UNIVERSIDADE FEDERAL DA GRANDE DOURADOS – UFGD

FACULDADE DE ENGENHARIA - FAEN

CURSO DE ENGENHARIA DE ENERGIA

CÉSAR AUGUSTO GOMES DE SOUZA

IDENTIFICAÇÃO E CLASSIFICAÇÃO DE CARGAS ATRAVÉS DO CONTEÚDO HARMÔNICO VIA REDES NEURAIS ARTIFICIAIS FAZENDO USO DE UM SISTEMA DE AQUISIÇÃO DE DADOS BASEADO EM ARDUINO

DOURADOS-MS

2016

UNIVERSIDADE FEDERAL DA GRANDE DOURADOS – UFGD
FACULDADE DE ENGENHARIA - FAEN
CURSO DE ENGENHARIA DE ENERGIA

CÉSAR AUGUSTO GOMES DE SOUZA

IDENTIFICAÇÃO E CLASSIFICAÇÃO DE CARGAS ATRAVÉS DO CONTEÚDO HARMÔNICO VIA REDES NEURAIS ARTIFICIAIS FAZENDO USO DE UM SISTEMA DE AQUISIÇÃO DE DADOS BASEADO EM ARDUINO

Trabalho de Conclusão de Curso de Graduação apresentado a banca examinadora da Faculdade de Engenharia da Universidade Federal da Grande Dourados para obtenção do Título de Bacharel em Engenharia de Energia, sob orientação do Prof. Dr. Etienne Biasotto

DOURADOS-MS
2016

Dados Internacionais de Catalogação na Publicação (CIP).

S729i Souza, Cesar Augusto Gomes De

Identificação e classificação de cargas através do conteúdo harmônico via redes neurais artificiais fazendo uso de um sistema de aquisição de dados baseado em Arduino. / Cesar Augusto Gomes De Souza -- Dourados: UFGD, 2016.

85f. : il. ; 30 cm.

Orientador: Etienne Biasotto

TCC (Graduação em Engenharia de Energia) - Faculdade de Engenharia, Universidade Federal da Grande Dourados.

Inclui bibliografia

1. Desagregação de cargas. 2. Harmônicos. 3. Redes neurais artificiais. I. Título.

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

CÉSAR AUGUSTO GOMES DE SOUZA

IDENTIFICAÇÃO E CLASSIFICAÇÃO DE CARGAS ATRAVÉS DO CONTEÚDO HARMÔNICO VIA REDES NEURAIS ARTIFICIAIS FAZENDO USO DE UM SISTEMA DE AQUISIÇÃO DE DADOS BASEADO EM ARDUINO

Trabalho de Conclusão de Curso de Graduação apresentado a banca examinadora da Faculdade de Engenharia da Universidade Federal da Grande Dourados para obtenção do Título de Bacharel em Engenharia de Energia, sob orientação do Prof. Dr. Etienne Biasotto

Aprovado em 07 de outubro de 2016

BANCA EXAMINADORA

_____

Prof. Dr. Etienne Biasotto – UFGD
Orientador

_____

Prof. Dr. Gerson Bessa Gibelli – UFGD
Examinador

_____

Prof. Dr. Eduardo Mirko Valenzuela Turdera – UFGD
Examinador

DOURADOS-MS

2016

AGRADECIMENTOS

Primeiramente à minha mãe, Idalina Gomes de Souza, por ser meu suporte e apoio ao longo destes longos anos de graduação, pelo amor, carinho, compreensão e gestos de apoio e incentivo durante toda minha educação. Por tudo que fez e faz, e sem a qual eu não estaria onde estou. À minha avó, Zulmira Paula de Souza por sempre estar ali para me socorrer em momentos de desespero e saber o que falar quando nem eu mesmo sabia o que queria ouvir. Sem elas eu não teria chego tão longe.

Aos meus grandes amigos Bárbara Lanzarini, Caio Camioli, Jéssica Olivi, Leonardo Freire, Milton Vasconcelos e Rita Andrade que me acompanharam durante a graduação, nos bons e maus momentos, sem os quais estes anos teriam sido muito mais difíceis. Obrigado por tornarem os dias mais leves. Um agradecimento especial à Karina Teixeira, Lucas Mora e Matheus Gomes, que mesmo não estando presentes até o fim comigo nunca me deixaram desistir, não só da graduação. Obrigado por me lembrarem que não estou sozinho.

Ao professor Dr. Etienne Biasotto pela orientação, por toda dedicação, atenção e paciência demonstrada durante a realização deste trabalho, e principalmente por todo conhecimento compartilhado comigo.

Ao professor Dr. Gerson Bessa Gibelli pelos conselhos e ensinamentos durante o desenvolvimento do trabalho

E a Faculdade de Engenharia da Universidade Federal da Grande Dourados, seu corpo docente, bem como outros que de alguma forma contribuíram para a minha formação.

RESUMO

O presente trabalho apresenta o desenvolvimento de um sistema de aquisição de dados baseado em Arduino para amostragem do sinal elétrico. O sistema é composto por um transformador de corrente (TC), um transformador de potencial (TP), circuitos de condicionamento do sinal e o Arduino como conversor analógico-digital e meio de comunicação com o computador. Os dados foram obtidos de três tipos de pares de lâmpadas diferentes: incandescentes, fluorescentes compactas e LEDs. Um *script* criado no *software* Matlab se encarregou de armazená-los. Os dados da corrente foram submetidos a análise de Fourier, através do algoritmo da *Fast Fourier Transform* (FFT), de forma que as harmônicas presentes no sinal fossem extraídas. Foi observado que as cinco primeiras harmônicas ímpares eram muito distintas entre as cargas. Partindo desse conjunto de dados a eficiência do uso de harmônicas como forma de classificação de cargas foi investigada através da ferramenta *Neural Network Pattern Recognition* do *software* Matlab, uma rede neural de classificação, onde foi constatado que de fato as harmônicas são uma fonte de dados que podem ser empregados para se classificar cargas.

Palavras-chave: desagregação de cargas, harmônicos, redes neurais artificiais.

ABSTRACT

This paper presents the development of a data acquisition system based on Arduino for signal sampling. The system consists of a current transformer (CT), a voltage transformer (VT), signal conditioning circuits, and the Arduino as the analog-to-digital converter and means of communication with the computer. Data were obtained from three different pairs of lamps: incandescent, compact fluorescent and LEDs. A script created in Matlab was in charge of storing the data. The current data were input in a fast Fourier transform (FFT) algorithm in order to extract the harmonics content. It was observed that the first five odd harmonics were quite distinct among the loads. From this set of data, the use of harmonics as a way for load classification was investigated by employing the *Neural Network Pattern Recognition* tool on Matlab, a classification neural network, where it was found that indeed the harmonics are a source of reliable data that can be used to classify loads.

Keywords: load disaggregation, harmonics, artificial neural networks.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ACEE	American Council for an Energy-Efficient Economy
ADC	Analog to Digital Converter
CC	Corrente Contínua
DFT	Discrete Fourier Transform
FFT	Fast Fourier Transform
HMM	Hidden Markov Model
IDE	Integrated Development Environment
IEC	International Electrotechnical Commission
LSB	Least Significant Bit
NIALM	Non-Intrusive Appliance Load Monitoring
NILM	Non-Intrusive Load Monitoring
rms	Root Mean Square
RNA	Rede Neural Artificial
ROC	Receiver Operating Caracteristic
SCG	Scaled Conjugate Gradient
TC	Transformador de Corrente
TF	Transformada de Fourier
TRS	Tip-Ring-Sleeve
TT	Transformador de Tensão

SUMÁRIO

1 INTRODUÇÃO

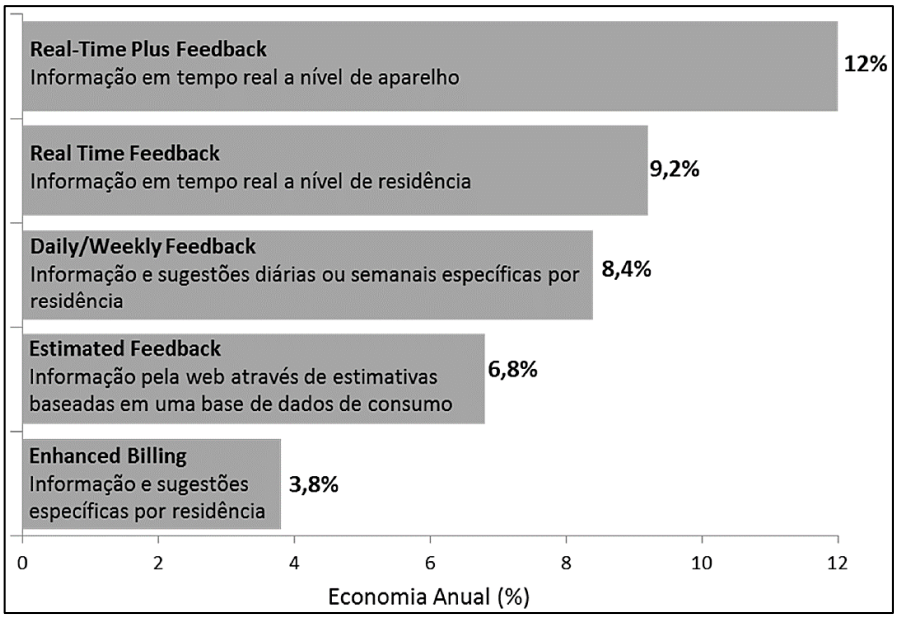
O *American Council for an Energy-Efficient Economy* – ACEEE afirma que o consumo de energia elétrica no mundo aumenta cerca de 1 % ao ano, fato preocupante visto que cerca de 80 a 90 % desta energia é obtida via queima de combustíveis fósseis, que são não renováveis e ainda causam sérios problemas ambientais, como o efeito estufa (YORK et al, 2013).

Com o intuito de reverter esse cenário, concessionárias de energia e agências do governo ao redor do mundo vêm implementando, mais expressivamente na última década, diversos programas de eficiência energética. Programas estes, que objetivam desenvolver soluções que tragam aos consumidores o poder de gerenciar de maneira mais eficiente o uso da energia e, por conseguinte, economizar em suas contas também (DUARTE et al, 2012).

A implementação de programas deste tipo, custa em média, três vezes menos do que a geração da mesma quantidade de energia economizada com esses programas (YORK et al, 2013). Fora os benefícios ambientais que os acompanham, como a redução de gases do efeito estufa, gerando benefícios para as indústrias geradoras, consumidores e meio ambiente.

No período de 1995 a 2010, a ACEEE realizou uma pesquisa em vários países e constatou que a energia economizada através da mudança de comportamento dos consumidores é proporcional à qualidade de informação que os mesmos recebem acerca do consumo. Dar ao consumidor uma maior compreensão de onde (em quais aparelhos) e quando (em quais períodos do dia) vem consumindo energia leva tanto a redução do consumo quanto a mudança dos períodos de ponta para os fora de ponta (EHRHARDT-MARTINEZ et al, 2010). Os resultados da pesquisa são mostrados na Figura 1, onde é possível perceber que a disponibilização de informação em tempo real com discriminação de consumo por aparelho (*Real-Time Plus Feedback*) é a que resulta em uma maior economia média. Análises mais recentes, entretanto, mostram que economias da ordem de até 19,5 %, com média de 3,8 % podem ser atingidas (YORK et al, 2013).

Figura 1. Economia média anual de energia elétrica em unidades residenciais, baseado em 36 estudos realizados entre 1995 e 2010.



Fonte: (EHRHARDT-MARTINEZ, et al, 2010).

Entregar esse tipo de informação aos usuários leva ao conceito de *smart metering* ou medição inteligente, que compreende um conjunto de tecnologias que possibilita ao consumidor acompanhar o próprio uso de energia em detalhes, dando-lhe conhecimento de como a vem utilizando, onde se encontram ineficiências e possibilidades de melhoria no uso (UENO et al., 2005). Embora já existam no mercado equipamentos medidores do tipo, estes ainda não abrangem toda a gama de funcionalidades desejadas sendo, portanto, fonte de pesquisa e desenvolvimento na área tanto por parte de empresas, quanto universidades e até mesmo desenvolvedores independentes.

Algumas soluções de *smart metering* já existentes para unidades residenciais são o myEragy Energy Monitoring, Opower Home Energy Report, Open Energy Monitor, People Power, Smart Energy Groups, ThingSpeak, entre outros. Embora todos estes forneçam ao usuário informações sobre o consumo e acerca de como economizar energia, nenhum oferece detalhamento do uso da energia por eletrodoméstico instalado na residência.

Empresas que criaram soluções do tipo, mas capazes de discriminar o consumo por aparelho através da medição em um único ponto também apareceram nos últimos anos, como as americanas: Bidgely (antiga MyEnerSave), LoadIQ Enable.EI, PlotWatt, Verdigris e Verlitics (antiga Emme); as inglesas: Navetas e Onzo; as francesas: Fludia e Wattseeker; as Irlandesas: Powersavvy e Wattics (antiga Veutility) e a alemã Yetu (BACURAU, 2014).

Estas soluções comerciais de sistemas de desagregação, em sua grande maioria, fazem uso apenas da potência ativa para discriminação de cargas. Quando esse tipo de dado é empregado para desagregação, o número de dispositivos que podem ser detectados é relativamente baixo, da ordem de 10 equipamentos, limitando-se aos de grande potência. Além destes sistemas terem sua resolução limitada a intervalos da ordem de um segundo a um minuto entre medições. O uso das harmônicas por outro lado permite que um número maior de dispositivos seja identificado, cerca de até 40, mas requer uma taxa de amostragem muito maior (da ordem de kHz), para o cálculo da Transformada de Fourier (TF) e um processador de sinal operando em tempo real (ARMEL et al., 2013).

Recentemente, entretanto, uma *startup* chamada Sense de Cambridge, Massachusetts desenvolveu um sistema que, diferentemente dos acima mencionados, é capaz de realizar a leitura do sinal 4 milhões de vezes por segundo, sendo assim capaz de identificar até mesmo equipamentos de baixa potência operando em tempo real, desagregando cerca de 80 % do consumo da residência. O módulo também é instalado no medidor, de maneira não invasiva. O diferencial de seu algoritmo é que ele não precisa ser treinado e no site está expresso que o sistema melhora com o tempo. Entretanto, mesmo o Sense apresenta algumas limitações, como funcionar apenas em redes de 120 V nos EUA, não ser compatível com todas as classes de medidores e não ser capaz de identificar todos os equipamentos em funcionamento na residência, embora seja mais capaz que os supracitados. (SENSE, 2016).

Enquanto soluções comerciais de sistemas de desagregação de cargas surgiram apenas nos últimos cinco ou seis anos, a ideia não é nova. Desde a década de 80 já existem pesquisas na área e cientistas vêm estudando sistemas, metodologias e algoritmos que possam medir e estimar o consumo de energia por eletrodoméstico. Os sistemas de discriminação do consumo por aparelho são divididos em dois grupos: intrusivos e não intrusivos (DUARTE et al, 2012).

O primeiro grupo, diz respeito a sistemas que medem o consumo de energia em cada carga individualmente, ou seja, cada carga monitorada possui um medidor e estes formam uma rede. Um aplicativo computacional recebe e processa estes dados para exibi-los aos usuários. Por conta da necessidade de vários medidores estes sistemas são relativamente caros e difíceis de instalar.

Devido as desvantagens apresentadas pelos sistemas intrusivos, o segundo grupo é o que se mostra mais promissor e é o mais visado em pesquisas.

Os sistemas de monitoramento não intrusivos (do inglês NILM, *Non-Intrusive Load Monitoring*, ou NIALM, *Non-Intrusive Appliance Load Monitoring*) se caracterizam por necessitarem de apenas um medidor que é instalado na entrada de energia da residência,

comumente na caixa de distribuição elétrica. Estes sistemas, em geral, são mais baratos e de instalação simples, entretanto ainda não são capazes de identificar todas as cargas existentes em uma residência moderna.

O princípio por trás do funcionamento destes sistemas é o conceito de assinatura de cargas, que é o conjunto de características elétricas únicas que cada aparelho possui, que podem ser medidas e usadas para desagregação. Mais detalhes sobre assinaturas de carga são apresentados no Capítulo 2.

Um sistema NILM compõe-se de dois módulos básicos: o medidor e o algoritmo de desagregação de cargas. O primeiro é o *hardware* responsável pela aquisição dos parâmetros elétricos do circuito que está sendo monitorado, usualmente é um conjunto de sensores conectado à um processador de sinal. Os parâmetros monitorados, são características do sinal que fazem parte de sua assinatura e podem ser: tensão e corrente, potência ativa, reativa e aparente, fator de potência, harmônicas do sinal de corrente, entre outras, também descritas no Capítulo 2. Serão estas as grandezas usadas como entrada do algoritmo de desagregação, responsável por identificar cada aparelho em uso em cada instante (BACURAU, 2014).

Como é possível perceber a efetividade de um sistema NILM depende da acurácia com que os dados de entrada do algoritmo são obtidos, os próprios dados empregados em si, além, é claro, de algoritmos eficientes e que produzam os resultados esperados.

1.1 Objetivos

O presente trabalho tem como objetivo desenvolver um sistema de aquisição e processamento de sinal simplificado, baseado em Arduino, para que se possa analisar a capacidade de uso das harmônicas da corrente para identificação de cargas via ferramenta *Neural Network Pattern Recognition*, uma rede neural de classificação, existente na ferramenta computacional *software* Matlab.

2 REVISÃO BIBLIOGRÁFICA

Este capítulo tem por objetivo apresentar os principais conceitos existentes na literatura por trás do tema desenvolvido.

2.1 Desagregação de cargas

Cada equipamento ou eletrodoméstico em uma residência possui um padrão de uso da eletricidade. A identificação destes padrões leva a desagregação de todo o consumo de energia, que ao ser apresentado ao usuário oferece um melhor entendimento de como a energia vem sendo utilizada. Também é possível ao consumidor identificar possibilidades para reduzir o consumo e o valor de sua conta, visto o crescente aumento das tarifas energéticas a cada dia que passa (ARDELEANU; DONCIU, 2012).

Pochacker, Egarter e Elmenreich (2016) NILM como uma técnica empregada para a compreensão da forma com que operam equipamentos em relação ao consumo energético, a partir de medições realizadas em apenas um ponto, como o quadro geral. Tal tecnologia é parte integrante do sistema de medição em uma casa inteligente (*smart home*), onde informações acerca de cada equipamento são de extremo interesse, mas monitorar cada equipamento individualmente é inviável. Assim dispositivos de baixo custo para monitoramento do consumo de energia residencial são um passo na integração dos lares na futura rede *smart grid*. Eles funcionam com base nas informações dos equipamentos envolvidos e nos cenários de consumo existentes, os algoritmos de desagregação então identificam os atributos dentre os dados e emitem conclusões acerca do uso da energia.

Diversas características podem ser empregadas para se identificar a assinatura das cargas.

A corrente de partida, que é diferente entre equipamentos, mesmo que seus estados de funcionamento sejam similares. A forma de onda da corrente, visto que é influenciada por parâmetros como amplitude, duração e constantes de tempo, portanto os algoritmos podem ser capazes de lidar e se adaptar a essas variações. O perfil do transiente, que tende geralmente a se manter, fazendo com que muitas cargas possuam perfis repetitivos ou pelo menos seções que se repetem, tornando a implementação mais simples, visto que não é preciso analisar o perfil como um todo, o que é muito mais complexo e exige mais recursos (CHANG, LIN e LEE, 2010).

As harmônicas da corrente também podem ser empregadas na determinação da assinatura dos equipamentos, como mostrado por Srinivasan, Ng e Liew (2006). Ao se analisar a amplitude e os valores das harmônicas, correspondências podem ser observadas entre o tipo de equipamento e estas. É possível dessa forma identificar equipamentos que consomem a mesma potência, mas cujas harmônicas presentes no sinal são completamente diferentes.

A forma de onda da corrente, da admitância instantânea, da potência instantânea, autovalores, todos são dados que também podem ser utilizados como entrada em algoritmos de desagregação de carga.

Dadas as particularidades de cada consumidor o método a ser implementado varia e muitas vezes a combinação de vários algoritmos pode ser utilizada.

Novamente, segundo Pochacker, Egarter e Elmenreich (2016) existem duas formas de se trabalhar a questão da desagregação, com medidas supervisionadas e não supervisionadas. Nas do primeiro tipo, dados identificados formam um conjunto de treinamento para um classificador ou algoritmos otimizados e/ou de reconhecimento de padrão. Nos otimizados o perfil de consumo e potência dos equipamentos é fornecido. Ao se comparar o perfil de consumo com os que compõem o banco de dados, aquele que mais se aproximar é escolhido. Já em sistemas de reconhecimento de padrão os dados podem ser classificados em forma de *clusters*, redes neurais, e algoritmos baseados em *Support Vector Machines* (SVM). A principal desvantagem desse tipo de solução é a necessidade prévia de informação. Este é o caso do presente trabalho.

Com relação aos algoritmos não supervisionados, que compõem o grupo de maior interesse para pesquisa atualmente, a necessidade de dados para treinamento, ou seja, de informações prévias do sistema, deixa de existir. Neste grupo se encontram algoritmos de *dynamic time warping*, clustering com separação de fonte aleatória, *Hidden Markov Models* (HMMs), fatorial de HMMs, além de outras variações, *temporal motif mining* e separação aleatória de fonte. Para todos estes, a distinção entre os equipamentos é não supervisionada, enquanto que a identificação de um equipamento não é feita automaticamente. Embora haja soluções que o façam baseadas em inferência Bayesiana e classificação semisupervisionada.

2.2 Características das assinaturas de cargas

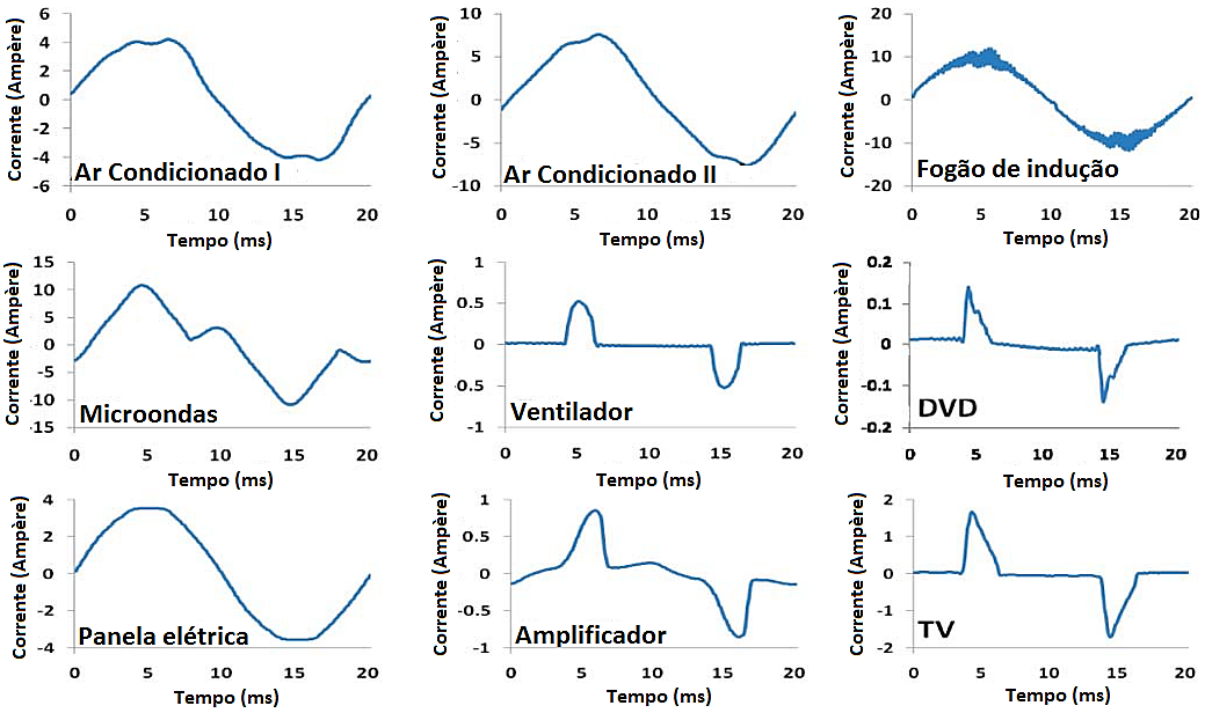
Em Liang et al (2010), a assinatura de uma carga é definida como o comportamento individual de um equipamento quando em operação, sendo que todo equipamento elétrico possui características únicas em seu padrão de consumo. Obviamente que tal comportamento é

limitado ao que pode ser monitorado no ponto de interesse, ou seja, no medidor, sendo usualmente tensão, corrente e potência.

Nesse ponto de fluxo de energia, o comportamento do que pode ser medido é coletivo e se refere a mais de um equipamento em funcionamento, constituindo uma carga composta. No geral esta é muito mais complexa em sua natureza do que a assinatura individual de suas constituintes, que se encontram misturadas e algumas vezes acabam mascaradas ou perdidas. Entretanto, quando se trata de um monitoramento não intrusivo das cargas, ainda mais sendo o ponto de observação o medidor de energia, este é o sinal de mais fácil acesso do lado dos equipamentos.

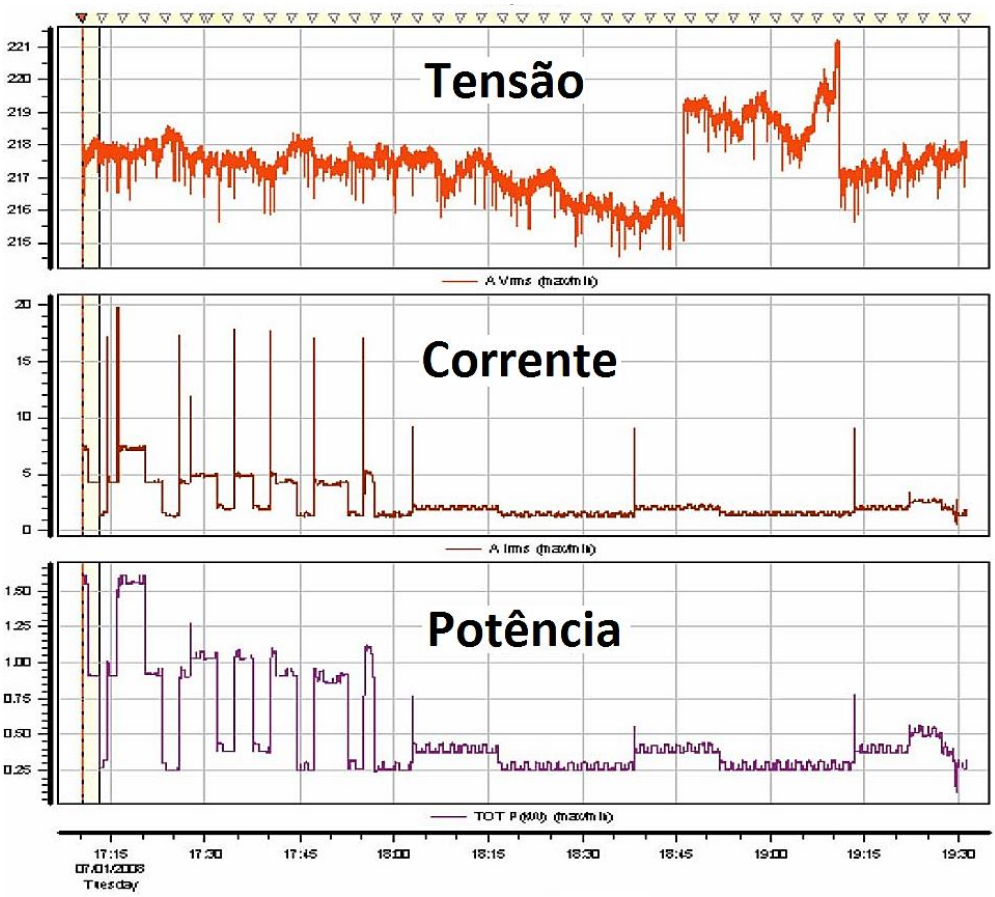
Quando aquisitando sinal para detectar a assinatura de cargas, seja individual ou composta, pode-se o fazer com um intervalo de amostragem menor que o período de um ciclo (nível micro, Figura 2) ou maior (nível macro, Figura 3) e dependendo de qual seja escolhido, as características observáveis variam. A nível micro é possível se ter detalhes muito distintos das formas de onda e da amplitude dos sinais. Enquanto em nível macro outros padrões, como o ciclo de operação dos equipamentos, são mais notáveis.

Figura 2. Exemplos de assinatura de carga em nível micro.



Fonte: (LIANG et al, 2010).

Figura 3. Exemplos de assinatura de carga em nível macro.



Fonte: (LIANG et al, 2010).

O sinal pode ser observado em sua forma instantânea, obtida em intervalos fixos de tempo, sendo que geralmente este corresponde a assinatura composta das cargas. Ou pode-se analisar a diferença entre duas formas instantâneas consecutivas. Caso o intervalo seja pequeno o suficiente é possível, por exemplo, inferir sobre um equipamento que foi ligado ou desligado entre cada observação. Sendo assim, neste segundo caso têm-se um comportamento mais próximo da assinatura de carga individual ao invés de uma observação do sinal composto.

Dentre as diversas características das cargas que podem ser empregadas para caracterizar sua assinatura, Liang et al (2010) destaca:

- Forma de onda da corrente

A forma de onda da corrente no domínio do tempo oferece o conjunto mais completo de informações para descrever o comportamento da carga. Sua grande vantagem vem da alta resolução do sinal que pode refletir de maneira detalhada as características, sendo notável a particularidade no formato de onda para os diferentes equipamentos. Por exemplo, um elemento resistivo tem uma onda de formato senoidal, equipamentos motorizados apresentam um

formato de onda senoidal torcida; eletrônicos, forma não senoidal e equipamentos indutivos apresentam uma alta porcentagem de harmônicas.

- Potência ativa e reativa

Potência ativa e reativa são frequentemente as medidas mais empregadas para se descrever o comportamento de uma carga. Entretanto, para valores de consumo pequenos, tais características tendem a se agrupar próximo a origem, tornando muito difícil empregar apenas estes para desagregação do consumo e identificação do aparelho ligado

- Harmônicas

Ao se aplicar a *Fast Fourier Transform* (FFT) sobre o sinal da corrente é possível obter as harmônicas geradas pela carga. Assim como no caso da forma de onda da corrente, harmônicas de diferentes ordens surgem dependendo da carga que está sendo alimentada. É sabido, por exemplo, que uma televisão e um ar condicionado apresentam harmônicas de baixa ordem, enquanto algo como um fogão à indução, de ordem elevada.

- Forma de onda da admitância instantânea

Definida como o quociente entre as formas de onda da corrente e tensão instantâneas, seu emprego em sistemas de desagregação se justifica pela conexão em paralelo encontrada na maioria dos equipamentos em uma casa.

- Forma de onda da potência instantânea

Produto das formas de onda da tensão e corrente instantâneas, assim como seus dados de origem, a forma de onda da potência instantânea apresenta formas diferentes dependendo da carga que se observa. Uma televisão possui uma forma de onda não senoidal, uma carga resistiva e outra motorizada são bem diferentes entre si e as cargas indutivas apresentam grande quantidade de harmônicas.

- Autovalores

A forma de onda da corrente de cargas dinâmicas, como a de um ar condicionado, pode variar de um ciclo para outro. Uma forma de se observar esse comportamento é rearranjar a série do tempo, da forma de onda da corrente, em formato matricial e aplicar uma análise de autovalores. A decomposição em valores é empregada para extrair a componente principal da forma de onda da corrente. Após a decomposição ciclo a ciclo da matriz da corrente em outras três, apenas a matriz diagonal é empregada na análise da assinatura da carga. É possível

observar que equipamentos com alto consumo energético apresentam o primeiro autovalor elevado. Outras, como o ar condicionado ou cargas indutivas, apresentam um efeito envelope ou alto conteúdo harmônico, apresentando boa correlação com o segundo e terceiro autovalor. Para demais equipamentos, como uma televisão ou cargas resistivas, autovalores além do primeiro são muito pequenos.

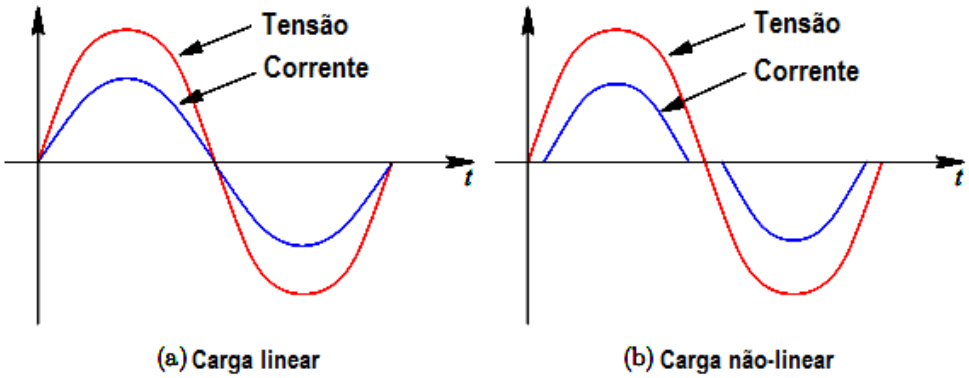
- Forma de onda do transiente

Todas as características acima mencionadas ocorrem em estado estacionário, mas fenômenos de transiente também oferecem boas informações acerca das cargas. Para se obter tal informação é possível calcular a potência instantânea para cada meio ciclo e empregar a forma de onda resultante como do transiente. E assim como as formas de onda já mencionadas aqui, o comportamento transiente também varia conforme a carga, nas resistivas cresce do zero ao estado estacionário gradualmente. Já equipamentos movidos a motor apresentam um pico crescente e depois caem até atingir o estado estacionário.

2.3 Harmônicas

As harmônicas são definidas como tensões ou correntes senoidais que possuem frequências múltiplas inteiras da fundamental, que para o sistema elétrico brasileiro é de 60Hz, logo as frequências de 180Hz, 300Hz, etc., correspondem as harmônicas de 3ª, 5ª, etc. ordem, respectivamente. Estas ocorrem continuamente no sistema e estão relacionadas à operação contínua de cargas que apresentam comportamento não linear, como mostrado na Figura 4 (FERNANDES, 2008). Este se deve a não conformidade da forma de onda da corrente quando relacionada com a forma de onda da tensão de alimentação da carga.

Figura 4. Diferença entre as formas de onda da tensão e corrente para uma carga linear e uma não-linear.



Fonte: (INGALE, 2014).

Uma carga linear apresenta a forma de onda da corrente senoidal proporcional a da tensão, também senoidal, como visto na Figura 4(a). Isso se deve ao fato de que cargas lineares não dependem da tensão para determinarem sua impedância em uma dada frequência e sempre obedecem a Lei de Ohm. Tais cargas não afetam a rede de transmissão a qual estão conectadas ou equipamentos de outros consumidores. Ao contrário, uma carga não-linear (Figura 4(b)) não consome energia de modo contínuo. Quando uma tensão senoidal é aplicada à uma carga desse tipo, a corrente não é senoidal e nem proporcional. Essa corrente não senoidal é causada pela impedância do equipamento variando ao longo de um ciclo da tensão. Estas cargas tem o potencial de distorcer a forma de onda da fonte de alimentação e podem também causar problemas para outras cargas. Como se pode observar na figura supracitada a corrente consumida pela carga é aproximadamente zero até que uma tensão de partida seja atingida. A partir daí a corrente irá aumentar até atingir o pico da senoide e então decai, o dispositivo desliga e a corrente vai a zero. Um segundo pulso negativo da corrente se inicia na segunda metade da senoide novamente. A corrente consumida é, portanto, uma série de pulsos positivos e negativos e não uma onda senoidal, como nas cargas lineares.

Quando presentes no sinal, as harmônicas se encontram sobrepostas a forma de onda da frequência fundamental, sendo as de 3ª à 25ª ordem as mais comuns no sistema de distribuição de energia. Além destas é possível que haja no sinal componentes que não são múltiplas inteiras da fundamental, chamadas de interharmônicas, aparecendo sempre que se esta lidando com sinais não-periódicos.

Segundo Ingale (2014) existem dois tipos de harmônicas, as pares e as ímpares. As primeiras são normalmente pequenas, geradas, por exemplo, por grandes conversores, transformadores sendo energizados, etc, ocorrendo apenas na presença de componente CC. Entretanto, por lei hoje nenhum equipamento deve gerar harmônicas pares, e de fato, medições realizadas nos pontos de alimentação mostram que as harmônicas do tipo par são muito pequenas. O efeito destas na forma de onda é a distorção que provocam entre os meio ciclos positivo e negativo fazendo com que deixem de ser simétricos. Se uma quantidade significativa se encontra presente no sinal, este não é mais simétrico com relação ao eixo zero.

O segundo tipo, as harmônicas ímpares, são normalmente as predominantes no sistema. Seu efeito é o de aumentar ou diminuir a amplitude do sinal em 10 %. O valor em rms (*root mean square*) aumenta pouco mais de 0,5 %, assim o fator de crista aumenta ou diminui por volta de 10 %. No geral é a harmônica de 3ª ordem que provoca a mudança no fator de crista. O efeito desta distorção é o mesmo para as metades positivas e negativas da senoide, desde que apenas harmônicas ímpares estejam presentes.

A Tabela 1 apresenta a distorção harmônicas total (DHT) da corrente causada por cargas residenciais e comerciais típicas

Tabela 1. Distorção harmônica total da corrente drenada por cargas residenciais e comerciais típicas.

Equipamento	**DHTi (%)**
Torradeira	2
Cafeteira	2
Lâmpadas fluorescentes (tipo reator)	36
Telefone sem fio	40
Forno de micro-ondas	46
Carregador de bateria	83
Aparelho de som com CD	104
Fonte de computador	111
Aparelho de ar condicionado	123

Fonte: (NDIAYE, 2006).

É possível observar que equipamentos que utilizam resistores para aquecimento, como as torradeiras e cafeteiras são puramente resistivos e apresentam baixa distorção de corrente. Por outro lado, correntes altamente distorcidas são verificadas em equipamentos que utilizam fontes chaveadas (computadores, televisores, etc).

Hart (1992) observou que uma carga que não é puramente resistiva é propensa a produzir correntes harmônicas, de forma que cada equipamento possui a sua própria assinatura de harmônicas. Essa assinatura, como descrito anteriormente, pode ser empregada para desagregação de cargas, sem haver dependência de alterações no padrão de consumo.

2.4 Redes neurais artificiais (RNAs)

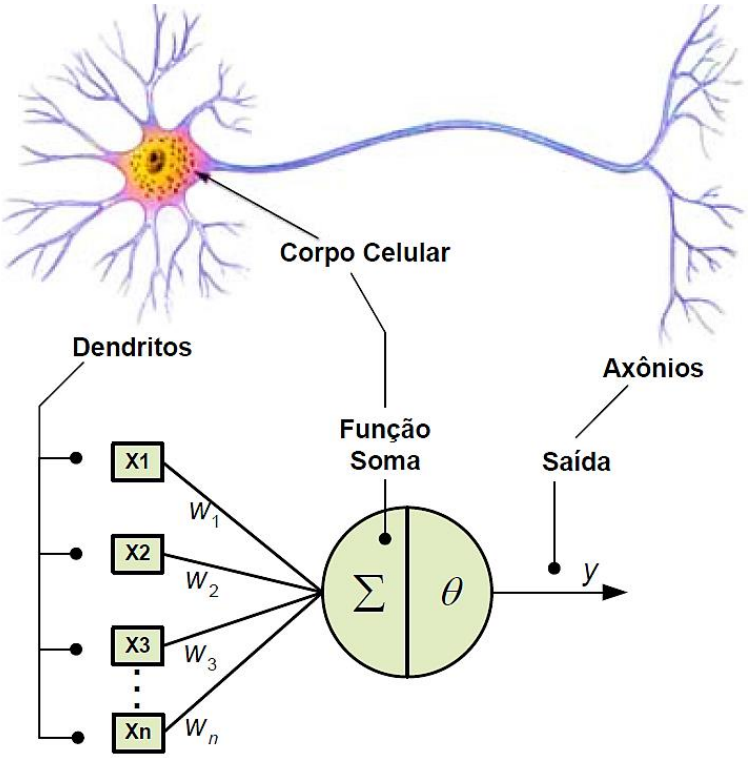
O cérebro humano tem uma capacidade de processamento de informação distinta de um computador digital convencional, extremamente complexo, não linear e paralelo, sendo assim capaz de organizar sua estrutura neuronal de forma a realizar certos processamentos, como reconhecer padrões, de maneira muita mais rápida. Sua grande habilidade reside em desenvolver suas próprias regras (OLIVEIRA, 2005).

Foi da forma com que o cérebro humano lida com dados que as RNAs foram desenvolvidas. Conceitualmente, Haykin (2001) define uma rede neural artificial como um

processador maciço paralelamente distribuído, constituído de unidades de processamento simples, que têm propensão natural para armazenar conhecimento experimental e torná-lo disponível para uso. Dentre as principais características que o autor destaca estão a possibilidade de considerar o comportamento não linear dos fenômenos físicos; a necessidade de pouco conhecimento estatístico sobre o ambiente onde a rede está inserida; a capacidade de aprendizagem, obtida através de exemplos entrada/saída que sejam representativos do ambiente; habilidade de aproximar qualquer mapeamento entrada/saída de natureza contínua; sua adaptabilidade, generalização e tolerância a falhas; informação contextual e capacidade de armazenar o conhecimento adquirido, através das forças de conexão entre neurônios, os chamados pesos sinápticos.

As RNAs vem sendo estudadas desde o início da segunda metade do século XX, mais precisamente desde 1943, quando Warren McCulloch, neuroanatomista e Walter Pitts, matemático, propuseram um modelo matemático que representasse o neurônio biológico (Figura 5)

Figura 5. Representação do neurônio artificial modelado por McCulloch e Pitts comparado a um neurônio biológico.



Fonte: (FERNANDES, 2008).

A operação do neurônio artificial pode ser descrita por meio dos seguintes passos:

1. Um conjunto de valores (sinais) é apresentado na entrada dos neurônios (*Xn*);
2. Cada entrada do neurônio é multiplicada pelo seu respectivo peso sináptico (*Wn*);
3. A soma ponderada dos sinais de entrada é calculada para obter o potencial de ativação;
4. A saída do neurônio é limitada por meio da aplicação de uma função de ativação (θ);
5. Por fim, a saída do neurônio é compilada a partir da função de ativação aplicada em relação ao seu potencial de ativação (*y*).

Matematicamente,

$$u = \sum_{i=1}^{n} W_i . X_i - \theta \quad (1)$$

$$y = g(u) \quad (2)$$

onde n é o número de entradas, W_i é o ganho (peso) associado à entrada i, θ é o limiar de ativação, X_i é a entrada e $g(.)$ é a função de ativação do neurônio artificial.

O modelo artificial apresentado por eles foi descrito com suas capacidades computacionais (acima detalhadas), porém, sem nenhuma técnica de aprendizado. Estas foram aparecer apenas em 1949, fruto do trabalho de Donald Hebb que demonstrou que a variação dos pesos de entrada poderia tornar o neurônio artificial capaz de aprender (FERNANDES, 2008).

Uma das mais importantes RNAs criadas foi o *perceptron* (Frank Rosenblatt, 1958) que tinha a estrutura do neurônio artificial proposto por McCulloch e Pitts, com algoritmo de treinamento capaz de ajustar os pesos da entrada, como demonstrado por Hebb. Esta RNA, entretanto, era limitada ao possibilitar apenas o aprendizado de padrões linearmente separáveis, como demonstrado em 1969 por Minsky e Papert.

Após um período de isolamento das pesquisas na área durante a década de 70, no início da década de 80 houve uma retomada dos trabalhos. Em 1989, Rumelhart et al. (1986) apresentaram aquele que viria a ser um dos mais importantes algoritmos de treinamento, contornando todas as limitações apresentadas por Minsky e Papert, o *backpropagation*, de forma que agora as redes *perceptrons* multicamadas eram passíveis de treinamento.

Da década de 80 em diante até os dias atuais, as pesquisas na área só vêm crescendo, bem como as aplicações nas quais o uso de RNAs se mostra vantajoso, isso principalmente pelo avanço tecnológico, o qual possibilita o processamento de algoritmos que necessitam de maior poder computacional (BRAGA; LUDERMIR; CARVALHO, 2000).

2.4.1 Arquiteturas das redes neurais artificias

A arquitetura de uma RNA está intimamente ligada ao problema a ser tratado pela rede, além de estar relacionada ao algoritmo de aprendizagem usado para treinamento. A forma com que os neurônios estão dispostos são o que a definem. Dentro de uma mesma arquitetura é possível existirem diferentes topologias, que refletem as diferentes composições estruturais (SILVA; SPATTI; FLAUZINO, 2010).

Em termos de arquitetura, a estrutura básica de uma RNA é:

a. Camada de entrada;
b. Camada(s) intermediária(s);
c. Camada de saída.

Dessa forma, quanto ao número de camadas a RNA pode ser de camada única ou múltiplas camadas, e dentro destas os neurônios podem estar totalmente ou parcialmente conectados. O número destes não segue uma regra, sendo determinados de forma experimental.

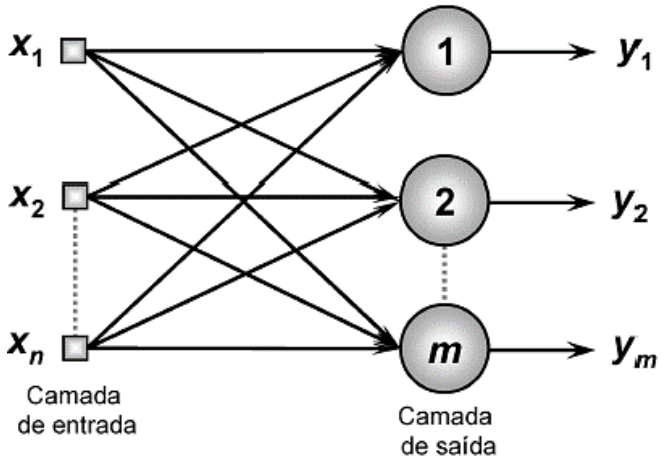
Quanto à topologia, dividem-se em *feedforward* e *feedback*.

Assim, destacam-se os seguintes tipos básicos de arquiteturas:

a. Rede *feedforward* camada simples

São RNAs que possuem apenas um agrupamento de neurônios (uma camada de entrada e uma de saída) que recebem informação simultaneamente (Figura 6). Esse tipo de rede também é acíclica, ou seja, não possui laços de realimentação. O fluxo de informação é unidirecional, indo da camada de entrada em direção à camada de saída (GIBELLI, 2016).

Figura 6. Rede *feedforward* de camada simples.



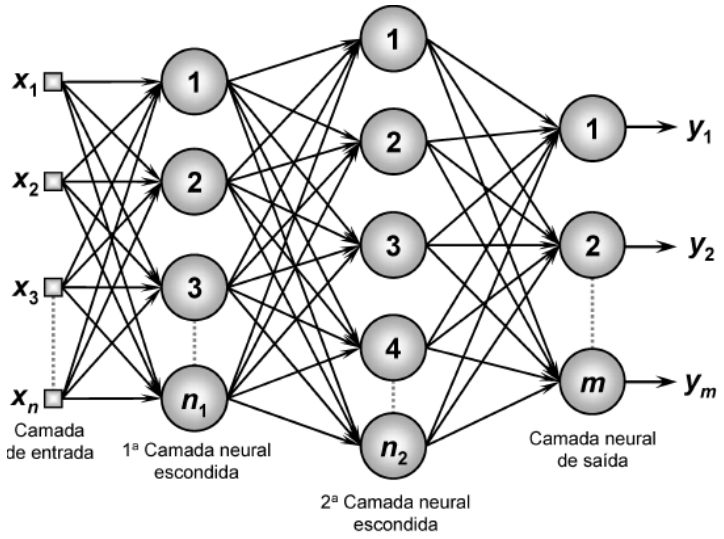
Fonte: (SILVA; SPATTI; FLAUZINO, 2010).

Redes do tipo são empregadas em problemas que envolvem a classificação de padrões e filtragem linear (GIBELLI, 2016). Também são empregados em telecomunicações, onde modens de alta velocidade de transmissão utilizam equalizadores adaptativos de linha e canceladores adaptativos de eco (WIDROW et al. 1994).

b. Rede *feedforward* de camadas múltiplas

RNAs de múltiplas camadas formadas por n sinais de entrada, duas camadas neurais intermediárias com n_1e n_2 neurônios, e uma camada de saída contendo m neurônios, que representam os valores de saída (Figura 7).

Figura 7. Rede *feedforward* de camadas múltiplas.



Fonte: (SILVA; SPATTI; FLAUZINO, 2010).

Assim como a arquitetura anterior, nesta o fluxo de informação também é unidirecional, porém, graças a presença de camadas ocultas, que formam um conjunto extra de conexões sinápticas e da riqueza de interações neurais, as camadas ocultas são capazes de extrair características complexas do ambiente em que atuam (HAYKIN, 2001).

As camadas intermediárias agem como extratoras de características, através de seus pesos elas são capazes de codificar as características apresentadas nos padrões de entrada, dessa forma a rede cria uma representação própria do problema, com mais riqueza e complexidade.

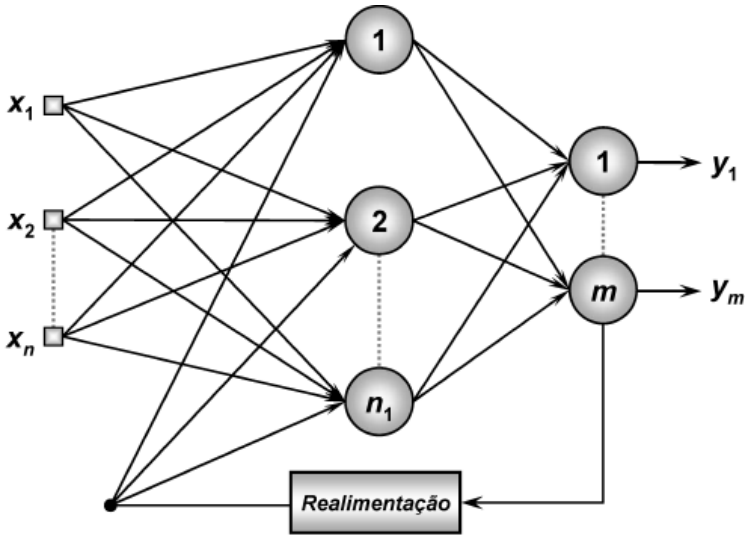
O número de camadas intermediárias necessárias para uma rede são duas, sendo capazes de reproduzir quaisquer mapeamentos, e aquelas com apenas uma, são suficientes para aproximar qualquer função contínua (CYBENKO, 1989).

Problemas de aproximação de funções, classificação de padrões, identificação de sistemas, otimização, robótica, controle de qualidade, etc. se beneficiam dessa arquitetura.

c. Rede recorrente (*feedback*)

Nessas redes o processamento da informação é dinâmico, possuindo ao menos um laço de realimentação (Figura 8). Esse laço de realimentação é o que torna a capacidade de aprendizagem e desempenho da rede dinâmico. Estes laços também envolvem o uso de ramos particulares compostos de elementos de atraso unitário, resultando em um comportamento dinâmico não linear, diferente das arquiteturas anteriores, desde que a se admita que a rede neural tenha unidades não-lineares (OLIVEIRA, 2005).

Figura 8. Rede recorrente.



Fonte: (SILVA; SPATTI; FLAUZINO, 2010).

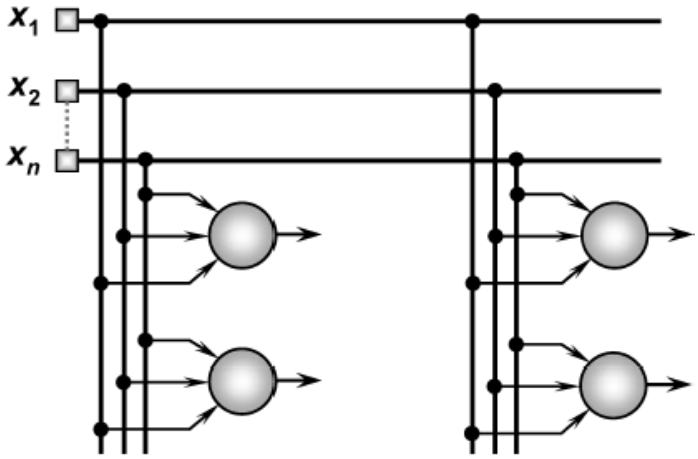
Exemplos de redes recorrentes famosas segundo Oliveira (2008) são:

- Rede de Hopfield: utilizada em reconhecimento de imagens;
- Rede de Kohonen: para classificação de padrões, otimização de problemas e simulações;
- Redes ART (*Adaptative Resonance Theory*): apenas um único neurônio artificial recebe a entrada de várias outras unidades semelhantes. Consiste na habilidade de se adaptar a novas entradas. Empregada no reconhecimento de sinais de radar e processamento de imagens.

d. Rede reticulada

O propósito desta rede é extrair características, sua aplicação se encontra em problemas de agrupamento, reconhecimento de padrões, otimização de sistemas, etc (GIBELLI, 2016). Uma ilustração para a arquitetura é apresentada na Figura 9.

Figura 9. Rede reticulada.



Fonte: (SILVA; SPATTI; FLAUZINO, 2010).

2.4.2 Função de ativação

A função de ativação do neurônio artificial tem como objetivo limitar a amplitude da saída dentro de um intervalo finito de valores, compatíveis com a resposta do neurônio. Existem vários tipos de funções de ativação disponíveis para as mais diversas aplicações, apresenta-se abaixo algumas das principais (SILVA; SPATTI; FLAUZINO, 2010).

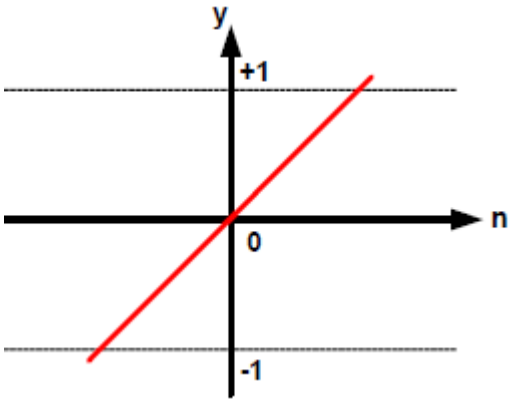
1. Função linear

   Para cada entrada, corresponde um valor real no intervalo -1 a 1.

$$y = n \quad (3)$$

   A Figura 10 apresenta a função de ativação do tipo linear

Figura 10. Função de ativação linear.



Fonte: (FERNANDES, 2008).

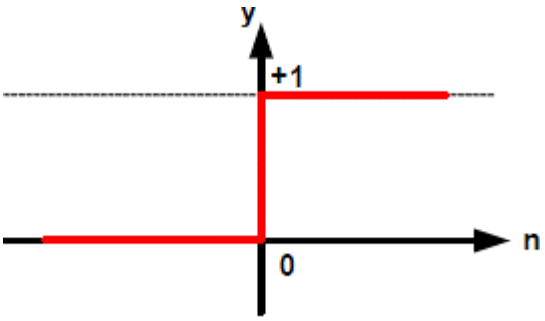
2. Função degrau

É uma função binária que apresenta os resultados com valores unitários positivos quando o potencial de ativação for maior ou igual a zero e valores nulos caso o potencial de ativação seja menor que zero.

$$y = \begin{cases} 1, se\ n \geq 0 \\ 0, se\ n < 0 \end{cases} \quad (4)$$

A Figura 11 ilustra a função degrau.

Figura 11. Função de ativação degrau.



Fonte: (FERNANDES, 2008).

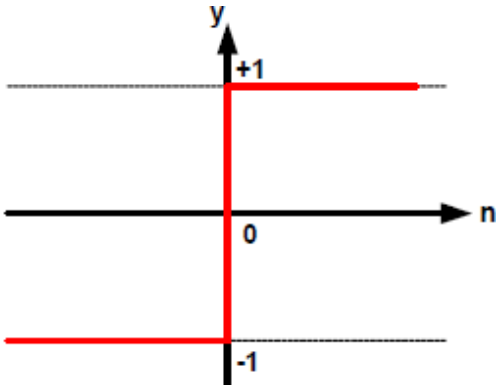
3. Função sinal (ou degrau bipolar)

Nesta função a saída pode assumir três valores, 1, 0 ou -1. Para o potencial de ativação maior que zero, a resposta será dada em valores unitários positivos. Quando zero, assumirá valor nulo, e para um potencial menor que zero, apresentará valores unitários negativos.

$$y = \begin{cases} 1, se\ n > 0 \\ 0, se\ n = 0 \\ -1, se\ n < 0 \end{cases} \quad (5)$$

A Figura 12 representa uma função do tipo sinal

Figura 12. Função de ativação sinal.



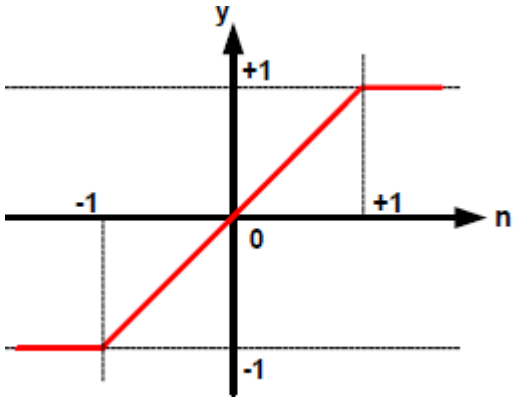
Fonte: (FERNANDES, 2008).

4. Função rampa simétrica

Para esta função os valores de saída são iguais aos valores da entrada para o intervalo $[-y, y]$. A Figura 13 representa essa função.

$$y = \begin{cases} 1, se\ n > 1 \\ n, se -1 \leq n \leq 1 \\ -1, se\ n < 0 \end{cases} \quad (6)$$

Figura 13. Função de ativação rampa simétrica.



Fonte: (FERNANDES, 2008).

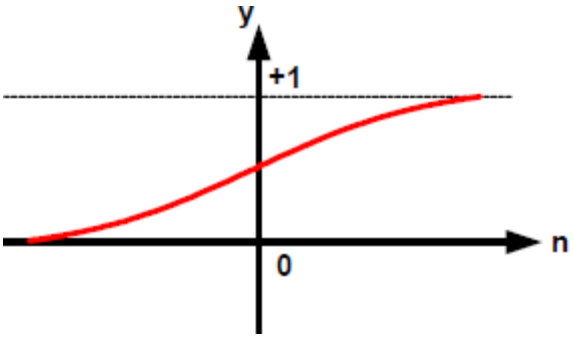
5. Função logística (ou sigmóide)

Esta função sempre assumirá valores reais entre 0 e 1.

$$y = \frac{1}{1 + e^{-\beta.n}} \quad (7)$$

onde β é uma constante real associada ao nível de inclinação da função logística frente ao seu ponto de inflexão, conforme mostrado pela Figura 14.

Figura 14. Função de ativação logística.



Fonte: (FERNANDES, 2008).

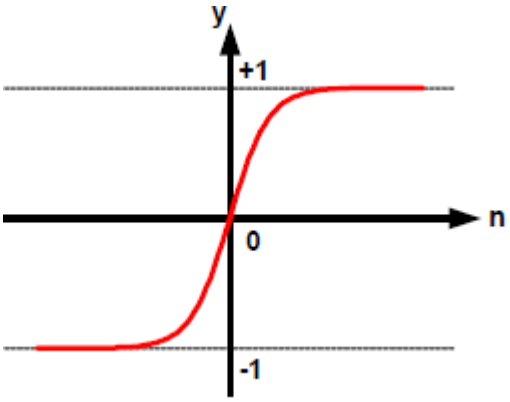
6. Função hiperbólica

A saída desta função será sempre valores reais entre -1 e 1.

$$y = \frac{1 - e^{-\beta.n}}{1 + e^{-\beta.n}} \quad (8)$$

onde β está associado ao nível de inclinação da função tangente hiperbólica em relação ao seu ponto de inflexão, conforme mostrado pela Figura 15.

Figura 15. Função de ativação tangente hiperbólica.



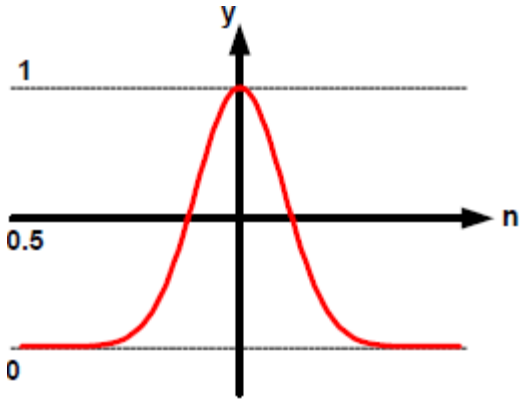
Fonte: (FERNANDES, 2008).

7. Função base radial (gaussiana)

Nesta função a saída é caracterizada por uma resposta que decresce (ou cresce) monotonicamente com a distância a um ponto central, como mostrado na Figura 16.

$$y = e^{-\left(\frac{(\sigma n)^2}{2\sigma^2}\right)} \quad (9)$$

Figura 16. Função base radial.



Fonte: (FERNANDES, 2008).

2.4.3 Aprendizagem

O processo de aprendizagem (ou treinamento) de uma RNA visa estabelecer qual a intensidade das conexões entre os neurônios desta. Este processo se dá em passos ordenados, de forma a sintonizar os pesos sinápticos e limiares de seus neurônios. Este conjunto de procedimentos é denominado algoritmo de aprendizagem (OLIVEIRA, 2005).

O objetivo deste algoritmo é generalizar as soluções apresentadas na saída. Para essa etapa o total de amostras é dividido em dois subconjuntos: treinamento e teste. O primeiro consiste em cerca de 60 a 90 % das amostras e é utilizado no processo de aprendizado da rede. A cada apresentação completa das amostras, os pesos sinápticos e limiares são ajustados, é a chamada época de treinamento. O segundo subconjunto, de teste, corresponde de 10 a 40 % do total de amostras, e é empregado para se verificar se os aspectos da generalização de soluções da rede estão em níveis aceitáveis, de forma que se possa validar a topologia utilizada (GIBELLI, 2016).

O treinamento pode ser do tipo supervisionado ou não supervisionado.

No supervisionado cada amostra dos sinais de entrada, associam-se amostra(s) conhecida(s) na(s) saída(s). Para cada par entrada-saída um sinal de erro é gerado e os pesos sinápticos são ajustados de forma a minimizar o erro. O algoritmo mais utilizado para treinamento supervisionado é o de retropropagação (*Backpropagation*). Problemas de aproximação de funções, modelagem de sistemas e classificação de dados são exemplos típicos de aprendizagem supervisionada (GIBELLI, 2016)

No treinamento não supervisionado, há um conjunto de amostra(s) na entrada, mas ao mesmo não está associado nenhuma saída. Dessa forma, a própria RNA se auto organiza via agrupamento do conjunto de treinamento em subconjuntos (*clusters*), com base na similaridade. Um cenário em que se busca identificar usuários de cartão de crédito com base nos seus perfis de compra, é um problema de categorização de dados em que a aplicação de aprendizagem não-supervisionada é necessária (WIDROW et al. 1994).

3 AQUISIÇÃO E PROCESSAMENTO DE DADOS

3.1 Sensor de corrente e circuito de condicionamento do sinal

O transformador de corrente (TC) YHDC produzido pela companhia Beijing YaoHuaDechang Electronics Ltda, modelo SCT-013-000 é um sensor de corrente alternada não invasivo (núcleo dividido) capaz de realizar medições até 100 A (Figura 17). O TC desenvolve tensão suficiente para utilizar de maneira total a resolução da entrada analógica do Arduino e a distorção na forma de onda devido a saturação do secundário é ínfima. Tais características aliadas ao seu baixo custo o tornaram adequado para o projeto.

O TC explora a lei da indução de Faraday e se mostra vantajoso frente aos demais sensores do tipo, pois o isolamento elétrico entre a corrente e o sinal de saída é inerente. Dessa forma é possível a medição de correntes em situações que a tensão é elevada e bastante variável. Transformadores de corrente correspondem a quase totalidade dos sistemas de medição de corrente (KIRKHAM, 2009), graças ao seu baixo custo e ao sinal de saída ser diretamente compatível com um conversor analógico-digital (ZIEGLER, et al., 2009).

O sinal de saída do TC empregado é em forma de corrente entre 0-50 mA, variando conforme a corrente do primário (0-100 A). O conector é do tipo TRS (do inglês *tip-ring-sleeve*, lit. ponta-anel-capa) de 3.5 mm

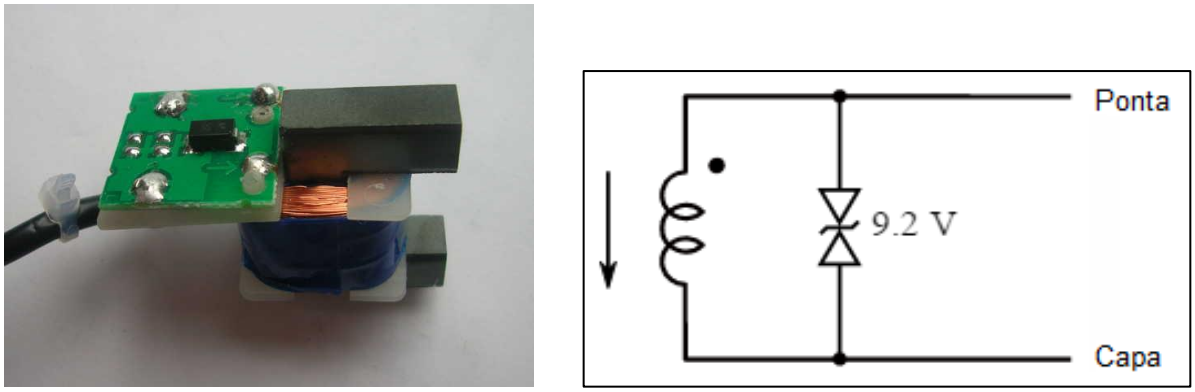
Figura 17. Transformador de corrente YHDC SCT-013-000.

Fonte: (OpenEnergyMonitor, 2015).

O sensor não possui uma resistência de carga interna, mas sim um transorb (*Transient Voltage Suppressor*), componente eletrônico destinado a absorver picos de sobre-tensões, para limitar a tensão que pode surgir no conector, caso o transformador seja desconectado enquanto a carga do primário ainda se encontre energizada.

Como é possível observar pelo diagrama do circuito (Figura 18), a ponta é positiva em relação a capa e o anel não está conectado.

Figura 18. Componentes internos e diagrama circuital do YHDC SCT-013-000.

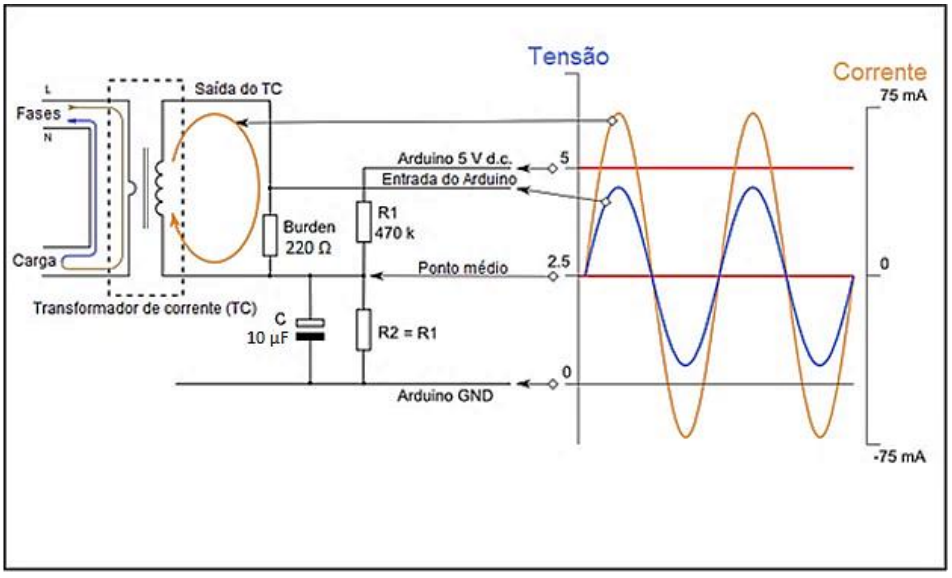


Fonte: (OpenEnergyMonitor, 2015).

Para conectar o sensor ao Arduino o sinal precisa ser condicionado para as entradas analógicas, visto que elas não operam com sinal de corrente, na forma de um sinal de tensão positivo entre 0 V e a tensão de referência do ADC (conversor analógico-digital), 5 V.

Para tanto foi empregado o circuito representado na Figura 19.

Figura 19. Diagrama do circuito de condicionamento do sinal de saída do TC.



Fonte: (OpenEnergyMonitor, 2015).

O resistor de carga (*burden*) é responsável pela conversão do sinal de corrente em um valor de tensão que possa ser enviado ao Arduino.

O valor deste foi determinado segundo a equação (10) abaixo.

$$Resistor\ de\ carga\ (\Omega) = \frac{(TRA \times VTC)}{(2\sqrt{2} \times I_{\max\_primário})} \quad (10)$$

Onde:

TRA, corresponde a tensão de referência da entrada analógica do Arduino em V;

VTC, número de voltas do secundário do transformador de corrente;

$I_{\max\_primário}$, corrente máxima do primário em A.

O princípio por de trás desse equacionamento é o de converter o valor máximo da corrente ($I_{\max\_primário}$) no primário em rms, através da multiplicação por $\sqrt{2}$. Dividir esse valor pelo número de voltas do secundário do TC para se obter a corrente de pico do secundário, e com esse valor calcular o resistor de carga ao dividir o valor da tensão de referência da entrada analógica do Arduino pela corrente no secundário.

Para o presente projeto a equação (10) resulta em:

$$Resistor\ de\ carga\ (\Omega) = \frac{(5\ V \times 2000\ voltas)}{(2\sqrt{2} \times 100\ A)} \cong 35{,}4\ \Omega \quad (11)$$

Como 35 Ω não é um valor usual haviam duas possíveis opções (39 Ω ou 33 Ω), o menor valor deve sempre ser escolhido para que a corrente máxima da carga não gere uma tensão superior à de referência da entrada analógica (OpenEnergyMonitor, 2015). Entretanto, como as cargas que foram testadas para este projeto se limitavam a um conjunto de 6 lâmpadas (2 incandescentes, 2 fluorescentes compactas e 2 LEDs) o valor de corrente gerado no secundário era muito pequeno (a de maior potência tinha uma corrente da ordem de 551 mA) para que o resistor de 33 Ω pudesse resultar em um valor de tensão alto o suficiente de forma que a digitalização do sinal fosse possível.

Para contornar esse problema, como descrito na página do projeto OpenEnergyMonitor (2015), um resistor de 220 Ω foi empregado. O valor é tal que se pudesse obter a melhor resolução possível, sem haver uma introdução de erro de fase considerável, problema que ocorre quando se lida com cargas muito pequenas e um valor muito elevado do resistor de carga.

As entradas analógicas do Arduino requerem também que o sinal seja sempre positivo, dessa forma ao invés de conectar o TC ao terra, este foi conectado a um divisor de tensão, com dois resistores de 470 kΩ, para que o consumo de energia fosse o menor possível, caso o sistema fosse alimentado via bateria., suprindo metade da tensão da fonte, que no caso são os 5 V de referência do Arduino. Sendo assim o valor do sinal de saída oscila acima e abaixo de 2,5 V, mas sempre positivo.

O capacitor de 10 µF age como um filtro no circuito.

3.2 Sensor de tensão e o circuito de condicionamento do sinal

Uma das soluções mais usuais para o sensoriamento da tensão é fazer uso de um transformador de potencial (TP) de forma a reduzir a tensão proveniente da rede para a faixa de valores de operação do microcontrolador.

A Comissão Eletrotécnica Internacional (IEC, na sigla em inglês) define o transformador de potencial como um equipamento estático, com dois ou mais enrolamentos, que, por indução eletromagnética, converte um sistema de tensão em outro, sendo este, geralmente, de tensão diferente, mas proporcional ao sistema original, mantida a frequência (IEC 60050-321, 1986).

Transformadores de potencial são baseados em três relações principais, são elas (CESÁRIO JÚNIOR, 2014):

- Relação nominal, entre os valores nominais das tensões primárias e secundárias, respectivamente, tensões estas para as quais o transformador foi projetado e construído;
- Relação real, entre o valor exato de uma tensão qualquer aplicada ao primário do transformador e o correspondente valor exato da tensão verificada no secundário;
- Fator de correção, razão entre a relação nominal e a relação real do transformador de potencial. Na prática mede-se o valor da tensão no secundário do transformador de potencial com um voltímetro e multiplica este valor pela relação nominal. Este valor representa o valor medido da tensão primária, e não seu valor exato.

Sendo assim, buscando tornar o processo o mais seguro possível foi empregado um adaptador AC-AC 127 V/12 V modelo JF41-1201000AU (Figura 20).

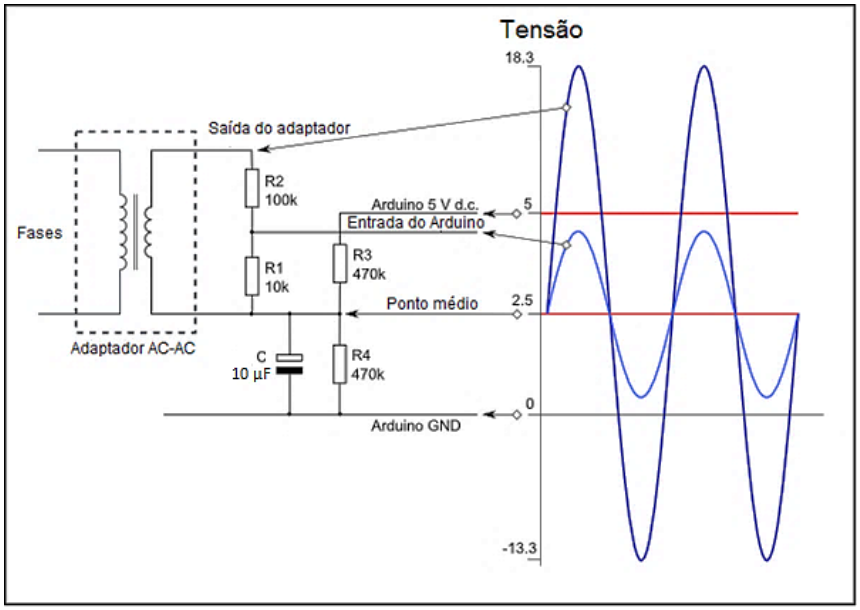
Figura 20. Transformador de tensão JF41-1201000AU.

Fonte: próprio autor.

O sinal de saída também deve ser condicionado antes da entrada no Arduino, assim como o aconteceu com o da corrente, ou seja, possuir um pico menor que os 5 V de referência do ADC e não apresentar componente negativa.

A figura abaixo apresenta o circuito para condicionamento do sinal.

Figura 21. Diagrama do circuito de condicionamento do sinal de saída do transformador de tensão.



Fonte: (OpenEnergyMonitor, 2015).

O circuito é composto por um divisor de tensão conectado aos terminais do adaptador para reduzir o valor do sinal de saída (R1 e R2) e outro para que o sinal de entrada seja sempre positivo (R3 e R4), da mesma forma que no circuito de condicionamento do sinal do TC.

Aqui o capacitor também age como filtro.

Para o cálculo de R1 e R2 se partiu do requisito de que o pico de tensão do sinal de saída fosse aproximadamente 1 V, aplicando a equação (12).

$$V_{pico\_saída} = \frac{R1}{(R1 + R2)} \times V_{pico\_entrada} \quad (12)$$

Onde:

$V_{pico\_saída}$, tensão pico em rms na saída do adaptador, cujo valor deve ser aproximadamente 1 V;

$V_{pico\_entrada}$, tensão pico em rms na entrada do adaptador;

$R1\ e\ R2$, valor dos resistores do divisor de tensão para redução do valor máximo do sinal de entrada.

Aplicada aos dados do presente projeto:

$$V_{pico\_saída} = \frac{10\ k\Omega}{(10\ k\Omega + 100\ k\Omega)} \times 12\ V = 1{,}091\ V \quad (13)$$

Os valores de R3 e R4 também são de 470 kΩ, pela mesma razão anteriormente exposta para o TC.

3.3 Arduino Uno R3

O Arduino (https://www.arduino.cc/) é uma plataforma aberta baseada em um conceito de hardware e software simples de se usar, programar ou alterar. As placas têm capacidade de ler sinais de entrada (um sensor, por exemplo) e torná-lo em um sinal de saída que pode controlar algum elemento, como um motor, ou exibir um resultado.

Segundo o fabricante, o Arduino Uno R3 (2016) é a terceira iteração da placa baseada no microcontrolador ATmega328P. Possui 14 entradas e saídas digitais, das quais 6 podem ser usadas como saídas PWM; 6 entradas analógicas; um cristal de 16 MHz, interface serial através de conector USB, que pode ser usado para alimentar a placa e/ou programar o microcontrolador;

um conector *power jack* de corrente contínua e um botão de reset, que permite reprogramá-lo (Figura 22)

Figura 22. Arduino Uno R3.

Fonte: (Arduino, 2016).

A natureza aberta, integrada aos componentes mínimos necessários para o funcionamento do microcontrolador, a facilidade para integração de novos elementos, sua vasta documentação e bibliotecas de suporte à sensores e dispositivos variados, além do preço acessível tornaram-no a plataforma a mais interessante para o desenvolvimento do projeto.

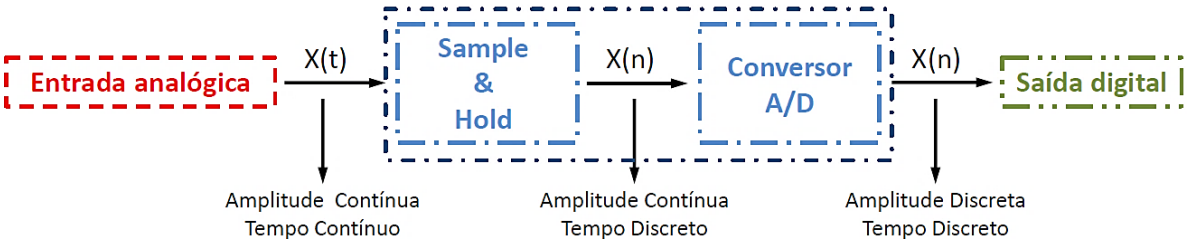
A placa é o principal componente do sistema desenvolvido, sendo responsável pela leitura, conversão e envio do sinal para computador.

3.3.1 O conversor analógico-digital (ADC) do ATmega 328P

Os sinais de corrente e tensão provenientes dos sensores são de natureza analógica e contínua no tempo. Consequentemente, para que sejam processados posteriormente, o principal papel do Arduino é realizar a conversão destes sinais para o formato digital, de forma que sejam representados por uma sequência de números discretizados, em que cada um deles representa um valor do sinal instantâneo. O processo de digitalização do sinal compreende a amostragem

do sinal e a quantização, sendo feito por meio de uma conversor analógico-digital, como demonstrado na Figura 23 (SEDRA; SMITH, 2000).

Figura 23. Processo de digitalização do sinal analógico.



Fonte: (GIBELLI, 2016).

Como o ADC opera via aproximação para se obter o valor da amostra analógica, quanto maior o número de bits (n), mais o valor da palavra quantizada se aproxima do valor da amostra analógica. O ADC presente no microcontrolador Atmel 328P, que faz parte do Arduino Uno R3, possui 10 bits.

A equação 14 define a saída de um ADC com n-bits (MADI-SETTI; WILLIAMS, 1998).

$$D = \frac{A_{sinal}}{FS} = \frac{b_n}{2^n} + \frac{b_{n-1}}{2^{n-1}} + \cdots + \frac{b_1}{2^1} \quad (14)$$

Onde:

D é a palavra digital quantizada;

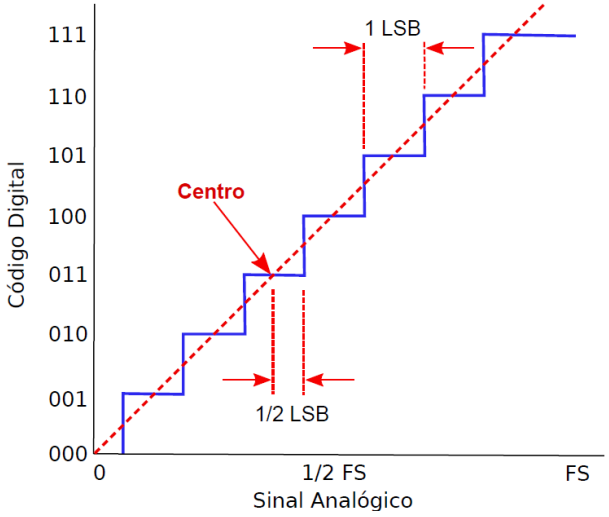
A_{sinal} é o sinal analógico;

FS é o fundo de escala do sinal

b_n é o valor digital 0 ou 1.

O *Least Significant Bit* (LSB), dentre os 10, é a largura da região quantizada, sendo que o valor ideal corresponde a linha que passa pelo centro de cada uma das regiões, da forma mostrada na Figura 24 para um ADC de 3 bits (MADISETTI; WILLIAMS, 1998).

Figura 24. Características da transferência ideal para um ADC de 3 bits.



Fonte: (GIBELLI, 2016).

A resolução do ADC é uma de suas características mais importantes e é definida como a menor distinção de alteração que pode ser produzida por meio do sinal analógico no conversor, dada pela equação 15 (MADISETTI; WILLIAMS, 1998).

$$\Delta A_{sinal} = \frac{FS}{2^n} \quad (15)$$

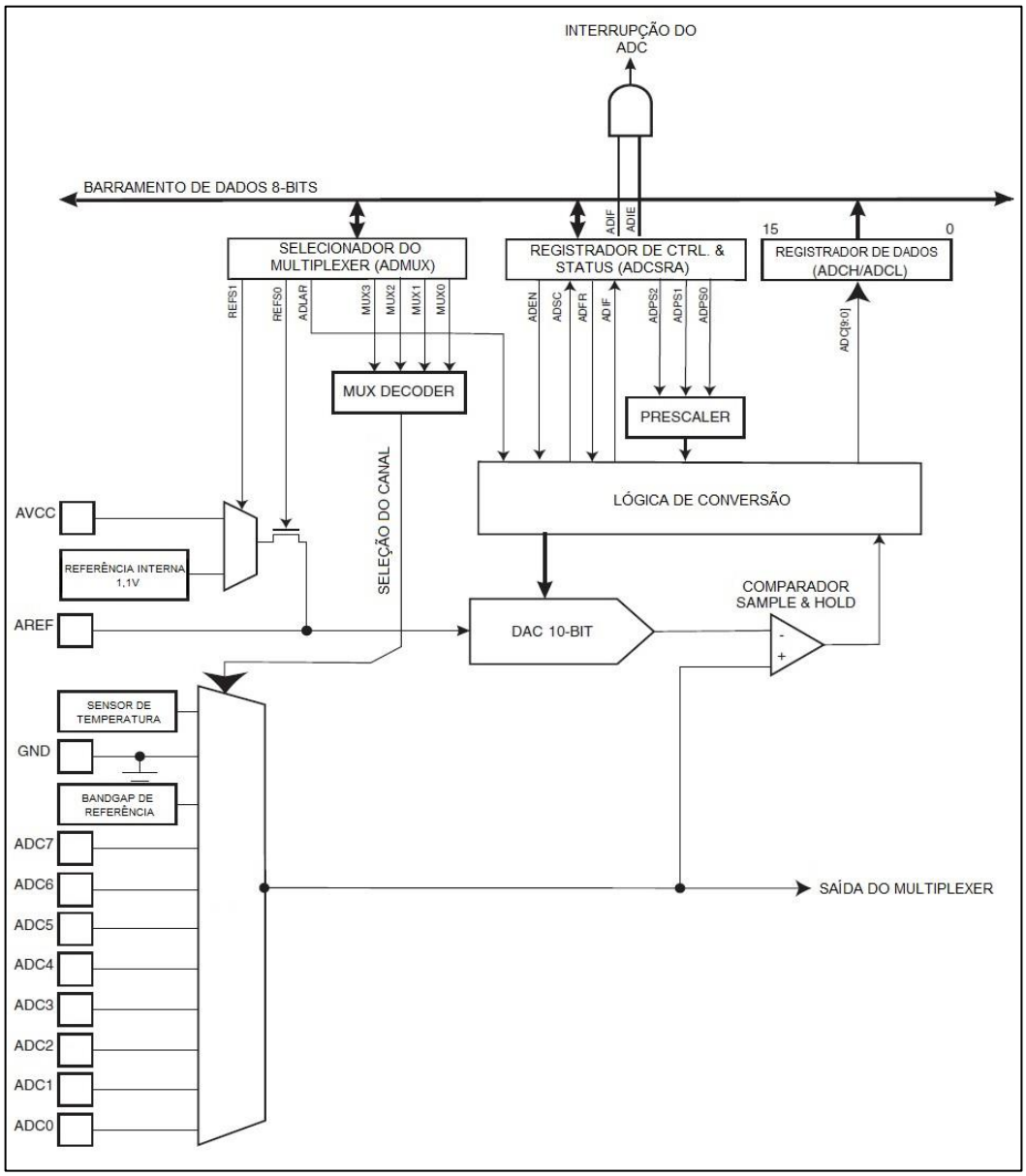
Na equação (15), ΔA_{sinal} representa o menor valor do sinal analógico reproduzível por um conversor com n-bits para um sinal com fundo de escala FS. No caso do ADC do Atmel 328P este valor corresponde a divisão da tensão de referência (V_{ref}) pela quantidade de níveis de quantização possíveis com 10 bits, que é de $2^{10} = 1024$ níveis, correspondendo à $\sim 5\ mV/divisão$ para uma V_{ref} de 5 V, com cada amostra sendo aproximada para o nível de quantização mais próximo do seu valor original. Devido a este processo de aproximação sucessiva há um erro associado à representação digital do sinal, que deve ser no máximo igual à metade de um nível de quantização (COURY; OLESKOVICZ; GIOVANINI, 1999). É possível notar que um maior número de bits, implica em um menor número de erros, tornando este um dos parâmetros de maior importância para que a representação do sinal seja a melhor possível.

Neste trabalho, embora o ADC possua 10 bits, a resolução foi reduzida para 8, diminuindo os níveis de quantização para 256 e, consequentemente, aumentando o valor de cada divisão para $0{,}02\ V$, elevando dessa forma o erro de quantização. Tal medida foi tomada

tendo em vista a frequência de operação do ADC de 500 kHz, muito acima da faixa de 50~200 kHz recomendada pela fabricante para operação com 10 bits (Atmel, 2015), além de problemas decorrentes da forma como o algoritmo de processamento do sinal no *software* Matlab interpretava os dados enviados.

A Figura 25 é o diagrama de blocos do ADC presente no ATmega 328P. Este é um ADC de aproximação sucessiva com 10 bits, onde o valor mínimo representa o GND e o valor máximo a tensão no pin AREF menos 1 LSB. O mesmo está, ainda, conectado a um multiplexador de 8 canais, que permite a conexão do ADC com as portas analógicas (Port A), uma por vez.

Figura 25. Diagrama de blocos do ADC presente no ATmega 328P.



Fonte: adaptado de (Atmel, 2015).

O circuito *Sample&Hold* presente garante que a tensão de entrada seja mantida em nível constante durante a conversão. Como tensão de referência é possível se empregar uma fonte externa conectada ao AREF, o pin de tensão individual (AV_{CC}) ou ainda a tensão de referência interna de 1,1 V, estas duas últimas devem estar conectadas via um capacitor em AREF para garantir a eliminação de ruídos. A seleção se dá via *set* dos bits REFSn no registrador ADMUX (Tabela 1).

A seleção de qual canal analógico será a entrada do ADC é feita via *set* dos bits MUXn no registrador ADMUX, sendo que apenas uma porta pode ser lida por vez (Tabela 2).

O bit ADEN no registrador ADCSRA precisa ser habilitado para que o ADC esteja disponível para uso, sendo que nenhuma alteração de canal ou tensão de referência será aplicada até que isso ocorra.

O resultado em 10 bits gerado é armazenado nos registradores de dados ADCH e ADCL, dois por conta da arquitetura do ATmega 328P ser 8 bits. Por padrão estes estão ajustados à direita, de forma que se deve ler ADCL primeiro e posteriormente o ADCH, garantindo assim que o conteúdo lido pertence a mesma conversão. Uma vez que ADCL é lido o acesso aos registradores de dados é bloqueado até que ADCH seja lido. Caso não seja necessária uma precisão maior que 8 bits é possível se ajustar os registradores à esquerda e realizar a leitura apenas do ADCH, como foi feito no presente trabalho, para tanto se deve dar *set* no bit ADLAR no ADMUX.

O ADC possui ainda a sua própria interrupção que se inicia sempre que uma conversão é completada.

Abaixo descrevem-se os registradores de maior importância e que foram diretamente manipulados para a execução deste trabalho (Figuras 26 à 28).

- ADMUX – Registrador de seleção do multiplexer

Figura 26. Registrador ADMUX.

Bit	7	6	5	4	3	2	1	0	
(0x7C)	**REFS1**	**REFS0**	**ADLAR**	**–**	**MUX3**	**MUX2**	**MUX1**	**MUX0**	**ADMUX**
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Fonte: (Atmel, 2015).

Bit 7:6 – REFS[1:0]: Bits de seleção da tensão de referência

Bits associados a seleção da tensão de referência do ADC, conforme mostrado na Tabela 2.

Tabela 2. Seleção da tensão de referência do ADC.

REFS1	REFS0	Tensão de referência
0	0	AREF, V_{ref} interna desligada
0	1	AV_{cc} com capacitor externo conectado ao pin AREF
1	0	Reservado
1	1	Tensão de referência interna de 1,1 V com capacitor externo conectado ao pin AREF

Fonte: (Atmel, 2015).

Bit 5 – ADLAR: Resultado ajustado à esquerda

Esse bit altera a forma com que o resultado será apresentado nos registradores de dados. Se a direita se tem 10 bits de precisão, se a esquerda 8 bits.

Bit 3:0 – MUX[3:0]: Bits de seleção da porta analógica

Os valores nestes bits indicam a qual porta analógica o ADC está conectado (Tabela 3).

Tabela 3. Seleção da porta analógica.

MUX3...0	Porta analógica
0000	ADC0
0001	ADC1
0010	ADC2
0011	ADC3
0100	ADC4
0101	ADC5
0110	ADC6
0111	ADC7
1000	ADC8*
1001	(Reservado)
1010	(Reservado)
1011	(Reservado)
1100	(Reservado)
1101	(Reservado)
1110	1,1V (V_{BG})
1111	0V (GND)

*Sensor de temperatura

Fonte: (Atmel, 2015).

- ADCSRA – Registrador A de controle e status

Figura 27. Registrador ADCSRA.

Bit	7	6	5	4	3	2	1	0	
(0x7A)	**ADEN**	**ADSC**	**ADATE**	**ADIF**	**ADIE**	**ADPS2**	**ADPS1**	**ADPS0**	**ADCSRA**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Fonte: (Atmel, 2015).

Bit 7 – ADEN: habilita o ADC

Quando 1 habilita o ADC; quando 0, o desliga. Desligar o ADC reduz o consumo de energia e caso ocorra durante uma conversão, esta é encerrada.

Bit 6 – ADSC: inicia uma conversão

Quando em modo de conversão única esteve bit deve ser 1. Quando em modo contínuo 1 inicia a primeira conversão. A primeira conversão após o *set* desse bit leva 25 ciclos de *clock* do ADC, ao invés dos 13, visto que a primeira conversão inicializa o ADC. Durante a conversão esse bit permanece como 1 e ao fim retorna a 0. Escrever 0 aqui não tem efeito algum.

Bit 5 – ADATE: habilita o *auto-trigger*

Quando 1, o *auto-trigger* é habilitado e o ADC irá iniciar uma conversão sempre na subida positiva do sinal de partida.

Bit 4 – ADIF: *flag* da interrupção

Este bit se altera sempre que uma conversão do ADC é completada e os registradores de dados são atualizados, desde que o bit ADIE, que habilita a interrupção de conversão completa e o bit I no SREG estejam *set*.

Bit 3 – ADIE: habilita a interrupção do ADC

Quando este bit e o I-bit no SREG estão *set* a interrupção de “conversão completa” do ADC é habilitada.

Bits 2:0 – ADPS [2:0]: bits de seleção do prescaler

São os bits que determinam qual será o fator de divisão entre a frequência de *clock* do sistema e a de entrada do ADC, sendo esta a razão entre 16 MHz (frequência de *clock* do sistema) e o fator de divisão (Tabela 4), que por padrão é 128. Para a correta operação do ADC é importante que este *clock* esteja entre 50 kHz e 200 kHz, embora caso uma precisão menor que 10 bits possa ser empregada, valores acima de 200 kHz são possíveis, caso deste trabalho.

Tabela 4. Seleção do prescaler do ADC.

ADPS2	**ADPS1**	**ADPS0**	**Fator de divisão**
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

Fonte: (Atmel, 2015).

Para o cálculo da frequência de operação do ADC emprega-se a equação abaixo.

$$f_{ADC} = \frac{16\ MHz}{F} \quad (16)$$

Onde:

f_{ADC} é a frequência de operação do ADC,

F é o fator de divisão.

A Equação 16 para o presente trabalho resulta em:

$$f_{ADC} = \frac{16\ MHz}{32} = 500\ kHz \quad (17)$$

- ADCL e ADCH – registradores de dados

Figura 28. Registradores de dados (ADCL e ADCH).

ADLAR = 0

Bit	15	14	13	12	11	10	9	8	
(0x79)	–	–	–	–	–	–	ADC9	ADC8	ADCH
(0x78)	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

ADLAR = 1

Bit	15	14	13	12	11	10	9	8	
(0x79)	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADCH
(0x78)	ADC1	ADC0	–	–	–	–	–	–	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

Fonte: (Atmel, 2015).

Estes são os registradores em que se encontram o resultado da conversão. Quando ADCL é lido o acesso aos registradores de dados é bloqueado até que ADCH seja lido também, isso para ADLAR=0, ou seja, com ajuste à direita e 10 bits de precisão. Quando ADLAR=1, presente caso, a precisão é reduzida para 8 bits e apenas ADCH precisa ser lido.

O resultado presente nestes registradores é obtido via Equação 18.

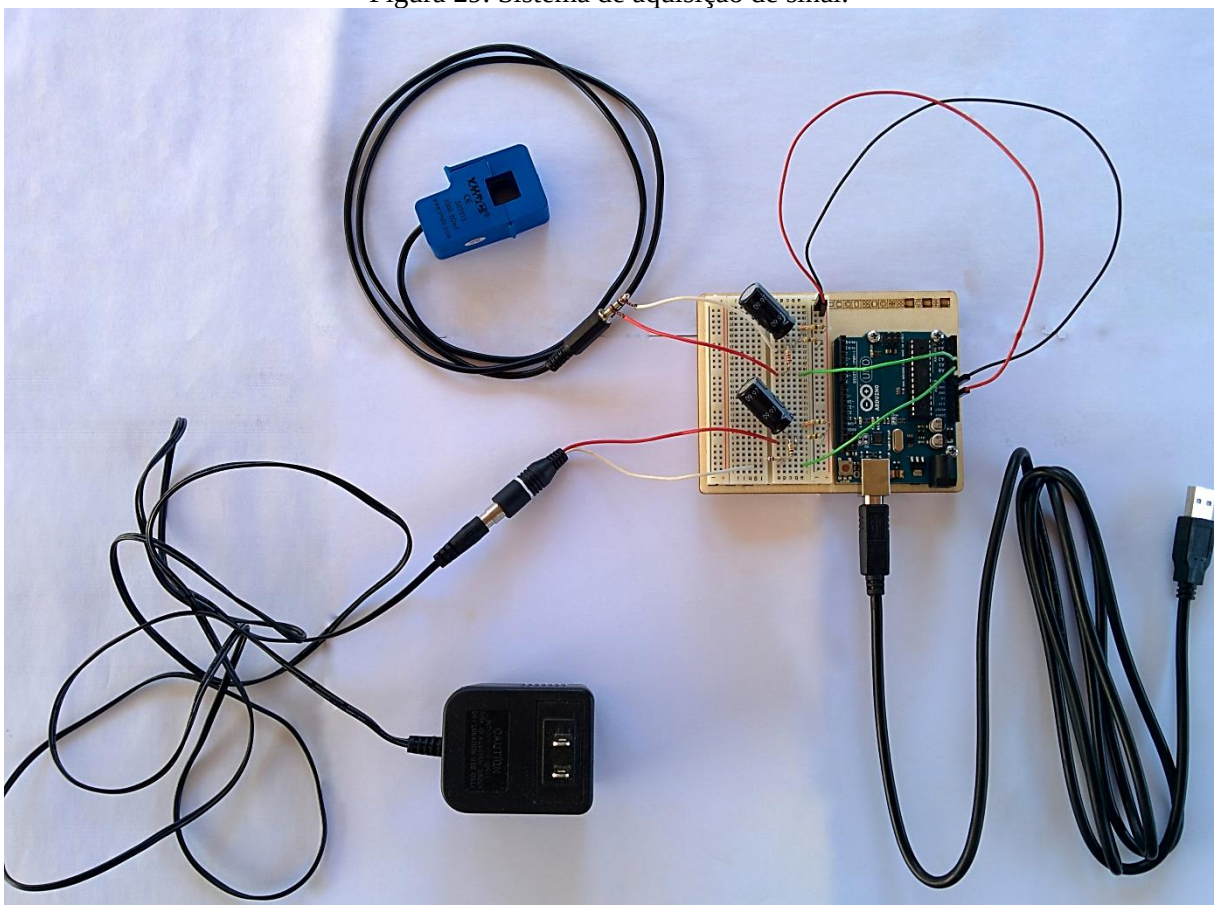
$$ADC = \frac{V_{IN} \times 2^n}{V_{REF}} \quad (18)$$

Na qual V_{IN} é a tensão na porta selecionada, V_{REF} a de referência selecionada menos um LSB e n, o número de bits do ADC.

3.4 O hardware para aquisição de sinal

A Figura 29 mostra o sistema completo desenvolvido para aquisição de sinal. Os 5 V de referência são providos pela interface USB.

Figura 29. Sistema de aquisição de sinal.



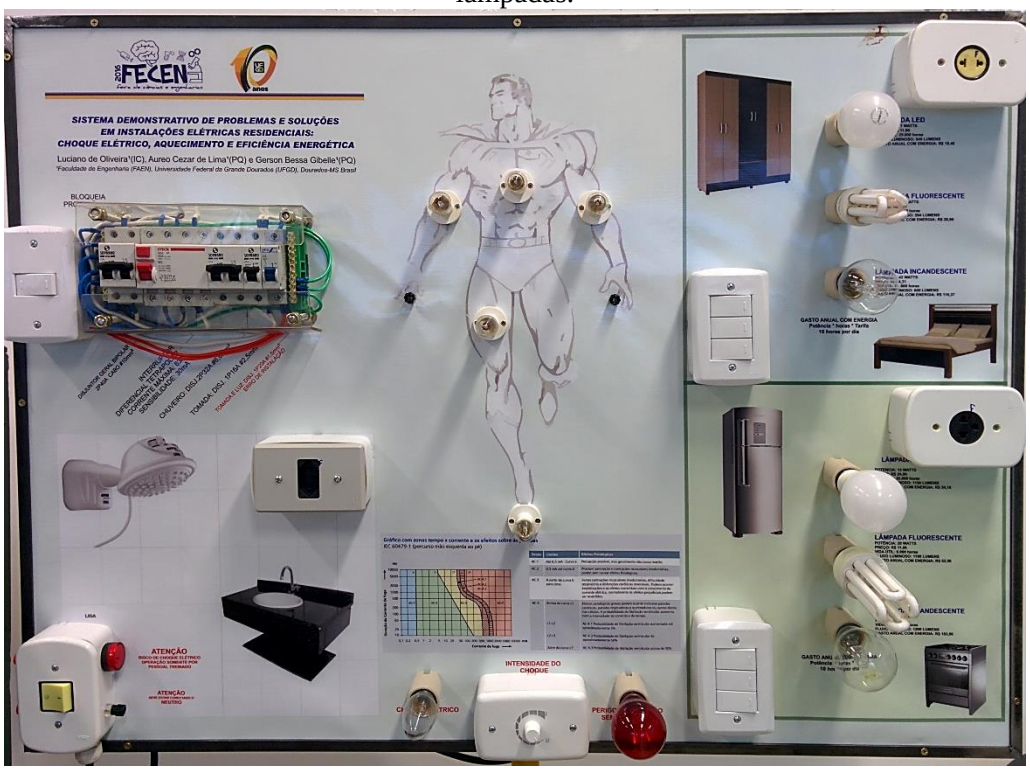
Fonte: próprio autor.

A frequência de amostragem para cada entrada analógica é de 15660 Hz.

Os dados foram coletados do painel didático desenvolvido em um projeto de ensino mostrado na Figura 30, onde foram realizadas 10 medições de tensão e corrente para cada par dos diferentes tipos de lâmpadas, com 5000 pontos para cada um dos sensores. O TP foi conectado a tomada superior e o TC aos dois fios vermelhos de alimentação das cargas (visíveis na imagem) no painel dos disjuntores. Os interruptores permitiram o acionamento individual de cada lâmpada, sendo que estas foram analisadas sempre em pares.

As lâmpadas utilizadas foram duas incandescentes, a superior com 42 watts e a inferior de 70 watts. Duas fluorescentes compactas de 11 watts e 20 watts, respectivamente. E por fim duas LEDs de 7 watts e 13 watts.

Figura 30. Painel utilizado para aquisição dos dados de tensão e corrente dos diferentes tipos de lâmpadas.



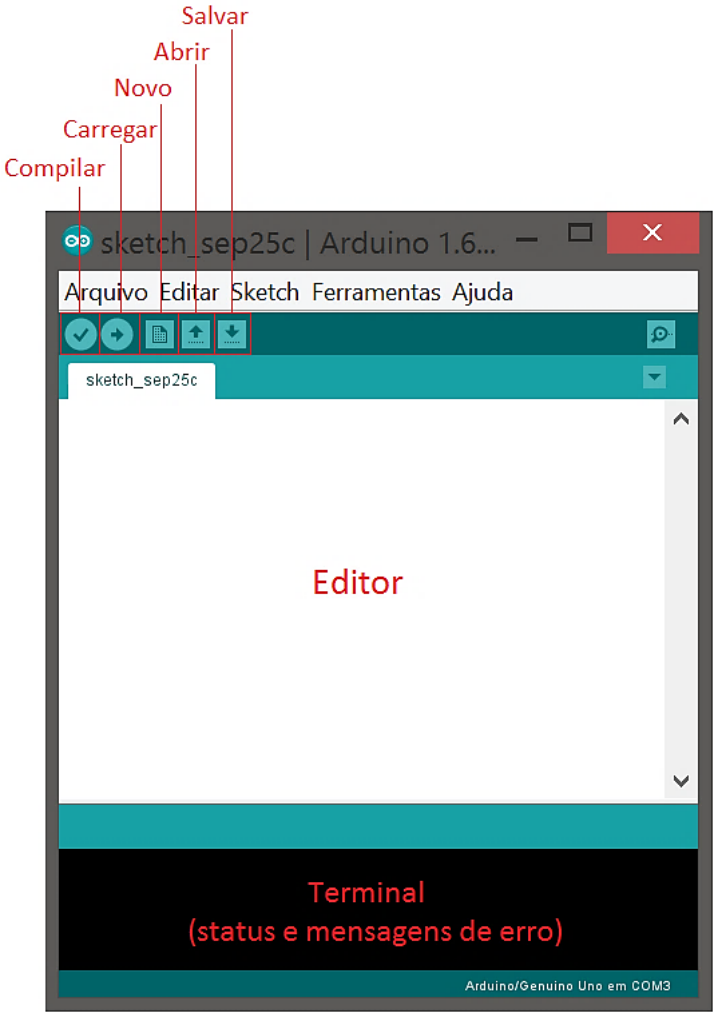
Fonte: próprio autor.

3.5 Código de aquisição de sinal no Arduino

A programação do Arduino é feita via o *Arduino Integrated Development Environment* (IDE), um ambiente de desenvolvimento que contém um editor de texto para a escrita do código, um terminal de mensagens, uma caixa de ferramentas com botões para as funções mais comuns e uma série de menus (Figura 31). Atráves dele *sketches*, os programas escritos usando o IDE, são enviados para o microcontrolador na placa.

A linguagem de programação é baseada no *Wiring*, um framework de programação em código aberto para microcontroladores que permite o desenvolvimento de código para diversas plataformas, e é bastante próxima ao C.

Figura 31. Arduino Integrated Development Environment (IDE).



Fonte: próprio autor.

O código possui duas partes essenciais sem as quais não funciona, são elas `setup()` e `loop()`. A primeira é executada apenas uma vez e é onde se encontram as instruções gerais que preparam o sistema para o loop principal. No caso do código empregado é onde se estabelece a velocidade da porta serial (*baud rate*), número de bits de dados, tipo de paridade (par ou ímpar) e número de *stop bits* (1 ou 2). Ainda nessa parte estão todas as configurações realizadas nos registradores do ADC, necessárias para o funcionamento do sistema de aquisição.

A segunda parte é a principal, sendo executada continuamente enquanto o Arduino estiver ligado. Aqui ela é responsável pelo envio dos dados na forma binária através da porta serial para o computador.

A parte mais importante deste código, entretanto, é a interrupção `ADC_vect` que é onde os dados são lidos diretamente do ADCH, os canais se alternam e se inicia uma nova conversão. Essa interrupção é chamada sempre que uma conversão do ADC termina, sendo

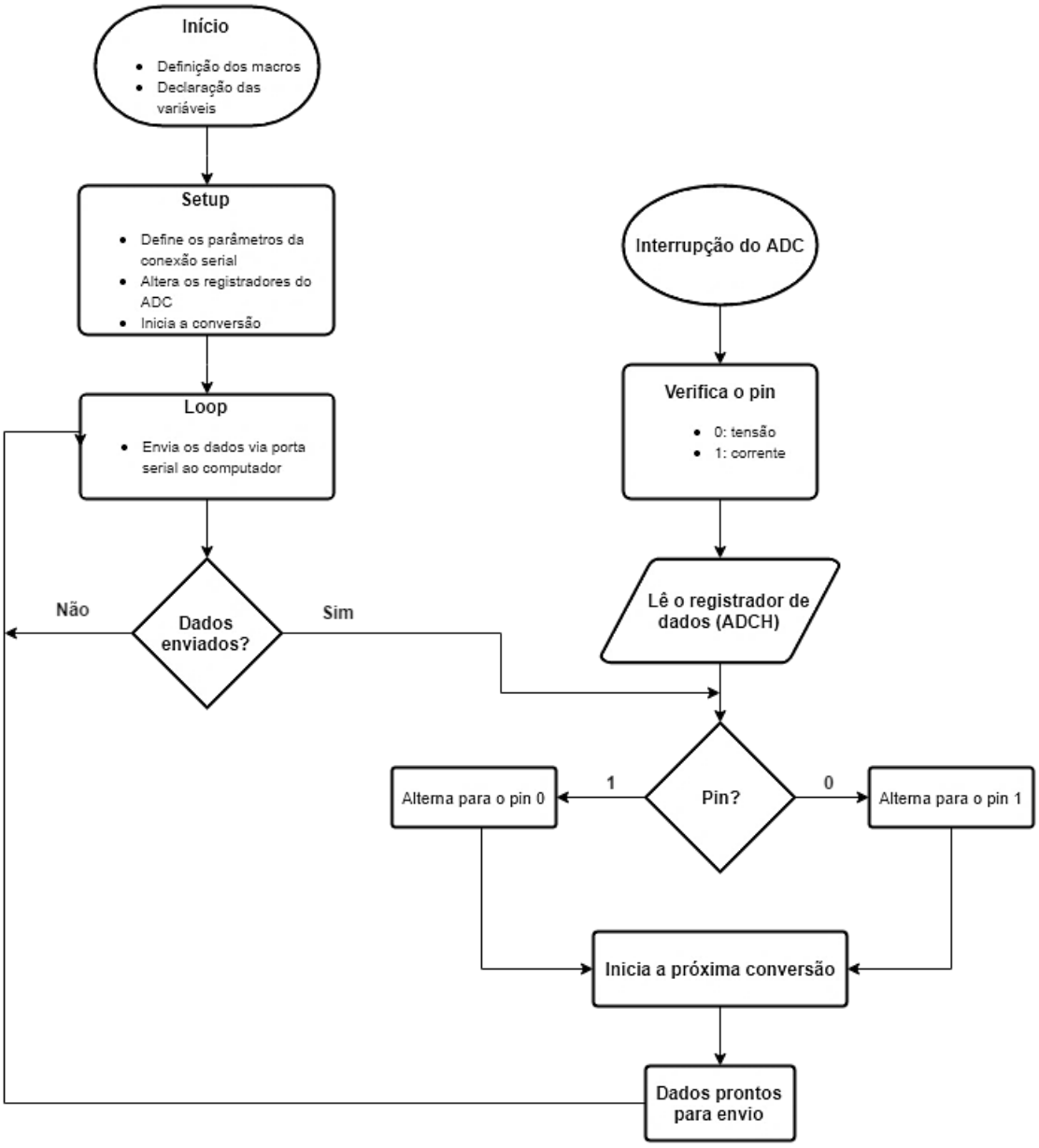
assim é possível amostrar dados a uma taxa mais elevada do que com as funções padrão criadas para o Arduino, mesmo que de certa forma a comunicação serial limite um pouco o sistema. Além do que a leitura dos dados diretamente do registrador elimina o delay que se tem ao convertar os dados da forma binária para ASCII antes do envio.

As variáveis e macros devem ser declarados antes do setup. Os macros aqui permitem a manipulação de bits individuais dos registradores e as variáveis são de um tipo especial (*volatile*), pois não são manipuladas pelo looping principal, mas sim dentro da interrupção.

A Figura 32 apresenta o fluxograma do algoritmo de aquisição de sinal descrito.

O código completo e comentado se encontra no APÊNDICE A.

Figura 32. Código de aquisição de sinal no Arduino.



Fonte: próprio autor.

3.6 *Script* de recebimento dos dados no *software* Matlab

O interfaceamento entre os sensores e o computador é realizado via porta USB-serial, o algoritmo no Arduino envia os dados em forma binária para a porta serial e no computador é necessário que haja um outro código de recepção e interpretação desses dados. Aqui, este código foi desenvolvido em *software* Matlab, dado o suporte existente nesta plataforma e de forma a facilitar o posterior processamento dos dados.

O script amostra 10000 pontos, um ponto a cada vez que o número de bytes especificado em `s.InputBufferSize` chega ao *buffer*, 1 byte=8bits, no caso. Como são duas portas analógicas diferentes, têm-se 5000 pontos para cada uma. Os dados, durante a leitura, ficam todos armazenados na matriz `Data`.

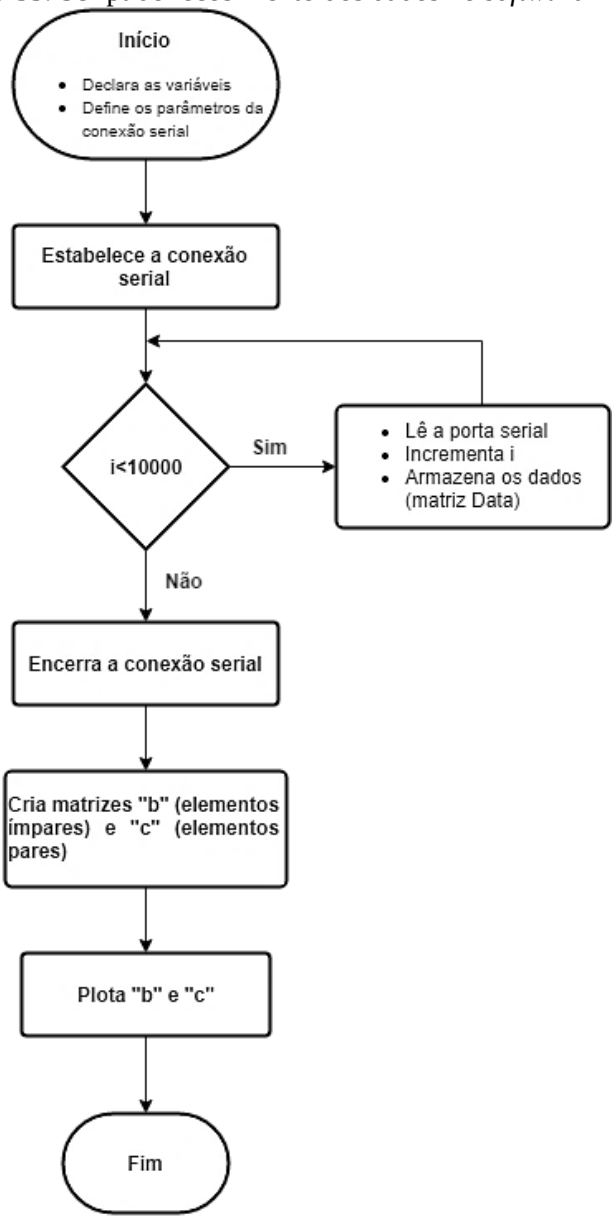
Por se estar trabalhando diretamente com o número máximo de bits do *buffer* como dados, não há margem para criar uma forma de identificação de cada um que chega, ou seja, não é possível identificar se o valor é de tensão (Pin A0) ou corrente (Pin A1) em dado instante, parte porque o IDE não permite uma configuração da porta serial de maneira mais granular, como estabelecer *stop bits* e paridade diferente para cada pin. Entretanto o fato de que eles chegam de maneira alternada, corrente-tensão-corrente ou tensão-corrente-tensão, torna possível ao menos separar os dados do mesmo tipo, e é isso que ocorre nas matrizes *b* e *c*, os dados das linhas ímpares e pares são separados, respectivamente, de forma que se tenha em cada matriz apenas os dados de tensão ou apenas os de corrente.

Por fim, o `plot` das duas matrizes é o que permite se identificar qual tipo de dado se encontra armazenado em cada uma, embora não seja um método eficiente e um tanto quanto manual, para a quantidade de dados com que se estava trabalhando, era viável.

A Figura 33 apresenta o fluxograma do algoritmo de aquisição de sinal descrito.

O código completo e comentado se encontra no APÊNDICE B.

Figura 33. Script de recebimento dos dados no *software* Matlab.



Fonte: próprio autor.

3.7 Processamento dos dados

Os sinais coletados de tensão e corrente para cada par de lâmpadas foram armazenados em matrizes individuais 5000x1, 10 para o sinal de corrente e 10 para o sinal de tensão para cada tipo de lâmpada e para o cenário com todas desligadas, totalizando 80 matrizes, de forma que ficasses mais organizados e o processo de tratamento dos dados fosse facilitado.

Primeiramente foram obtidas as harmônicas presentes em cada sinal de corrente para as 10 amostras de cada par de lâmpadas, além do sistema com todas desligadas, via FFT no *software* Matlab. Os dados das harmônicas de 3ª, 5ª, 7ª e 9ª ordem, além da fundamental, foram

utilizados para montar o vetor de entrada da RNA, uma vez que são as que mais apresentaram variação.

Por fim, a foi empregada a RNA existente na *Neural Network Pattern Recognition Tool* do *software* Matlab para classificação dos dados. A entrada da rede é uma matriz 40x5 em que cada coluna corresponde a uma ordem de harmônicas, com 40 amostras totais em cada coluna. A saída, uma matriz 40x4 indica a classe à qual pertence cada grupo de 10 amostras: classe 1, LEDs; classe 2, fluorescentes compactas; classe 3, incandescentes; classe 4, lâmpadas desligadas.

3.7.1 *Fast Fourier Transform* (FFT)

A DFT (*Discrete Fourier Transform*) é uma fórmula matemática que relaciona um sinal amostrado no tempo ou espaço, ao domínio da frequência. Para um vetor x que tem contém n pontos amostrados, a equação 19 expressa a DFT de x.

$$y_{k+1} = \sum_{j=0}^{n-1} \omega^{jk} x_{j+1} \quad (19)$$

Na qual, i é a unidade imaginária, $\omega = e^{-2\pi i/n}$ é uma das n raízes complexas da unidade, e j e k são índices que variam de 0 à n-1.

Para o cálculo de cada elemento de y, um número de n^2 operações de ponto flutuante é necessário, sendo computacionalmente custosa.

Neste contexto entra a FFT, um algoritmo para o cálculo da DFT de maneira mais eficiente. Ao usar FFT ao invés de DFT a complexidade computacional se reduz de n^2 para $n.\log n$. E há ainda algumas implementações especializadas de FFT, como as que tem um aumento de eficiência quando n é uma potência de 2 (MathWorks, 2016).

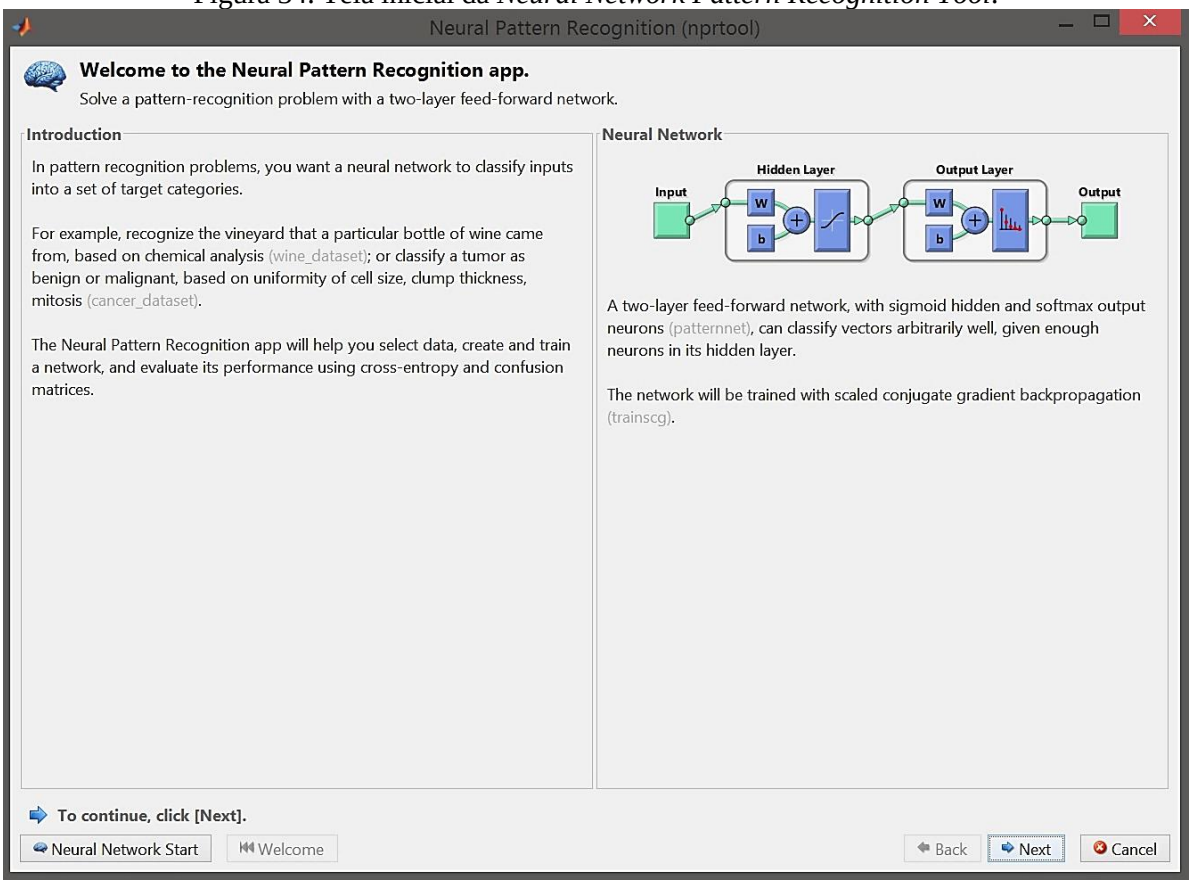
A FFT pode ser, portanto, empregada para análise do conteúdo harmônico de um sinal senoidal, separando-as em par, ímpar e interhamônicas, resultando em suas magnitudes e fases.

A função `fft` no *software* Matlab retorna o resultado da transformada rápida de Fourier em 1 dimensão para um conjunto de dados e foi empregada para a obtenção das harmônicas do sinal amostrado, conforme script descrito no APÊNDICE C.

3.7.2 *Neural Network Pattern Recognition Tool*

A *Neural Network Pattern Recognition Tool* é uma aplicação interna do *software* Matlab especializada em empregar uma rede neural para classificar um conjunto de dados de entrada em um grupo de categorias. A arquitetura da rede é *feedforward* com uma camada escondida com função de ativação sigmóide e outra de saída com função de ativação softmax. A rede é treinada com um algoritmo backpropagation de gradiente conjugado escalonado (Figura 34).

Figura 34. Tela inicial da *Neural Network Pattern Recognition Tool*.



Fonte: próprio autor.

Os dados de entrada da rede foram organizados na forma de uma matriz 40x5, onde cada linha representa o conjunto das cinco primeiras harmônicas ímpares (colunas) para cada medição, e cada conjunto de 10 linhas corresponde a um tipo de lâmpada, exceto as 10 primeiras que são o conjunto de dados em que todas estão desligadas. O *target*, ou seja, as classes em que se deseja que os dados sejam classificados é uma matriz 40x4, onde cada coluna diz respeito a uma das classes: lâmpadas desligadas, lâmpadas incandescentes ligadas, lâmpadas fluorescentes ligadas e lâmpadas LED ligadas. Nessa matriz todas as células são zeradas, exceto

às que representam a classe para a qual o conjunto de dados pertence, estas recebem 1. A Figura 35 mostra parte das matrizes de entrada e saída construídas.

Figura 35. Input e output da RNA.

40x5 double

	1	2	3	4	5	6
1	0.8728	0.6688	0.5595	0.5197	0.2975	
2	0.3935	0.3533	0.2697	0.1668	0.0863	
3	0.8027	0.5070	0.4680	0.6343	0.4603	
4	0.2393	0.2196	0.1912	0.1444	0.1059	
5	0.7525	0.7149	0.6141	0.4392	0.2738	
6	0.6316	0.5634	0.4053	0.2302	0.1328	
7	0.7374	0.6539	0.4887	0.3093	0.1634	
8	0.7490	0.7044	0.5946	0.4281	0.2568	
9	0.4326	0.2727	0.0689	0.1720	0.3461	
10	0.3756	0.2269	0.0561	0.2726	0.3743	
11	6.4511	0.4340	0.6055	0.2499	0.1888	
12	6.5964	0.1838	0.3982	0.0884	0.0435	
13	6.0560	0.4972	0.7589	0.4077	0.2661	
14	6.4769	0.3428	0.3431	0.1590	0.2755	
15	6.3181	0.5931	0.6653	0.4835	0.4171	
16	6.8977	0.5845	0.7197	0.3396	0.1727	
17	6.8697	0.8221	0.2685	0.2101	0.3504	
18	6.8116	0.2748	0.2541	0.3756	0.4947	
19	6.1810	0.4666	0.5863	0.5180	0.3890	
20	6.1255	0.5009	0.4818	0.2543	0.1694	

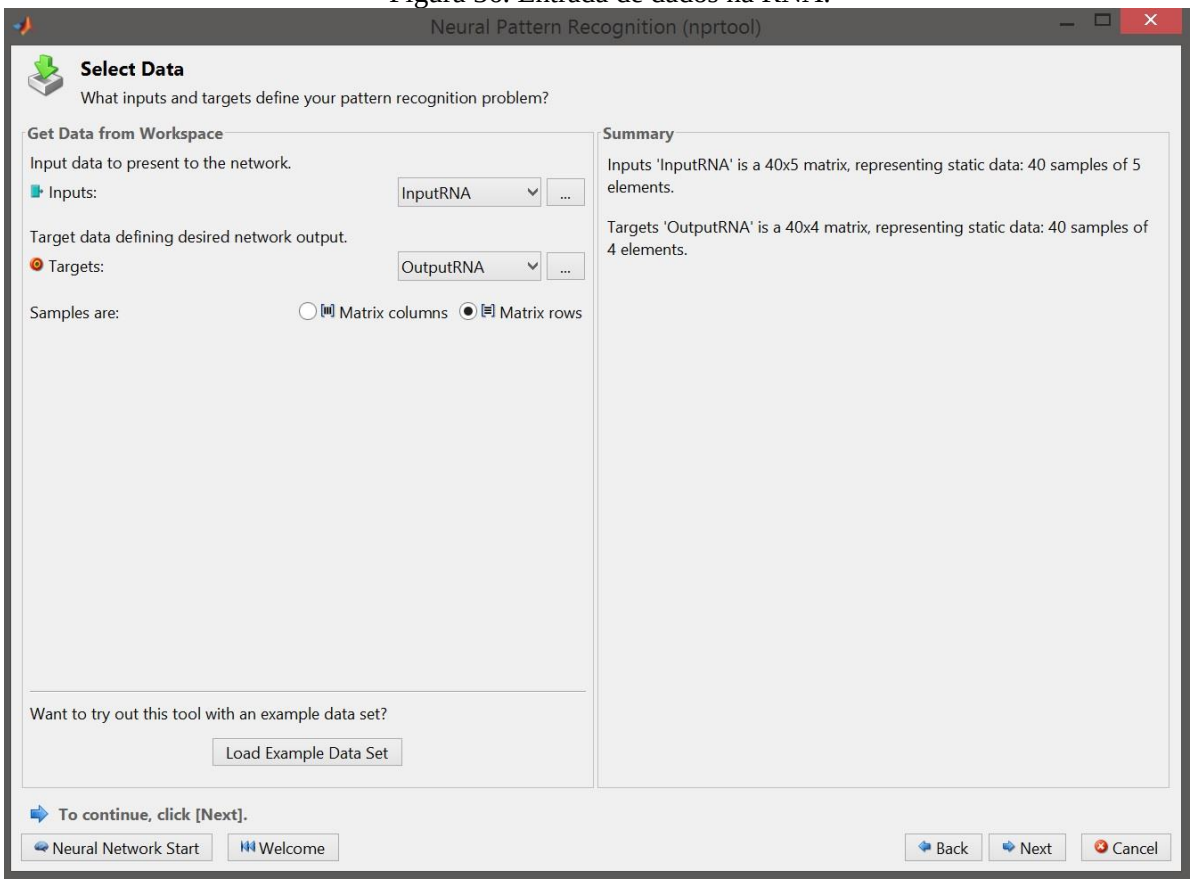
40x4 double

	1	2	3	4	5
1	0	0	0	1	
2	0	0	0	1	
3	0	0	0	1	
4	0	0	0	1	
5	0	0	0	1	
6	0	0	0	1	
7	0	0	0	1	
8	0	0	0	1	
9	0	0	0	1	
10	0	0	0	1	
11	0	0	1	0	
12	0	0	1	0	
13	0	0	1	0	
14	0	0	1	0	
15	0	0	1	0	
16	0	0	1	0	
17	0	0	1	0	
18	0	0	1	0	
19	0	0	1	0	
20	0	0	1	0	

Fonte: próprio autor.

A classe 1 corresponde às lâmpadas LEDs, a 2 às fluorescentes, a 3 às incandescentes e por fim a classe 4 são as lâmpadas desligadas. A Figura 36 mostra o painel de entrada dos dados.

Figura 36. Entrada de dados na RNA.



Fonte: próprio autor.

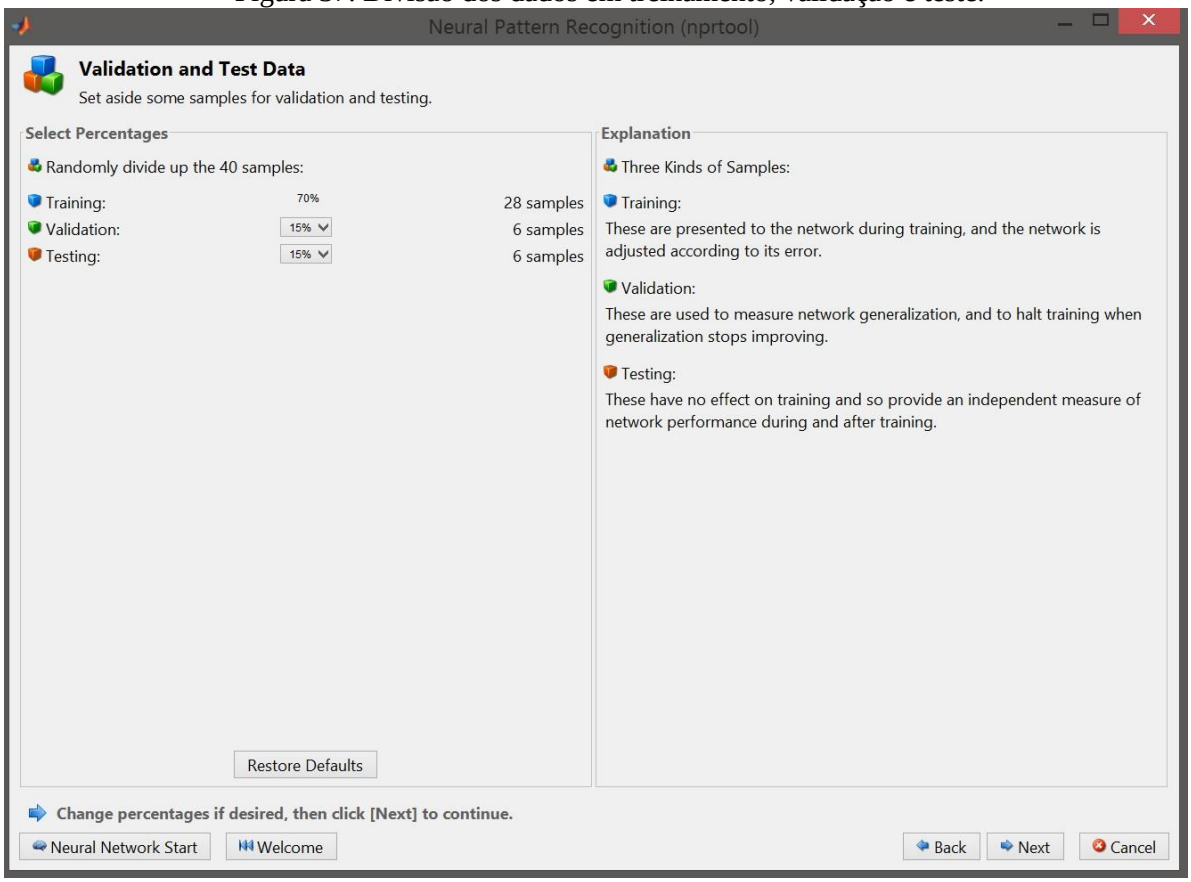
A forma com que os dados estão organizados é muito importante, sendo possível selecionar se as amostras são colunas ou linhas das matrizes de dados, caso haja divergência com relação a essa característica não é possível prosseguir, mas um aviso aparece dizendo que as dimensões não estão corretas.

Uma vez que os dados são apresentados deve-se dividi-los em três grupos: treinamento, validação e teste. Os dados de treinamento são empregados para o cálculo do gradiente e atualização dos pesos e bias da rede. O segundo grupo, o de validação, é utilizado para medir o grau de generalização da rede, seu erro é monitorado durante todo o processo de treinamento e durante a fase inicial normalmente decai. Entretanto, caso a rede esteja fazendo *overfitting* dos dados, este erro torna a crescer, significando que a rede memorizou os exemplos de treinamento, mas não aprendeu a generalizar para novas situações. E por fim, o grupo de testes corresponde a um conjunto de dados que não foi empregado no treinamento, mas é usado para se comparar os diferentes modelos (MathWorks, 2016).

A aplicação permite que se escolha qual a porcentagem de dados pertence a cada um desses conjuntos, sendo possível variar as porcentagens de 5 % a 35 %, ou seja, é possível

tornar o treinamento tão pequeno quanto 30 %, ou tão grande quanto 90 % (Figura 37). Por padrão a divisão dos dados é feita de maneira aleatória.

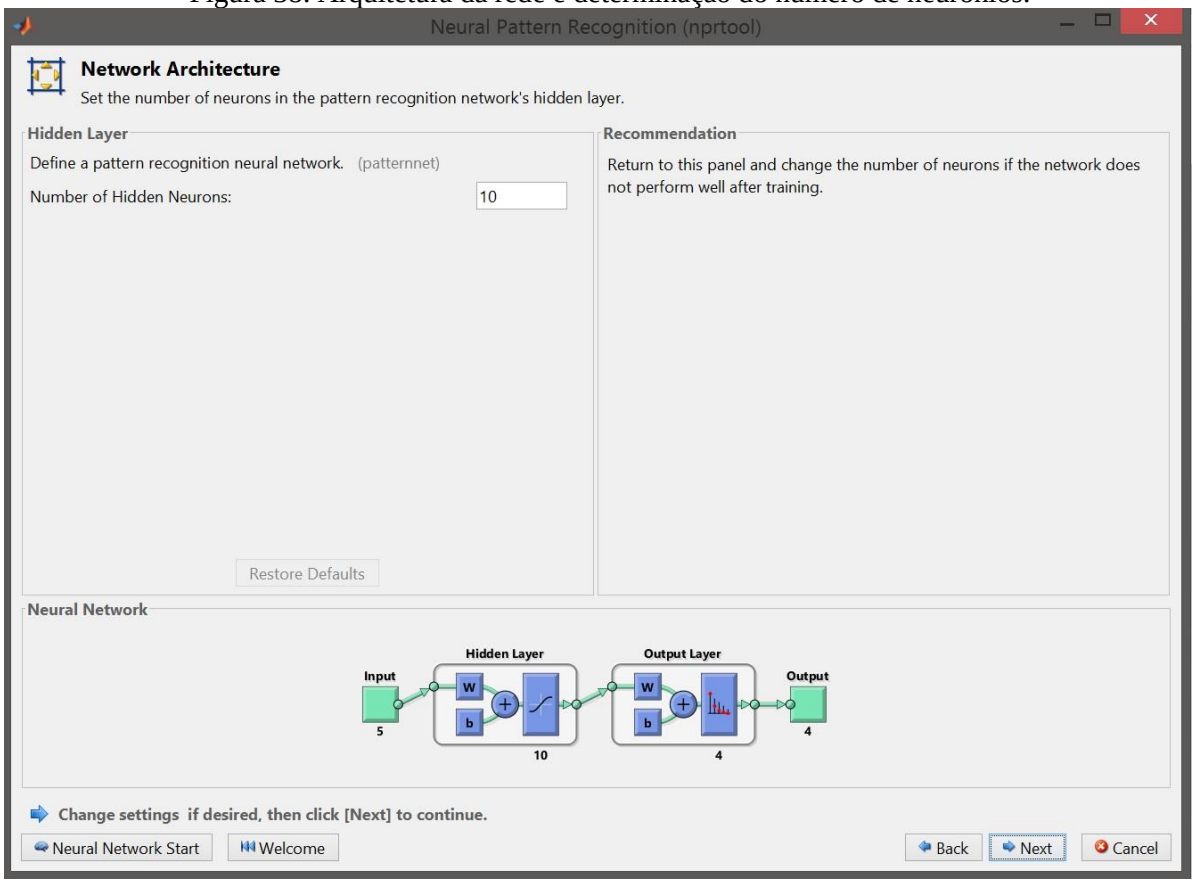
Figura 37. Divisão dos dados em treinamento, validação e teste.



Fonte: próprio autor.

A quantidade de neurônios da camada escondida pode ser ajustada de forma que se adeque ao problema a ser resolvido (Figura 38), caso o treinamento corra bem, mas a performance no grupo de teste tenha sido significantemente ruim, reduzir o número de neurônios pode ser necessário, visto que a rede está fazendo *overfitting*.

Figura 38. Arquitetura da rede e determinação do número de neurônios.

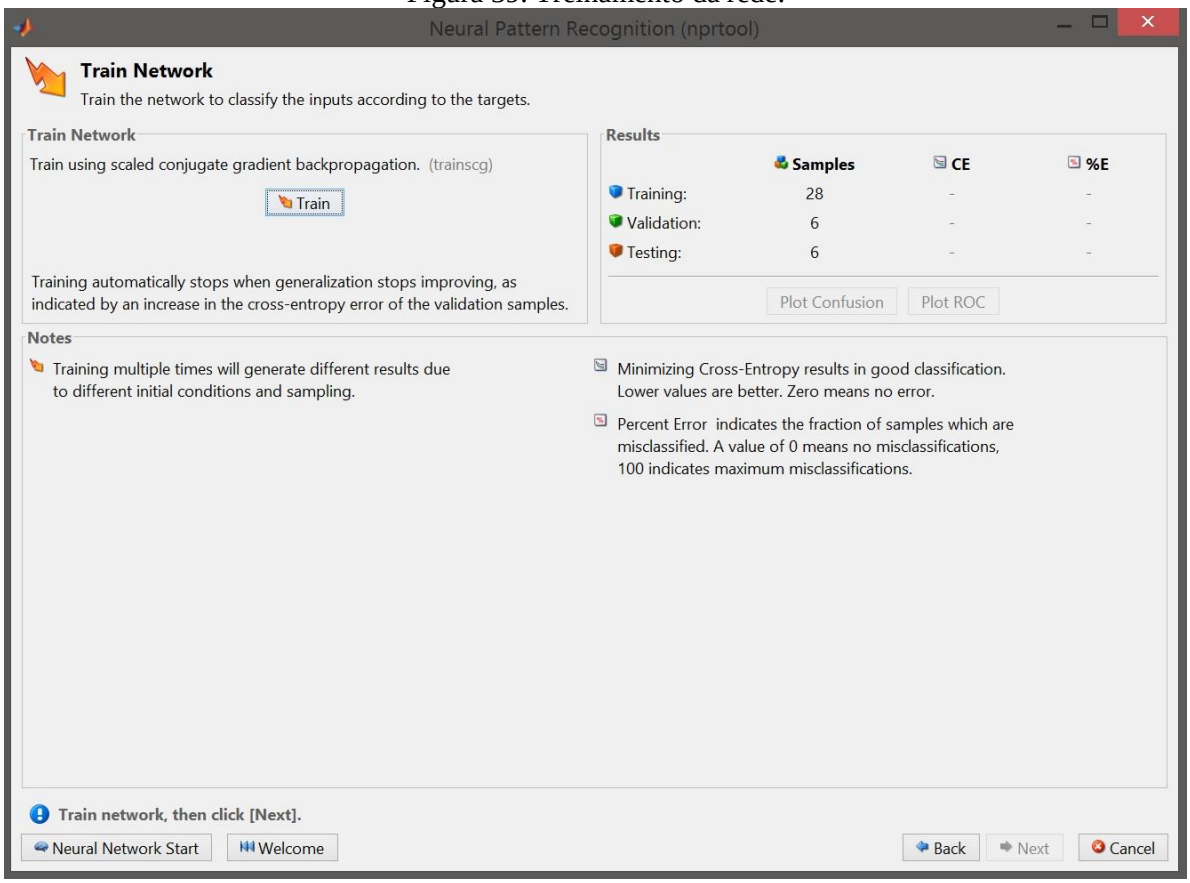


Fonte: próprio autor.

Uma vez que todos os parâmetros estejam determinados parte-se para o treinamento da rede (Figura 39). O algoritmo de treinamento empregado aqui é o *backpropagation* de gradiente conjugado escalonado (SCG). Este algoritmo atualiza os pesos e bias da rede de acordo com o método do gradiente conjugado escalonado, ou seja, ele é capaz de treinar a rede desde que os pesos, entrada e funções de transferência tenham funções derivadas. Métodos deste tipo são técnicas de segunda ordem que permitem minimizar funções com muitas variáveis. O uso de derivadas de segunda ordem, no geral, permite obter o mínimo de uma função de forma melhor do que os de primeira ordem, embora exijam maior poder computacional.

Assim como no algoritmo de *backpropagation* padrão, o método SCG também busca se aproximar do mínimo de maneira iterativa, entretanto enquanto o primeiro se desloca para baixo no gradiente de erro da função, o SCG se move na direção que é o conjugado das direções dos passos anteriores, sendo assim o passo seguinte não desfaz o anterior, sendo consideravelmente mais rápido que os demais (MACHE, 1995).

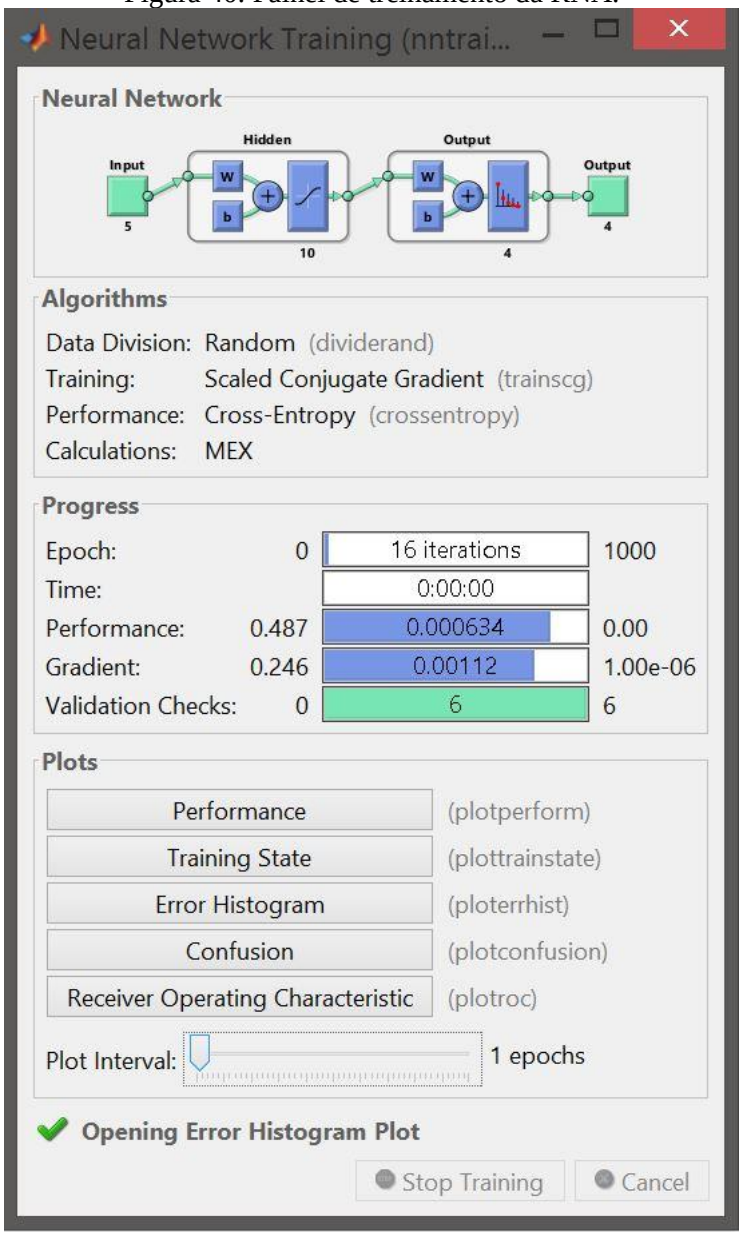
Figura 39. Treinamento da rede.



Fonte: próprio autor.

Uma vez treinada a rede (Figura 40), é possível se verificar o seu desempenho, via gráfico de performance, histograma de erros, painel de confusão ou ROC (*Receiver Operating Caracteristic*).

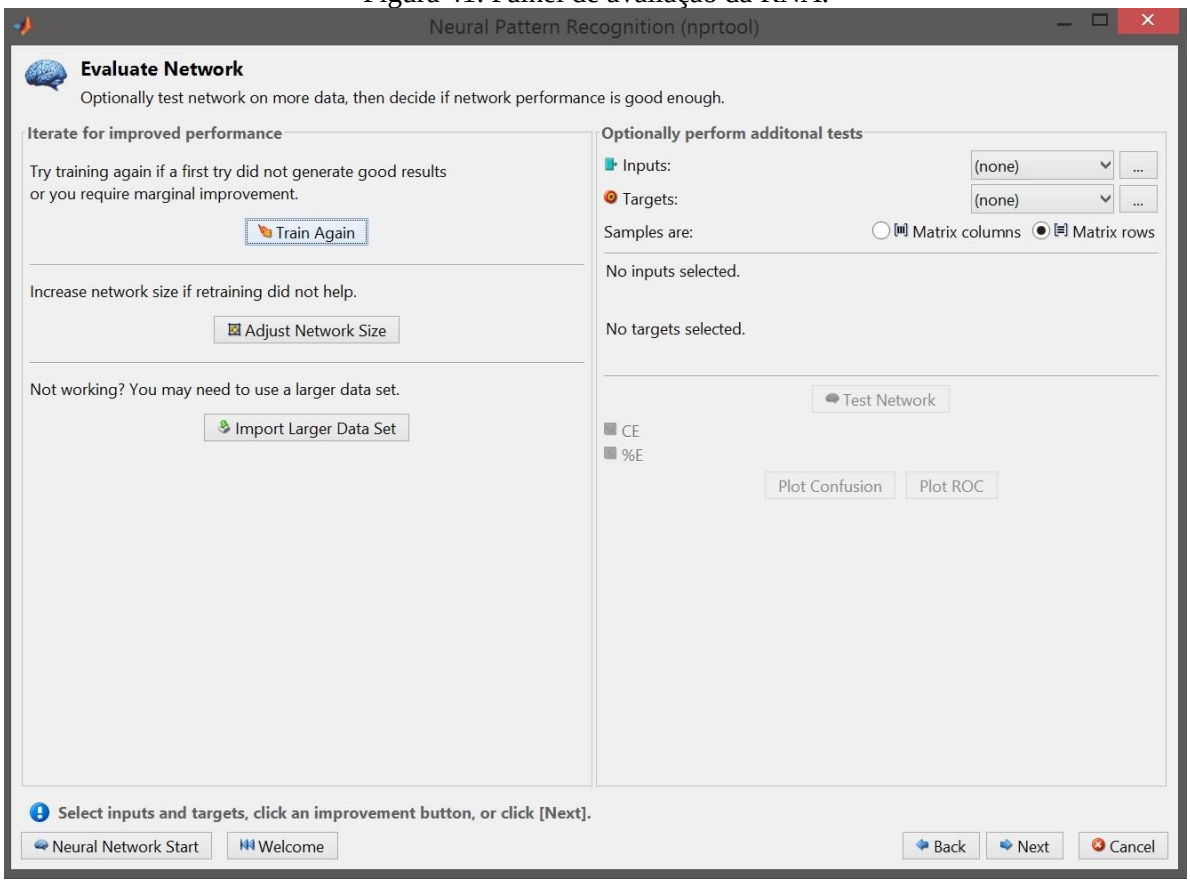
Figura 40. Painel de treinamento da RNA.



Fonte: próprio autor.

Após a avaliação é possível se alterar os parâmetros previamente definidos, sendo que a rede pode ser treinada novamente, o número de neurônios ajustado ou um universo maior de dados inserido. Caso os resultados tenham sido satisfatórios, mas se deseja testar a rede em um conjunto independente de dados, também é possível (Figura 41).

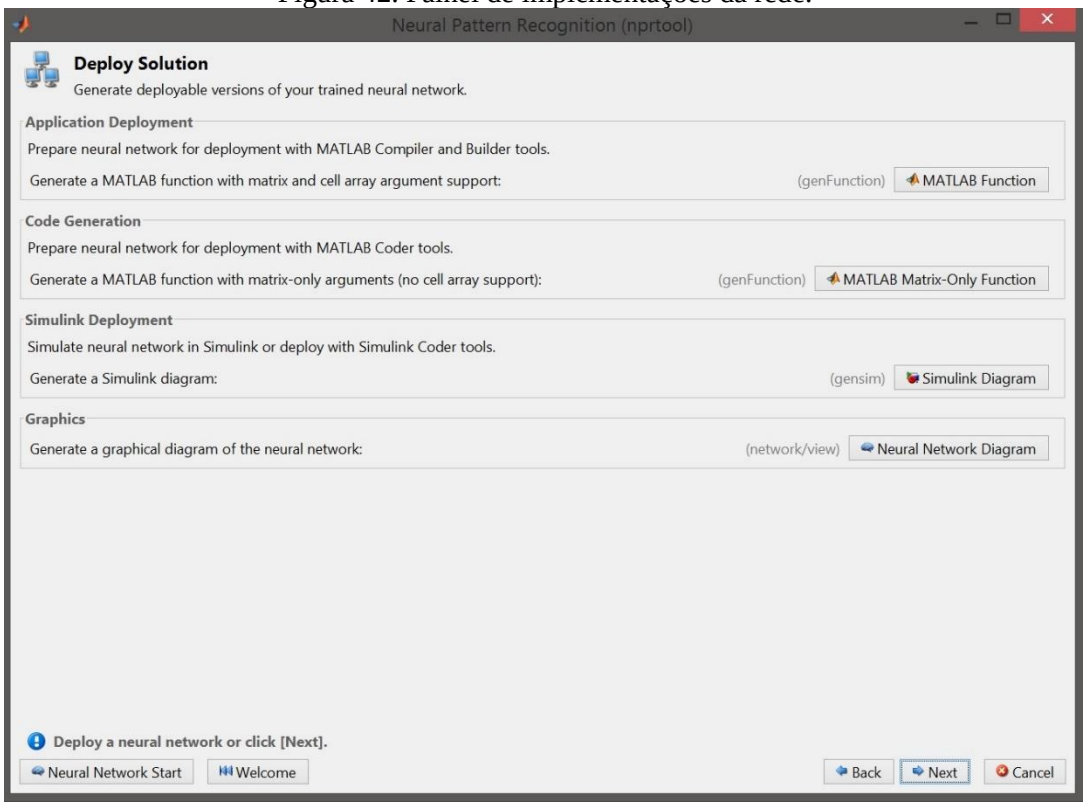
Figura 41. Painel de avaliação da RNA.



Fonte: próprio autor.

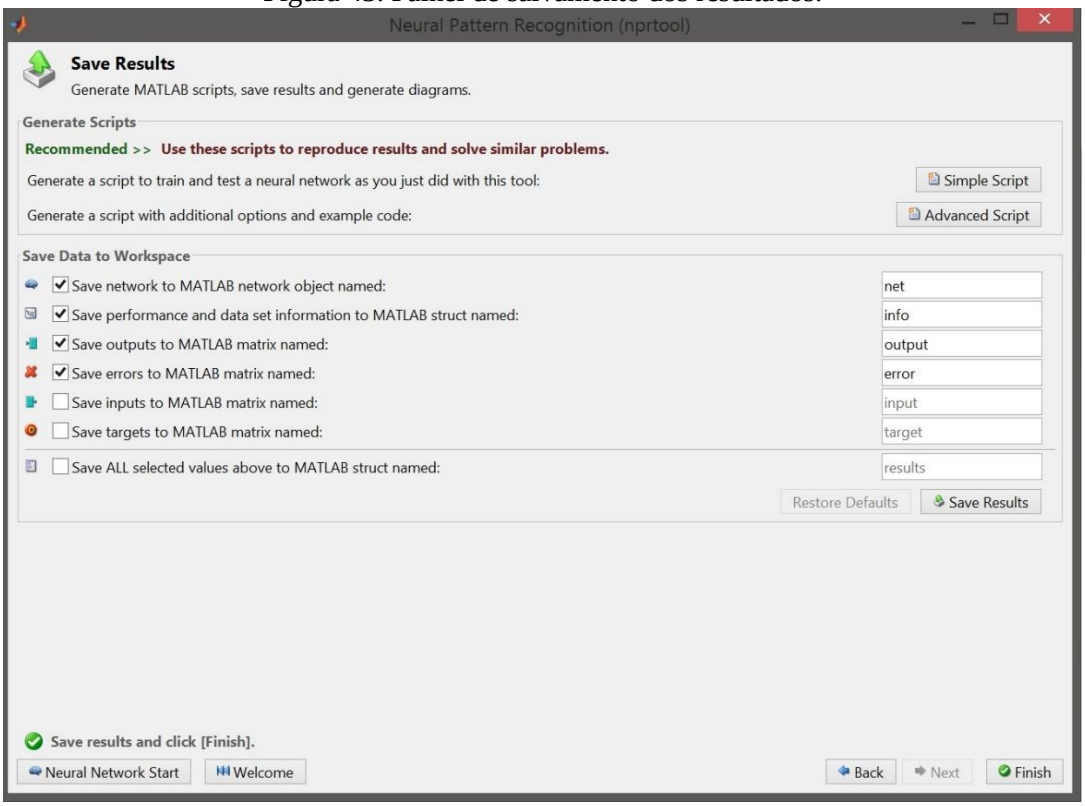
Uma vez que a rede atingiu os objetivos especificados se pode gerar versões implementadas como função do *software* Matlab ou diagrama do Simulink para a rede já treinada (Figura 42). Por fim é possível salvar os dados gerados, como as saídas, entradas, erros, etc e gerar scripts (simples ou avançado) de forma que se possa reproduzir o que foi feito acima via linhas de comando, dando uma maior liberdade na manipulação dos parâmetros da rede (Figura 43).

Figura 42. Painel de implementações da rede.



Fonte: próprio autor.

Figura 43. Painel de salvamento dos resultados.



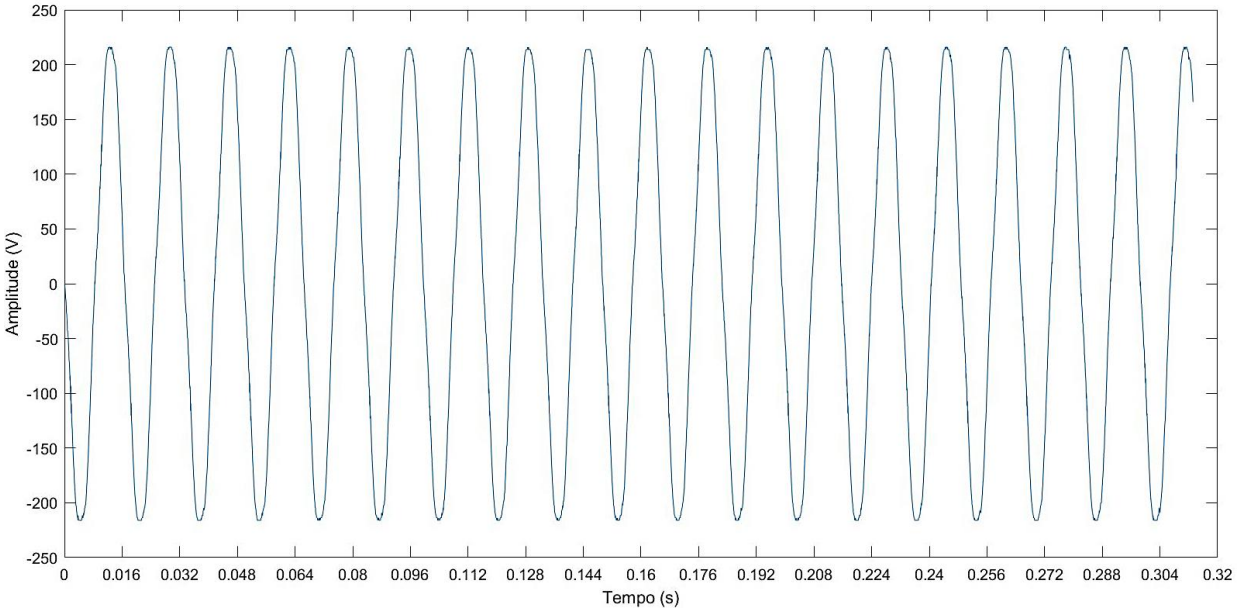
Fonte: próprio autor.

4 RESULTADOS E DISCUSSÕES

Em primeira instância, mostrar-se-á o desempenho do sistema de aquisição de dados em ser capaz de reconstruir o sinal corretamente, além de apresentar a característica das harmônicas presentes em cada lâmpada.

Com os dados aquisitados foi possível obter a forma de onda da tensão reamostrada no tempo (Figura 44). As 5000 amostras tornaram possível reconstruir 0,32 s do sinal, com aproximadamente 261 amostras por ciclo, mais do que o dobro do mínimo necessário para a digitalização (120 amostras), mostrando que de fato é possível empregar o Arduino como um sistema de aquisição de dados.

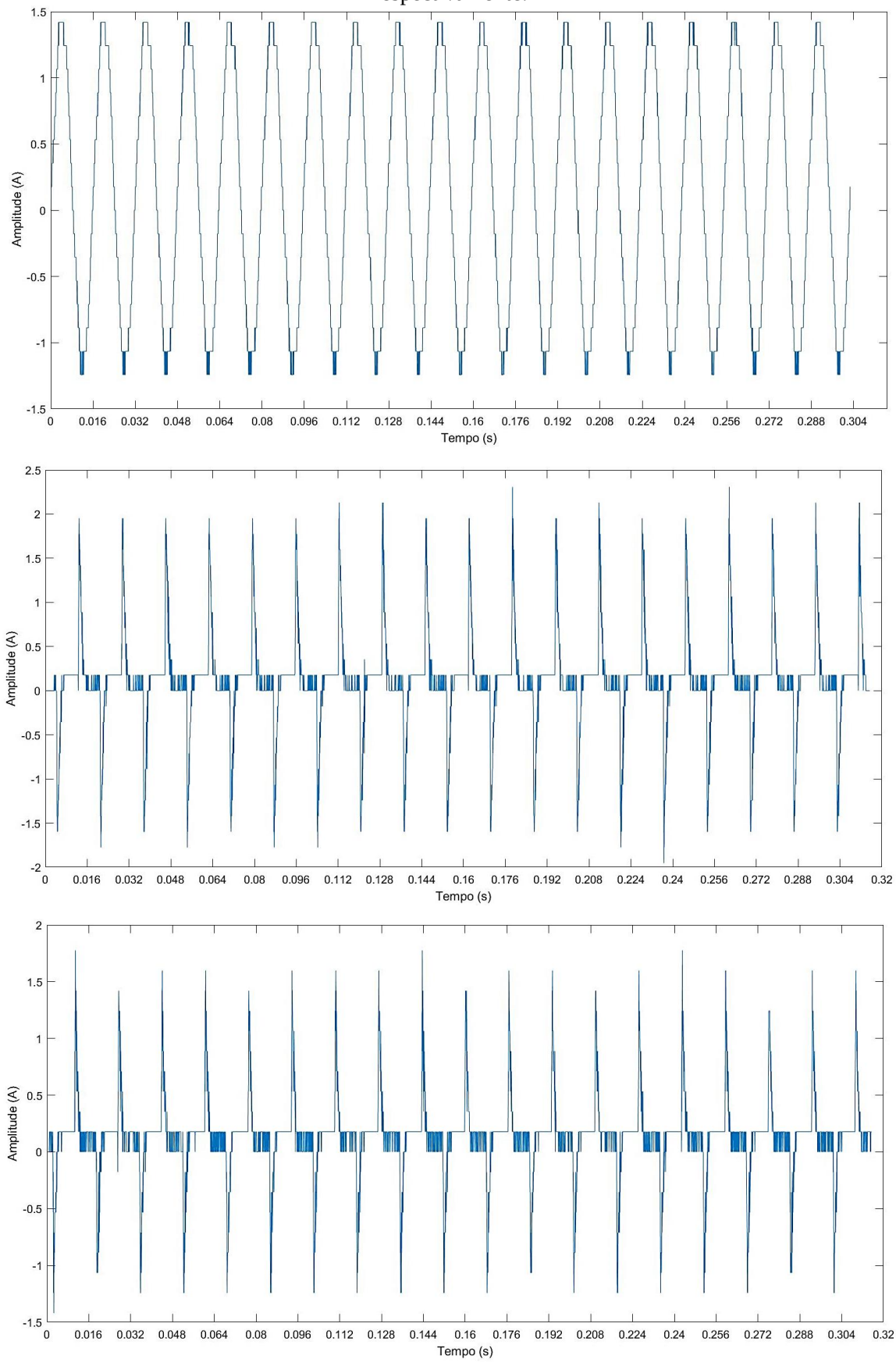
Figura 44. Forma de onda característica da tensão.



Fonte: próprio autor.

Esse resultado também pode ser conferido para as formas de onda da corrente de cada uma das lâmpadas utilizadas nos testes. O que essas curvas mostram é que mesmo com uma resolução limitada do ADC, de 8 bits, a característica de forma da onda não foi comprometida, sendo condizente com o que é esperado de cada carga (Figura 45).

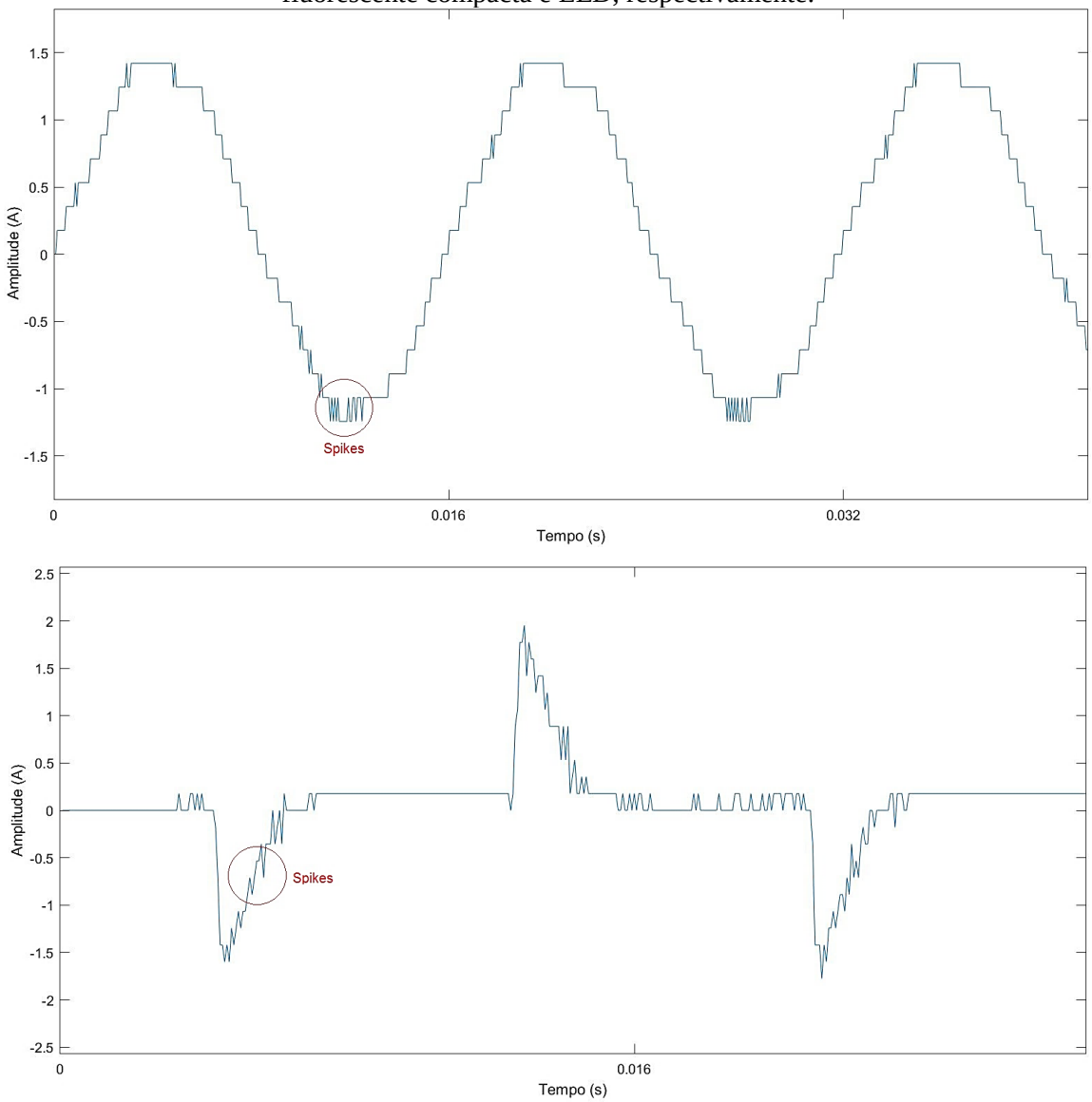
Figura 45. Forma de onda da corrente de uma lâmpada incandescente, fluorescente compacta e LED, respectivamente.

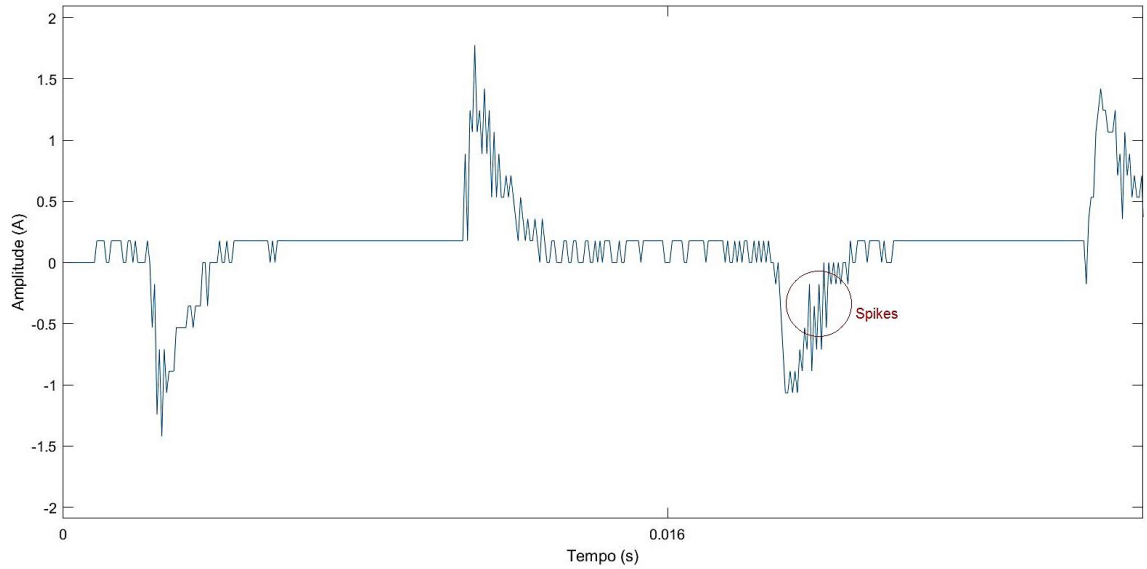


Fonte: próprio autor.

Por outro lado, ao se observar um ciclo individualmente fica claro o efeito do número reduzido de bits (Figura 46). Quanto menos níveis de quantização se tem, mais em forma de escada a curva se torna, visto que pequenas variações não são detectadas e acabam classificadas dentro do mesmo nível.

Figura 46. Efeitos da quantização na forma de onda da corrente de uma lâmpada incandescente, fluorescente compacta e LED, respectivamente.



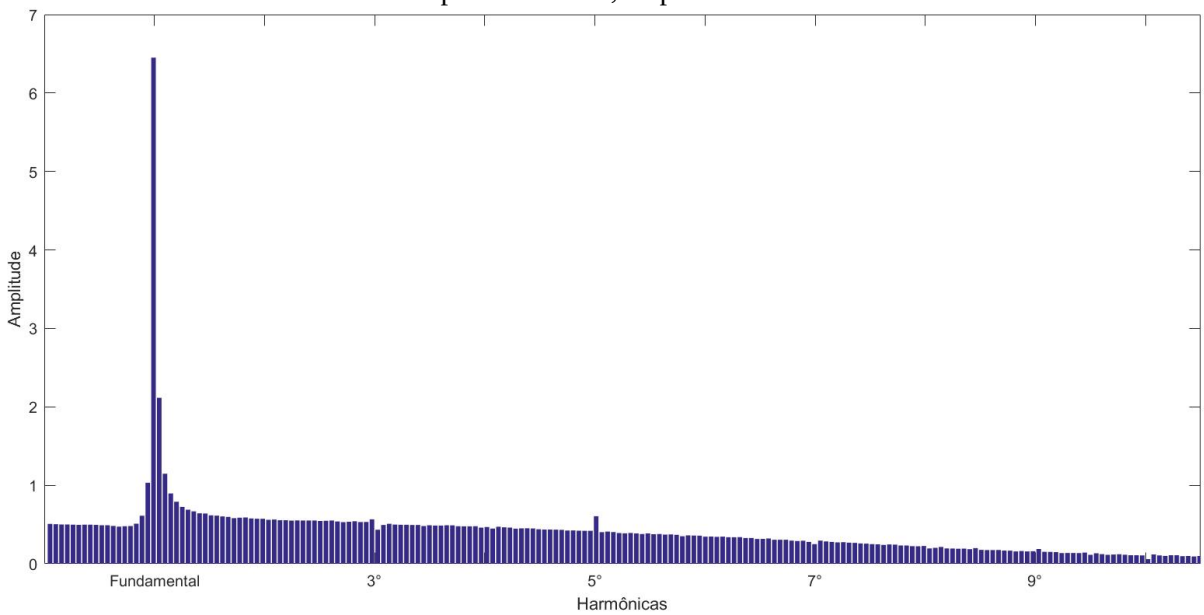


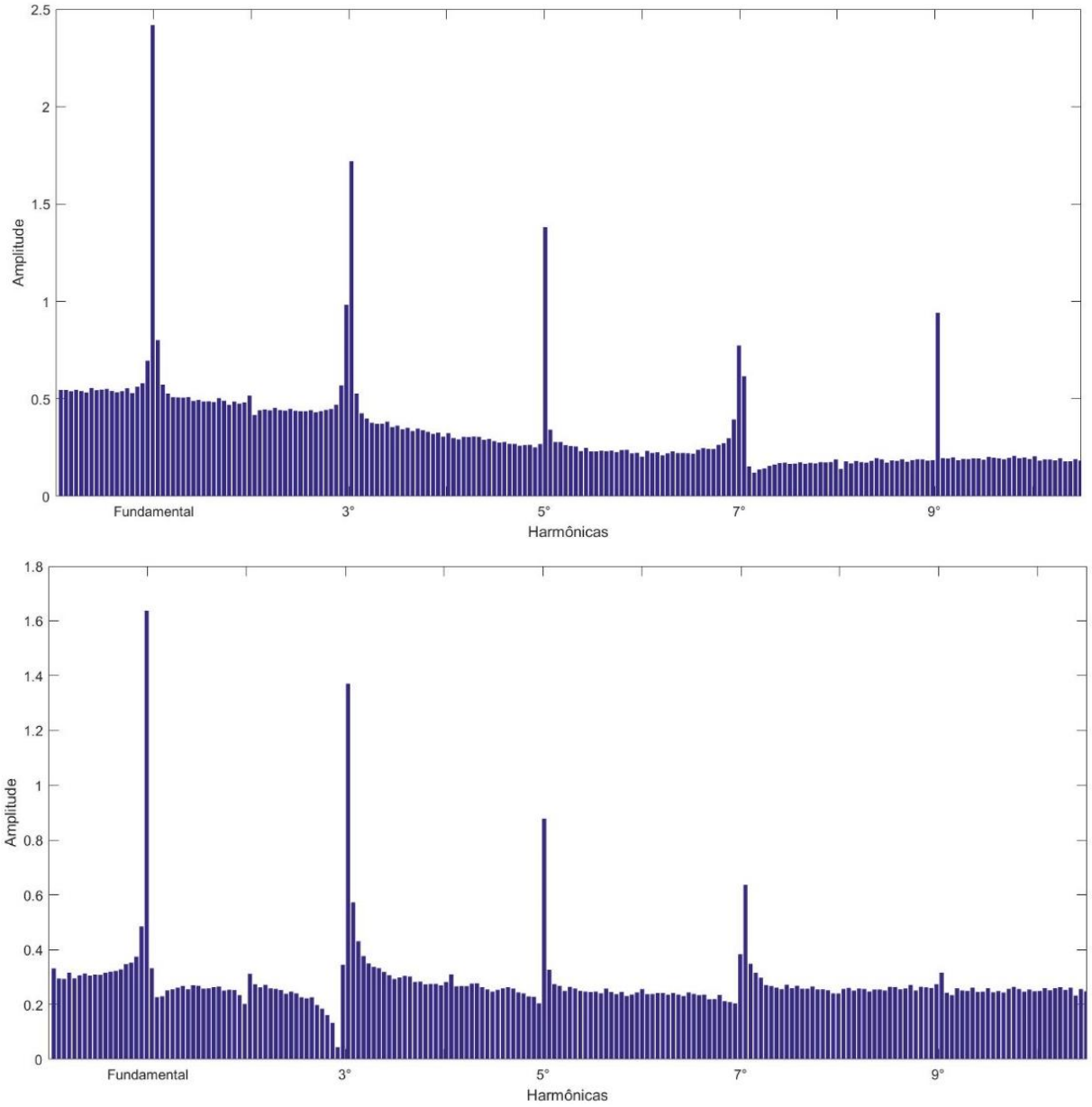
Fonte: próprio autor.

É interessante notar ainda aqui os erros de precisão presentes ao se operar o ADC em uma frequência elevada, que se assemelham à *spikes* no sinal.

Os dados obtidos foram submetidos ao algoritmo de FFT e os resultados plotados na Figura 47, abaixo.

Figura 47. Harmônicas presentes no sinal de corrente das lâmpadas incandescentes, fluorescentes compactas e LEDs, respectivamente.



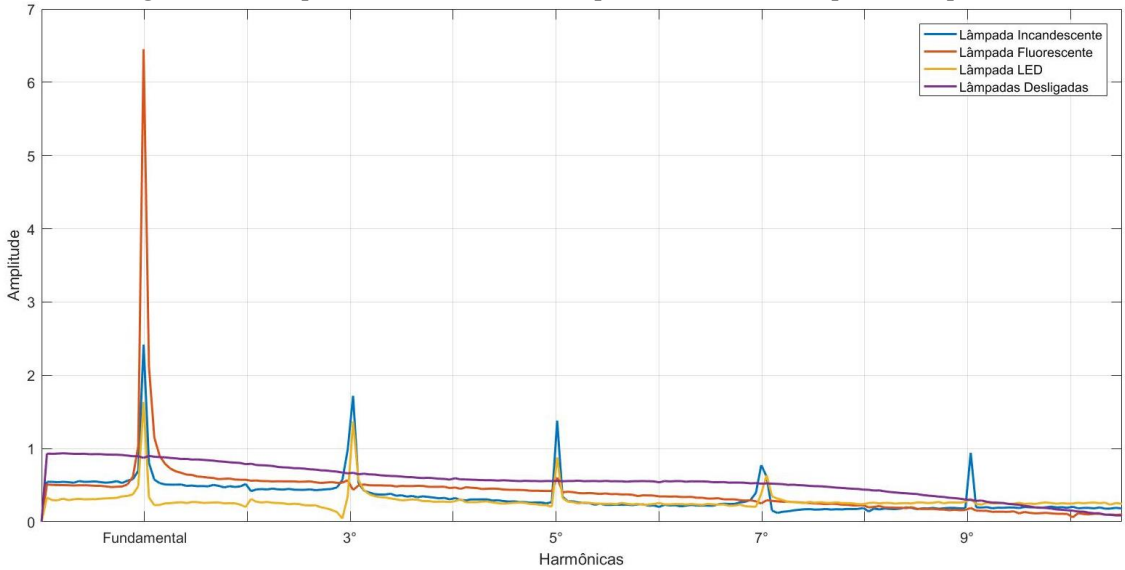


Fonte: próprio autor.

Como é possível observar o conteúdo harmônico presente nos diferentes tipos de lâmpadas é bastante distinto entre si com relação às cinco primeiras harmônicas, razão pela qual esta assinatura de carga foi escolhida como dado de entrada da RNA. Nas incandescentes há apenas a componente fundamental, dada sua natureza resistiva e nas fluorescentes compactas e LEDs há ainda harmônicas de 3ª, 5ª, 7ª e 9ª ordem, já que possuem componentes eletrônicos.

A Figura 48 traz um comparativo entre elas, onde a diferença se torna mais visível ainda, sendo possível observar como diferem não só com relação a quais harmônicas estão presentes, mas também em como a magnitude varia, corroborando o seu uso para caracterização da carga e consequentemente possibilitando a classificação destas.

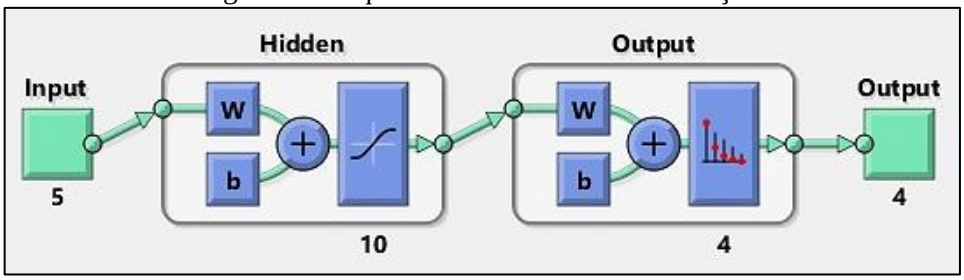
Figura 48. Comparativo das harmônicas presentes em cada tipo de lâmpada.



Fonte: próprio autor.

Todas as RNAs empregadas neste trabalho foram desenvolvidas com o auxílio da *Neural Network Pattern Recognition Tool* no *software* Matlab, como exposto anteriormente (Figura 49). Para testar a capacidade desta rede em classificar as harmônicas das diferentes lâmpadas, considerando o conjunto limitado de dados, apenas 40 amostras, 10 por classe, testou-se a rede com 10 neurônios em sua camada escondida e variação nas porcentagens das amostras de treinamento, validação e teste. Os cenários avaliados foram: 60 % das amostras para treinamento, 20 % para validação e 20 % para teste; 70 % das amostras para treinamento, 15 % para validação e 15 % para testes e 80 % das amostras para treinamento, 10 % para validação e 10 % para teste. As redes foram treinadas apenas uma vez para cada conjunto de dados, logo a que obteve a melhor generalização foi considerada ótima.

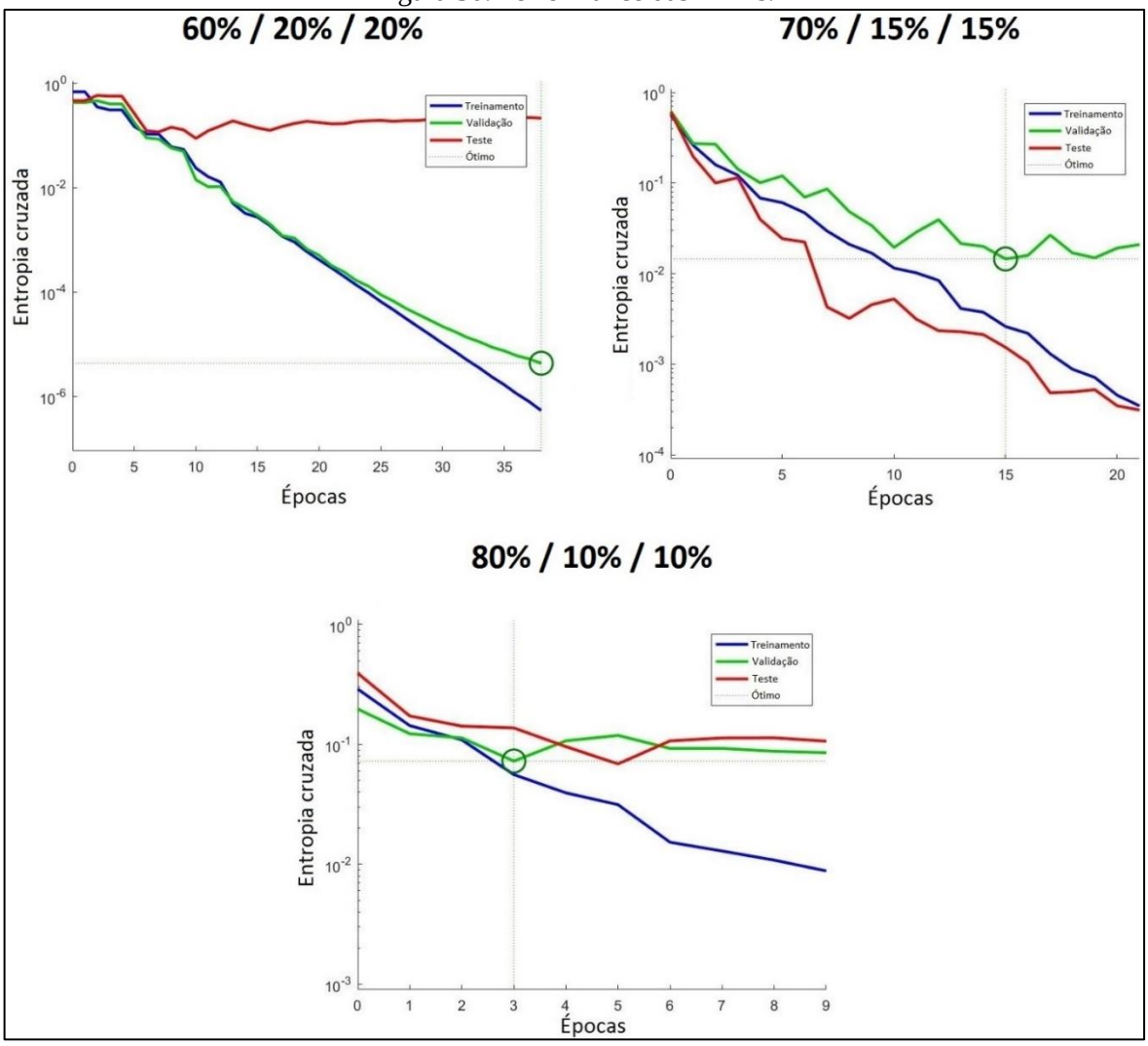
Figura 49. Arquitetura da RNA de classificação.



Fonte: próprio autor.

A Figura 50 apresenta a performance das RNAs testadas, que corresponde ao gráfico dos erros para cada um dos grupos (treinamento, validação e teste) e o comportamento está dentro do esperado. Com o tempo o erro diminuiu, consequentemente a curva do treinamento é descendente em todos os casos e o erro final é pequeno. Entretanto, apenas o terceiro cenário apresenta um comportamento geral dentro do esperado: treinamento descendente com o passar do tempo (erro diminuindo), e curva de validação e teste bastante próximas, o que indica uma boa capacidade da rede em generalizar os dados. Para o primeiro cenário, o perfil da curva de teste indica que pode estar havendo *overfitting*, o que pode ser comprovado pelo valor pequeno do erro de treinamento, enquanto o valor para o grupo de testes é muito alto em comparação. A solução para este caso seria reduzir a quantidade de neurônios, buscando um valor ótimo, de forma a não haver margem para *overfitting* (MathWorks, 2016). Infelizmente, não há como saber de antemão o tamanho da rede para uma determinada aplicação, deve-se ir testando.

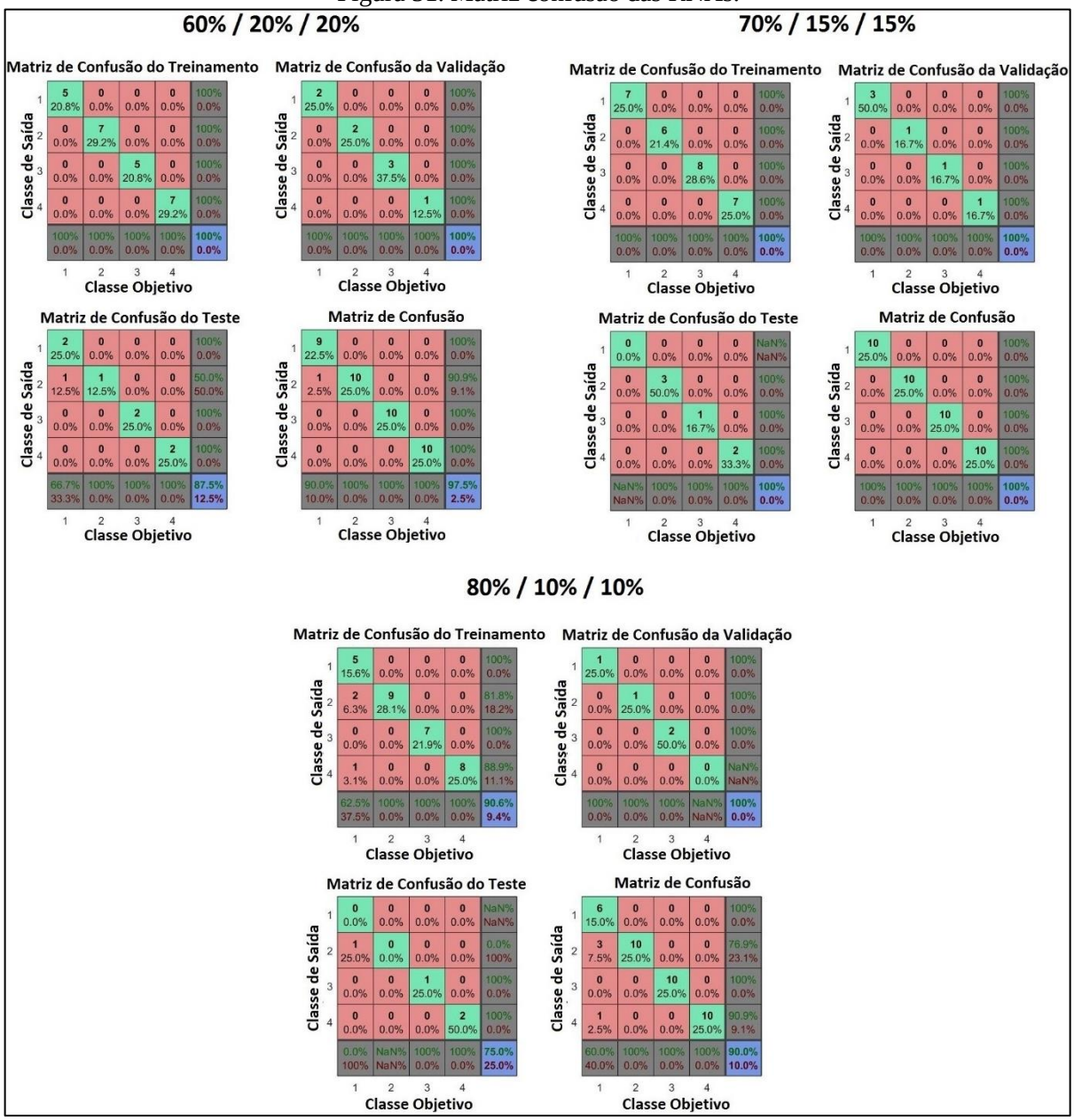
Figura 50. Performance das RNAs.



Fonte: próprio autor.

Para uma RNA de classificação, como a empregada, mais importante até do que apenas a minimização do erro é o resultado da classificação apresentado pela matriz de confusão (Figura 51). Aqui, os valores da diagonal (em verde) mostram o número de casos que foram corretamente classificados, fora da diagonal (em vermelho) os que foram classificados de maneira errônea. A célula em azul no canto inferior direito, a porcentagem total de acertos e erros.

Figura 51. Matriz confusão das RNAs.



Fonte: próprio autor.

De forma geral as RNAs testadas apresentaram bons resultados no reconhecimento dos padrões de harmônicos para cada tipo de lâmpada, com porcentagens totais de acertos maiores ou iguais a 90 %. Tal resultado, deixando de lado características da rede, mostra que as harmônicas escolhidas de fato são capazes de serem empregadas para classificação, dada a magnitude bastante distinta entre elas, como já mostrado anteriormente, e quais ordens estão presentes ou não conforme a natureza da carga.

5 CONCLUSÕES

5.1 Conclusões gerais

Este trabalho foi realizado em duas etapas: primeiramente o desenvolvimento de um sistema para aquisição de dados baseado no Arduino Uno R3. Em seguida, com a assinatura harmônica da corrente para os diferentes tipos de lâmpadas, obtida dos sinais aquisitados, o potencial para classificação de cargas destes dados foi testado através da ferramenta *Neural Network Pattern Recognition* do *software* Matlab.

O sistema de aquisição desenvolvido se mostrou limitado para a coleta dos dados, mas considerando sua simplicidade o resultado obtido pode ser considerado bom. Foi possível coletar uma quantidade de dados suficiente para a aplicação da transformada rápida de Fourier e para a caracterização do sinal, indicando uma elevada taxa de amostragem. Entretanto, dada a limitação da resolução do ADC (8 bits) imposta pela alta frequência de operação deste, a quantização resultou em curvas com formato de escada, em especial para a corrente, dado o pequeno valor desta. E embora não houve comprometimento da forma de onda da corrente, sua magnitude não ficou representada de maneira correta.

Verificou-se pela transformada rápida de Fourier que para as cargas analisadas, 6 lâmpadas: 2 incandescentes, 2 fluorescentes compactas e 2 LEDs, as cinco primeiras harmônicas ímpares (1ª, 3ª, 5ª, 7ª e 9ª) apresentam a maior parte da informação útil para a identificação de cargas. Essa constatação foi muito importante pois reduziu os atributos de entrada das RNAs para 5, garantindo um menor esforço computacional.

Os resultados experimentais obtidos com as RNAs testadas empregando o conjunto de dados supracitado se mostrou excelente, na casa dos 90 %, provando que há um grande potencial no uso de RNAs para identificação de cargas e que as harmônicas da corrente são, de fato, fonte de informação confiável para classificação de equipamentos.

5.2 Trabalhos futuros

Há muitas possibilidades para serem desenvolvidas nessa área de identificação de cargas. Os primeiros projetos partindo deste podem começar eliminando as limitações existentes.

Como para identificação de cargas a forma de onda da corrente contém a maioria das informações, é possível operar o sistema aquisitando dados apenas de uma porta analógica

empregando os 10 bits, ou até mesmo ampliar a resolução via *oversampling*. Pode-se ainda repensar os circuitos de condicionamento de sinal e testar outros tipos de sensores de corrente.

Trabalhos seguintes podem ainda testar outros métodos de classificação e identificação de cargas, outros tipos de RNAs, ou até mesmo planejar o desenvolvimento de uma por completo, ao invés de se fazer uso da ferramenta.

Este projeto pode ainda ser o ponto de partida para o desenvolvimento de um analisador de qualidade de energia barato ou mesmo de um identificador de cargas que funcione de maneira independente.

REFERÊNCIAS BIBLIOGRÁFICAS

ARDELEANU, A. S.; DONCIU, C. **Nonintrusive Load Detection Algorithm Based on Variations in Power Consumption**. International Conference and Exposition an Electrical and Power Engineering, 2012, Iasi, Roménia. Proceedings: EPE, p. 309 - 313.

Arduino Board Uno. 2016. Disponível em: <https://www.arduino.cc/en/Main/ArduinoBoardUno>. Acesso em 15/06/2016.

ARMEL, K. C.; GUPTA, A.; SHRIMALI, G.; ALBERT, A. **Is Disaggregation the holy Grail of Energy Efficiency? The Case of Electricity**. Energy Policy vol. 52, 2013.

Atmel. Atmel 8-Bit Microcontroller with 4/8/16/32kbytes In-System Programmable Flash – Datasheet. 2015.

BACURAU, R. M. **Medidor de Energia Inteligente para Discriminação de Consumo por Aparelho Através de Assinatura de Cargas.** 2014. 114 f. Dissertação (Mestrado) - Curso de Engenharia Elétrica, Universidade Estadual de Campinas, Campinas, 2014.

BRAGA, A. P., LUDERMIR, T. B. e CARVALHO, A. C. P. L. F. **Redes Neurais Artificiais: Teoria e Aplicações**. LTC. 2000.

CESÁRIO JÚNIOR, J. M. **Um medidor de energia elétrica integrado em redes de comunicações.** 2014. 95 f. Dissertação (Mestrado) - Curso de Mestrado em Tecnologia, Universidade Estadual de Campinas, Limeira, 2014.

CHANG, H. H.; LIN, C. L.; LEE, J. K. **Load Identification in Nonintrusive Load Monitoring Using Steady-State and Turn-on Transient Energy Algorithms**. Proceedings: 14th International Conference on Computer Supported Cooperative Work in Design, 2010, pp. 27-32, 2010.

COURY, D. V.; OLESKOVICZ, M.; GIOVANINI, R. **Proteção Digital de Sistemas Elétricos de Potência: dos Relés Eletromecânicos aos Microprocessados Inteligentes**. São Paulo: São Carlos: Universidade de São Paulo, 1999. ISBN 978-85-85205-78-2.

CYBENKO, G. **Approximation by Superpositions of a Sigmoid Function**, Mathematics of Control, Signals and Systems 2, pp.303-314, 1989.

DUARTE, L. F. C.; FERREIRA, E. C.; DIAS, J. A. S. **Measurement Techniques for Energy Efficiency Programs. In: EISSA, M. Energy Efficiency - The Innovative Ways for Smart Energy, the Future Towards Modern Utilities**. 1. ed. [S.l.]: InTech, v. 1, 2012. p. 193-208.

EHRHARDT-MARTINEZ, K.; DONNELLY, K. A.; LAITNER, J. A. **Advanced Metering Initiatives and Residential Feedback Programs: A Meta-Review for Household Electricity-Saving Opportunities**. ACEEE. Washington D.C. 2010.

FERNANDES, R. A. S. (2008). **Identificação de Fontes de Correntes Harmônicas por Redes Neurais Artificiais.** Dissertação (Mestrado) – Escola de Engenharia de São Carlos, Universidade de São Paulo, 2008.

GIBELLI, G. B. **Metodologia para o Diagnóstico e Análise da Influência dos Afundamentos e Interrupções de Tensão nos Motores de Indução Trifásicos**. 2016. 174 f. Tese (Doutorado) - Curso de Engenharia Elétrica, Universidade de São Paulo, São Carlos, 2016.

HART, G. **Nonintrusive appliance load monitoring**. Proceedings of the IEEE, vol. 80, no. 12, pp. 1870 –1891, Dez. 1992.

HAYKIN, S. **Redes Neurais: Princípios e Prática**. Editora Bookman, 2001.

IEC - INTERNATIONAL ELECTROTECHNICAL COMMISSION.**60050-321**: **International Electrotechnical Vocabulary. Chapter 321: Instrument transformers.** Genebra: IEC, 1986.

INGALE, R. **Harmonic Analysis Using FFT and STFT**. International Journal of Signal Processing, Image Processing and Pattern Recognition. Latur, p. 345-362. jul. 2014.

KIRKHAM, H. **Current measurement methods for the smart grid**. Proceedings. IEEE PES General Meeting, Calgary, AB, Canada, pp. 1-7, 2009.

MACHE, N. **Scaled Conjugate Gradient (SCG)**, 1995. Disponível em: <http://www.ra.cs.uni-tuebingen.de/SNNS/UserManual/node241.html>. Acesso em: 26/09/2016.

MADISETTI, V. K.; WILLIAMS, D. B. **The Digital Signal Processing Handbook**. Florida: CRC Press, 1998.

MathWorks. **Divide Data for Optimal Neural Network Training**. Disponível em: < https://www.mathworks.com/help/nnet/ug/divide-data-for-optimal-neural-network-training.html >. Acesso em: 28/09/2016.

MathWorks. **Fourier Transforms**. Disponível em: <https://www.mathworks.com/help/matlab/math/fourier-transforms.html>. Acesso em: 28/09/2016.

MathWorks. **Improve Neural Network Generalization and Avoid Overfitting**. Disponível em: < http://www.mathworks.com/help/nnet/ug/improve-neural-network-generalization-and-avoid-overfitting.html >. Acesso em: 28/09/2016.

MCCULLOCH, W. S. e PITTS, W. (1943). **A Logical Calculus of the Ideas Immanent in Nervous Activity**. Bulletin of Mathematical Biophysics, No. 5, pp. 115-133.

NDIAYE, M. S. (2006). **Modelagem de cargas não-lineares por fontes de corrente sincronizadas**. Dissertação (Mestrado). – Curso de Engenharia Elétrica, Universidade Federal do Rio de Janeiro, 2006.

OLIVEIRA, Â. R. **Redes Neurais Artificiais Aplicadas na Detecção, Classificação e Localização de Defeitos em Linhas de Transmissão** [Juiz de Fora] 2005. XI, 132 p. 29,7 cm. (UFJF, M. Sc, Engenharia Elétrica, 2005)

OpenEnergyMonitor Project. Disponível em: <https://openenergymonitor.org/emon/buildingblocks>. Acesso em 29/06/2016.

POCHACKER, M.; EGARTER, D.; ELMENREICH, W**. Proficiency of Power Values for Load Disaggregation**. IEEE Transactions.on Instrumentation and Measurement. v. 65, n. 1, p.46-55, jan. 2016. Institute of Electrical & Electronics Engineers (IEEE).

RUMELHART, D. E. e MCCLELLAND, J. L. **Parallel Distributed Processing.** MIT Press. 1986

SEDRA, A. S.; SMITH, K. C. **Microeletrônica**. 4. ed. São Paulo: MAKRON Books, 2000. ISBN 85-346-1044-4.
Sense, 2016. Disponível em:<www.sense.com>. Acesso em: 20/08/2016

SILVA, N. I.; SPATTI, D. H.; FLAUZINO, R. A**. Redes Neurais Artificiais para engenharia e ciências aplicadas**. São Paulo: Artliber Editora, 2010. ISBN 978-85-88098-53-4.

SRINIVASAN, D; NG, W. S.; LIEW, A. C. **Neural-network-based signature recognition for harmonic source identification**. IEEE Transactions on Power Delivery, vol. 21, pp. 398-405, 2006.

UENO, T.; INADA, R.; SAEKI, O. E TSUJI, K. **Effectiveness of displaying energy consumption data in residential houses. Analysis of how the residents respond**. ECEEE 2005 Summer Study Proc., vol. 3, pp. 1289-1301, 2005.

WIDROW, B.; RUMELHART, D.E.; LEHR, M.A. **Neural Networks: Applications in Industry, Business and Science.** Communications of ACM, Vol.37, N.3, March, 1994.

YORK, D. et al. **Next Generation Programs Reach for High Energy Savings**. ACEEE. Washington -D.C. 2013.

ZIEGLER, S.; WOODWARD, R. C.; IU, H. H.; BORLE, L. J. **Current Sensing Techniques: A Review**. IEEE Sensors Journal.Crawley, p. 354-376. abr. 2009.

APÊNDICES

APÊNDICE A – Código fonte de aquisição de sinal no Arduino

```
//Macros que habilitam o uso das funçoes CBI (clear bit in) e SBI (set bit
in), que permitem dar set e reset em bit específicos dentro dos
registradores
#define cbi(sfr, bit) (_SFR_BYTE(sfr) &= ~_BV(bit))
#define sbi(sfr, bit) (_SFR_BYTE(sfr) |= _BV(bit))

//Declaração das variáveis
volatile unsigned char badc;
volatile unsigned short channel;
volatile unsigned char done = 0;

//Observação
//Channel 0 -- Sinal CA da tensão (Pin A0)
//Channel 1 -- Sinal CA da corrente (Pin A1)

void setup() {

Serial.begin(1000000,SERIAL_8O1); //Estabelece uma conexão serial com baud
rate de 1000000 baud/s, número de bits de dados (8), paridade ímpar (O) e
número de stop bits (1)

  //Reseta o conteúdo dos registradores
  ADCSRA = 0;
  ADCSRB = 0;
  ADMUX = 0;
  //Seleciona a tensão de referência do ADC para Vcc=5V
  ADMUX |= (1 << REFS0);
  //Alinha o resultado do ADC à esquerda, reduzindo a resolução para 8 bits
  ADMUX |= (1 << ADLAR);
  //Habilita a interrupção do ADC
  ADCSRA |= (1 << ADIE);
  //Seleciona o fator pré-escalar para 32
  ADCSRA |= (1 << ADPS0 | 1 << ADPS2);
  //Habilita o ADC
  ADCSRA |= (1 << ADEN);
  //Inicia a conversão
  ADCSRA |= (1 << ADSC);
}

void loop() {

  if(done == 1) {
   //Envia os dados em formato binário para porta serial (8 bits)
   Serial.write(badc);
   //Envio concluído
   done = 0;

  }
}

//Interrupção do ADC
ISR(ADC_vect) {

  //Verifica qual canal está sendo lido
  channel = ADMUX & 0x01;
  //Recebe os dados do registrador de dados ADCH
  badc = ADCH;
```

```
  //Verifica o status do envio
  if(done == 0) {
    //Verifica qual canal está sendo lido
    //Se a última conversão foi para tensão, a próxima é para corrente,
alternando infinitamente
    if(channel == 0) {

       //Amostra o valor da corrente no Pin A1
       sbi(ADMUX,MUX0);
       cbi(ADMUX,MUX1);
       cbi(ADMUX,MUX2);
       cbi(ADMUX,MUX3);

    } else if(channel == 1) {

       //Amostra o valor da tensão no Pin A0
       cbi(ADMUX,MUX0);
       cbi(ADMUX,MUX1);
       cbi(ADMUX,MUX2);
       cbi(ADMUX,MUX3);

    }
    //Inicia a próxima conversão
    sbi(ADCSRA,ADSC);
    //Resultado disponível para envio
    done = 1;

  }
}
```

APÊNDICE B – *Script* de recebimento dos dados no *software* Matlab

```
% Contador
i=0;
% Número de pontos a ser amostrado
iter=10000;
% Matriz dos pontos amostrados
Data=zeros(10000,1);
% Cria uma conexão com a porta serial especificada e as propriedades
definidas
s = serial('COM3', 'BaudRate',1000000,'Parity','odd','StopBits',1);
% Configura o número total de bytes que podem ser armazenados no
input buffer durante operação de leitura
s.InputBufferSize=1;
% Estabele a conexão serial
fopen(s);
% Amostra o sinal pelo número de vezes estabelecido e armazena o
dado lido na matriz Data
while (i<iter);
a=fread(s);
i=i+1;
Data(i,:)=a;
end
% Encerra a conexão serial
fclose(s);
% Cria duas matrizes compostas pelo elementos ímpares e pares da
matriz Data, respectivamente
b=Data(1:2:end,:);
c=Data(2:2:end,:);
% Plota a forma de onda dos dados em b e c
plot(b);
hold;
plot(c,'color','r');
```

APÊNDICE C – *Fast Fourier Transform* (FFT)

```
% Frequência de amostragem
FS=15660;
% Período de amostragem
T=1/FS;
% Comprimento do sinal (número de amostras)
L=length(b);
% Vetor tempo
t=(0:L-1)*T;
% Remoção da componente DC
b=b-mean(b);
% Fast Fourier Transform (FFT)
Y=fft(b);
% Calcula os dois lados do espectro
P2=abs(Y/L);
% Calcula um lado do espectro
P1=P2(1:L/2+1);
P1(2:end-1)=2*P1(2:end-1);
% Define o domínio da frequência
f=FS*(0:(L/2))/L;
```