLP, que está mostrado no item 4.3 a seguir.

4.3 Procedimento para extração e tratamento dos dados do GLP

Para a extração de características das amostras de gás GLP, foram usados dois procedimentos distintos. O primeiro procedimento foi o de coleta do gás GLP e o confinamento do mesmo em uma seringa. Este procedimento foi diferente do usado na extração do vapor de álcool, pois não podíamos extrair a amostra de gás GLP diretamente do botijão de gás disponível do laboratório, e com isso, foi necessário o uso do artifício da bexiga. O segundo procedimento foi a obtenção das medidas do gás GLP com a injeção da amostra da seringa na câmara de sensores. Esses dois procedimentos estão descritos nos itens 4.3.1 e 4.3.2 a seguir.

4.3.1 Procedimento para coleta da amostra de gás GLP

O procedimento experimental adotado para extrair os dados de uma amostra de gás combustível do sistema de reconhecimento foi baseado em uma câmara mostrada na Fig. 4.3.1a:

Fig. 4.3.1a: Câmara de medidas.

A purga ou limpeza da câmara dos sensores foi feita através de uma bomba de sucção de ar. As válvulas 1 e 2 foram mantidas fechadas (com as suas “alavancas” posicionadas na vertical), para isolar a câmara dos sensores do meio externo. A câmara foi tampada com uma tampa de borracha por onde passa o tubo de sucção do ar feito

pela bomba. A inserção da amostra de gás foi feita injetando a amostra de gás a ser analisado por meio de uma seringa, espetada nesta tampa de borracha.

Os seis sensores utilizados foram os mesmos do item 4.2 anterior. Eles foram alimentados com 12Vcc. A tensão dos aquecedores dos sensores foi ajustada com 5Vcc. Os fios de alimentação da placa com o sensor de umidade foram alimentados com 5Vcc. O conector DB9 foi ligado a uma placa de aquisição de sinais, que por sua vez foi ligado ao equipamento PXI.

O sistema de coleta do gás foi feito por meio de uma seringa e uma bexiga. O procedimento de coleta está ilustrado na Fig. 4.3.1b abaixo:

Fig. 4.3.1b: Sistema de coleta do gás

A coleta da amostra dos gases foi feita seguindo o seguinte procedimento operacional:

- 1) Ligar a fonte de tensão para alimentar o controlador de fluxo (MKS186A) de gás, e espere 5 minutos para o controlador aquecer até atingir a precisão necessária.
- 2) Desconectar o tubo de borracha com a bexiga do tubo de plástico, para assegurar que resíduos de gás não fiquem nos tubos de acesso à bexiga.
- 3) Abrir a VI “medidas” (Fig 4.2c). Se este programa começar enviando uma mensagem de erro, seguir o procedimento sugerido na mesma mensagem.
- 4) Ligar a válvula V3, abrindo o fluxo do gás nitrogênio. Ligar a válvula V4 até que o manômetro ligado no botijão do gás GLP atinja valor estável. Fechar a válvula V4, para que seja usado somente o gás GLP contido na mangueira, e também por questões de segurança. Após esvaziar completamente a bexiga, re-encaixar o tubo de borracha no tubo de plástico.

- 5) Na VI “medidas” coloque o fluxo, em sccm (scc=standart cubic centimeter per minute; 1000sccm = 1litro/min) de gás, e o tempo durante o qual este fluxo deve estar entrando na bexiga. Seguir o mesmo procedimento com o Nitrogênio. Recomenda-se utilizar um mesmo intervalo de tempo tanto para o gás quanto para o Nitrogênio. Os valores que se mostram na Fig. 4.3.1c correspondem a uma proporção de 5:1. Recomenda-se 1000sccm de nitrogênio e a faixa de fluxo de 10 a 500sccm de GLP. Os tempos recomendados de injeção para cada um dos gases são de 10 segundos. Com a opção “Preparação de Amostra” selecionada, pressione o botão “OK”. Uma vez que a bexiga tenha enchido um pouco, a amostra de gás pode ser extraída com a seringa, espetando-o no tubo de borracha.

Fig. 4.3.1c: VI que controla a mistura de gás com nitrogênio.

Cada extração de gás poderá fornecer de duas a três medidas sequenciais. Após três medidas no máximo, esvaziar a bexiga e coletar novamente a mistura de gás executando o aplicativo VI “medidas”. Caso se queira medir a resposta das resistências dos sensores ao gás puro, é necessário somente colocar 0sccm no campo nitrogênio. Para medir as respostas das resistências dos sensores quando expostos ao gás nitrogênio puro, coloque 0sccm no campo Gás.

4.3.2 Procedimento para efetuar as medições de gás GLP

O procedimento aplicado para as medições foi o seguinte:

- 1) Ligar o equipamento PXI, executar o programa NarizEletrônico Cliente, clicar no botão Conectar, e inserir o login e password.

- 2) Efetuar a limpeza da câmara. Para isso, ligue a bomba de sucção de ar que irá extrair o ar dentro da câmara. Observe no PXI a resposta dos sensores. Espere os mesmos se estabilizarem (considera-se que os sensores estão estabilizados, quando a variação das resistências dos mesmos não ultrapasse a porcentagem de 0.05% da variação medida entre os 15 pontos sucessivos (para uma amostragem de 0.5s, equivale a um tempo de 7.5s) para iniciar as medidas).
- 3) Desligue a bomba de vácuo. Espere os sensores se estabilizarem novamente, observando suas respostas no PXI.
- 4) Extrair uma amostra de gás combustível, seguindo o “Procedimento para Coleta de amostra de gás”, descrito anteriormente. Espetar a seringa na mangueira de borracha, e extrair a amostra do gás (recomenda-se retirar um volume de 1ml). Pare a execução na VI “medidas”, clicando em “STOP”.
- 5) No PXI escolha um nome de arquivo em formato texto no padrão:

<data>_<fluxo>sccmN2_<fluxo>sccmGAS<tempo>s_<volume>ml-
injetado_T<temperatura>UR<umidade relativa>_med<número da medida>.txt

Por exemplo, para uma medição de uma amostra feita no dia 08 de agosto de 2003, segundo 200sccm de gás, 1000sccm de nitrogênio por um tempo de 10s, volume de injeção de 1ml, temperatura de 20°C, Umidade Relativa de 14%, e número da medida 15, o nome do arquivo adotado para a próxima medição foi:

08-08-03_200sccmGAS_1000sccmN2_10s_1ml-injetado_T20_UR14_med16.txt.

- 6) Clique em gravar, e insira o gás da seringa na câmara com os sensores. Espere estabilizar a resposta dos mesmos observando o PXI, e clique em Parar.
- 7) Para efetuar uma nova leitura, repita os passos de 2 a 6.
- 8) Após terminar as medições, assegure-se de desligar os registros do gás e do Nitrogênio. Além disso, desligue a fonte de alimentação do controlador de fluxo (MKS186A). Porém não desligue a fonte que alimenta os sensores de gás.

5 Pré-Processamento dos Dados

O aplicativo Data Sculptor foi usado neste trabalho para um prévio estudo do efeito do pré-processamento dos dados na etapa de treinamento e na característica de generalização da rede neural. Para realizar este estudo, foi montado o esquema da Fig. 5a utilizando os objetos do Data Sculptor. As caixas são os objetos do Data Sculptor que têm funções distintas. As flechas que as interligam representam o fluxo dos dados que trafegam no esquema montado.

Fig. 5a: O aplicativo Data Sculptor

Os dados utilizados como entrada foram amostras sintéticas baseadas em experiências feitas com sensores de álcool etílico. Foram estipuladas faixas de resposta para cada sensor submetido a amostras de álcool com qualidade boa (álcool de concentrações entre 94.3 e 98.4%), e ruim (álcool com concentrações fora desta faixa). Esses dados de entrada são inseridos no sistema através do objeto “IN” nomeado como “Dados Crus” na Fig. 5a. Os dados foram colocados em gráficos listados nos objetos “GRAPH”s nomeados como “PAR1 x SAÍDAS”; “PAR2 x SAIDAS”, etc.

O banco de dados resultante foi composto de 50 amostras ("medidas" dos oito sensores considerados, obtendo-se oito atributos) para treino e mais 50 amostras para teste. Cada atributo foi obtido dividindo-se os valores de tensão dos sensores depois da exposição dos mesmos ao álcool (em regime permanente) pelos valores de tensão antes da exposição dos sensores ao álcool. Para o álcool de boa qualidade foram gerados 50 amostras sintéticas, e o mesmo número foi gerada para o álcool de qualidade ruim. Mas para submeter esses dados ao aplicativo Data Sculptor, eles foram ajuntados em um arquivo avulso contendo 100 amostras sintéticas. Para cada amostra foram designados dois campos (colunas na Tabela 5a do banco de dados) destinados a saída desejada, sendo um dado enumerado:

Tabela 5a: Valores Desejados adotados para as saídas

	Saída Desejada 1	Saída Desejada 2
Álcool Bom	1	0
Álcool Ruim	0	1

Como os gráficos obtidos para os oito atributos versus a saída ficaram parecidos, por conveniência, abaixo na Fig. 5b é mostrado um exemplo de gráfico com os dados “crus” para um dos sensores, obtido pelo objeto GRAPH (PAR1 X SAIDAS):

Fig. 5b: Gráfico obtido com o Data Sculptor antes do pré-processamento

Para testar o caráter seletivo do aplicativo Data Sculptor, foram introduzidos algumas amostras redundantes e inconsistentes ao banco de dados. Duas amostras são

consideradas redundantes quando os seus atributos e saídas desejadas são exatamente os mesmos, ou seja, as duas amostras são idênticas. É necessário remover essas amostras do banco de dados, pois elas atrapalham o treinamento da rede neural. Duas amostras são inconsistentes quando seus atributos são idênticos, porém suas saídas desejadas não o são. Para remover essas amostras foram utilizados dois objetos, o objeto Sort (ORDEM_ASCEND) e o Sieve (INCONS_REDUND).

Para o objeto Sort foi selecionado o atributo 1 do banco de dados (primeira coluna), e todas as amostras foram ordenadas baseadas no valor crescente desse atributo. Depois esses dados passaram pelo objeto Sieve, que fez uma filtragem nesse banco de dados. Foi utilizada a seguinte expressão para a filtragem:

```
!REC_COMP(PARAMETRO1, 109, 1)
```

Sendo que o símbolo “!” representa uma negação, ou seja, as amostras diferentes são as que continuarão no fluxo de dados, e as amostras idênticas ou inconsistentes serão filtradas. Depois o banco de dados foi submetido a um objeto Detail (ADP). Nesse objeto foi utilizada a ferramenta ADP (Automatic Data Preprocessing). Essa ferramenta realiza automaticamente a escolha das funções matemáticas que são usadas para transformar as amostras que estão sendo injetados nele. O ADP possui três modos de transformação de acordo com o nível crescente de complexidade das funções matemáticas obtidas pelo mesmo: o BRIEF, o MODERATE, o THROUGHOUT e o EXPERT. O usuário deverá escolher o modo de transformação de acordo com a sua percepção da complexidade do problema a ser resolvido.

Para o modo BRIEF, foram obtidas as seguintes funções para cada atributo (PARAMETRO's):

```
PARAMETRO10 = TRNSCALE(PARAMETRO1,0,1,0.838042,1.21017)
```

```
PARAMETRO20 = TRNSCALE(PARAMETRO2,0,1,0.213979,2.42972)
```

```
(...)
```

```
PARAMETRO80 = TRNSCALE(PARAMETRO8 ...)
```

```
SAIDA10 = TRNSCALE(SAIDA1,0,1,0,1)
```

```
SAIDA20 = TRNSCALE(SAIDA2,0,1,0,1)
```

Para o modo MODERATE, foram obtidas as seguintes funções:

```
PARAMETRO1000 = TRNSQR(PARAMETRO1,0,1,0.814189,1.21017)
```

```
PARAMETRO1001 = TRNSCALE(PARAMETRO1,0,1,0.84485,1.21017)
```

```
(...)
```

SAIDA10 = TRNSCALE(SAIDA1,0,1,0,1)

SAIDA20 = TRNSCALE(SAIDA2,0,1,0,1)

Para o modo THROUGHOUT, foram obtidas as funções:

PARAMETRO10000 = TRNSQR(PARAMETRO1,0,1,0.814189,1.21017)

PARAMETRO10001 = TRNFUZZYLEFT(PARAMETRO10000,0,0.142857)

PARAMETRO10002 = TRNFUZZYCENTER(PARAMETRO10000,0,0.142857,0.428571)

PARAMETRO10003 = TRNFUZZYCENTER(PARAMETRO10000,0,0.285714,0.571429)

(...)

PARAMETRO10007 = TRNFUZZYCENTER(PARAMETRO10000,0,0.714286,1)

PARAMETRO10008 = TRNFUZZYRIGHT(PARAMETRO10000,0,0.857143,1)

SAIDA10 = TRNSCALE(SAIDA1,0,1,0,1)

SAIDA20 = TRNSCALE(SAIDA2,0,1,0,1)

E, finalmente para o modo EXPERT, as funções obtidas foram:

PARAMETRO10000 = TRNSQR(PARAMETRO1,0,1,0.814189,1.21017)

PARAMETRO10001 = TRNFUZZYLEFT(PARAMETRO10000,0,0.142857)

PARAMETRO10002 = TRNFUZZYCENTER(PARAMETRO10000,0,0.142857,0.428571)

PARAMETRO10003 = TRNFUZZYCENTER(PARAMETRO10000,0,0.285714,0.571429)

(...)

PARAMETRO10007 = TRNFUZZYCENTER(PARAMETRO10000,0,0.714286,1)

PARAMETRO10008 = TRNFUZZYRIGHT(PARAMETRO10000,0,0.857143,1)

PARAMETRO10009 = TRNSCALE(PARAMETRO1,0,1,0,0.844815)

SAIDA10 = TRNSCALE(SAIDA1,0,1,0,1)

SAIDA20 = TRNSCALE(SAIDA2,0,1,0,1)

Neste trabalho foi escolhido o modo “brief”, pois os dados sintéticos disponíveis eram linearmente separáveis. Para o modo BRIEF, após a transformação dos dados feita pelo ADP, o gráfico resultante para o objeto GRAPH (PAR10 X SAIDAS) é mostrado na Fig. 5c. Vemos por esse gráfico que o Data Sculptor ajustou uma escala para as amostras randômicas, fazendo com que todos os registros se enquadrassem no intervalo entre 0 a 1. Isso porque a intenção foi usar os dados para treinar uma rede segundo a função de ativação de sigmóide, cujo intervalo vai de 0 a 1. Caso a rede a ser treinada fosse usar a função de ativação de tangente hiperbólica, o intervalo definido na caixa de diálogo do ADP seria de -0.8 a +0.8, e conseqüentemente, a escala do gráfico resultante se ajustaria ao mesmo intervalo de -0.8 a +0.8. O objetivo desse ajuste é melhorar a

performance do treinamento da rede neural, que irá enxergar dados mais compatíveis, e visivelmente mais separados, acelerando o processo de aprendizado da rede.

Fig. 5c: Gráfico obtido com o Data Sculptor após o pré-processamento

A próxima etapa foi ordenar randomicamente as amostras no banco de dados. Para isso foi utilizado um objeto SORT (RANDOMIZE) ordenando aleatoriamente as amostras baseando-se no PARAMETRO1. Depois foi utilizado um objeto DETAIL (SEP_CONJ) para separar as amostras em três conjuntos de amostras, um conjunto de dados de validação, outro de treino, e por último um conjunto para teste. A expressão utilizada para separar os três conjuntos no objeto DETAIL (modo BRIEF) foi:

SEP_CONJ = DATASET (PARAMETRO10,8,92,50)

Esse comando criou um novo atributo chamado SEP_CONJ, que contém o resultado da seleção feita por esse objeto. Isto é, as atribuições a essa variável são feitas segundo a regra mostrada na Tabela 5b:

Tabela 5b: Separação dos dados feita pelo aplicativo Data Sculptor

Dados destinados a:	Valor atribuído a variável SEP_CONJ
Nenhum banco de dados	0
Banco de dados de validação	1
Banco de dados de treino	2
Banco de dados de teste	3

Nesse exemplo, a ordenação aleatória das amostras foi baseada nos valores do primeiro atributo, ou seja, PARAMETRO10. Os parâmetros que se seguem desse comando representam respectivamente, a porcentagem do total do banco de dados que será destinado ao conjunto de validação (8%), a porcentagem da soma dos conjuntos de treino e teste com relação ao número total do banco de dados (92%, ou seja 0% das amostras não pertencerá a nenhum conjunto), e finalmente o último parâmetro representa a razão entre o número de dados de treino versus teste. Nesse caso 50%, ou seja, metade dos 92% do banco de dados para cada conjunto. Porém para o treinamento da rede com os dados pré-processados, foi utilizado somente o conjunto de treino. O conjunto de teste foi usado para testar a rede já treinada.

O objeto SIEVE para cada conjunto serve para separar efetivamente os três bancos de dados. As expressões utilizadas em cada bloco SIEVE para a filtragem de cada conjunto foram as seguintes:

Para o conjunto de validação: $SEP_CONJ == 1$

Para o conjunto de teste: $SEP_CONJ == 2$

Para o conjunto de treino: $SEP_CONJ == 3$

Implementação de análise de componentes principais no Matlab

A implementação da análise de componentes principais dos dados de treinamento no aplicativo Matlab é feita com dois comandos do toolbox de Redes Neurais Artificiais: **premnmx** e **prepca**. Eles estão detalhados a seguir.

[pn,meanp,stdp] = premnmx(<Dados de Treino>);

Os dados de treino necessitam ser inicialmente normalizados pelo comando “premnmx” para que eles tenham média 0 (zero) e desvio padrão igual a 1. Em pn estará contido o vetor normalizado dos dados de treino. A variável meanp conterá a média das amostras de treino e stdp conterá o desvio padrão dos mesmos. E, na linha:

[ptrans,transMat] = prepca(pn,0.5);

O segundo argumento passado para o comando “prepca” determina que aqueles componentes principais de “pn” que contribuírem com menos de 0,5 % da variação total no conjunto de dados de treinamento serão eliminados.

A matriz “ptrans” contém os vetores de entrada transformados, e a matriz “transMat” contém a matriz que foi obtida da análise dos componentes principais das

amostras de treino da rede neural artificial. Portanto, multiplicando-se os dados de entrada com a matriz `transMat`, obtém-se a matriz `ptrans`.

Na implementação de uma rede neural do tipo `feedforward`, necessitamos fornecer a matriz “PR”, que nada mais é do que uma matriz $R \times 2$, onde R é o número de entradas da rede neural (no nosso caso, 8 entradas).

Foi usado o comando `premnmx` ao invés do comando `prepca`, pois ao contrário deste último que normaliza os dados de treinamento em um intervalo entre 0 e 1, o comando `premnmx` normaliza os dados no intervalo entre -1 e 1 . Com isso, foi possível o uso das funções de ativação de tangente hiperbólica, ao invés da sigmóide.

Os dados de teste necessitaram de transformações para serem usados nas redes neurais treinadas com os dados pré-processados e com a análise de componentes principais. A normalização dos dados de teste foi feita com o comando `tramnmx`. Foram utilizados os mesmos valores de máximos e mínimos dos dados de treino da RNA. A matriz com os dados de teste normalizados foi transformada com o comando `trapca`, utilizando a matriz de transformação `transMat` obtida dos dados de treinamento.

Após a simulação da rede neural com os dados de teste transformados por esses dois comandos `tramnmx` e `trapca`, o resultado da rede neural ainda se encontrava no intervalo entre -1 e 1 . Foi necessário utilizar o comando `postmnmx` para que os dados resultantes ficassem no intervalo entre 0 e 1; e pudessem ser comparados com os dados de teste com as saídas desejadas.

5.1 Reconhecimento da qualidade de álcool com Redes Neurais

Para o reconhecimento da qualidade do álcool combustível, foi usado o mesmo sistema de sensores Taguchi. Foram considerados dois padrões de álcool: de boa qualidade (com concentrações de 92.6% a 94.3%) e o de má qualidade (com outras concentrações). O álcool combustível foi injetado na câmara de sensores por meio de uma seringa. O sistema usado por rotâmetros foi substituído por um sistema de tubos. A limpeza da câmara foi feita com um ventilador e um filtro de ar contendo sílica gel, para além de retirar resíduos de álcool, retirar também a umidade. O sistema de aquisição de dados foi o mesmo.

Foram testados dois algoritmos de aprendizagem de redes neurais – `backpropagation` e `LVQ` – para reconhecer os padrões dos combustíveis gasosos, seja

com ou sem a análise de componentes principais. Sendo que neste último caso, a rede neural teve um número menor de entradas, em consequência da redução de dimensionalidade.

As redes neurais sem ACP tiveram 6 neurônios de entrada, 6 neurônios na camada escondida, e dois neurônios na camada de saída. Já as redes neurais com ACP tiveram 4 entradas, 4 neurônios na camada escondida e dois neurônios na camada de saída. O número de neurônios usados nas camadas escondidas foi obtido a partir de treinamentos prévios das redes neurais, e análises das capacidades de generalização para cada um dos casos, obtendo-se um valor ótimo empírico de número de neurônios escondidos.

Vários treinamentos foram realizados. O parâmetro para a interrupção do treinamento foi o erro RMS com valor de 0.01, ou o gradiente do erro de 0.001. Os resultados obtidos estão demonstrados na Tabela 5.1a.

A Fig. 5.1a mostra um gráfico obtido de um dos sensores com os dados experimentais do álcool combustível. Os pontos azuis escuros representam o padrão de álcool bom, enquanto os pontos em rosa representam os dados do álcool ruim. Pode-se notar que há sobreposição de alguns dados do álcool bom nos dados do álcool ruim, o que pode explicar os erros obtidos na generalização da rede neural.

Um estudo do efeito do pré-processamento nos dados de entrada com o aplicativo DataSculptor foi feito com dados sintéticos simulando a injeção de álcool na câmara de ensaios. O critério de convergência da rede foi o erro médio quadrático a 0.01. O método de pré-processamento escolhido foi o “brief”, pois não havia sobreposição dos dados sintéticos dos dois padrões, e além disso os dados dos dois padrões eram linearmente separáveis. O DataSculptor aplicou uma escala nos dados e separou-os randomicamente em conjuntos de treino e teste. A rede foi treinada dez vezes com os dados crus, e depois mais dez vezes com os dados pré-processados. Foi observado que a rede com os dados pré-processados foi treinada mais rapidamente, tendo uma razão média de épocas necessárias para o treinamento de 0.15 com relação a rede treinada com os dados sem o pré-processamento.

Tabela 5.1a: Resultados obtidos das RNA's no reconhecimento de álcool

	Com Pré-Processamento						Sem Pré-Processamento					
	Traingdx		Trainlm		Learnlvq		Traingdx		Trainlm		Learnlvq	
	Épocas	Acertos	Épocas	Acertos	Épocas	Acertos	Épocas	Acertos	Épocas	Acertos	Épocas	Acertos
Máximo	774	25	961	34	200	31	6395	28	119	25	200	31
Mínimo	170	15	2	15	200	31	1309	18	1	18	200	31
Média	369.95	20.19	221.33	25.38	200	31	2501.63	20.18	32.91	21.13	200	31
Desvio Padrão	159.04	3.39	218.19	5.97	0	0	1396.59	2.99	31.45	2.86	0	0

Fig. 5.1a: Variação da resistência versus resistência inicial (sensor 6)

5.2 Implementação da RNA em DSP (Digital Signal Processor)

Foi utilizado o software Neural Works Professional II Plus para construir o código em C da rede neural a ser utilizada. O procedimento da criação da rede neural artificial foi a seguinte:

Com o comando InstaNet [33], foi escolhido a rede neural do tipo Back Propagation. A seguir, foi escolhido o número de neurônios utilizados: oito entradas, duas saídas e três camadas escondidas (camadas 3, 4 e 5) com 8 neurônios em cada uma. Como regra de aprendizado, foi utilizado a backpropagation. Como função de transferência foi utilizada a função sigmóide (cuja expressão é $f(x) = 1/(1+\exp(-x))$). Para os bancos de dados de treino e de teste foram utilizados respectivamente os

arquivos `treino.nna` e `teste.nna` (para os dados crus, sem serem submetidos ao Data Sculptor), e depois os arquivos `cnj_trn.nna` e `cnj_tst.nna`.

O critério de convergência da rede foi escolhido como $\text{RMS Error} < 0,01$.

A tabela de aprendizado/teste de parâmetros da rede neural utilizada está mostrada na Tabela 5.2a:

Tabela 5.2a: Coeficientes utilizados no aplicativo Neural Works

Tabela de aprendizado/teste	
Learning Count	50000
Temperature	0,000
Learning Rate	0,9
Momentum	0,6
Error Tolerance	0
Weight Decay	0
Coefficient5	0,002
Coefficient6	0
Coefficient7	0
Coefficient8	0
Coefficient9	0

A tabela das várias tentativas e iterações de treinamento da rede utilizando-se as amostras pré-processadas e as amostras cruas é mostrada na Tabela 5.2b:

Tabela 5.2b: Iterações necessárias para o treinamento da rede neural

Iterações necessárias para treinar a rede backpropagation:	
Com Pré Processamento	Sem Pré Processamento
4162	9429
2158	28658
3700	26700
2658	25200
3658	9666
5158	29200
2656	31300
1700	23204
2429	9704

Pode-se notar que as amostras Pré-Processadas possibilitaram que a rede fosse treinada com um Erro RMS menor que 0,01, em um número de iterações menor do que se a rede fosse treinada com as amostras cruas. Os valores de iterações em **negrito** correspondem à rede resultante que foi utilizada para realizar o código em C da rede neural a ser implementada no kit de DSP.

Foi utilizado um kit de DSP da Analog Devices com o processador ADSP-21061 para implementar a rede neural já treinada, e gravados os dados de teste na memória EPROM do kit para posteriormente observar as suas saídas resultantes. Para isso, o banco de dados com as amostras sintéticas de teste foi utilizado como um arquivo *.h de biblioteca. Foi utilizado também um sinal de interrupção externa (IRQ1) ligado a um botão de pressionamento para que fosse possível o monitoramento da entrada dos sinais e das saídas resultantes.

O arquivo utilizado foi o “testev.h”. O código em C da rede neural foi obtida automaticamente pelo programa Neural Works Professional II Plus. Este código em C da rede neural gerada pelo Neural Works foi adaptado ao algoritmo e implementado no DSP.

5.2.1 O programa implementado no DSP

Primeiramente foram feitas as definições das várias variáveis do programa, bem como da matriz `cnj_teste` que contém o banco de dados `testev.h` com amostras cruas. A declaração `pm` na linha “float pm cnj_teste[NUM_REGISTROS][NUM_ATRIBUTOS]”, significa que a matriz será colocada na memória de programa do DSP. As declarações “volatile” significam que essas variáveis são globais e que podem mudar a qualquer instante.

Depois foi feito o protótipo da função `NN_Recall` e foram implementadas as funções que serão utilizadas no programa `main`. Trata-se de duas rotinas de interrupção, em que a primeira, “void timer_handler(int signal)” trata de uma interrupção por time-out, isto é, após um determinado tempo (definido pela contagem do número de ciclos da função `timer_set((unsigned int)10000, (unsigned int)10000)`; isto é, 10000 ciclos), a interrupção é gerada. A segunda rotina, void `irq1_handler(int signal)` trata-se de uma interrupção externa causada por um botão de pressionamento externo soldado no kit.

O código gerado pelo Neural Works já inclui os pesos resultantes da rede treinada. Essa função `NN_Recall`, não possui retropropagação, somente o passo de feed-forward. Ela tem como parâmetros de entrada as variáveis `yin[NUM_ATRIBUTOS (=8)]`, e `yout[NUM_SAIDAS (=2)]`. Essa função retorna zero caso sua execução for bem sucedida. Como o parâmetro `yout` é um arranjo (ou vetor) de dados do tipo float, a função ao modificar esse vetor internamente, automaticamente estará mudando os seus valores resultantes, que serão apresentados no término da execução dessa função.

Já a rotina principal `main` realiza o seguinte algoritmo:

1. Seta as variáveis de interrupções (time-out e o botão de pressionamento) para zero;
2. Configura as rotinas de serviço de interrupções (ISR's) para responder as interrupções;
3. Configura o timer para um time-out de 1000 ciclos de relógio;

4. Inicializa os LEDs e entra em um laço infinito sendo apresentadas todas as amostras seqüencialmente;
5. Espera ocorrer uma interrupção (instrução `idle()`);
6. Se ocorrer a interrupção do pressionamento do botão segue os seguintes passos:
7. Desliga o flag da interrupção `IRQ1`;
8. Sinaliza que a interrupção ocorreu pela mudança do `FLAG3` (`LED6`);
9. Atribui uma amostra do banco de dados a variável `yin`;
10. Insere o `yin` na função `NN_Recall` e é extraído o valor de `yout`;
11. Compara os valores de `yout[0]` e `yout[1]`. De acordo com a diferença entre esses valores, o álcool será classificado como bom ou ruim. A diferenciação é feita através dos leds `D3` e `D4`.
12. Volta para o passo 5

5.2.2 Compilação e Linkagem

Para compilar e linkar o programa do DSP, foi utilizado o comando:

```
g21k -o rna4.21k rna4.c
```

Onde o `g21k` é o compilador e linkador propriamente dito. A opção `-o` significa colocar a saída (arquivo compilado) em um nome especificado. No caso foi o `rna4.21k`.

Esse comando gerou o arquivo `rna4.21k`, que mais tarde foi submetido a um programa de interface do computador com o kit de DSP. Esse programa carrega o arquivo `rna4.21k` na placa do kit de DSP. E com isso, a placa começou a executar o programa sozinho.

Outro comando utilizado foi o seguinte:

```
g21k -g rna4.c -o rna4.exe
```

Esse comando é similar ao anterior, porém ele gera um arquivo chamado `rna4.exe`; e além disso a opção `-g` produz um código debugável para ser usado com o CBUG. O CBUG é um simulador que pode ser usado para testar o programa do projetista antes mesmo de ele ser carregado no kit.

5.2.3 Implementação da Rede Neural em Hardware

A rede neural artificial treinada com o programa `NeuralWorks` foi implementada em um kit de DSP da Analog Devices. Este programa extraiu os pesos de cada sinapse da rede neural, e eles foram transformados em um programa em linguagem C. Foi

considerado que o hardware não iria aprender por si próprio. Com isso, a função que implementa o passo forward da rede foi composta basicamente de somas e multiplicações. A maior complexidade computacional encontrada foi o cálculo da resposta da função de ativação dos neurônios, que tem caráter não-linear.

O sistema implementado inicialmente foi o de reconhecimento da qualidade do álcool combustível. Como uma primeira abordagem, o conjunto de dados de teste foi incluído no programa do kit do DSP. Com um botão do próprio kit, foi possível varrer todos os dados de teste, e observar o resultado da rede neural através de dois led's do próprio kit. Houve consistência dos resultados simulados pelo programa NeuralWorks, com os resultados observados nas respostas dos led's do kit.

Um hardware dedicado ao reconhecimento da qualidade de álcool composto por um microcontrolador foi implementado em nosso laboratório. O programa da rede neural foi armazenado na memória EEPROM, e os pesos sinápticos foram armazenados na memória FLASH. Como a memória disponível era pequena, o programa da rede neural gerado pelo NeuralWorks teve de ser otimizado. O sistema somente trata os valores iniciais e finais da resposta do sensor em regime permanente. Com esses valores, ele realizou os cálculos e apresentou a resposta da rede neural através dos led's soldados no cartão.

5.2.4 Resultados do programa implementado no DSP

Os resultados obtidos estão mostrados na Tabela 5.2.4a.

Como se pode observar, houve um erro no reconhecimento da amostra. Vemos também que se tratou do único resultado em que os valores de `yout[0]` e `yout[1]` ficaram bem parecidos. Ao invés de retornar um valor errado na saída, pode-se atribuir um valor de dúvida, para que erros sejam evitados no reconhecimento do álcool em questão. Isso mostra que a rede backpropagation é efetiva, porém não é perfeita. Há alguns erros que essa topologia deixa passar. Uma outra possibilidade para melhorar o reconhecimento é a escolha de um outro tipo de rede, como a Kohonen, LVQ, ADALINE, etc.

Tabela 5.2.4a: Resultados obtidos nas simulações das amostras de álcool

# Reg.	Arquivo testev.h		Simulações		Experimento - kit	
	Yout[0]	yout[1]	Yout[0]	yout[1]	yout[0]	yout[1]
0	1	0	1,02076	-0,0176	1	0
1	0	1	-0,0001	1,00072	0	1
2	1	0	1,01205	-0,0115	1	0
3	0	1	0,04834	0,95376	0	1
4	1	0	1,02130	-0,0178	1	0
5	0	1	-0,0014	1,0020	0	1
6	1	0	0,95314	0,03707	1	0
7	0	1	-0,0072	1,0075	0	1
8	0	1	-0,0071	1,0074	0	1
9	0	1	0,01681	0,98457	0	1
10	0	1	0,13379	0,86803	0	1
11	0	1	-0,0076	1,00792	0	1
12	0	1	-0,0065	1,00689	0	1
13	0	1	0,00455	0,99635	0	1
14	0	1	-0,0065	1,00681	0	1
15	0	1	0,00238	0,99842	0	1
16	1	0	0,99615	0,00079	1	0
17	1	0	1,01458	-0,0134	1	0
18	1	0	0,99666	0,00039	1	0
19	0	1	0,05668	0,94552	0	1
20	0	1	-0,0033	1,00382	0	1
21	1	0	1,01041	-0,0128	1	0
22	0	1	-0,0068	1,00717	0	1
23	0	1	0,02828	0,97343	0	1
24	1	0	0,96694	0,02505	1	0
25	0	1	-0,0017	1,00235	0	1
26	0	1	0,04369	0,95834	0	1
27	0	1	0,51529	0,47285	1	0
28	0	1	-0,0011	1,00175	0	1
29	0	1	0,00898	0,99211	0	1
30	0	1	-0,0059	1,00629	0	1
31	0	1	-0,0081	1,00832	0	1
32	0	1	0,07265	0,92965	0	1
33	0	1	-0,0074	1,00773	0	1
34	1	0	1,02107	-0,0177	1	0
35	0	1	-0,0082	1,00844	0	1
36	1	0	1,01699	-0,0151	1	0
37	1	0	1,01859	-0,0162	1	0
38	0	1	-0,0074	1,00768	0	1
39	0	1	-0,0038	1,00432	0	1
40	1	0	1,02141	-0,0178	1	0
41	1	0	1,02174	-0,0179	1	0
42	0	1	-0,0076	1,00787	0	1
43	1	0	1,02065	-0,0175	1	0
44	1	0	1,01292	-0,0121	1	0
45	0	1	0,00223	0,99856	0	1
46	1	0	1,01541	-0,0139	1	0
47	0	1	0,27583	0,72194	0	1
48	1	0	1,02168	-0,0179	1	0
49	0	1	-0,0082	1,00847	0	1

5.3 Implementação do Sistema de Inferência Fuzzy

Foram implementados dois sistemas FIS resolver o problema de classificação do poder calorífico do gás combustível, sendo o primeiro utilizando dados sem Pré-Processamento de dados e Análise de Componentes Principais, e o segundo utilizando essas duas ferramentas.

A abordagem utilizada para implementar o sistema FIS foi o uso da clusterização subtrativa, método ilustrado na Fig. 5.3a abaixo. Essa figura mostra os dados de entrada e as funções de pertinência ajustadas para as duas dimensões dos dados de entrada. A partir do conhecimento prévio de qual padrão pertence cada cluster de dados e também das funções de pertinência, é gerada a base de conhecimento do sistema FIS. Maiores detalhes da implementação desta abordagem estão descritos nas seções seguintes.

Fig. 5.3a: Clusterização Subtrativa utilizada no HS

5.3.1 FIS com os dados sem pré-processamento e PCA

As bases de conhecimento dos FIS foram extraídas a partir dos próprios dados sintéticos. Segundo [34], as regras fuzzy são extraídas das estimativas de clusters nos dados; sendo que cada cluster representa uma regra que relaciona uma região do espaço de entrada a uma classe de saída. A extração das regras fuzzy foi feita através da clusterização subtrativa, e com isso foram obtidos centros de cluster para cada padrão e sensor. Os atributos analisados foram a resistência inicial dos sensores e a variação normalizada dos mesmos. A Fig. 5.3.1a mostra os dados sintéticos usados como entradas do sistema FIS. Os centros dos clusters estão assinalados com símbolos negros. Os pontos coloridos são os padrões de combustível simulados, sendo vermelho o padrão

com maior poder calorífico, azul o padrão com menor poder calorífico, e o verde o padrão de combustível com poder calorífico intermediário.

Para a extração destes clusters foram usadas as mesmas amostras de treino das Redes Neurais Artificiais. Após a construção e ajuste da base de conhecimento, as amostras de teste foram usadas no FIS para testar sua capacidade de generalização. Os resultados e metodologias estão detalhados nas seções posteriores. O sistema Fuzzy implementado está mostrado na Fig. 5.3.1b.

Fig. 5.3.1a: Dados sintéticos de cada sensor usados no FIS.

Fig. 5.3.1b: Sistema Fuzzy com os dados sem pré-processamento e PCA

5.3.1.1 Modelamento das Variáveis Linguísticas

As variáveis linguísticas usadas para cada sensor foram a Resistência Inicial de cada sensor, e a sua resposta à injeção do gás representado pela Resistência Normalizada. As funções de pertinência foram ajustadas manualmente até obter um desempenho satisfatório.

Entrada – Resistência Inicial

A Fig. 5.3.1.1a mostra as funções de pertinência relacionada com a variável linguística **Resistência Inicial**. Esta variável representa a Resistência Inicial do Sensor no instante anterior à injeção do gás combustível na câmara com os sensores de gás. Esta função foi usada para todos os sensores utilizados.

- Variável Linguística: Resistência Inicial
- Universo de Discurso: 0 a 50 Ohms (valores normalizados para o intervalo [0;1])
- Valores Linguísticos: baixo, médio e alto

Fig. 5.3.1.1a: Função de Pertinência da Variável Resistência Inicial

Entrada – Resistência Normalizada para o Sensor 1 a 6

A Fig. 5.3.1.1b mostra as funções de pertinência relacionadas com as variáveis lingüísticas **Resistência Normalizada** para os sensores 1 a 6. Esta variável representa o resultado do cálculo relacionado com as resistências final e inicial dos sensores em regime permanente. O cálculo é feito através da equação (4a). Estas funções de pertinência foram ajustadas manualmente.

- Variável Lingüística: Resistência Normalizada
- Universo de Discurso: 0 a 1
- Valores Lingüísticos: baixo, médio e alto

Saídas

A Fig. 5.3.1.1c mostra as funções de pertinência relacionadas com a variável de saída de cada sistema para cada sensor.

- Variável Lingüística: Saída
- Universo de Discurso: 0 a 1.
- Valores Lingüísticos: Classe1, Classe2 e Classe3

Fig. 5.3.1.1b: Funções de Pertinência ajustadas para cada sensor.

Fig. 5.3.1.1c: Funções de pertinência da saída do FIS.

5.3.1.2 Base de Conhecimento do FIS

As regras utilizadas para o reconhecimento dos padrões foram baseadas nos dados de treino, e nos clusters obtidos. Para todos os sensores as regras foram as mesmas. Por exemplo, para o sensor número 5, as regras foram:

1. SE (Ri5 é baixo) E (Atr5 é baixo) ENTÃO (output5 é classe2)
2. SE (Ri5 é médio) E (Atr5 é baixo) ENTÃO (output5 é classe2)
3. SE (Ri5 é alto) E (Atr5 é baixo) ENTÃO (output5 é classe2)
4. SE (Ri5 é baixo) E (Atr5 é médio) ENTÃO (output5 é classe3)
5. SE (Ri5 é médio) E (Atr5 é médio) ENTÃO (output5 é classe3)
6. SE (Ri5 é alto) E (Atr5 é médio) ENTÃO (output5 é classe3)
7. SE (Ri5 é baixo) E (Atr5 é alto) ENTÃO (output5 é classe1)
8. SE (Ri5 é médio) E (Atr5 é alto) ENTÃO (output5 é classe1)
9. SE (Ri5 é alto) E (Atr5 é alto) ENTÃO (output5 é classe1)

5.3.1.3 Metodologia utilizada para realizar os tratamentos dos dados

Para criar o ambiente necessário para realizar os tratamentos de dados no Sistema de Inferência Fuzzy foi feito o seguinte procedimento:

- 1) Criar as amostras de treino e de teste através da execução arquivo “AmSint1.m”.

Este arquivo criará amostras sintéticas relacionadas com o gás combustível. Ele gerará os arquivos com as 300 amostras sintéticas que serão usados tanto na Rede Neural Artificial quanto no Sistema de Inferência Fuzzy (FIS).

- 2) Para modelar o FIS

O arquivo utilizado do FIS foi fzrulext__1.mdl, feito através do Simulink (ver Fig. 5.3.1b). Neste arquivo foram modeladas todas as funções de pertinência de entrada e saída referentes a todos os componentes principais, além das regras fuzzy usados no FIS, descritos na seção anterior.

- 3) Executar o arquivo “fre.m” (extração de regras fuzzy)

Este arquivo criará todas as variáveis de entrada do FIS, usando os arquivos de treino e teste obtidos anteriormente. Ele também inserirá os FIS dos 6 sensores na área de trabalho do Matlab. Atribuir valor zero à variável **PreProcessamento**.

- 4) Executar a simulação do arquivo do Simulink fzrulext__1.mdl.

A execução do FIS no Simulink criará os arquivos 'resultadosensor(i)' sendo i, a denominação dos sensores de 1 a 6.

- 5) Executar o arquivo res_g_fis (resultado global do fis)

Este arquivo criará os resultados globais obtidos dos 6 FIS criados separadamente. Os pontos resultantes foram colocados no gráfico da Fig. 5.3.1.4b.

5.3.1.4 Resultados Obtidos com o Sistema de Inferência Fuzzy

Os resultados do FIS sem otimizações nas funções de pertinência não foram muito satisfatórios. As 150 amostras foram submetidas ao FIS sequencialmente, uma a cada segundo, sendo as primeiras 50 primeiras correspondentes ao primeiro padrão, as 50 subsequentes pertencentes ao padrão 3, e as últimas 50 correspondentes ao padrão 2. Foram estabelecidos limites para as saídas do FIS: valores de 0 a 0,25 reconhece o padrão 1; valores de 0,25 a 0,75 reconhece o padrão 3, e valores de 0,75 a 1,0 reconhece o padrão 2.

As curvas de resposta de cada subsistema FIS para cada sensor estão mostradas na Fig. 5.3.1.4a. Pode-se observar que somente o sensor 2 teve um resultado satisfatório. Isso porque os padrões estavam bem definidos e sem sobreposições. Pode-se notar que houve muitos erros no reconhecimento do padrão 3, mesmo com o ajuste manual das funções de pertinência. Isto porque os padrões observados nos outros sensores ficaram sobrepostos, como se pode observar na Fig. 5.3.1a.

Na Fig. 5.3.1.4b pode ser observada a resposta global do sistema fuzzy, isto é, o resultado da votação de todos os sistemas fuzzy implementados. Os valores de saída correspondem à classe reconhecida, por exemplo, o valor 1 corresponde à classe 1, 2 corresponde à classe 2 e o valor 3 corresponde à classe 3. O valor 4 é obtido quando a votação das três classes obtiveram o mesmo número de votos. São, portanto, os casos em que houve as maiores dúvidas. Valores intermediários de 1,5 e 2,5 representam que o sistema ficou com resposta duvidosa entre os padrões 1 e 2, e os padrões 2 e 3 respectivamente.

Fig. 5.3.1.4a: Resultados Obtidos dos FIS para cada sensor.

Fig. 5.3.1.4b: Resultado final da votação obtida para a resposta global do FIS.

Houve 74% de acertos de reconhecimento dentre as amostras de teste. Pode ser observado que o FIS não tem uma boa performance de reconhecimento quando os padrões a serem reconhecidos estão sobrepostos. A implementação do FIS também é trabalhosa, pois é necessário realizar vários ajustes nas funções de pertinência para as entradas e saídas até que se pudesse obter um resultado satisfatório.

Uma etapa de pré-processamento de dados e análise de componentes principais nos dados de entrada de treino e de teste pode ser usada para melhorar a taxa de acertos do sistema de inferência fuzzy utilizado. E esta abordagem foi implementada na seção seguinte.

5.3.2 FIS utilizando dados com pré-processamento e PCA

Os dados sintéticos foram os mesmos usados na seção anterior. Porém eles foram submetidos a um pré-processamento e análise de componentes principais. Para isto, os dados de treino foram normalizados de tal modo que eles ficassem no intervalo $[-1;1]$ através da função do toolbox do programa Matlab **premnmx**.

Em seguida, foi usado o comando **prepca** para que somente os componentes dos dados com a fração mínima de variância de valor 0,07 com relação a variação total do conjunto de dados, fosse incluído no conjunto de dados transformado. Em seguida, foi utilizado o comando **subclust** para extrair os clusters subtrativos dos dados já transformados. O conjunto de dados de treino foi reduzido de 6 atributos para somente dois atributos, que foram colocados num gráfico atributo1 “versus” atributo2 mostrado na Fig. 5.3.2a. Os dados em preto representam o primeiro padrão de gás GLP com o maior poder calorífico. Os dados em vermelho representam o terceiro padrão de GLP com o menor poder calorífico, e os dados em azul representam o segundo padrão de GLP com um valor intermediário de poder calorífico. Os dados de cor verde representam os clusters obtidos pelo processo de clusterização subtrativa. Pode-se notar que as classes 2 e 3 foram invertidas com relação ao experimento anterior, mas isso não mudou a metodologia, e o tratamento dos dados e resultados experimentais não ficaram comprometidos.

Após a construção e ajuste da base de conhecimento, as amostras de teste foram transformadas utilizando as mesmas transformações sofridas pelo conjunto de treino. E os dados resultantes de treino foram usados no FIS para testar sua capacidade de generalização. Os dados de teste resultantes estão mostrados na Fig. 5.3.2b. Os

resultados e metodologias estão detalhados nas seções posteriores. O sistema Fuzzy está mostrado na Fig. 5.3.2c.

Fig. 5.3.2a: Gráfico dos dois Componentes Principais com os Clusters dos dados de treino

Fig. 5.3.2b: Dados sintéticos de teste usados no FIS.

Fig. 5.3.2c: Sistema FIS utilizado com os dados de treino submetidos ao pré-processados e análise de componentes principais

5.3.2.1 Modelamento das Variáveis Lingüísticas

As variáveis lingüísticas usadas cada sensor foram os dois componentes principais dos dados de treino, denominados **PrinComp1** e **PrinComp2** respectivamente. As funções de pertinência foram ajustadas manualmente até obter um desempenho satisfatório.

Entrada – Componente Principal 1

A Fig. 5.3.2.1a mostra as funções de pertinência relacionada com a variável lingüística **PrinComp1**. Esta variável representa o primeiro componente principal da análise feita nos dados de treino.

- Variável Lingüística: PrinComp1
- Universo de Discurso: -2 a 2
- Valores Lingüísticos: baixo1, médio1 e alto1

Fig. 5.3.2.1a: Função de Pertinência da Variável Lingüística PrinComp1

Entrada – Componente Principal 2

A Fig. 5.3.2.1b mostra a função de pertinência relacionadas com a variável lingüística **PrinComp2** para o segundo componente principal dos dados de treino. Esta variável representa o resultado do cálculo relacionado com as resistências inicial e final dos sensores em regime permanente. O cálculo é feito através da equação (4a). Estas funções de pertinência foram ajustadas manualmente.

- Variável Lingüística: Resistência Normalizada
- Universo de Discurso: 0 a 1
- Valores Lingüísticos: baixo2, médio2 e alto2

Fig. 5.3.2.1b: Função de Pertinência da Variável Lingüística PrinComp2

Saída – Separação das Classes

A Fig. 5.3.2.1c mostra as funções de pertinência relacionadas com a variável de saída de cada sistema para cada sensor.

- Variável Lingüística: Saída
- Universo de Discurso: 0 a 1.
- Valores Lingüísticos: Classe1, Classe2 e Classe3

Fig. 5.3.2.1c: Funções de pertinência da saída do FIS.

Base de Conhecimento do FIS

As regras utilizadas para o reconhecimento dos padrões foram baseadas nos dados de treino, e nos clusters obtidos. As regras utilizadas foram:

1. SE (PrincComp1 é baixo1) E (PrinComp2 é baixo2) ENTÃO (output1 é classe1)
2. SE (PrincComp1 é baixo1) E (PrinComp2 é médio2) ENTÃO (output1 é classe1)
3. SE (PrincComp1 é baixo1) E (PrinComp2 é alto2) ENTÃO (output1 é classe1)
4. SE (PrincComp1 é médio1) E (PrinComp2 é baixo2) ENTÃO (output1 é classe2)
5. SE (PrincComp1 é médio1) E (PrinComp2 é médio2) ENTÃO (output1 é classe2)
6. SE (PrincComp1 é médio1) E (PrinComp2 é alto2) ENTÃO (output1 é classe2)
7. SE (PrincComp1 é alto1) E (PrinComp2 é baixo2) ENTÃO (output1 é classe3)
8. SE (PrincComp1 é alto1) E (PrinComp2 é médio2) ENTÃO (output1 é classe3)
9. SE (PrincComp1 é alto1) E (PrinComp2 é alto2) ENTÃO (output1 é classe3)

5.3.2.2 Metodologia utilizada para realizar os tratamentos dos dados

Para criar o ambiente necessário para realizar os tratamentos de dados no Sistema de Inferência Fuzzy foi feito o seguinte procedimento:

- 1) Criar as amostras de treino e de teste através do arquivo “AmSint1.m”.

Este arquivo criará amostras sintéticas relacionadas com o gás combustível. Ele gerará os arquivos “treino.txt” e “teste.txt” que serão usados tanto na Rede Neural Artificial quanto no Sistema de Inferência Fuzzy (FIS).

- 2) Para modelar o FIS

O arquivo utilizado do FIS foi FreFis.mdl, feito através do Simulink (ver Fig. 5.3.2c). Neste arquivo foram modeladas todas as funções de pertinência de entrada e saída referentes a todos os sensores, além das regras fuzzy usados no FIS, descritos na seção anterior.

- 3) Executar o arquivo fre.m (extração de regras fuzzy versão 1.0)

Este arquivo criará todas as variáveis de entrada do FIS, usando os arquivos de treino e teste obtidos anteriormente. Ele também inserirá o FIS correspondente aos dados com pré-processamento e análise de componentes principais na área de trabalho do Matlab. Atribuir valor um à variável **PreProcessamento**.

4) Executar o modelo do Simulink FreFis.mdl

A execução do FIS no Simulink criará a variável ResultadoFrePCA na área de trabalho no Matlab.

5) Analisar o resultado obtido no passo anterior

Foram contadas as amostras em que o reconhecimento foi errôneo, através da análise da Fig. 5.3.2.3a da seção seguinte.

5.3.2.3 Resultados Obtidos com o Sistema de Inferência Fuzzy

Como foi obtido somente um controlador Fuzzy, não foram necessárias as etapas de combinação e votação dos vários sistemas FIS, e com isso, foi obtido somente uma curva de resposta do FIS que está mostrada na Fig. 5.3.2.3a. Para obter esta curva de resposta, foram utilizadas as mesmas amostras de teste da seção anterior, para comparação, e também para avaliar a capacidade de generalização do sistema FIS. Pode-se observar que houve uma melhora de reconhecimento com relação ao sistema FIS implementado sem o pré-processamento e análise de componentes principais, obtidas na seção 5.3.1 desta Dissertação.

Os dados foram agrupados do seguinte modo: as amostras de número 1 a 50 pertencem ao padrão 1, as amostras de número 51 a 100 pertencem ao padrão 3, e finalmente as amostras de número 101 a 150 pertencem ao padrão 2. Por causa das condições iniciais impostas pelo aplicativo Simulink do programa Matlab, a amostra de número 0 não foi considerado no cálculo das taxas de erros e acertos do reconhecimento. Cada amostra foi inserida no sistema a uma taxa de amostragem de 1s. A simulação do programa Matlab foi feita entre os tempos 0 e 155s, portanto, tempos superiores a 151s também não foram considerados no cálculo.

Fig. 5.3.2.3a: Resultado do FIS para os dados com pré-processamento e análise de componentes principais.

Considerando os limiares de 0,3 para o reconhecimento entre os padrões 1 e 3; e 0,7 para os padrões 2 e 3, pode-se observar que houveram 12 erros de reconhecimento em um universo de 150 amostras. Portanto, foi obtida uma taxa de acertos de 92,0% no reconhecimento das amostras de teste. Pode ser observado que o pré-processamento dos dados e a análise de componentes principais nos dados de entrada fizeram com que o número de amostras dos padrões sobrepostas fosse diminuído. Isso fez com que o desempenho do FIS para a etapa de reconhecimento melhorasse. A implementação do FIS deste item foi menos trabalhosa que a da seção anterior, pois foram necessários ajustes nas funções de pertinência das entradas e da saída de somente um sistema FIS, até que se pudesse obter um resultado satisfatório. O fato dos dados de entrada estarem bem separados e limitados também facilitou a criação das funções de pertinência.

5.4 Reconhecimento do poder calorífico do GLP em Redes Neurais

Foram testadas dois algoritmos de aprendizado de redes neurais – a backpropagation e LVQ – para reconhecer os padrões dos combustíveis gasosos. Para a rede backpropagation, foram usadas duas funções de treinamento, a backpropagation com o gradiente decrescente com momentum adaptativo (função `traingdx` do Matlab) e

a função backpropagation Levenberg-Marquardt (trainlm do Matlab), seja com ou sem a análise de componentes principais. Sendo que neste último caso, a rede neural teve um número menor de entradas, em consequência da redução de dimensionalidade.

Os dados de entrada utilizados para o treino e teste foram os dados sintéticos obtidos segundo o procedimento descrito no item 4.2. Os 300 dados obtidos foram divididos em dois conjuntos de 150 amostras de treino e 150 amostras teste. Cada conjunto tinha 50 amostras para cada um dos três padrões de gás GLP.

As redes neurais backpropagation sem ACP tiveram 6 neurônios na camada de entrada, 6 neurônios na camada escondida, e três neurônios na camada de saída. Já as redes neurais com ACP tiveram 4 entradas, 4 neurônios na camada escondida e três neurônios na camada de saída.

Para cada rede, foram feitos 10 treinamentos. Os parâmetros para a parada do treinamento foi o erro RMS de 0.01, e o gradiente do erro de 0.001. Os resultados obtidos com as médias e variâncias das redes estão demonstrados na Tabela 5.4a a seguir.

O número de acertos foi obtido com testes das redes neurais usando as amostras de teste. Pode-se notar que o desempenho da rede neural backpropagation usando a função 'traingdx' teve um maior índice de acertos, com menor desvio padrão. A rede backpropagation 'trainlm', apesar de ter o maior índice de acertos com muitas poucas épocas de treinamento, houve muitos casos com poucos acertos que não foram computados nesta tabela. O treinamento da rede LVQ foi limitado a 200 épocas, porque foi observado que o erro RMS oscilava após a época 75. Portanto a rede neural treinada no Matlab eleita foi a backpropagation com a função de treinamento 'traingdx' sem pré-processamento.

Tabela 5.4a: Treinamentos das RNA's com Matlab

	Com Análise de Componentes Principais						Sem Análise de Componentes Principais					
	Traingdx		Trainlm		Learnlvq		Traingdx		Trainlm		Learnlvq	
	Épocas	Acertos	Épocas	Acertos	Épocas	Acertos	Épocas	Acertos	Épocas	Acertos	Épocas	Acertos
Máx.	633	140	1028	142	200	134	915	144	139	148	200	135
Mín.	262	135	25	129	200	134	285	128	14	138	200	134
Média	389.45	137.94	117.34	137.28	200	134	444.49	139.97	45.31	143.17	200	134
Desvio Padrão	87.64	1.25	148.96	2.90	0	0	116.52	2.85	31.64	2.35	0	134.7

Outro aplicativo utilizado para o treinamento da rede neural foi o NeuralWorks. Os mesmos conjuntos de treino e de teste da rede foram submetidos ao programa. A rede neural teve seis neurônios na camada de entrada e na camada escondida, e mais três neurônios na camada de saída. Foram feitos dez treinamentos para a rede neural MLP com o algoritmo de aprendizado backpropagation sem a tabela “MinMax”, mais dez treinamentos com essa tabela. Segundo [20], a tabela “MinMax” armazena máximos e mínimos de cada dado de entrada. Ela é usada para realizar um pré-processamento que aplica uma escala nos dados de entrada. Assim, ela previne a saturação da função de ativação que faz com que o neurônio pare de aprender. Os critérios de parada do treinamento foram a máxima iteração de 50000 épocas, ou o erro RMS menor que 0.001. O máximo número de iterações foi fixado para que a rede não sofresse “overfitting”.

Neste mesmo aplicativo foi treinada uma rede LVQ. A regra de treinamento para as 4500 primeiras iterações usou um fator de consciência de 1.0, o qual encorajou todos os neurônios no aprendizado. Para outras 2250 iterações, a regra de treinamento fez um refinamento de limites entre as classes. Os resultados dos treinamentos da rede neural com o aplicativo NeuralWorks estão demonstrados na Tabela 5.4b. A análise de componentes principais não foi utilizado neste programa.

Os resultados de generalização da rede neural treinada pelo NeuralWorks foram melhores. Portanto, a rede escolhida para implementar a rede neural em hardware foi a criada pelo programa NeuralWorks com a tabela MinMax. Uma possível implementação da Rede Neural para o reconhecimento de gás GLP no hardware do DSP seria muito parecida com o apresentado no item 5.2.

Tabela 5.4b: Treinamento das RNA's com Neural Works

	Rede Backpropagation Sem Tabela “MinMax”		Rede Backpropagation Com Tabela “MinMax”		Rede LVQ	
	Épocas	Acertos	Épocas	Acertos	Épocas	Acertos
Máximo	50000	142	50000	147	6750	134
Mínimo	50000	141	50000	145	2545	135
Média	50000	141.6	50000	146.1	5711.6	134
Desvio Padrão	0	0.52	0	0.74	1565.5	134.7

Porém, neste trabalho foram utilizadas amostras de gases em dois padrões. O primeiro padrão de gás GLP foi a injeção de 200ml de gás na câmara de sensores. Já o segundo padrão de gás GLP foi injetado 1000ml de uma mistura de GLP com gás

Nitrogênio, numa proporção de 1:5. Essa mistura foi obtida por meio do ajuste dos fluxos dos gases GLP em 200sccm e Nitrogênio em 1000sccm. A metodologia adotada para a extração dessas amostras de gases já foram discutidas nos itens 4.2 e 4.3.2 deste trabalho.

Foram obtidas 36 amostras da injeção de 200ml do gás GLP e 43 amostras da injeção da mistura de 200sccm de GLP com 1000sccm de gás Nitrogênio. O tratamento desses dados obtido experimentalmente está descrito nos itens posteriores. Os dados medidos para cada sensor estão mostrados na Fig. 5.4b.

5.4.1 Tratamento dos dados experimentais do GLP

O tratamento dos dados experimentais foi feito no programa MATLAB. Foram feitas medidas com o gás GLP puro e a mistura de gás GLP e Nitrogênio. Sendo que esta última estava simulando um tipo de combustível com menor poder calorífico que o primeiro. O volume injetado do gás GLP puro foi de 0.2ml, enquanto que o volume injetado da mistura foi de 1ml. Essa mistura foi obtida com a injeção de um fluxo de 200sccm de gás GLP para 1000sccm de Nitrogênio. Ou seja, a proporção da mistura foi de 1 parte de GLP para 5 partes de gás Nitrogênio.

Os valores iniciais e finais de cada medida foram submetidos a um programa feito em Matlab (arquivo “amostras_1.m”), destinado a tratar os dados e realizar o reconhecimento do poder calorífico de cada padrão de gás combustível.

Fig. 5.4b: Amostras do Gás GLP e da mistura GLP com Nitrogênio.

Os atributos correspondentes à resistência normalizada dos sensores foram calculados segundo a equação (4a). O tratamento dos dados foi feito segundo a execução do arquivo em Matlab “ExtAtributos1.m”. As amostras de cada padrão foram divididas em dois conjuntos aproximadamente iguais, sendo um destinado ao treino do sistema reconhecedor, e o outro foi destinado ao teste. Como o número de amostras era reduzido, foi optado não criar um novo conjunto de validação, para que o treinamento

não fosse comprometido com poucos exemplos a serem aprendidos. Isto é, os dados foram divididos em somente dois subconjuntos, um de treino e outro de teste.

Os dados extraídos foram submetidos a sistemas de reconhecimento baseados em redes neurais artificiais e em sistemas de inferência fuzzy, sendo este último detalhado no capítulo 5.3. A primeira abordagem foi o treinamento da rede neural e o ajuste de um sistema de inferência fuzzy com os dados obtidos com todos os sensores funcionando corretamente.

A segunda abordagem adotada foi a simulação de sensores com falha nos dados medidos. Foi observado empiricamente que um sensor com falha não tem sensibilidade alguma quando exposto ao gás a ser medido. Como os valores iniciais e finais não mudam, o valor submetido ao sistema reconhecedor será zero, por causa da normalização adotada para a obtenção dos dados medidos.

5.4.2 Primeiro Experimento: Rede Neural Simples

Uma arquitetura de rede neural artificial foi treinada 100 vezes até obter um melhor resultado de generalização. Deste modo, foi escolhida uma rede tal que o erro de reconhecimento das amostras de teste foi o menor de todos. Os dados utilizados para o treinamento dessas redes foram os próprios dados medidos experimentalmente, considerando que todos os sensores estavam funcionando corretamente. A rede neural está mostrada na Fig. 5.4.2a. Um exemplo de curva de erro RMS obtida versus número de iterações está mostrado na Fig 5.4.2b.

E para isso, foi executada a função **RNAMelhorGeneralizacao.m** 10 vezes, com o número de 10 experimentos até encontrar a melhor rede neural com melhor generalização. A seleção do tipo de tratamento de dados de entrada utilizados (com ou sem pré-processamento e análise de componentes principais) foi selecionado através de um parâmetro de entrada na execução da função **RNAMelhorGeneralizacao.m**.

As características da Rede Neural usadas nesse experimento estão descritas na Tabela 5.4.2a:

Sem Pré-Processamento e Análise de Componentes Principais:

Tabela 5.4.2a: Propriedades da RNA sem PCA

epochs	Goal	■	■ dec	■ inc	max_fail	Max perf inc	Mc	min_grad	show	time
10000	0	0.01	0.7	1.05	5	1.04	0.9	1e-7	25	inf

Tipo de Rede	Função de Treinamento	Função de Adaptação	Função de Performance (erro)	Número de Camadas
Backpropagation	traingdx	LEARNGDM	Erro médio quadrático	2

Propriedades da camada:	Número de Neurônios	Transfer Function
1 (entrada)	8	-
2 (escondido)	10	sigmóide logarítmica
3 (saída)	2	Sigmóide logarítmica

Fig. 5.4.2a: RNA usada no Reconhecimento dos dados experimentais

Fig. 5.4.2b: Exemplo de curva de erro obtido dos dados experimentais

Para os dados dos sensores sem pré-processamento, os resultados de generalização da melhor rede neural foram de 100% acertos para um determinado conjunto de amostras de teste. Porém, o tempo de treinamento foi elevado, com 10 mil iterações para cada rede neural treinada, e o erro foi de pouco mais de 0,01.

Com Pré-Processamento e Análise de Componentes Principais:

Foi utilizada a mesma metodologia da seção anterior, porém os dados de treino e teste foram submetidos a uma análise de componentes principais. A arquitetura de rede neural artificial foi treinada 10 vezes até obter um melhor resultado de generalização. Os dados utilizados para o treinamento dessas redes foram os dados medidos experimentalmente, considerando que todos os sensores estavam funcionando corretamente. A rede neural está mostrada na Fig. 5.4.2c. Um exemplo de curva de erro RMS obtida versus número de iterações está mostrado na Fig 5.4.2d.

Os parâmetros da rede estão mostrados na Tabela 5.4.2b:

Tabela 5.4.2b: Parâmetros utilizados na RNA com PCA

epochs	goal	■	■ dec	■ inc	max_fail	Max perf inc	Mc	min_grad	show	time
10000	0	0.01	0.7	1.05	5	1.04	0.9	1e-20	25	inf

Tipo de Rede	Função de Treinamento	Função de Adaptação	Função de Performance (erro)	Número de Camadas
backpropagation	traindx	LEARNGDM	Erro médio quadrático	2

Propriedades da camada:	Número de Neurônios	Transfer Function
1 (entrada)	6	-
2 (escondido)	3	Sigmóide logarítmica
3 (saída)	2	Sigmóide logarítmica

Fig. 5.4.2c: Rede Neural com PCA treinada com os dados experimentais

Fig. 5.4.2d: Exemplo de curva de erro da RNA com dados experimentais

Submetendo os dados a um pré-processamento e uma análise de componentes principais, foi observado que a rede também reconheceu 100% das amostras de teste. Além disso, foi observado um tempo de treinamento muito menor do que o caso anterior: com aproximadamente 1000 iterações, a rede neural convergia para erros RMS menores que 10^{-20} .

5.4.3 Tratamento dos dados que simulam um sensor com falha

A partir dos dados obtidos experimentalmente, foram gerados dados que simulavam a falha de cada um dos sensores individualmente. Deste modo, para cada sensor com falha, os seus valores para todas as amostras de treino foram zerados. Isto porque foi observado experimentalmente que a falha do sensor faz com que o sensor pare de responder a estímulos dos vapores de gases. Assim, o valor de resistência extraído do sensor terá uma variação normalizada nula.

O conjunto de dados de treinamento foi multiplicado pelo número de sensores mais um. Como os sensores de temperatura e umidade não foram considerados como possíveis de ter falhas, o novo conjunto de dados de treino teve um tamanho de $6+1=7$ vezes do tamanho original. O conjunto dos dados de teste também foi submetido a essa geração de dados de sensores com falha e, portanto, teve o seu tamanho também multiplicado por esse fator.

5.4.4 Segundo Experimento: RNA de sistema com um sensor em falha

Nesta abordagem, uma rede neural com as mesmas características da rede anterior foi treinada com todos os dados originais ou com os dados que simulavam a falha de um sensor. No caso dos dados que simulavam a falha de um sensor, sem pré-processamento e análise de componentes principais, houve um caso ótimo em que foram reconhecidos 260 das 273 amostras dos dados de teste. Todos os treinamentos acabaram com 10mil épocas, que foi o critério máximo adotado. Com o pré-processamento e análise de componentes principais de dados houve um resultado ótimo em que foram reconhecidos 266 das 273 amostras, utilizando uma média de 1000 épocas de treinamento. Os resultados estão detalhados na Tabela 5.4.6a na coluna denominada “RNNa Simples”.

A geração dos dados sintéticos que simulavam a falha do sensor foi feita por meio da execução do arquivo “CriaDadosRuidosos.m”. A criação e o uso da rede neural que utiliza esses dados sintéticos foram feitos por meio da execução da rotina “Experimento1.m”.

Para tentar melhorar a generalização do sistema reconhecedor, foram utilizadas as máquinas de comitê, detalhadas nas seções seguintes.

5.4.5 Terceiro Experimento: Implementação das Máquinas de Comitê

Na segunda abordagem, foi proposta a máquina de comitê mostrado na Fig. 5.4.5a abaixo. Esta máquina teve sete redes neurais, sendo que seis redes foram treinadas, levando em conta os sensores de gases com falha. Os sensores de temperatura e umidade relativa não foram considerados nessas simulações. A primeira rede neural foi treinada levando em conta que o sensor 1 estava com falha, e desta maneira, ela simplesmente ignorou os dados do sensor 1 para o seu treinamento. A segunda rede neural ignorava os dados do sensor 2, e assim por diante até a sexta rede neural. A sétima rede neural foi treinada levando-se em conta que todos os sensores estavam funcionando normalmente.

O arquivo correspondente à implementação desta abordagem de sistema reconhecedor foi denominado de “Teste1.m”.

Fig. 5.4.5a: Primeira proposta de máquina de comitê

As redes neurais foram treinadas individualmente para depois serem juntadas neste sistema. Todas foram submetidas a 10 treinamentos com valores iniciais aleatórias dos pesos. As redes neurais com melhores generalizações foram escolhidas para fazerem parte do sistema.

A etapa de Seleção faz o direcionamento do dado inserido no sistema para a rede neural incumbida de fazer o seu reconhecimento. A escolha foi feita da seguinte maneira: se os dados de todos os sensores fossem diferentes de zero, foi considerado que todos os sensores estavam em bom estado e, portanto, esta amostra foi submetida à sétima rede neural. Caso a resposta do sensor 1 for nulo, ela foi submetida a rede neural 1, em que foi considerado o sensor 1 com falha. E assim por diante para os demais sensores.

No caso da seleção não cega do sensor com falha, isto é, quando a informação da falha de um determinado sensor é passada previamente para o sistema, o mesmo obteve um resultado ótimo de 271 acertos dentro do universo de 273 amostras, no caso das redes neurais utilizarem os dados crus. Isto é, houve uma taxa de acertos de 99,3%. Já o sistema em que os dados foram submetidos a um pré-processamento e análise de componentes principais, o resultado da generalização do sistema foi de 270 acertos dentro do universo de 273 amostras de teste. Isso resultou em uma taxa de acerto de 98,9%.

No caso da seleção cega, isto é, quando a informação da falha de um determinado sensor não era passada para o sistema reconhecedor e o sistema inferia por si só a falha de um dos sensores analisando os dados de entrada, houve um caso ótimo de 262 acertos sem pré-processamento e análise de componentes principais, e 263 acertos no caso em que foram usados o pré-processamento e análise de componentes principais.

5.4.6 Quarto Experimento: Redes Ensemble

Foi implementado um sistema de dez redes neurais em paralelo, de modo que suas saídas foram combinadas linearmente para produzir a resposta com maior probabilidade de reconhecimento do sistema. O sistema construído está demonstrado na Fig. 5.4.6a a seguir. O arquivo correspondente a implementação das redes ensemble em Matlab foi denominado “Ensemble.m”.

Fig. 5.4.6a: Máquina de Comitê Ensemble

Todas as redes foram treinadas com condições iniciais aleatórias e diferentes entre si. Elas foram combinadas simplesmente somando as suas respostas, dividindo por dez, e comparando com as respostas desejadas.

As redes ensemble que utilizavam os dados sem o pré-processamento e análise de componentes principais tiveram um caso ótimo em que houveram 250 acertos dentro do universo de 273 amostras. Já o resultado ótimo de generalização do sistema de redes neurais com pré-processamento de dados e análise de componentes principais foi de 264 acertos dentro do universo de 273 amostras.

Tabela 5.4.6a: Resultados de RNA simples ou máquinas de comitê, que simulavam a falha de um dos sensores.

	RNA Simples		Máquina de Comitê Substitutiva (Seleção Cega)		Máquina de Comitê Substitutiva (Seleção Não Cega)		Máquina de Comitê Ensemble	
	Sem PP e PCA	Com PP e PCA	Sem PP e PCA	Com PP e PCA	Sem PP e PCA	Com PP e PCA	Sem PP e PCA	Com PP e PCA
Média	235,91	257,73	260,00	262,50	267,33	266,67	236,00	262,00
Máximo	260,00	266,00	262,00	263,00	271,00	270,00	250,00	264,00
Mínimo	223,00	246,00	256,00	262,00	263,00	264,00	231,00	251,00
Desvio Padrão	9,48	6,72	3,46	0,71	4,04	3,06	5,92	3,74

6 Conclusões e Perspectivas Futuras

Foi demonstrado que é possível implementar um sistema embarcado para o reconhecimento do poder calorífico de um gás combustível ou da qualidade do álcool combustível, tanto com redes neurais quanto com um FIS. A rede neural teve uma alta taxa de acertos, comprovando a sua capacidade de generalização.

A análise de componentes principais é uma ferramenta importante na redução da dimensionalidade do espaço de entrada do sistema. Mas vimos que ela pode prejudicar a capacidade de generalização da rede, pois sensores que aparentemente não influem nos dados podem ter participação no reconhecimento. Também foi observado que o pré-processamento dos dados ajuda na velocidade de convergência da rede neural.

Um sistema FIS pode ser uma alternativa para a solução do problema proposto. A sua vantagem é que a modelagem do sistema é acessível. O modelo da rede neural tem propriedades teóricas poderosas [8], mas há problemas graves no ajuste da complexidade do modelo da rede. Isto porque o próprio número de parâmetros livres da rede não é obtido diretamente, dependendo das características de generalização desejadas da aproximação do modelo. A desvantagem do FIS é que ele não consegue separar padrões sobrepostos.

Para melhorar a generalização do FIS, foi utilizada uma etapa de pré-processamento e análise de componentes principais nos dados de treino submetidos ao FIS. Isso fez com que o número de sensores que tinham uma variância maior fossem considerados, e com isso, o número de atributos passou de 6 para 2. O resultado do sistema foi melhorado significativamente com esse tratamento. E, portanto, podemos considerar o sistema FIS como uma ótima alternativa de reconhecimento, sob determinadas condições dos dados de entrada.

Foi observado que também é possível implementar um sistema reconhecedor de modo a aumentar a sua robustez à perda de sensores. E para este fim, pode-se usar as máquinas de comitê estáticas do tipo Ensemble ou um sistema formado de redes neurais treinadas com subconjuntos dos dados. Dentre as abordagens discutidas, a sistema que obteve a melhor taxa de acertos das amostras de teste (generalização) foi a obtida no Terceiro Experimento do item 5.4.5, em que foram usadas sete redes neurais treinadas com conjuntos diferentes dos dados, de acordo com o sensor com falha que era

considerado. Mas mesmo esta abordagem pode ter duas possibilidades: se o sistema reconhecedor tiver um meio de detectar a falha do sensor, ela terá uma taxa de reconhecimento melhor se comparado com um sistema em que o mesmo tem de inferir a falha do sensor por si só.

Neste trabalho foi estimada uma maior facilidade na implementação em hardware da rede neural do que o sistema de inferência fuzzy para o problema de reconhecimento de padrões. Isto porque a própria arquitetura da rede neural MLP é mais simples que a arquitetura necessária para implementar um FIS. A função matemática mais complicada na implementação da rede neural em hardware é a função de ativação, que normalmente é não linear. Já o sistema fuzzy necessita de vários cálculos nas etapas de fuzzificação, inferência, e defuzzificação.

As abordagens de sistemas de aumento de robustez do sistema ainda poderão ser aplicadas recursivamente. Ou seja, dada a determinação do sensor com falha e a escolha do sistema adequado para tratar a perda deste sensor, a metodologia adotada neste trabalho ainda poderá ser usado novamente para o caso em que se detectar a falha de um outro sensor, diferente do primeiro.

A metodologia adotada para aumentar a robustez do sistema ainda poderá submeter os dados a outros sistemas estáticos de máquinas de comitê, a Máquina Reforço ou a Máquina AdaBoost [8], em que o treinamento do sistema prevê a pouca disponibilidade de amostras de treino, e com isso, é previsto que se possa obter resultados melhores com um conjunto reduzido de amostras. Máquinas de Comitê Dinâmicas também poderão ser uma outra alternativa para resolver o problema.

A implementação do hardware da melhor solução encontrada ainda necessitará ser estudada e discutida. Como a implementação do problema de reconhecimento no Kit do DSP da Analog Devices é muito limitada, por esta ter somente uma entrada analógica estéreo, será necessária a discussão da implementação em outro sistema baseado em microprocessador, já desenvolvida no laboratório do LME.

7 Bibliografia:

- [1] PAULSSON, N.; LARSSON, E.; WINQUIST, F.; **Extraction and selection of parameters for evaluation of breath alcohol measurement with an electronic nose**, Sensors and Actuators 84 (2000) 187-197.
- [2] DRAKE, M. A. et. al, **Application of an electronic nose to correlate with descriptive sensory analysis of aged Cheddar cheese**, Lebensm.-Wiss. Technologie 36(1):13-20, 2003; IFT Annual Meeting Technical Program Abstracts. 15C-37 p 28, 2001.
- [3] KELLER, P. E. et al., **Electronic noses and their applications**, IEEE Technical Applications Conference and Workshop: Conference Record, Institute of Electrical and Electronic Engineers – October (1995) 116-119.
- [4] YANG, Y.; YANG, P.; WANG, X.; **Electronic nose based on SAWS array and its odor identification capability**, Sensors and Actuators B 66 (2000), 167-170.
- [5] GARDNER, J., SHIN H., HINES E., **An electronic nose to diagnose illness**, Sensors and Actuators B 70 (2000), 19-24.
- [6] GARDNER, J. et al., **An electronic nose system for monitoring the quality of potable water**, Sensors and Actuators B 69 (2000), 336-341.
- [7] HIRAYAMA, V., RAMIREZ-FERNANDES, F. J., **Embedded System to recognize the heat power of a fuel gas and to classificate the quality of alcohol fuel**, 2003 IEEE International Symposium on Industrial Electronics, Rio de Janeiro Brazil.
- [8] HAYKIN, S. **Redes Neurais – Princípios e Prática**, Segunda Edição, Porto Alegre, Editora Bookman, 1999, 900p.
- [9] ULBIG, R.; HOBURG D.; **Determination of the calorific value of natural gas by different methods**, Thermochimica Acta 382 (2000), 27-35.
- [10] PARDO, M., SBERVEGLIERI, G., **Learning From Data: A Tutorial With Emphasis on Modern Pattern Recognition Methods**, IEEE SENSORS JOURNAL, VOL. 2, NO. 3, JUNE 2002, 203-217.
- [11] KARTALOPOULOS, S.V., **Understanding Neural Networks and Fuzzy Logic**, Wiley-IEEE Press, 1995, 232 p

- [12] FIGARO; **General Information for TGS Sensors**, site da internet www.figarosensors.com.
- [13] VEGA, M.L.B.P., **Sistema Inteligente para Identificar Gases**, Tese de Mestrado, Universidade de São Paulo, Brasil. 1998, 109p.
- [14] RAKOW, N.A.; SUSLICK, K. S.; **A colorimetric sensor array for odour visualization**, Department of Chemistry, University of Illinois – USA. Revista “Nature”, Vol.406, 17 de agosto de 2000, páginas 710 a 713.
- [15] LUNDSTRÖM, I.; **Picture The Smell**; Revista “Nature”, Vol. 406, 17 de agosto de 2000, páginas 682 a 683.
- [16] ORTH, A.; STEMMER, M. R.; **Sistema de visão aplicado ao Monitoramento do desgaste de ferramentas de corte: Uma estratégia para medir o desgaste de flanco (Vb)**, UFSC – Universidade Federal de Santa Catarina – SC – Brasil. 2001.
- [17] CASTILHO, D.; **Analizador de Gases**, Tese de Mestrado – USP – Universidade de São Paulo – SP – Brasil. 1996, 155p.
- [18] KELLER, P.E. et al; **Transmission of Olfactory Information for Telemedicine**, Pacific Northwest Laboratory, USA. IOS Press, Vol. 18, January 1995.
- [19] KELLER, P. E.; KOUZES, R. T., KANGAS, L. J.; **Three Neural Network Based Sensor Systems for Environment Monitoring**, Northwest College and University Association for Science (Washington State University), USA. Miller Freeman, Inc., May 1994, 377-382.
- [20] WU, H.; SIEGEL, M.; **Odor-Based Incontinence Sensor**, Robotics Institute, School of Computer Science, Carnelie Mellon University, Pittsburgh. Proceedings of the 17th IEEE Instrumentation and Measurement Technology Conference, Vol. 1, May 2000, pp.63-68.
- [21] LU, Y.; BIAN L.; YANG, P.; **Quantitative Artificial Neural Network For Electronic Noses**, Department of Chemistry, Fudan University, Shanghai, China. Analytica Chimica Acta. 417:101-110, 2000
- [22] FLEXER, A.; **Statistical Evaluation of Neural Network Experiments: Minimum Requirements and Current Practice**, Austrian Research Institute for Artificial Intelligence; Vienna – Austria. 1995

- [23] LAZZERINI, B.; MAGGIORE, A; MARCELLONI, F.; **FROS: a fuzzy logic-based recognizer of olfactory signals**, University of Pisa, Pisa, Italy. Pattern Recognition 34(11): 2215-2226 (2001)
- [24] JUNIOR, E. F. C., e Grupo Comunicações Homem - Máquina, **Redes Neurais Artificiais, um curso teórico e prático para engenheiros e cientistas**, Escola Politécnica da USP (Universidade de São Paulo), Departamento de Engenharia Eletrônica e Laboratório de Comunicações de Sinais (PCS). EDUSP, 1999.
- [25] STORK, D. G.; DUDA, R. O.; HART, P. E., **Pattern Recognition**, Wiley-Interscience; 2 edition (October 2000) , 654p.
- [26] MATHWORKS, The Inc. , **Documentação do Software MATLAB Versão 6.0**.
- [27] ROSEN, B.E., **Ensemble learning using decorrelated neural networks**, Connection Science, 8, 3-4, pp. 373-384, 1996.
- [28] PARDO, M. et al., **Decompositive classification models for electronic noses**, Anal. Chimica Acta, 2001. pp. 223-232.
- [29] ZADEH, L. A., **Fuzzy Sets**, Information and Control 8, pg. 338-353 (1965).
- [30] LEE, C. C., **Fuzzy Logic in Control Systems: Fuzzy Logic Controller – Part I**, IEEE Transactions on Systems, man, and cybernetics, vol.10, number 2, 1990, 404-418.
- [31] LEE, C. C., **Fuzzy Logic in Control Systems: Fuzzy Logic Controller – Part II**, IEEE Transactions on Systems, man, and cybernetics, vol.20, number 2, 1990, 419-435.
- [32] BARATTO, G.; **Classificador de Aromas Com Redes Neurais Artificiais para um Nariz Eletrônico**, Tese de Doutorado – USP – Universidade de São Paulo – SP – Brasil. 1997.
- [33] NEURALWARE INC., **Using NeuralWorks - NeuralWorks Professional II/PLUS and NeuralWorks Explorer**, NeuralWare, Inc., 1995;
- [34] DUBOIS, D.; PRADE H.; YAGER R., **Fuzzy Information Engineering: A Guided Tour of Applications**; Capítulo 9: Extracting Fuzzy Rules from data for function approximation and pattern recognition (CHIU, S. L.). Wiley; (November 1996), 149-162.
- [35] NEURALWARE INC., **Pre-processing Data with Data Sculptor**, NeuralWare, Inc., 1995;

[36] Mathworks, The Inc. , **Software MATLAB.**