

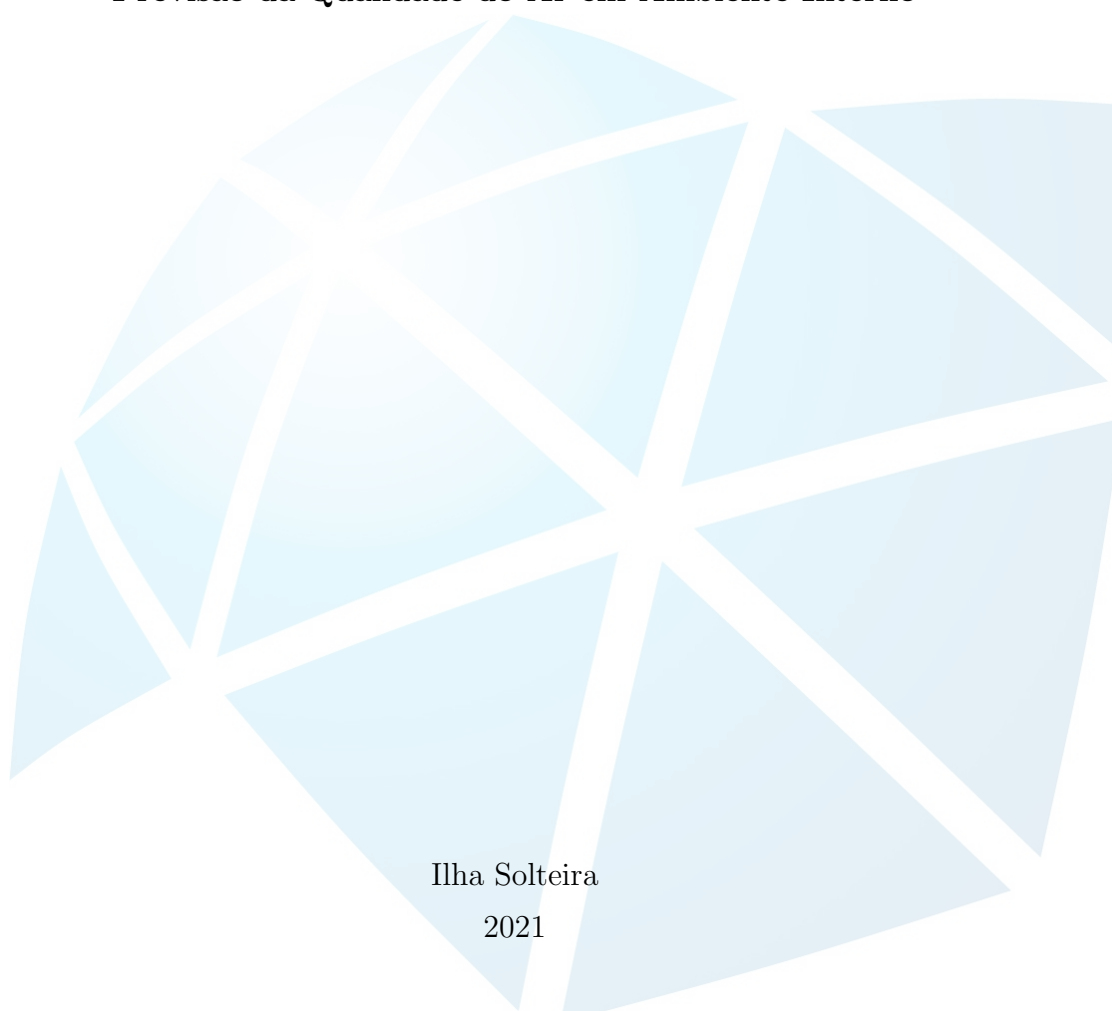
UNIVERSIDADE ESTADUAL PAULISTA “JÚLIO DE MESQUITA FILHO”

FACULDADE DE ENGENHARIA

CÂMPUS DE ILHA SOLTEIRA

WILLIAN DE ASSIS PEDROBON FERREIRA

**Rede Neural ARTMAP *Fuzzy* Implementada em Hardware Aplicada na
Previsão da Qualidade do Ar em Ambiente Interno**



Ilha Solteira

2021

WILLIAN DE ASSIS PEDROBON FERREIRA

**Rede Neural ARTMAP *Fuzzy* Implementada em Hardware Aplicada na
Previsão da Qualidade do Ar em Ambiente Interno**

Tese apresentada à Faculdade de Engenharia –
UNESP – Câmpus de Ilha Solteira como parte
dos requisitos exigidos para obtenção do título
de Doutor em Engenharia Elétrica.
Área de Conhecimento: Automação.

Prof. Dr. Alexandre César Rodrigues da Silva
Orientador

Prof. Dr. Ian Andrew Grout
University of Limerick - Irlanda
Coorientador

Ilha Solteira
2021

FICHA CATALOGRÁFICA

Desenvolvido pelo Serviço Técnico de Biblioteca e Documentação

F383r Ferreira, Willian de Assis Pedrobon.
Rede neural ARTMAP Fuzzy implementada em hardware aplicada na
previsão da qualidade do ar em ambiente interno / Willian de Assis Pedrobon
Ferreira. -- Ilha Solteira: [s.n.], 2021
100 f. : il.

Tese (Doutorado em Engenharia Elétrica) - Universidade Estadual Paulista.
Faculdade de Engenharia de Ilha Solteira. Área de conhecimento: Automação,
2021

Orientador: Alexandre César Rodrigues Da Silva
Co-orientador: Ian Andrew Grout
Inclui bibliografia

1. Rede neural ARTMAP Fuzzy . 2. Internet das coisas. 3. Qualidade do ar. 4.
Tempo real. 5. Hardware. 6. FPGA.


Raiane da Silva Santos

CERTIFICADO DE APROVAÇÃO

TÍTULO DA TESE: Rede Neural ARTMAP Fuzzy Implementada em Hardware Aplicada na Previsão da Qualidade do Ar em Ambiente Interno

AUTOR: WILLIAN DE ASSIS PEDROBON FERREIRA

ORIENTADOR: ALEXANDRE CESAR RODRIGUES DA SILVA

Aprovado como parte das exigências para obtenção do Título de Doutor em ENGENHARIA ELÉTRICA, área: Automação pela Comissão Examinadora:

Prof. Dr. ALEXANDRE CESAR RODRIGUES DA SILVA (Participação Virtual)
Departamento de Engenharia Elétrica / Faculdade de Engenharia de Ilha Solteira - UNESP

Prof. Dr. GUILHERME DE AZEVEDO E MELO (Participação Virtual)
Departamento de Engenharia Elétrica / Faculdade de Engenharia de Ilha Solteira - UNESP

Profª. Drª. MARA LUCIA MARTINS LOPES (Participação Virtual)
Departamento de Matemática / Faculdade de Engenharia de Ilha Solteira - UNESP

Profª. Drª. YARA DE SOUZA TADANO (Participação Virtual)
Departamento de Matemática / Universidade Tecnológica Federal do Paraná - UTFPR

Prof. Dr. MAICON APARECIDO SARTIN (Participação Virtual)
Departamento de Ciência da Computação / Universidade do Estado do Mato Grosso - UNEMAT

Ilha Solteira, 25 de agosto de 2021



Alexandre César Rodrigues da Silva

AGRADECIMENTOS

Agradeço a Deus, por me conceder saúde e por não me deixar faltar nada.

Aos meus pais, Emídio e Solange, pelo apoio e suporte.

Ao meu irmão, Tarcísio, companheiro sempre presente.

Ao meu orientador Prof. Dr. Alexandre César Rodrigues da Silva e coorientador Prof. Dr. Ian Andrew Grout da Universidade de Limerick, Irlanda, por terem possibilitado o desenvolvido deste trabalho.

Aos membros da banca, por se disponibilizarem para auxiliar na avaliação deste trabalho.

Aos amigos do Laboratório de Processamento de Sinais e Sistemas Digitais, pela cooperação e longa amizade construída durante todos esses anos.

Ao Thiago Costa de Paiva, pelo auxílio em etapas primordiais do desenvolvimento da descrição do hardware.

À Poliana, pelos incentivos, paciência e companhia em todos os momentos.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

RESUMO

Tecnologias elaboradas na Internet das Coisas e em áreas relacionadas proporcionaram vários avanços no desenvolvimento de sensores e de interfaces computacionais miniaturizadas, que são empregadas em serviços de monitoramento e auxiliam diversos cenários da sociedade moderna. Com o uso de algoritmos de aprendizado de máquina para processar informações amostradas por sensores, a atuação automática dos sistemas computacionais em resposta aos comportamentos existentes beneficiou vários processos, trazendo otimização, segurança e bem-estar aos cidadãos assistidos. Nesse contexto insere-se esta pesquisa, cujo objetivo foi a implementação de um sistema embarcado com treinamento on-line, possibilitando uma aprendizagem contínua para realizar previsões em tempo real. A aplicação alvo foi o monitoramento e a previsão de séries temporais com dados da qualidade de ar em ambiente interno utilizando-se a rede neural ARTMAP *fuzzy*, que emprega conceitos da Teoria da Ressonância Adaptativa. Sensores de baixo custo amostraram os seguintes componentes em um ambiente doméstico utilizado como quarto e escritório: índice da qualidade de ar interno calculado automaticamente por uma biblioteca da Bosch, temperatura, umidade, dióxido de carbono, compostos orgânicos voláteis e material particulado. A melhor acurácia das previsões univariadas foi a da temperatura, com porcentagem de erro médio absoluto igual a 2,8 na previsão 1 hora à frente, e de 6,4 nas previsões 24 horas à frente. Para obter previsões do índice da qualidade do ar com maior acurácia, foram verificadas algumas combinações multivariadas como entradas da ARTMAP *fuzzy*, sendo o melhor resultado da previsão 1 hora à frente obtido com a temperatura e umidade, com 31,3 de porcentagem de erro médio absoluto, que foi 15,2% menor em comparação à previsão univariada desse índice. Com o intuito de aprimorar o tempo de resposta da rede neural ARTMAP *fuzzy*, foi projetada uma arquitetura de hardware em uma FPGA (*field-programmable gate array*). Foi proposto um novo gerenciamento da manutenção das categorias da rede neural ARTMAP *fuzzy*, que economizou 70% do uso de memória do estudo de caso em hardware. Considerando ambos os tempos de treinamento e predição, a FPGA foi em média 84 e 683 vezes mais rápida que um laptop com um processador Intel Core-i5 8250U e um Raspberry Pi 3, respectivamente, que processaram o algoritmo da ARTMAP *fuzzy* implementado em Python.

Palavras-chave: rede neural ARTMAP *fuzzy*. internet das coisas. qualidade do ar. tempo real. hardware. FPGA.

ABSTRACT

Internet of Things and related technological areas have provided several advances in the development of sensors and miniaturized computer interfaces, which are employed in monitoring services and help various scenarios of modern society. With the use of machine learning algorithms to process sensor data, the automatic computer systems feedback in response to existing behaviors benefited several processes, bringing optimization, safety, and well-being to assisted citizens. However, the association of machine learning algorithms with dense data volumes requires high computational processing rates. In this context, this research is included, whose objective was the implementation of an embedded system with online training, enabling continuous learning to perform real-time predictions. The target application was the monitoring and prediction of indoor air quality time series data using the fuzzy ARTMAP neural network, which employs concepts of Adaptive Resonance Theory. Low-cost sensors sampled the following components in a domestic indoor environment used as a bedroom and office: indoor air quality index automatically calculated by a Bosch library, temperature, humidity, carbon dioxide, volatile organic compounds, and particulate matter. The best univariate prediction accuracy was the temperature, with a mean absolute percentage error equal to 2.8 for 1 hour ahead prediction, and 6.4 for 24 hours ahead predictions. To obtain more accurate air quality index prediction, some multivariate combinations were verified as fuzzy ARTMAP input, being the best 1 hour ahead prediction result obtained by temperature and humidity with a mean absolute error percentage of 31.3, which was 15.2% lower compared to the univariate prediction of this index. To improve the fuzzy ARTMAP neural network, a hardware field-programmable gate array (FPGA) architecture was designed. A new proposed ARTMAP fuzzy neural network clusters management saved 70% memory usage of the hardware study case. Considering both training and prediction operations, the FPGA was on average 84 and 683 times faster than a laptop with an Intel Core-i5 8250U processor and a Raspberry Pi 3, respectively, which processed a fuzzy ARTMAP Python implementation.

Keywords: fuzzy ARTMAP neural network. internet of things. air quality. real-time. hardware. FPGA.

LISTA DE FIGURAS

Figura 1 – Etapas realizadas neste trabalho	16
Figura 2 – Paradigmas de computação anteriores à IoT	17
Figura 3 – Aprimoramento de sistemas IoT com técnicas de IA	18
Figura 4 – Camadas de processamento aplicadas em sistemas IoT	20
Figura 5 – Estrutura elementar da rede neural ART	27
Figura 6 – Arquitetura da rede neural ART <i>fuzzy</i>	29
Figura 7 – Fluxograma da rede neural ART <i>fuzzy</i>	32
Figura 8 – Arquitetura da rede neural ARTMAP <i>fuzzy</i>	34
Figura 9 – Fluxograma da rede neural ARTMAP <i>fuzzy</i> com treinamento off-line	35
Figura 10 – Arquitetura da rede neural ARTMAP <i>fuzzy</i> com treinamento on-line	37
Figura 11 – Fluxograma da rede neural ARTMAP <i>fuzzy</i> com treinamento on-line	39
Figura 12 – Fluxograma do cálculo de uma predição da rede ARTMAP <i>fuzzy</i>	41
Figura 13 – Diagrama de ligação dos sensores ao Raspberry Pi 3	44
Figura 14 – Sensores utilizados nessa pesquisa	46
Figura 15 – Séries temporais dos dados amostrados	47
Figura 16 – Série temporal modelada em janelas deslizantes sobrepostas	49
Figura 17 – Diferentes metodologias de normalização dos padrões de entrada da rede neural ARTMAP <i>fuzzy</i>	51
Figura 18 – Previsões univariadas com normalização por linha	53
Figura 19 – Previsões univariadas com normalização por coluna	55
Figura 20 – Matrizes de correlação das séries temporais com atraso de até 3 <i>lags</i>	57
Figura 21 – Previsões univariadas 24 horas à frente	60
Figura 22 – Matriz de correlação dos compostos do ar	63

Figura 23 – Previsões do IAQ aplicando normalização por linha e séries temporais multivariadas	64
Figura 24 – Previsão do IAQ aplicando normalização por coluna e séries temporais multivariadas	64
Figura 25 – Taxa de dados produzidos por várias aplicações	69
Figura 26 – Fluxograma da rede neural ARTMAP <i>fuzzy</i> com o novo gerenciamento de categorias	73
Figura 27 – Diagrama de blocos da entidade <i>ARTMAP_fuzzy</i>	76
Figura 28 – Diagrama de blocos da entidade <i>ART_a</i>	77
Figura 29 – Exemplo de uso do vetor <i>flag_cluster_a</i>	78
Figura 30 – Diagrama de blocos da entidade <i>WTA</i>	79
Figura 31 – Diferentes metodologias de mapeamentos Inter-ART	80
Figura 32 – Diagrama de blocos da entidade <i>IACM</i>	81
Figura 33 – Diagrama de blocos da entidade <i>pred_pattern</i>	82
Figura 34 – Diagrama de sequência da rede ARTMAP <i>fuzzy</i> em hardware	83
Figura 35 – Número de categorias criadas durante o treinamento on-line da rede neural ARTMAP <i>fuzzy</i> em hardware	85
Figura 36 – Tempos de treinamento e predição executados em diferentes dispositivos	86
Figura 37 – Dados da qualidade do ar atualizados em tempo real	99
Figura 38 – Configuração do servidor Node-RED	99

LISTA DE QUADROS

Quadro 1 – Poluentes típicos do ar em ambientes internos	24
Quadro 2 – Informações dos sensores utilizados nesse trabalho	43
Quadro 3 – Classificação do IAQ calculado pelo BSEC	44
Quadro 4 – Parâmetros da ARTMAP <i>fuzzy</i> com normalização por linha	52
Quadro 5 – Avaliação das previsões univariadas	56
Quadro 6 – Número de categorias criadas nas avaliações univariadas	59
Quadro 7 – Avaliação das previsões univariadas 24 horas à frente	61
Quadro 8 – Avaliação das previsões do IAQ com séries temporais multivariadas	65
Quadro 9 – Avaliação da rede ARTMAP <i>fuzzy</i> com diferentes números de categorias	72
Quadro 10 – Hardware da FPGA utilizado na descrição em VHDL	84
Quadro 11 – Comparação da acurácia da rede neural ARTMAP <i>fuzzy</i> implementada em software e em hardware	85
Quadro 12 – Tempos de execução em segundos de diferentes dispositivos	86
Quadro 13 – Relação de tempos de execução da FPGA com os dos dispositivos baseados em software	87
Quadro 14 – Avaliação das previsões do IAQ 24 horas à frente com séries temporais multivariadas	100
Quadro 15 – Número de categorias criadas nas avaliações de séries temporais multivariadas	100

LISTA DE ABREVIATURAS E SIGLAS

$\mu\text{g}/\text{m}^3$	Micrograma por metro cúbico
AmI	<i>Ambiental intelligence</i>
ARMA	<i>Autoregressive Moving Average</i>
ART	<i>Adaptive Resonance Theory</i>
ASIC	<i>Application-specific integrated circuit</i>
BSEC	<i>Bosch Sensortec Environmental Cluster</i>
CO	Monóxido de carbono
CO ₂	Dióxido de carbono
DL	<i>Deep learning</i>
DSP	<i>Digital signal processor</i>
EPA	<i>Environmental Protection Agency</i>
FPGA	<i>Field-programmable gate array</i>
FSM	<i>Finite state machine</i>
GPU	<i>Graphics processing unit</i>
HLS	<i>High-level synthesis</i>
HVAC	<i>Heating, ventilating and air conditioning</i>
IA	Inteligência artificial
IACM	<i>Inter-ART cluster manager</i>
IAQ	<i>Indoor air quality</i>
IoT	<i>Internet of Things</i>
IP	<i>Intellectual property</i>

LSTM	<i>Long short-term memory</i>
LUT	<i>Look-up table</i>
MAE	<i>Mean absolute error</i>
MAPE	<i>Mean absolute percentage error</i>
MENS	<i>Micro-electromechanical systems</i>
ML	<i>Machine learning</i>
NO ₂	Dióxido de nitrogênio
NO _x	Óxidos nitrosos
O ₃	Ozônio
OS-ELM	<i>On-line Sequential Extreme Learning Machine</i>
PLD	<i>Programmable logic device</i>
PM	<i>Particulate matter</i>
PM ₁₀	Partículas com diâmetros de até 10 micrômetros
PM _{2.5}	Partículas com diâmetros de até 2,5 micrômetros
ppb	Partes por bilhão
ppm	Partes por milhão
RFID	<i>Radio-frequency identification</i>
RMSE	<i>Root mean squared error</i>
Rn	Radônio
RNA	Rede neural artificial
SO ₂	Dióxido de enxofre
TPU	<i>Tensor processing unit</i>
VOC	<i>Volatile organic compound</i>
WTA	<i>Winner takes all</i>

SUMÁRIO

1	INTRODUÇÃO	13
1.1	OBJETIVOS	15
2	ASSOCIAÇÃO DE ALGORITMOS DE <i>MACHINE LEARNING</i> À APLICAÇÕES IoT	17
2.1	PLATAFORMAS IoT BASEADAS EM SOFTWARE E HARDWARE	19
2.2	FERRAMENTAS EMPREGADAS NO MONITORAMENTO E NA PREVISÃO DA QUALIDADE DE AR	22
3	TEORIA DA RESSONÂNCIA ADAPTATIVA	27
3.1	REDE NEURAL ART <i>FUZZY</i>	29
3.2	REDE NEURAL ARTMAP <i>FUZZY</i>	33
3.2.1	Rede neural ARTMAP <i>fuzzy</i> com treinamento on-line	37
3.2.2	Cálculo de uma predição com a rede neural ARTMAP <i>fuzzy</i>	40
4	PREVISÃO DE COMPONENTES DA QUALIDADE DE AR COM A REDE NEURAL ARTMAP <i>FUZZY</i>	42
4.1	SISTEMA PROPOSTO	42
4.1.1	Amostragem de dados	44
4.2	REDE NEURAL ARTMAP <i>FUZZY</i> APLICADA NA PREVISÃO DE SÉRIES TEMPORAIS	48
4.2.1	Avaliações de séries temporais univariadas	51
4.2.2	Previsão do IAQ aplicando séries temporais multivariadas	62
5	IMPLEMENTAÇÃO DA REDE NEURAL ARTMAP <i>FUZZY</i> EM HARDWARE	68

5.1	PROPOSTA DE UM NOVO GERENCIAMENTO DE CATEGORIAS	70
5.2	OPERAÇÕES ARITMÉTICAS MODELADAS EM PONTO FIXO	74
5.3	ARQUITETURA DA REDE NEURAL ARTMAP <i>FUZZY</i> EM HARDWARE	75
5.4	AVALIAÇÃO DA DESCRIÇÃO EM HARDWARE	83
6	CONCLUSÕES	89
6.1	TRABALHOS FUTUROS	90
	REFERÊNCIAS	92
	APÊNDICE A - <i>DASHBOARD</i> PARA VISUALIZAÇÃO DOS DADOS EM TEMPO REAL	99
	APÊNDICE B - AVALIAÇÃO DAS PREVISÕES DO IAQ APLICANDO SÉRIES TEMPORAIS MULTIVARIADAS	100

1 INTRODUÇÃO

A tecnologia está cada vez mais presente no cotidiano e tornou-se fundamental para promover melhorias na qualidade de vida da sociedade moderna. Atualmente existem vários serviços fornecidos por dispositivos eletrônicos, muitas vezes transparentes aos usuários, que auxiliam na execução de atividades em diversas aplicações. Tarefas executadas manualmente vêm sendo automatizadas por sistemas computacionais autônomos, proporcionando conforto e segurança às pessoas.

Para possibilitar a interação homem-máquina, é indispensável estudar comportamentos existentes com a finalidade de identificar padrões das ações e dos ambientes monitorados. O sensoriamento dessas características foi amplamente promovido após a ascensão da Internet das Coisas, ou IoT (*Internet of Things*), que auxiliou no desenvolvimento de novas tecnologias empregadas na implementação de vários tipos de sensores.

Em centros urbanos, por exemplo, há grande interesse na coleta de informações em tempo real de diversos contextos, como dados ambientais (qualidade de ar, ruído sonoro), dados de infraestrutura e informações da dinâmica das cidades (tráfego de veículos e fluxo de pessoas) (WANG *et al.*, 2019). Além dos fenômenos físicos, estuda-se também o comportamento das pessoas, que pode ser realizado por meio de sensores móveis e sensores corporais, como acelerômetros e giroscópios (CHEN *et al.*, 2019).

Um dos objetivos de obter informações que caracterizam comportamentos de ambientes e pessoas é a inserção de atuadores nas situações monitoradas, cujo papel é influenciar, de alguma forma, a execução dos procedimentos existentes no contexto analisado. Considerando essas mudanças obtidas com a densa aplicação de sensores e atuadores em centros urbanos, surgiu o conceito de cidades inteligentes, cujo fundamento é a obtenção de melhorias no gerenciamento de recursos e de processos baseando-se no monitoramento para gerar serviços eficientes que auxiliem os cidadãos (WANG *et al.*, 2019; DU *et al.*, 2019).

Diante da grande quantidade de dados gerados pelas cidades inteligentes, é necessário interpretar e identificar padrões das informações para que os sistemas IoT possam atuar e fornecer as respostas desejadas de acordo com o contexto analisado. Esta necessidade foi

amplamente atendida aplicando-se conceitos da inteligência artificial (IA) para desenvolver algoritmos de aprendizagem capazes de extrair conhecimento dos dados amostrados. Além disso, os algoritmos inteligentes se adaptam automaticamente de acordo com as mudanças e comportamentos existentes, fazendo com que as respostas destes sistemas permaneçam atualizadas (DU *et al.*, 2019).

O aprendizado de máquina, ou ML (*machine learning*), é um campo da IA de grande interesse no estudo e desenvolvimento de algoritmos inteligentes. Adicionalmente, uma importante extensão do ML é o aprendizado profundo, ou DL (*deep learning*). O objetivo dessas linhas de pesquisa é desenvolver sistemas computacionais capazes de aprender a partir de padrões dos dados processados. Com o conhecimento adquirido pela interação do algoritmo com os estímulos do ambiente, criam-se modelos matemáticos generalizados capazes de reconhecer padrões e comportamentos existentes, possibilitando o cálculo de previsões nos contextos observados.

O processamento da densa quantidade de dados aplicando algoritmos de ML e DL pode demandar altas capacidades computacionais. Considerando aplicações IoT, é comum enviar dados de sensores via Internet para serviços de computação em nuvem. Todo processamento é realizado remotamente em servidores com alta capacidade computacional, e os resultados são transmitidos de volta aos dispositivos IoT (SAMIE; BAUER; HENKEL, 2019). Contudo, podem surgir algumas limitações nesta abordagem, como alta latência e consequentemente baixa taxa de transferência de dados.

Uma alternativa para se obter processamento e respostas em tempo real, geralmente exigidos em sistemas IoT, é migrar o processamento remoto para camadas mais próximas aos terminais de amostragem de dados, isto é, interpretar os dados em aplicações de computação de borda (SHAFIQUE *et al.*, 2018). Assim, o processamento de dados é realizado mais próximo dos sensores, o que minimiza gargalos de latência e aumenta a privacidade das informações (VERHELST; MOONS, 2017). Outra possibilidade de realizar análises localmente é a implementação de dispositivos IoT inteligentes, cujo processamento é realizado no mesmo dispositivo que os sensores são conectados.

A implementação de aplicações IoT com processamento embarcado fundamentalmente utiliza dispositivos digitais baseados em software e hardware (VIEL *et al.*, 2018). Sistemas com nicho em software utilizam processadores para executar instruções de linguagem de programação, sendo compreendidos desde dispositivos minimalistas, como microcontroladores, até sistemas complexos, como DSPs (*digital signal processors*). Por outro lado, a abordagem de hardware é baseada em dispositivos lógicos programáveis, ou

PLDs (*programmable logic devices*), que possibilitam projetar arquiteturas personalizáveis por meio de circuitos digitais reconfiguráveis. Há vantagens e desvantagens no uso de software ou hardware, considerando parâmetros específicos, que devem ser analisados para se obter a eficiência necessária pelos sistemas inteligentes embarcados.

Entre os dispositivos de hardware, o recurso de executar cálculos paralelamente da FPGA (*field-programmable gate array*) as tornam uma opção viável de hardware para desenvolver sistemas ML complexos (DINIZ *et al.*, 2017). Da mesma forma, no DL, a possibilidade de implementar sistemas adaptáveis e reconfiguráveis capacita as FPGAs, que fornecem integridade e desempenho necessário para executar o treinamento e predições em sistemas embarcados (STORNAIUOLO; SANTAMBROGIO; SCIUTO, 2018).

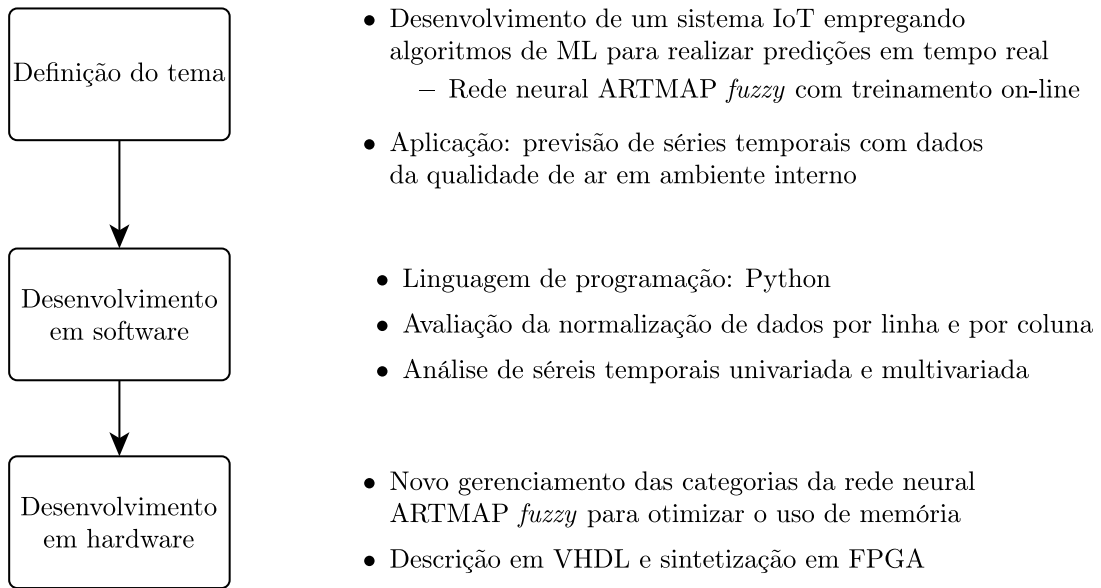
Em relação aos contextos dinâmicos das cidades inteligentes, para que sejam feitas análises atualizadas, o monitoramento é realizado aplicando-se amostragem contínua de dados. Dessa forma, é necessário que os algoritmos de ML aplicados em sistemas IoT sejam treinados constantemente para que forneçam respostas em tempo real (PÉREZ-SÁNCHEZ; FONTENLA-ROMERO; GUIJARRO-BERDIÑAS, 2018). Esta abordagem de treinamento também é conhecida como aprendizagem on-line.

1.1 OBJETIVOS

Diante do contexto apresentado, o objetivo deste trabalho foi aprimorar a capacidade de atuação de dispositivos IoT embarcados por meio do desenvolvimento de um algoritmo de ML empregando treinamento on-line. Dessa forma, uma das principais contribuições desta pesquisa foi propor um sistema que beneficia aplicações em tempo real, pois a abordagem de treinamento on-line possibilita incrementar o conhecimento do algoritmo de ML constantemente com dados recém amostrados. As etapas desenvolvidas durante o desenvolvimento deste trabalho estão apresentadas na Figura 1.

O algoritmo de ML considerado foi a rede neural ARTMAP *fuzzy*, que emprega conceitos da Teoria da Ressonância Adaptativa. Um dos principais diferenciais dessa rede neural é a possibilidade de incrementar o seu conhecimento por meio do armazenamento de novas informações, mas sem afetar o conhecimento previamente adquirido.

O sistema foi aplicado na previsão da qualidade de ar em ambiente interno. A amostragem de dados foi realizada em um quarto onde reside uma pessoa, que também foi utilizado como escritório de trabalho remoto durante a pandemia. Utilizando sensores de baixo custo, foram monitoradas as seguintes variáveis: umidade, temperatura,

Figura 1 – Etapas realizadas neste trabalho

Fonte: Elaboração do próprio autor

gás carbônico, compostos orgânicos voláteis e material particulado. Adicionalmente, considerou-se o IAQ (*Indoor Air Quality*), que é um índice calculado automaticamente por uma biblioteca da Bosch que mensura a qualidade do ar. Os dados foram modelados em séries temporais por meio de janelas deslizes sobrepostas, possibilitando realizar o treinamento supervisionado da rede neural ARTMAP *fuzzy* para calcular a previsão até 24 horas à frente.

O sistema foi implementado na linguagem de programação Python. Além da abordagem em software, outro objetivo deste trabalho foi o desenvolvimento em hardware da ARTMAP *fuzzy* com treinamento on-line, que foi descrita em VHDL e sintetizada em uma FPGA. Visando otimizar o uso de memória de sistemas embarcados, foi proposta uma nova metodologia da manutenção das categorias dessa rede neural, possibilitando reduzir o consumo de recursos computacionais sem afetar a acurácia das previsões.

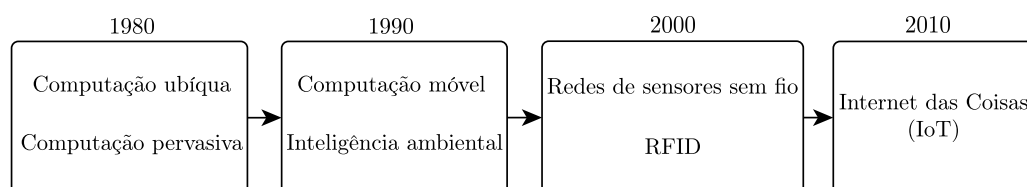
O restante do texto está organizado como segue. No Capítulo 2 apresenta-se uma revisão sobre algoritmos de ML empregados em aplicações IoT e sobre os conceitos aplicados em estudos da qualidade de ar. No Capítulo 3 apresenta-se uma introdução sobre a Teoria da Ressonância Adaptativa, incluindo a extensão de alguns operadores da rede neural ARTMAP *fuzzy* para aprimorar o treinamento on-line. O desenvolvimento do sistema proposto em software está apresentado no Capítulo 4, e o aprimoramento realizado em hardware está apresentado no Capítulo 5. Por fim, no Capítulo 6 apresentam-se as conclusões deste trabalho e possíveis perspectivas para sua continuidade.

2 ASSOCIAÇÃO DE ALGORITMOS DE *MACHINE LEARNING* À APLICAÇÕES IoT

A idealização de utilizar entidades computacionais para interagir com as pessoas surgiu na década de 1980, quando pesquisadores identificaram a possibilidade de inserir tecnologias no cotidiano. Entretanto, o nível tecnológico daquele período não permitiu que interfaces computacionais fossem incorporadas às aplicações, devido às limitações de tamanho e desempenho computacional, além de questões relacionadas ao fornecimento de energia. Mesmo assim, Mark Weiser (WEISER, 1991) foi um dos pioneiros a visualizar os benefícios que poderiam ser obtidos por meio dos conceitos que se tornariam a IoT.

Apresentam-se, na Figura 2, as áreas que contribuíram para o desenvolvimento da IoT. Inicialmente na computação ubíqua (WEISER; GOLD; BROWN, 1999) e pervasiva, definiram-se novos conceitos de software e hardware para inserir sistemas computacionais de forma transparente nos ambientes. O principal objetivo foi realizar processamento de informações em segundo plano para fornecer feedback automáticos aos usuários.

Figura 2 – Paradigmas de computação anteriores à IoT



Fonte: Adaptado de Sezer, Dogdu e Ozbayoglu (2018)

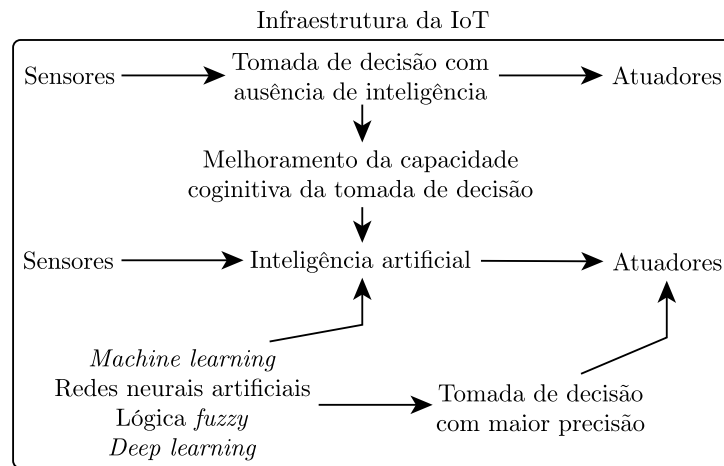
Avanços na fabricação de semicondutores durante a década de 1990 incrementaram a capacidade de extração de características em vários contextos. Com o desenvolvimento de sistemas microeletromecânicos, ou MEMS (*micro-electromechanical systems*), surgiram novos sensores e atuadores embarcados empregados na inteligência ambiental, ou Aml (*ambiental intelligence*), cujo foco é desenvolver ambientes que se adaptam para atender às necessidades dos usuários, de maneira não intrusiva (LINDWER *et al.*, 2003).

Posteriormente, o interesse na identificação por radiofrequência, ou RFID (*radio-frequency identification*), deu origem ao termo IoT, sendo mencionado primeiramente por Kevin Ashton (ASHTON, 2009). Paralelamente, avanços nas redes de sensores sem fio foram fundamentais no suporte às inovações introduzidas pela IoT e auxiliaram na

integração dos seus componentes.

Desde o surgimento das áreas relacionadas à IoT, discutiu-se a possibilidade de fornecer, aos dispositivos computacionais, uma função maior do que apenas capturar dados e atuar por meio de processos simplificados. Para isso, a tomada de decisão dos sistemas IoT foi aprimorada com técnicas de IA, como está apresentado na Figura 3. Consequentemente, dispositivos IoT se tornaram qualificados para agirem de maneira inteligente, refletindo o aprendizado obtido durante a interação com os agentes externos.

Figura 3 – Aprimoramento de sistemas IoT com técnicas de IA



Fonte: Adaptado de Alam *et al.* (2017)

Para adquirir conhecimento, os algoritmos de ML passam por processos de aprendizado realizados na fase de treinamento. Essa etapa é baseada em técnicas de treinamento supervisionado, não supervisionado e por reforço.

Na abordagem supervisionada, rotulam-se os dados de entrada para que os modelos de aprendizado produzam padrões de saídas desejados com base nas características dos dados de treinamento (SEZER; DOGDU; OZBAYOGLU, 2018). Quando os dados não possuem rótulos, empregam-se algoritmos de aprendizado não supervisionado, cuja essência é agrupar informações em conjuntos, seguindo critérios de semelhança. No aprendizado por reforço, a interação entre o agente e o ambiente permite que o algoritmo aprenda com as ações executadas para melhorar o propósito programado (VALLADARES *et al.*, 2019).

Uma importante extensão do ML é o DL. Esses modelos matemáticos são implementados por arquiteturas de redes neurais artificiais (RNA) com várias camadas que executam processamentos não lineares e permitem a extração de características hierárquicas, reproduzindo a capacidade cognitiva do cérebro humano (MOHAMMADI *et al.*, 2018).

Diversas aplicações da sociedade moderna foram beneficiadas por sistemas IoT inteligentes. Considerando centros urbanos, pode-se citar serviços inteligentes que abrangem cidadãos, economia, governo, transporte e gerenciamento sustentável do ambiente (KIRIMTAT *et al.*, 2020). A diversidade de sensores utilizados nas situações monitoradas gera dados heterogêneos, e essa abundância de dados é essencial para extrair conhecimento e calcular previsões nas cidades inteligentes (ALAM *et al.*, 2017).

A IoT também oferece ferramentas para a agricultura (LIU *et al.*, 2020) e indústria (MUHURI; SHUKLA; ABRAHAM, 2019). Pesquisadores consideram as melhorias obtidas com a utilização da IoT e áreas relacionadas (IA e *big data*) como sendo a quarta revolução desses ramos, que são denominadas agricultura 4.0 e indústria 4.0. Com o aprimoramento do gerenciamento de recursos por meio do monitoramento e das tomadas de decisões inteligentes, espera-se que a produtividade dessas áreas aumente.

A grande variedade de contextos monitorados por tecnologias derivadas da IoT e IA produz quantidades significativas de dados. Estas informações são geralmente gerenciadas como dados em formato de *big data* ou *streaming* (MOHAMMADI *et al.*, 2018). Na abordagem de *big data*, armazenam-se grandes bancos de dados que demandam altas taxas de processamento computacional e, usualmente, não exigem respostas instantâneas. Por outro lado, dados em *streaming* não são armazenados e são processados após a amostragem, sendo esperadas respostas em curtos períodos.

Para atender o desempenho de sistemas IoT envolvendo operações de IA, pode-se realizar o processamento e o gerenciamento de dados em diversos dispositivos computacionais baseados em software e em hardware. Na próxima seção, apresenta-se um estudo comparativo entre essas duas metodologias.

2.1 PLATAFORMAS IoT BASEADAS EM SOFTWARE E HARDWARE

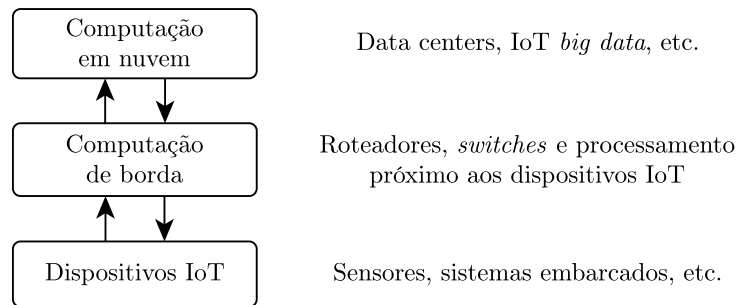
Para que as respostas de sistemas IoT sejam calculadas nos tempos estipulados, é essencial considerar as configurações da infraestrutura que controla os dados amostrados. Nesse processo são analisadas questões de comunicação entre os sensores e as demais entidades computacionais que armazenam e processam as informações.

Atualmente, existem diferentes metodologias de software e hardware para otimizar processos computacionais. Tradicionalmente, o uso de hardware na implementação de sistemas IoT associa componentes físicos como sensores, atuadores e interfaces computacionais baseadas em microprocessadores, que processam software. No entanto,

o uso de hardware também se refere à aplicação de dispositivos com arquiteturas especializadas que oferecem grandes capacidades de processamento computacional. Os principais dispositivos baseados em hardware são GPUs (*graphics processing unit*), ASICs (*application-specific integrated circuit*) e FPGAs.

Determinar a abordagem ideal de software ou de hardware para implementar sistemas IoT inteligentes requer uma análise prévia das características da aplicação. Consumo de energia, técnicas de amostragem, armazenamento de dados e tempo de resposta são alguns dos pontos que definem as melhores metodologias a serem usadas para garantir o desempenho necessário. Nesse processo, é comum considerar diferentes camadas de processamento, como está apresentado na Figura 4. Em todas as camadas é possível utilizar plataformas baseadas em software e em hardware.

Figura 4 – Camadas de processamento aplicadas em sistemas IoT



Fonte: Elaboração do próprio autor

Uma abordagem comum para processar grandes bancos de dados da IoT é o uso de serviços em nuvem. Além da grande capacidade de armazenamento, a computação em nuvem contém vários núcleos de processamento e compartilha informações entre vários servidores, permitindo o uso do processamento distribuído. Atualmente, existem serviços de computação em nuvem com processamento em hardware, como a TPU (*tensor processing unit*) do Google, que é um ASIC projetado para realizar cálculos de algoritmos de ML e DL.

Aplicações que requerem baixo tempo de resposta podem ser afetadas se forem executadas na nuvem, devido às possíveis altas latências na comunicação. Esta limitação pode ser amenizada realizando processamento mais próximo aos dispositivos IoT por meio de dispositivos presentes na computação de borda. Nesta camada de processamento há sistemas baseados em software, como microservidores e tablets (PACHECO *et al.*, 2018), e plataformas de hardware, como as FPGAs (DAMLJANOVIC; LANZA-GUTIERREZ, 2018).

O tempo de resposta de aplicações IoT é ainda mais crítico em contextos que exigem

atuação em tempo real. Uma possibilidade de otimizar o tempo de resposta é realizar processamento de dados nos próprios dispositivos que os sensores são conectados, surgindo o conceito de sensores inteligentes.

Entre os sistemas embarcados baseados em software que processam dados da IoT, os microcontroladores frequentemente gerenciam a comunicação com sensores e executam ações simples, pois possuem capacidades de armazenamento e processamento limitados. Há alternativas de sistemas baseados em software com capacidade computacional superior, como microprocessadores (AMATO *et al.*, 2017) e smartphones (ZEBIN *et al.*, 2019).

Mesmo com desempenho computacional aprimorado, sistemas embarcados baseados em software podem sofrer gargalos em aplicações em tempo real. Fatores como processamento sequencial podem limitar a taxa de processamento e de transferência de dados. Esse cenário torna-se mais crítico quando há vários sensores operando simultaneamente, pois são produzidos mais dados e há maior necessidade de gerenciamento e processamento eficiente para atender às restrições de tempo de resposta.

Uma possibilidade de aumentar a taxa de transferência de dados é utilizar dispositivos com processamento paralelo, que é uma das especialidades do hardware. Com arquiteturas projetadas para executar cálculos simultâneos, GPUs, ASICs e FPGAs podem otimizar a taxa de operações e da transferência de dados, diminuindo o tempo de resposta de sistemas IoT inteligentes.

Com foco em sistemas embarcados, as GPUs não são muito atraentes pois consomem muita energia. ASICs possuem menor consumo, no entanto, apresentam arquitetura fixa (LEE; SHIN; YOO, 2017) e o custo financeiro de implementação é alto. Por sua vez, a FPGA é uma opção mais balanceada considerando-se consumo de energia e flexibilidade para projetar diferentes arquiteturas de hardware, pois possui arquitetura reconfigurável (LACEY; TAYLOR; AREIBI, 2016). Adicionalmente, outros recursos das FPGAs que permitem otimizar algoritmos de ML são alto desempenho de cálculos, paralelismo, alto rendimento e processamento em tempo real (MOLANES *et al.*, 2018).

Uma importante etapa do projeto de arquiteturas de hardware em FPGAs é a organização de dados. Os padrões de acesso à memória devem ser analisados com o intuito de otimizar as operações matemáticas. Considerando algoritmos de ML e DL, utilizam-se vários conceitos da álgebra linear, como operações com matriz. Portanto, um aspecto importante para otimizar algoritmos de IA em FPGAs é aprimorar operações matriciais (WOEHRLE; KIRCHNER, 2018).

Uma alternativa que suporta álgebra matricial é a álgebra tensorial, cujo objetivo

é generalizar representações matriciais. Os tensores também são um tópico de pesquisa para otimizar algoritmos de DL embarcados em hardware (IBRAHIM; VALLE, 2018)

Outra flexibilidade das FPGAs é a possibilidade de modelar diferentes operações aritméticas. Para acelerar os cálculos de algoritmos inteligentes, vários pesquisadores exploram conceitos de DSP nas FPGAs. Uma metodologia comum é realizar quantização de ponto flutuante em ponto fixo. Nesse processo, é possível representar números utilizando menos bits (GUO *et al.*, 2018). Na modelagem em ponto fixo, deve-se realizar um estudo com o objetivo de determinar a quantidade necessária de bits para armazenar corretamente os resultados das operações matemáticas, como no cálculo de filtros (OSTA *et al.*, 2018).

Como descrito, as FPGAs são dispositivos digitais baseados em hardware com vários recursos que podem ser amplamente explorados em sistemas IoT embarcados e inteligentes, em que todos os estágios dos algoritmos de ML e DL são realizados. Uma das vantagens de executar o processamento no dispositivo IoT é a obtenção de respostas mais rapidamente, e é uma área que ainda precisa ser mais estudada e explorada (MOHAMMADI *et al.*, 2018).

Atualmente, um importante serviço desenvolvido na IoT é o monitoramento e a previsão da qualidade do ar. Na próxima seção apresentam-se as técnicas empregadas no estudo da qualidade do ar com foco em ambientes internos. Também é apresentada uma revisão de técnicas de IA, empregadas para analisar e prever componentes relacionados às condições do ar.

2.2 FERRAMENTAS EMPREGADAS NO MONITORAMENTO E NA PREVISÃO DA QUALIDADE DE AR

A constante evolução tecnológica traz inúmeros benefícios à vida humana. No entanto, o consumo descontrolado dos recursos naturais vem desgastando o meio ambiente, e uma das consequências é a poluição ambiental. Entre os tipos de poluição, a má qualidade do ar é uma grande ameaça à saúde e aos ecossistemas (EUROPEAN ENVIRONMENT AGENCY, 2018). O estudo da qualidade do ar tornou-se uma das preocupações ambientais mais discutidas, dando origem à várias políticas e diretrizes para fornecer informações sobre a qualidade do ar em ambientes internos e externos (PIERPAOLI; RUELLO, 2018).

A qualidade do ar em determinados ambientes pode ser medida baseando-se em

valores padronizados ou por diretrizes da qualidade do ar. Os padrões são definidos por autoridades e agências reguladoras, e as diretrizes são obtidas por meio de experimentos científicos para garantir saúde e segurança dentro das especificações consideradas (ABDUL-WAHAB *et al.*, 2015). Atualmente, existem várias entidades dedicadas ao estudo da qualidade do ar, como a Organização Mundial de Saúde, a Agência de Proteção Ambiental, ou EPA (*Environmental Protection Agency*), e a Comissão Europeia. Cada instituição pode fornecer diferentes conjuntos de informações e definições, que são adotados por diferentes regiões ou países.

Os níveis de poluentes do ar são expressos em diferentes unidades de medida, sendo os mais comuns microgramas por metro cúbico ($\mu\text{g}/\text{m}^3$), partes por milhão (ppm) e partes por bilhão (ppb) (WILLIAMS *et al.*, 2014). Alguns dos poluentes mais comuns considerados pelos sistemas de monitoramento são gás ozônio (O_3), material particulado, ou PM (*particulate matter*), monóxido de carbono (CO), dióxido de nitrogênio (NO_2), dióxido de enxofre (SO_2), chumbo (Pb), gás radônio (Rn) e compostos orgânicos voláteis, ou VOC (*volatile organic compound*).

Para classificar a qualidade do ar como segura, além de intervalos estipulados para as concentrações dos poluentes, é necessário determinar um período máximo de exposição para o cenário analisado. Existem diferentes intervalos de tempo, como os adotados pela EPA, que consistem em períodos médios de horas (1, 8 e 24) e períodos médios de meses (3 e 12) (WORLD HEALTH ORGANIZATION, 2010).

É comum apenas haver preocupação com a qualidade do ar em ambientes externos (SEGUEL *et al.*, 2017). No entanto, controlar a qualidade do ar em ambientes internos tem sido igualmente importante. Várias conferências são realizadas atualmente pela comunidade científica e agências governamentais para discutir a necessidade de melhores condições do índice da qualidade ar de ambientes internos, também mencionada como IAQ (*indoor air quality*) (BAHNFLETH, 2017).

A necessidade imediata de estudar o IAQ se deve à tendência das pessoas ficarem em ambientes fechados por até 90% do tempo (KLEPEIS *et al.*, 2001). A poluição interna pode se originar de fontes presentes nos próprios ambientes fechados, que também são suscetíveis à poluição do ar externo (WORLD HEALTH ORGANIZATION, 2010). Exemplos de ambientes internos incluem casas, locais de trabalho (escritórios), edifícios públicos (hospitais, escolas, teatros, bibliotecas e restaurantes) e cabines de veículos (TANABE, 2017).

Os danos à saúde causados pela exposição de ar poluído em ambientes internos podem

ser notados rapidamente ou a longo prazo. Os efeitos imediatos aparecem logo após o primeiro contato, podendo causar irritação nos olhos, narinas, dor de cabeça, tontura e fadiga. Esses sintomas variam de pessoa para pessoa e o tratamento pode ser feito simplesmente eliminando a fonte de poluentes do ar. Os efeitos a longo prazo aparecem após exposições repetitivas e duradouras, causando consequências mais graves como problemas respiratórios, cardíacos e podem até se tornarem fatais (ENVIRONMENTAL PROTECTION AGENCY, 2017). Apresentam-se, no Quadro 1, alguns dos poluentes mais comuns presentes em ambientes internos, as principais fontes e efeitos colaterais.

Quadro 1 – Poluentes típicos do ar em ambientes internos

Poluentes	Fonte e danos à saúde
Dióxido de carbono (CO ₂)	Fontes: aparelhos a gás e querosene não ventilados; produção de produtos de combustão; respiração humana. Danos à saúde: dificuldade de concentração; sonolência; aumento da taxa respiratória.
Monóxido e carbono (CO)	Fontes: fumaça de cigarro; combustão de combustíveis fósseis. Danos à saúde: tonturas; dor de cabeça; náusea; cianose; efeitos cardiovasculares; morte.
Formaldeído	Fontes: desgaseificação do isolamento de espuma e de madeira compensada; carpetes e tecidos; colas e adesivos; combustão de produtos, incluindo fumaça de cigarro. Danos à saúde: reações hipersensíveis ou alérgicas; erupções cutâneas; irritação dos olhos, de membranas respiratórias e mucosas; odor desagradável.
Óxidos nitrosos (NO _x)	Fontes: combustão de fornos e aparelhos a gás; fumaça de cigarro; exaustores de solda; motores a gás e a diesel. Danos à saúde: irritação dos olhos e de membranas mucosas e nasais.
Ozônio (O ₃)	Fontes: fotocopiadoras; purificadores de ar eletrostáticos; arco elétrico; poluição atmosférica. Danos à saúde: irritação dos olhos, trato respiratório e membranas mucosas; agravamento de doenças respiratórias crônicas.
Radônio (Rn)	Fontes: subsolos de edifícios; materiais de construção; águas subterrâneas. Danos à saúde: Não são conhecidos efeitos agudos para a saúde, mas a exposição prolongada pode levar ao aumento do risco de câncer de pulmão devido à radiação alfa.
Compostos orgânicos voláteis (VOCs)	Fontes: tintas; produtos de limpeza; naftalina; colas; fotocopiadoras; materiais de calafetagem de silicone; inseticidas; herbicidas; combustíveis; asfalto; vapores de gasolina; fumaça de cigarro; ralos; cosméticos e outros produtos de uso pessoal. Danos à saúde: náuseas; tontura; irritação dos olhos, trato respiratório e membranas mucosas; dor de cabeça; fadiga.

Fonte: Adaptado de Occupational Safety and Health Administration (1999)

Também é importante considerar o monitoramento interno de PM, que é composto por partículas sólidas e líquidas de diversos tamanhos dispersos no ar. Nas análises, é comum a mensuração de partículas com diâmetro de até 2,5 μm e 10 μm , que são respectivamente denominadas PM_{2.5} e PM₁₀.

Materiais particulados em ambientes internos são provenientes de atividades como culinária, combustão (queima de velas, lareiras, aquecedores sem ventilação adequada e fumaça de cigarro) e podem ser biológicos. Podem causar irritação nos olhos, nariz e garganta, agravamento dos sintomas de doenças coronárias e respiratórias e morte prematura em pessoas com doenças cardíacas ou pulmonares (ENVIRONMENTAL PROTECTION AGENCY, 2017).

Além de poluentes, outros componentes que afetam o IAQ é a temperatura e a umidade relativa do ar. Esses dois parâmetros refletem o nível de conforto térmico que está diretamente relacionado à produtividade em ambientes internos (ABDUL-WAHAB *et al.*, 2015). Sistemas de aquecimento, ventilação e ar condicionado, ou HVAC (*heating, ventilating and air conditioning*) são fundamentais para controlar o IAQ, pois influenciam o controle de temperatura e umidade, e contribuem para a diluição de poluentes em ambientes internos (FONSECA *et al.*, 2018).

A análise da qualidade do ar requer o uso de estações de monitoramento com sensores e equipamentos especializados. Tradicionalmente, estações de monitoramento da qualidade do ar em ambientes externos baseiam-se em equipamentos grandes, pesados e significativamente caros. Esses sistemas geralmente realizam poucas amostragens por hora. Entretanto, o comportamento dinâmico dos centros urbanos faz com que os níveis de poluição mudem rapidamente, o que pode tornar o monitoramento da qualidade do ar ineficiente. Outra dificuldade é aumentar a cobertura de monitoramento, pois é necessário realocar as estações ou instalar mais unidades (YI *et al.*, 2015).

No entanto, o desenvolvimento de sensores MEMS de baixo custo alterou o cenário de monitoramento e a análise da qualidade do ar (KUMAR *et al.*, 2015). A aquisição de sensores tornou-se mais acessível, auxiliando o monitoramento local para fornecer informações mais realistas em comparação às previsões meteorológicas realizadas com dados regionais (AYYALASOMAYAJULA *et al.*, 2016). Também há interesse no desenvolvimento de sensores de baixo custo voltados para o monitoramento de ambientes internos (CHOJER *et al.*, 2020; SAINI; DUTTA; MARQUES, 2021).

A diferença no custo destes sensores em relação às opções mais caras também reflete na acurácia das medições. A escolha correta dos sensores utilizados na implementação de um sistema de monitoramento da qualidade do ar deve considerar as necessidades da aplicação. As características dos sensores que devem ser analisadas incluem detecção de poluentes, alcance de medição, precisão e polarização, requisitos de calibração, tempo de resposta, durabilidade e custo (WILLIAMS *et al.*, 2014).

Há diferentes tecnologias empregadas na implementação de sensores aplicados no monitoramento da qualidade do ar. A detecção de gás geralmente é realizada por sensores eletroquímicos, catalíticos, de estado sólido (semicondutores), por absorção de radiação infravermelha e por detector de fotoionização (CHOU, 2000). Para medir PM, os sensores mais comuns são monitores de atenuação de radiação beta, microbalanças oscilantes de elemento cônico e sensores óticos (BADURA *et al.*, 2018).

A ampla disponibilidade de sensores de baixo custo empregados no monitoramento de ar em ambientes internos possibilitou o desenvolvimento de pesquisas que compreendem vários contextos atualmente. Para interpretar as informações, pode-se aplicar algoritmos de ML, pois oferecem grande capacidade de análise e reconhecimento de padrões considerando dados de IAQ (WEI *et al.*, 2019). Com o conhecimento adquirido pelos algoritmos de aprendizado é possível, por exemplo, estudar como o comportamento humano é influenciado pelo IAQ (LIN *et al.*, 2017) ou ainda, controlar atuadores para gerenciar o conforto térmico e a economia de energia em ambientes internos (VALLADARES *et al.*, 2019).

Outro serviço que envolve o estudo da qualidade de ar em ambientes internos e que sistemas IoT inteligentes são fundamentais é o cálculo de previsões dos parâmetros monitorados. Uma possibilidade para organizar dados de IAQ monitorados continuamente é por meio de séries temporais. Essa técnica é amplamente aplicada em várias abordagens de sistemas de aprendizado, como *k-means* (CARON *et al.*, 2019), RNAs tradicionais baseadas na *backpropagation* (CHEN; MIHARA; WEN, 2018) e também em redes neurais propostas no DL, como a arquitetura LSTM (*long short-term memory*) (LOY-BENITEZ *et al.*, 2019).

Assim, IoT e algoritmos de ML são áreas essenciais empregadas na implementação de sistemas de monitoramento e previsão da qualidade do ar. Mesmo com várias pesquisas explorando esse tema na atualidade, ainda é necessário investigar e desenvolver metodologias para garantir a confiabilidade dos sistemas de monitoramento da qualidade do ar em tempo real (MARQUES; PITARMA, 2019).

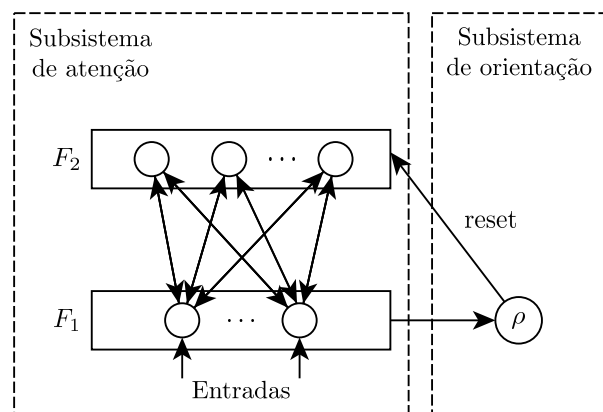
Como uma das principais contribuições desta pesquisa, propõe-se implementar e avaliar o desempenho de uma rede neural desenvolvida aplicando-se os conceitos da Teoria da Ressonância Adaptativa, mais especificamente a rede neural ARTMAP *fuzzy*, para calcular a previsão de componentes da qualidade de ar em ambientes internos. No próximo capítulo apresenta-se uma introdução dessa teoria.

3 TEORIA DA RESSONÂNCIA ADAPTATIVA

A Teoria da Ressonância Adaptativa, ou ART (*Adaptive Resonance Theory*) foi proposta em 1976 por Stephen Grossberg. É um modelo matemático inspirado pela capacidade cogitativa de sistemas biológicos baseando-se em dois dilemas: plasticidade e estabilidade. A plasticidade representa a habilidade de aprender novos padrões à medida que o sistema interage com os agentes externos do ambiente, unindo novos padrões de entradas em grupos específicos. Por outro lado, a estabilidade garante a manutenção do conhecimento previamente adquirido, separando os novos padrões criados dos já existentes (CARPENTER; GROSSBERG, 1987b).

As RNAs da família ART são modelos auto-organizáveis, que aplicam um processo de competição e busca para agrupar padrões de entrada. As ações executadas durante o aprendizado da rede ART são controladas por dois subsistemas, o subsistema de atenção e o subsistema de orientação (CARPENTER; GROSSBERG, 1987b). A arquitetura dessa RNA está apresentada na Figura 5.

Figura 5 – Estrutura elementar da rede neural ART



Fonte: Adaptado de Carpenter e Grossberg (1987b)

Os neurônios artificiais da rede ART são organizados em duas camadas localizadas no subsistema de atenção. A camada F_1 é responsável pelo processamento das informações recebidas na entrada, e a camada F_2 armazena os padrões reconhecidos em categorias. As camadas F_1 e F_2 são conectadas por vetores de pesos que armazenam o conhecimento adquirido durante a fase de treinamento.

Ao receber um padrão de entrada, inicia-se o reconhecimento dessas informações com

o cálculo da função de escolha para analisar a similaridade entre a entrada e todas as categorias existentes. No processo de busca, seleciona-se a categoria que gerar a maior função de escolha, sendo considerada a hipótese mais apta para representar o padrão de entrada.

A verificação de similaridade é realizada na comparação aplicando-se o parâmetro de vigilância, que aceita ou recusa a associação do padrão de entrada à categoria vencedora na busca. Caso o critério de vigilância não seja aceito, a categoria é excluída do processo de busca aplicando-se o *reset* por meio do subsistema de orientação.

As etapas de comparação e busca são repetidas até que uma categoria passe no teste de vigilância, fazendo com que a rede entre em ressonância e seja atualizada, isto é, treine os pesos da categoria selecionada e associada ao padrão de entrada. Quando todas as categorias são testadas e nenhuma satisfaz o teste de vigilância, uma categoria é criada para representar o novo padrão de entrada.

Dessa forma, os subsistemas de atenção e orientação operam de maneira complementar durante o treinamento da rede neural ART. O subsistema de atenção lida com informações já conhecidas, garantindo a estabilização da representação dos padrões existentes. Por outro lado, o gerenciamento de novos padrões é executado pelo subsistema de orientação, que é essencial para determinar quando é necessário criar novas categorias (CARPENTER; GROSSBERG, 1987b). Essas relações entre os subsistemas das redes ART traduzem os dilemas da plasticidade e estabilidade, que garantem a manutenção do aprendizado sem perder o conhecimento adquirido.

A primeira RNA da família ART foi apresentada em 1987, denominada ART-1 (CARPENTER; GROSSBERG, 1987b). Esse modelo processa apenas dados binários, aplicando treinamento não supervisionado para agrupar os padrões de entradas apresentados à rede. Adicionalmente, outras arquiteturas com treinamento não supervisionado foram propostas, como a ART-2, que processa tanto dados binários como analógicos (CARPENTER; GROSSBERG, 1987a). A teoria da ressonância adaptativa também foi explorada na metodologia de treinamento supervisionado por meio da rede ARTMAP (CARPENTER; GROSSBERG; REYNOLDS, 1991).

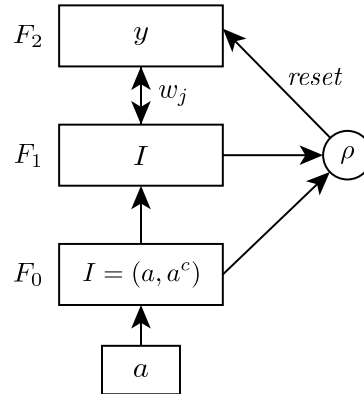
Uma importante extensão da rede neural ART surgiu com a associação de propriedades da lógica *fuzzy* (ZADEH, 1965) para obter a arquitetura ART *fuzzy* (CARPENTER; GROSSBERG; ROSEN, 1991). Esta rede é capaz de processar dados analógicos e binários, e foi aplicada na implementação da ARTMAP, surgindo a ARTMAP *fuzzy* (CARPENTER *et al.*, 1992). Nas próximas seções são apresentados detalhes dos algoritmos de algumas

RNAs da família ART.

3.1 REDE NEURAL ART *FUZZY*

Entre os modelos da família de redes neurais ART, a ART *fuzzy* é a mais difundida (SILVA; ELNABARAWY; WUNSCH, 2019). A arquitetura da ART *fuzzy* apresentada na Figura 6 é semelhante à ART-1, alterando-se apenas alguns operadores devido à agregação de propriedades da lógica *fuzzy*. A diferença básica entre as operações das redes ART-1 e ART *fuzzy* é a substituição da interseção binária, isto é, a operação AND ($\cap$), pelo operador AND da lógica *fuzzy* ($\wedge$), o que proporciona a capacidade de processar dados analógicos e binários. Pode-se inserir uma camada adicional na ART *fuzzy* (F_0), responsável pelo pré-processamento dos padrões de entrada. Os dados que compõem os padrões de entrada (a) das redes ARTs são organizados em vetores linha.

Figura 6 – Arquitetura da rede neural ART *fuzzy*



Fonte: Adaptado de Lopes (2005)

O pré-processamento da ART *fuzzy* consiste na normalização dos padrões de entrada, padronizando todos os valores no intervalo $[0,1]$. Esta operação é necessária para evitar a proliferação de categorias (CARPENTER; GROSSBERG; ROSEN, 1991). A normalização é aplicada separadamente em cada variável que forma os padrões de entrada, com o objetivo de não desconfigurar a amplitude de diferentes tipos de informações. Considerando um conjunto com n padrões de entrada, cada um com M variáveis, a normalização de uma variável pode ser realizada agrupando em um vetor v todos os n valores da variável em questão e aplicar a equação 1:

$$\hat{v} = \frac{v}{|v|} \quad (1)$$

em que $|v|$ é a norma de um vetor:

$$|v| = \sum_{i=1}^n |v_i| \quad (2)$$

Outra operação necessária no pré-processamento é a aplicação da codificação complemento definida por:

$$I = (\hat{a}, \hat{a}^c) = [\underbrace{\hat{a}_1, \dots, \hat{a}_M}_{\hat{a}}, \underbrace{1 - \hat{a}_1, \dots, 1 - \hat{a}_M}_{\hat{a}^c}] \quad (3)$$

produzindo vetores com o dobro da dimensão original ($2M$). Esta codificação preserva e unifica a amplitude dos padrões de entrada, pois a norma de todos os vetores se torna a mesma:

$$|I| = \sum_{i=1}^M a_i + \sum_{i=1}^M (1 - a_i) = M \quad (4)$$

O algoritmo ART é iniciado pela comparação da similaridade entre a entrada I e cada categoria j armazenada na camada F_2 por meio da função de escolha, que é calculada por:

$$T_j(I) = \frac{|I \wedge w_j|}{\alpha + |w_j|} \quad (5)$$

sendo w_j a matriz de pesos que conecta as camadas F_1 e F_2 , a qual armazena o conhecimento da rede neural, α ($\alpha > 0$) o parâmetro de escolha, que influencia a sequência de busca das categorias, e $(\wedge)$ o operador AND da lógica *fuzzy* definido por:

$$(x \wedge y) = \min(x_i, y_i) \quad (6)$$

A busca por uma categoria suficientemente semelhante ao padrão de entrada capaz de representá-lo é iniciada selecionando-se o valor T_j com maior valor e menor índice:

$$T_J = \max \{T_j : j = 1, \dots, N\} \quad (7)$$

A similaridade entre a categoria J e o padrão de entrada é verificada por meio do teste de vigilância calculado por:

$$\frac{|I \wedge w_J|}{|I|} \geq \rho \quad (8)$$

sendo ρ ($\rho \in [0, 1]$) o parâmetro de vigilância, que controla a ocorrência da ressonância verificando a combinação dos valores da entrada e dos pesos armazenados em w_J (LOPES, 2005). Quanto menor for o valor de ρ , maior é o grau de generalização da rede, produzindo

uma menor quantidade de categorias. Contudo, maiores valores de ρ diminuem a generalização e geram mais categorias com melhor qualidade de representação dos padrões de entradas (CARPENTER; GROSSBERG; ROSEN, 1991).

Caso a condição da inequação 8 seja satisfeita, a rede entra no estado de ressonância e ativa a atualização dos pesos w_J , calculados por:

$$w_J = \beta(I \wedge w_J) + (1 - \beta)w_J \quad (9)$$

sendo β a taxa de treinamento ($\beta \in [0, 1]$). Utilizando-se $\beta = 1$, o treinamento é dito rápido, e a atualização dos pesos ocorre em apenas um ciclo de treinamento. Quando tem-se valores $\beta < 1$, são necessários mais ciclos de aprendizagem considerando cada padrão de entrada para atualizar os pesos, o que caracteriza o treinamento lento.

Nas ocasiões em que uma categoria não satisfaz o teste de vigilância, aplica-se o *reset* ($T_J = 0$) para excluí-la da busca e o próximo maior valor T_J é o novo candidato a ser testado. Esse processo se repete até ocorrer a ressonância com alguma categoria existente. Caso contrário, o padrão de entrada ainda desconhecido pela rede é associado à uma nova categoria.

Na etapa subsequente, indica-se qual foi o neurônio vencedor selecionado para representar o padrão de entrada por meio do vetor de atividade da camada F_2 :

$$y_j = \begin{cases} 1, j = J \\ 0, j \neq J \end{cases} \quad (10)$$

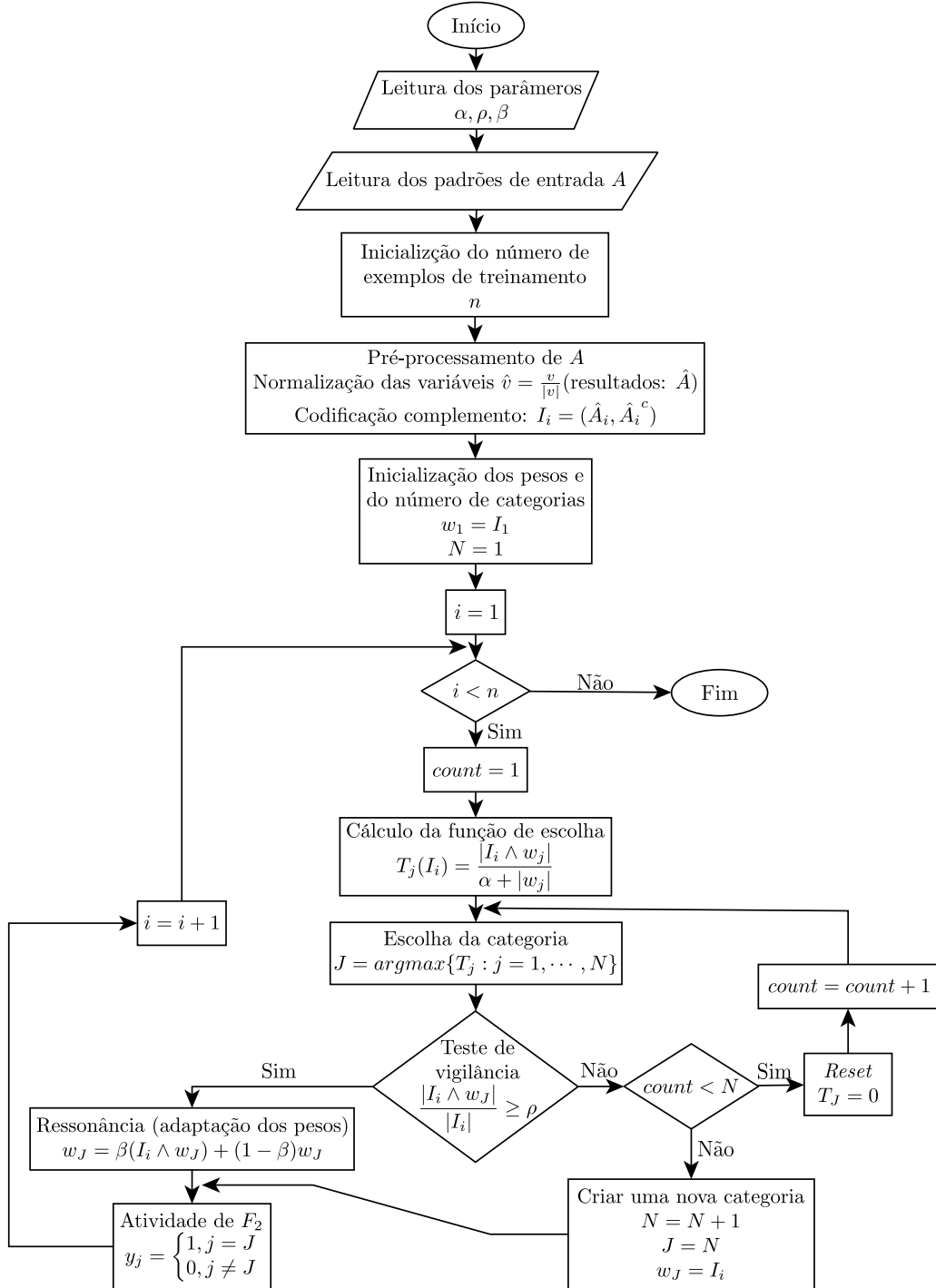
Uma alternativa otimizada para implementar a rede ART *fuzzy* é a adição de novas categorias de acordo com a necessidade. Dessa forma, no início do treinamento há apenas uma categoria ($N = 1$) com os pesos w_1 inicializados com os valores da primeira entrada. A cada entrada apresentada à rede, utiliza-se um contador ($count = 1$) para armazenar a quantidade de categorias verificadas durante a busca, até que haja ressonância.

Caso o padrão de entrada seja inédito, isto é, quando nenhum neurônio da camada F_2 passar no teste de vigilância, o novo padrão de entrada é adicionado ao conhecimento da rede criando-se uma categoria. Consequentemente, insere-se uma linha na matriz de pesos w_j com os valores da entrada, e o número de categorias é incrementado em uma unidade.

Apresenta-se, na Figura 7, o fluxograma das operações aplicadas no treinamento da ART *fuzzy*. A leitura de n padrões de entrada, os quais correspondem aos exemplos de treinamento, é realizada apenas uma vez e os dados são armazenados na matriz A , que

possui n linhas e M colunas. Foi considerada a metodologia de iniciar o algoritmo com apenas uma categoria aplicando o treinamento rápido. Um contador auxiliar (i) verifica a quantidade de padrões apresentados à rede até a finalização do treinamento.

Figura 7 – Fluxograma da rede neural ART *fuzzy*



Fonte: Adaptado de Lopes (2005)

É importante ressaltar que o treinamento das redes neurais ARTs não se limita a um grupo fixo de exemplos de treinamento, pois os dilemas da plasticidade e da estabilidade

garantem simultaneamente a adição de novas categorias e a manutenção do conhecimento adquirido. Logo, essas redes são capazes de aprenderem e reconhecerem novos padrões continuamente.

Mesmo empregando treinamento não-supervisionado, a arquitetura da ART *fuzzy* gera a saída y na camada F_2 , que indica o índice da categoria ativa caso ocorra ressonância. Contudo, esse processo não caracteriza efetivamente uma classificação do modo que ocorre nos algoritmos de ML com treinamento supervisionado, cujas respostas baseiam-se em valores utilizados durante o treinamento para identificar os padrões reconhecidos.

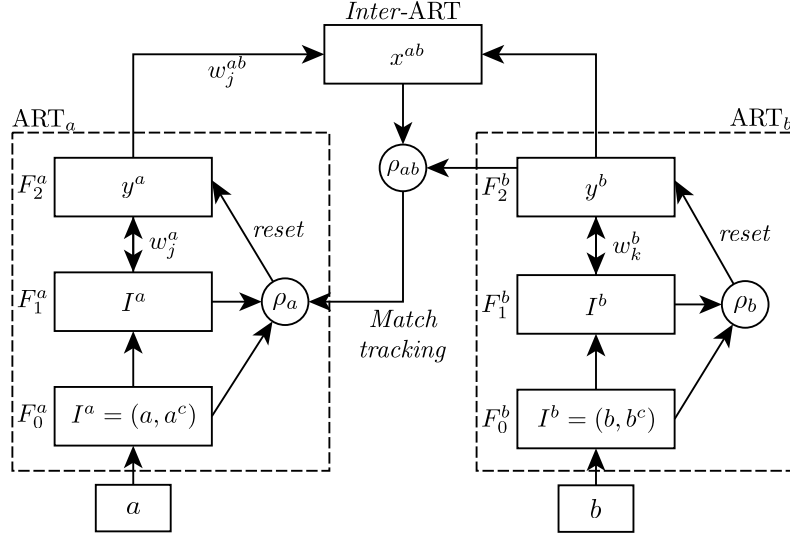
A família das redes ARTs também possui uma arquitetura baseada no treinamento supervisionado, que é denominada ARTMAP. Assim como a ART *fuzzy*, também foram associadas operações da lógica *fuzzy* às operações da ARTMAP, surgindo a ARTMAP *fuzzy*. Na sequência apresentam-se mais detalhes desta rede neural.

3.2 REDE NEURAL ARTMAP FUZZY

A ARTMAP *fuzzy* é um modelo de RNA auto-organizável que emprega a abordagem de treinamento supervisionado para reconhecer sucessões aleatórias de padrões de entrada. Utilizando-se um método de mapeamento que combina a atividade de dois módulos ART *fuzzy*, este sistema oferece uma aprendizagem estável e incremental (CARPENTER *et al.*, 1992).

Apresenta-se, na Figura 8, a arquitetura da rede ARTMAP *fuzzy*. Os módulos ART_a e ART_b são responsáveis por aprenderem e agruparem os padrões de entrada e saída, respectivamente, considerando parâmetros de vigilância ρ_a e ρ_b e empregando o algoritmo da rede ART *fuzzy*, que podem ser processados paralelamente. Sempre que a rede recebe um novo padrão de entrada, o parâmetro de vigilância da ART_a é inicializado com o valor base $\overline{\rho_a}$.

Durante o treinamento da ARTMAP *fuzzy*, apresentam-se pares (a, b) à rede, sendo a a entrada e b a saída associada. Com as classificações disponíveis nas camadas de saída F_2^a e F_2^b , o módulo Inter-ART recebe essas informações para gerenciar o mapeamento entre as categorias de entrada e saída ativas por meio de um processo denominado *match tracking*. Entre a camada F_2^a e o módulo Inter-ART, adiciona-se uma matriz de pesos (w_j^{ab}) , a qual é atualizada para armazenar o mapeamento entre as categorias ativas de entrada (T_J) e saída (T_K) , o que só acontece se o teste de vigilância aplicado no *match tracking* ocorrer, que é calculado por:

Figura 8 – Arquitetura da rede neural ARTMAP *fuzzy*

Fonte: Adaptado de Carpenter *et al.* (1992)

$$\frac{|y^b \wedge w_j^{ab}|}{|y^b|} \geq \rho_{ab} \quad (11)$$

sendo y^b o vetor de atividade da camada F_2^b e ρ_{ab} o parâmetro de vigilância do módulo Inter-ART. Quando a inequação 11 não é satisfeita, aplica-se um *reset* na categoria J e o parâmetro ρ_a é incrementado ligeiramente, calculado por:

$$\rho_a = \frac{|I^a \wedge w_j^a|}{|I^a|} + \varepsilon \quad (12)$$

em que ε é o incremento aplicado, que corresponde a um valor positivo próximo de zero. Esta operação é realizada para que não ocorra novamente a classificação incorreta da entrada atual no módulo ART_a . Inicia-se então a busca por uma categoria que se associe corretamente à saída desejada da ART_b . Caso nenhuma categoria existente seja associada ao padrão de entrada, cria-se uma categoria na camada F_2^a , adicionando uma linha à matriz de pesos w^{ab} para mapear os novos dados à saída armazenada na ART_b . Quando se obtém uma classificação correta entre os padrões de entrada e saída, as conexões de mapeamento destas categorias são atualizadas calculando-se:

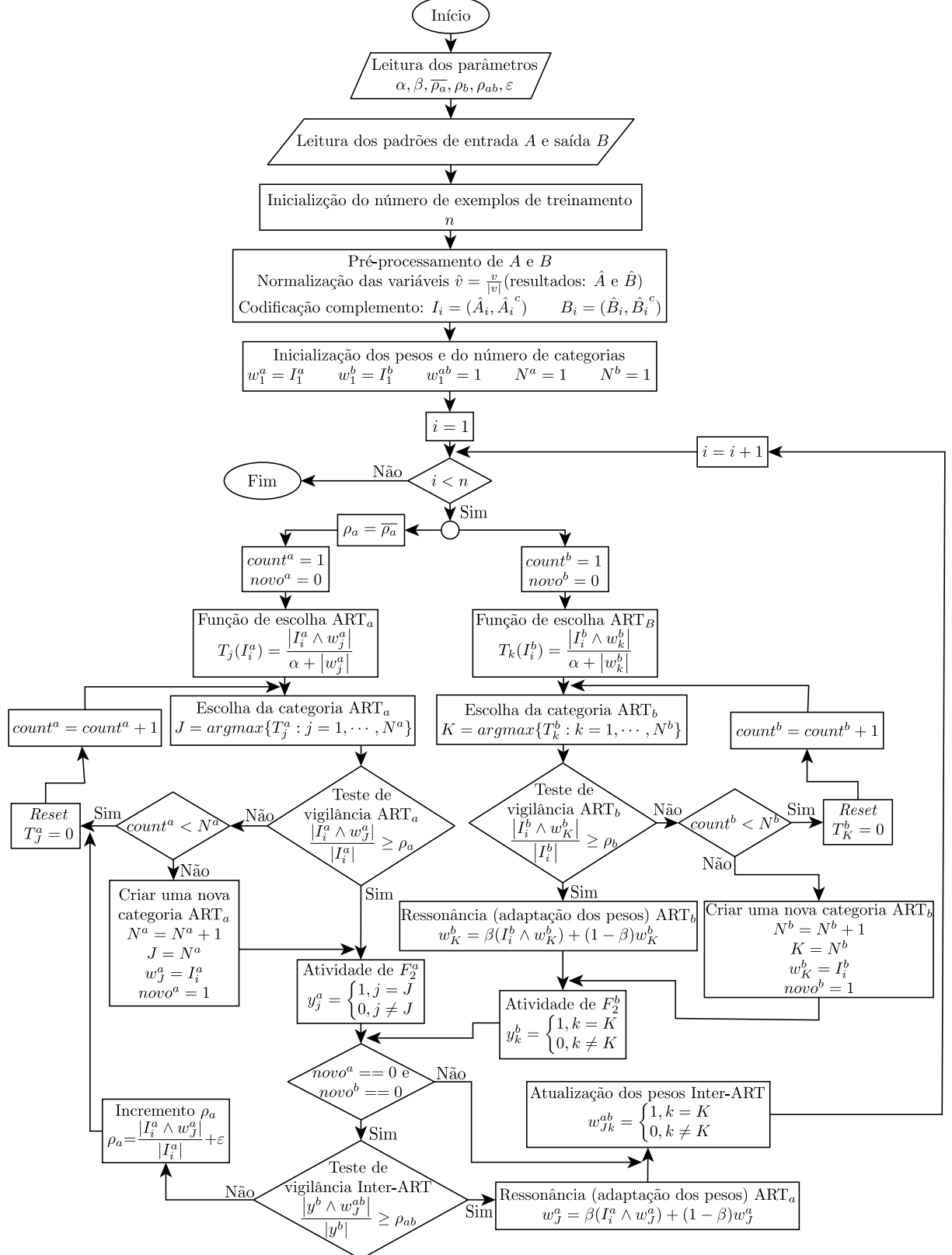
$$w_{Jk}^{ab} = \begin{cases} 1, k = K \\ 0, k \neq K \end{cases} \quad (13)$$

fazendo com que a conexão entre as categorias ativas (J e K) seja preenchida com o valor 1 e os demais índices da linha J de w^{ab} sejam 0.

Apresenta-se, na Figura 9, o fluxograma do treinamento da rede ARTMAP *fuzzy*

considerado neste trabalho. Esta metodologia de aprendizado possui algumas diferenças em relação ao algoritmo original proposto por Carpenter *et al.* (1992).

Figura 9 – Fluxograma da rede neural ARTMAP *fuzzy* com treinamento off-line



Fonte: Adaptado de Barros (2013)

Assim como na rede ART *fuzzy* apresentada, considerou-se a inicialização da aprendizagem da ARTMAP *fuzzy* com apenas uma categoria ($N^a = 1$ e $N^b = 1$) com os dados da primeira entrada ($w_1^a = I_1^a$) e da respectiva saída ($w_1^b = I_1^b$), incrementando esses parâmetros conforme a ocorrência de novos padrões de entrada. Os pesos da matriz w^{ab} são inicializados com o valor 1, indicando que a categoria recém-criada ainda não foi associada a nenhum padrão de saída. Os exemplos de treinamento são organizados em duas matrizes, que contém os padrões de entrada (A) e as respectivas saídas (B).

Outra mudança ocorre na aplicação do teste de vigilância do módulo Inter-ART, que sempre é verificado no algoritmo tradicional da ARTMAP *fuzzy*. Contudo, a rede utilizada neste trabalho somente ativa o módulo Inter-ART se a busca por categorias para representarem o padrão de entrada considerado não criar nenhuma categoria nos módulos ART *fuzzy* (BARROS, 2013). Essa verificação é realizada por meio de variáveis auxiliares ($novo^a$ e $novo^b$), que indicam quando categorias são criadas.

A aprendizagem apresentada no fluxograma da Figura 9 pode ser caracterizada como uma abordagem de treinamento off-line, pois considera um conjunto fixo de exemplos de treinamento. Logo, sistemas de ML baseados nessa técnica de aprendizagem costumam calcular predições isoladamente do treinamento do algoritmo, que é realizado apenas uma vez com dados previamente amostrados.

O treinamento off-line pode não garantir boas predições em aplicações cujos dados são amostrados em tempo real, pois os exemplos de treinamento podem tornarem-se obsoletos e não representarem corretamente possíveis mudanças no comportamento dos sinais ao decorrer do tempo.

Uma alternativa para manter o algoritmo de predição sempre atualizado é realizar a aquisição de conhecimento constantemente, que é uma característica do treinamento on-line. Nessa metodologia, pode-se utilizar dados amostrados antes do início da operação do algoritmo, assim como é feito no treinamento off-line, contudo a aprendizagem continua durante a operação do sistema em tempo real. Dessa forma, o sistema inteligente realiza predições e simultaneamente incrementa o conhecimento por meio do treinamento considerando os padrões de entrada recém amostrados.

A ARTMAP *fuzzy* é uma RNA interessante para implementar sistemas de aprendizagem on-line, pois armazena o conhecimento adquirido e possibilita a criação de novas categorias para incrementar a capacidade de classificação de novos padrões.

Embora a arquitetura tradicional da ARTMAP *fuzzy*, apresentada na Figura 8, suporte o processamento do treinamento on-line, apresenta-se na próxima subseção o

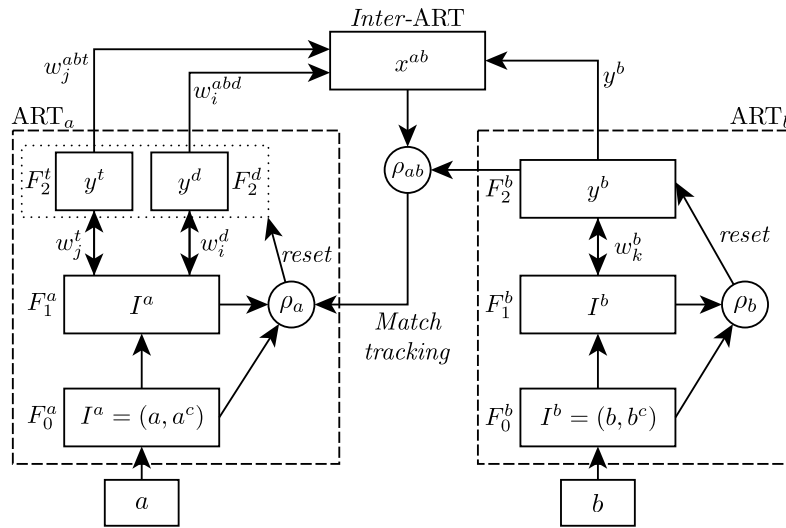
desenvolvimento de uma variação da rede ARTMAP *fuzzy* para aprimorar o treinamento on-line e a predição de aplicações em tempo real.

3.2.1 Rede neural ARTMAP *fuzzy* com treinamento on-line

Sempre que a rede neural ARTMAP *fuzzy* recebe um padrão de entrada desconhecido durante a aprendizagem, cria-se uma categoria para incorporar essas novas informações à arquitetura da rede. Esta característica é essencial para manter o conhecimento do sistema sempre atualizado, mas pode tornar-se inconveniente dependendo da operação do sistema. Um exemplo é considerar a implementação da ARTMAP *fuzzy* com treinamento on-line em um sistema embarcado. O processamento constante durante extensos períodos produziria um elevado número de categorias, exigindo quantidades de memórias significativas, o que pode ser um recurso limitado em sistemas embarcados.

Uma alternativa para amenizar esse problema é a realização de um gerenciamento mais adequado para permitir a criação de novos neurônios na ARTMAP *fuzzy*. Foi com este objetivo que Marchiori *et al.* (2011) propôs uma mudança na camada F_2^a , separando as categorias que armazenam as informações dos padrões de entrada em dois grupos: categorias temporárias (F_2^t) e categorias definitivas (F_2^d), como pode ser visualizado na arquitetura apresentada na Figura 10. Cada conjunto de categorias possui distintas matrizes de pesos entre as camadas que estão interligadas.

Figura 10 – Arquitetura da rede neural ARTMAP *fuzzy* com treinamento on-line



Fonte: Elaboração do próprio autor

No grupo temporário, são alocadas as categorias recém-criadas no módulo ART_a . O critério para uma categoria temporária tornar-se definitiva é que ela seja selecionada

um número mínimo de vezes ($Nmin$) para representar padrões de entrada durante o treinamento. Por outro lado, excluem-se as categorias temporárias que não forem selecionadas para classificar os padrões de entrada após a rede receber um determinado número de entradas ($Nmax$), já que são consideradas como informações irrelevantes.

Aplicando esse gerenciamento na ARTMAP *fuzzy*, garante-se a constante atualização do conhecimento da rede neural, mantendo-se as informações mais importantes no grupo de categorias definitivas e, simultaneamente, se faz um uso mais otimizado de memória, pois descartam-se as informações com pouca frequência de uso.

As etapas dessa abordagem de treinamento estão apresentadas no fluxograma da Figura 11. Se houver a disponibilidade de dados amostrados anteriormente, o treinamento da ARTMAP *fuzzy* pode começar na abordagem off-line, como demonstrada na Figura 9.

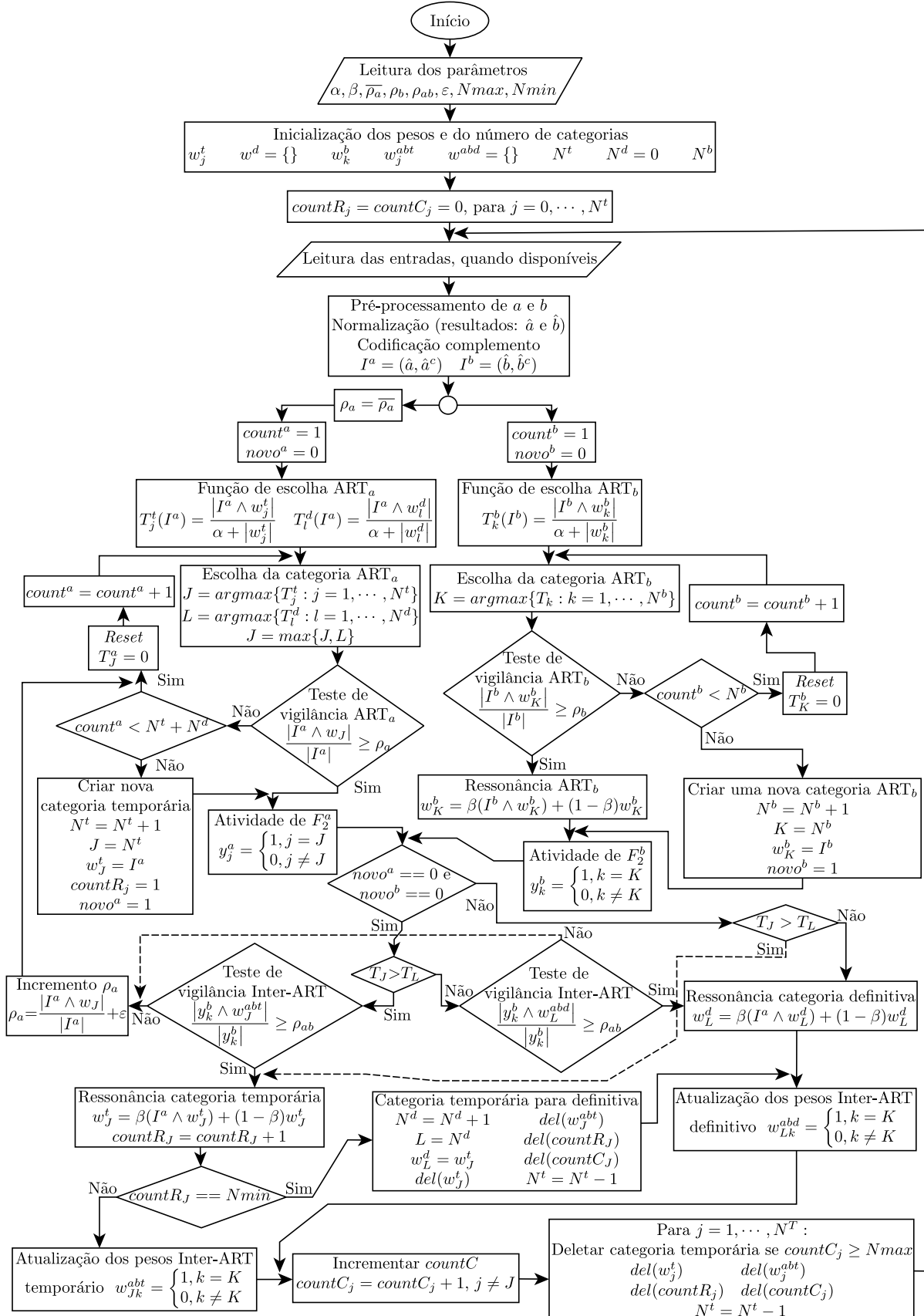
No início da fase on-line, o conhecimento obtido durante o treinamento off-line é alocado no grupo de categorias temporárias. Por outro lado, se a aprendizagem for iniciada diretamente no modo on-line, a rede não possui nenhum conhecimento temporário. Independentemente da inicialização das categorias temporárias consideradas, o treinamento on-line sempre começa sem neurônio definitivo, sendo o número de categorias definitivas nulo ($N^d = 0$) e as matrizes de pesos relacionadas definidas como vazias ($w^d = w^{abd} = \{\}$).

Para realizar o gerenciamento das categorias da camada F_2^a , dois novos contadores são implementados por vetores que possuem o número de posições igual ao número de categorias temporárias (N^t). O $countR_j$ armazena o número de ressonâncias que cada categoria temporária obteve, o qual é comparado com o parâmetro $Nmin$ para tornar-se definitivo. Já o contador $countC_j$ armazena a quantidade de ciclos de treinamento, isto é, o número de entradas apresentadas à rede, que as categorias não foram escolhidas no teste de vigilância para excluir as informações com pouca frequência de uso.

No treinamento on-line, o processamento do módulo ART_b é o mesmo aplicado na aprendizagem off-line. Na ART_a , a busca por uma categoria que atenda ao parâmetro de vigilância ρ_a considera todas as categorias temporárias e definitivas, sendo escolhida a que possuir a maior função de escolha. Quando nenhum neurônio da camada F_2^a classifica um novo padrão de entrada, cria-se uma categoria temporária com estas informações e adiciona-se uma posição no vetor $countR$, inicializada com o valor 1.

Na etapa subsequente, o módulo Inter-ART é ativado se nenhuma categoria for criada nos módulos ART_a e ART_b durante a busca por neurônios que classifiquem as entradas da ARTMAP *fuzzy*. Caso não haja nenhuma categoria nova, é verificado se o neurônio

Figura 11 – Fluxograma da rede neural ARTMAP *fuzzy* com treinamento on-line



da ART_a selecionado na busca é temporário ou definitivo, comparando-se os valores máximos das funções de escolha (T_J e T_L) para considerar as variáveis corretas no teste de vigilância Inter-ART. Quando a condição estipulada por ρ_{ab} não é satisfeita, incrementa-se o parâmetro ρ_a , aplica-se o *reset* e a busca por outra categoria da camada F_2^a continua até que todas sejam verificadas, ou uma nova seja adicionada.

Quando o teste de vigilância Inter-ART é satisfeito, realiza-se a ressonância na categoria ativa da rede ART_a , a qual já foi classificada como temporária ou definitiva antes dos cálculos executados no módulo Inter-ART. Contudo, se uma categoria foi criada durante o processamento dos módulos ART *fuzzy*, isto é, ao menos uma das variáveis *novo^a* ou *novo^b* é diferente de zero, também é necessário verificar os valores máximos das funções de escolha para identificar se a ressonância da ART_a será realizada em uma categoria temporária ou definitiva.

Com a finalização do processamento dos módulos ARTs, se a categoria ativa da ART_a for definitiva, a atualização dos pesos do módulo Inter-ART para realizar o mapeamento entre as atividades dos módulos ART_a *fuzzy* e ART_b *fuzzy* é realizada automaticamente. Por outro lado, quando a categoria ativa é temporária, incrementa-se o contador de ressonâncias *countR* no índice correspondente e é verificado se essa categoria obteve o número mínimo *Nmin* de ressonâncias para tornar-se definitiva. Se o neurônio permanecer no grupo temporário, apenas atualiza-se os pesos Inter-ART temporários relacionados. Mas se foi obtido o número de ressonâncias necessárias, a categoria torna-se definitiva e aplica-se a atualização dos pesos Inter-ART correspondentes no grupo definitivo.

A etapa subsequente à atualização dos pesos Inter-ART é atualizar a quantidade de ciclos das categorias temporárias não ativadas e verificar quais serão excluídas por terem ficado *Nmax* ciclos sem classificar uma entrada. Sempre que um padrão de entrada é apresentado à rede, as etapas da aprendizagem on-line são executadas.

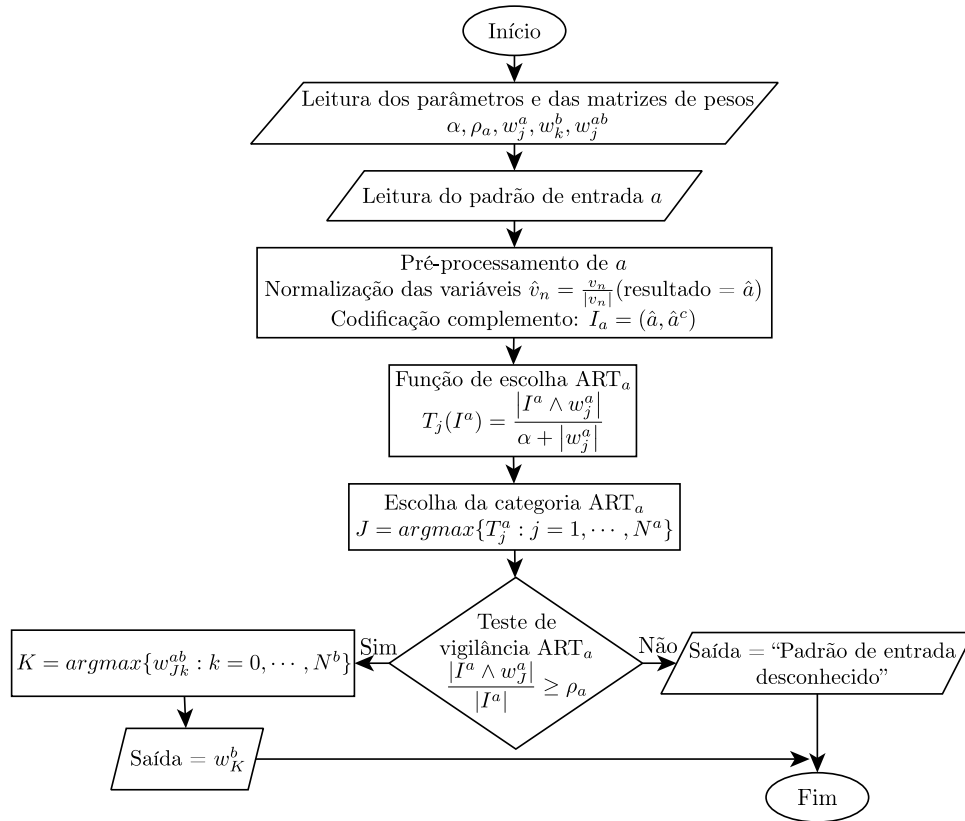
3.2.2 Cálculo de uma predição com a rede neural ARTMAP *fuzzy*

Para realizar predições com a ARTMAP *fuzzy*, é necessário apenas do vetor de entrada *a*. Nesse caso, o sistema de classificação interno executado no algoritmo do módulo ART_a *fuzzy* associa o padrão de entrada à categoria *J* da camada F_2^a com maior função de escolha para verificar o teste de vigilância ART_a . Caso o teste de vigilância seja satisfeito, o resultado da predição é a categoria *K* do módulo ART_b *fuzzy* que foi mapeada à categoria *J* durante o treinamento. A verificação do mapeamento é realizada na linha *J* da matriz w^{ab} , sendo a categoria *K* identificada pela posição que possuir o valor 1 dessa linha. A

resposta da rede neural ao padrão de entrada corresponde à linha K da matriz de pesos w_b . Se o teste de vigilância não for satisfeito durante a classificação, significa que o padrão de entrada é novidade para o conhecimento da rede, e nenhuma resposta é obtida.

O fluxograma das etapas empregadas na predição da rede ARTMAP *fuzzy* está apresentado na Figura 12. Ressalta-se que na abordagem de treinamento on-line as predições são calculadas em tempo real e concorrentemente ao treinamento da rede. Logo, é possível utilizar a mesma implementação para processar o treinamento e o cálculo das predições, sendo necessário identificar a situação que os dados de entrada representam.

Figura 12 – Fluxograma do cálculo de uma predição da rede ARTMAP *fuzzy*



Fonte: Elaboração do próprio autor

Na Figura 12 não foi representada a separação das categorias do módulo ART_a *fuzzy* que ocorre na arquitetura especializada aplicada no treinamento on-line. Quando há a distinção de categorias definitivas e temporárias da ART_a *fuzzy*, na predição ambas as matrizes de pesos w_j^t e w_l^d são consideradas nos cálculos da função de escolha para verificar a associação do padrão de entrada com uma categoria do módulo ART_a *fuzzy*.

Apresenta-se no próximo capítulo a implementação da rede neural ARTMAP *fuzzy* com treinamento on-line para realizar previsões em tempo real da qualidade de ar de ambientes internos.

4 PREVISÃO DE COMPONENTES DA QUALIDADE DE AR COM A REDE NEURAL ARTMAP *FUZZY*

O desenvolvimento de sistemas IoT baseados em sensores ambientais permite realizar a análise de várias características físicas e verificar a existência de componentes com potencial de risco à saúde e ao bem-estar. Com monitoramento constante, pode-se emitir informações e alertas em tempo real, contendo estatísticas dos compostos monitorados. Esses sistemas podem ser aprimorados aplicando-se algoritmos de ML para processarem os dados amostrados. Consequentemente, a extração do conhecimento é realizada com maior eficácia, tornando possível a implementação de serviços de previsões baseando-se nas tendências das situações estudadas.

Um dos objetivos desse trabalho foi o desenvolvimento de um algoritmo de ML, com treinamento on-line, capaz de realizar a previsão de dados ambientais em tempo real. Com foco na qualidade de ar em ambientes internos, propõe-se utilizar sensores de baixo custo para amostrar sinais do ambiente monitorado e uma rede neural ARTMAP *fuzzy* com treinamento on-line para calcular as previsões. Apresentam-se, na sequência, os dispositivos e as configurações do sistema projetado.

4.1 SISTEMA PROPOSTO

Para realizar a análise da qualidade de ar em ambientes internos, foram considerados alguns dos componentes mais usuais examinados pela literatura especializada. Dessa forma, os seguintes sensores foram utilizados:

- BME680: desenvolvido pela Bosch, é um sensor digital para medição de gás, pressão, umidade e temperatura com alta acurácia (BOSCH SENSORTEC, 2019);
- MH-Z19B: sensor da Winsen Electronics Technology que realiza medições de CO₂ por meio de raio infravermelho não dispersivo (WINSEN, 2016);
- SGP30: produzido pela Sensirion (SENSIRION, 2020a), é um sensor de gás especializado na leitura de VOC;
- PMS7003: sensor de PM produzido pela Plantower (PLANTOWER, 2016). Realiza a leitura de material particulado com diâmetros de até 1, 2,5 e 10 micrômetros.

Apresenta-se, no Quadro 2, a tecnologia de fabricação, a acurácia de leitura e a faixa de detecção dos sensores citados. A acurácia dos sensores MH-Z19B e do PMS7003 são compostas por dois valores, sendo o primeiro uma porcentagem de variação em relação ao valor da leitura e o segundo uma variação adicional considerando o composto amostrado. O sensor SGP30 é sensível a vários gases, sendo os principais o etanol e hidrogênio, com 10% e 15% de acurácia de medição, respectivamente.

Quadro 2 – Informações dos sensores utilizados nesse trabalho

Sensor	Tecnologia de fabricação	Leitura	Acurácia	Faixa de detecção
BME680	Óxido metálico	IAQ Temperatura Umidade Pressão	- 0~65°C: $\pm 1^\circ\text{C}$ 20~80% de umidade a 25°C: $\pm 3\%$ 0~65°C: $\pm 0,6 \text{ hPa}$	0~500 -40°C~85°C 0%~100% de umidade 300~1100 hPa
MH-Z19B	Infravermelho	CO_2	3% + $\pm 50 \text{ ppm}$	0~5000 ppm
SGP30	Óxido metálico	VOC	10% a 15%	0~60000 ppb
PMS7003	Difração a laser	Material particulado	$\text{PM}_{2.5}$: 10% + $\pm 10 \mu\text{g}/\text{m}^3$	$\text{PM}_{2.5}$: 0~500 $\mu\text{g}/\text{m}^3$

Fonte: Elaboração do próprio autor

Com relação ao IAQ medido pelo BME680, trata-se de um índice da qualidade do ar calculado pelo BSEC (Bosch *Sensortec Environmental Cluster*), que é um software de código fechado de propriedade da Bosch. Como resultado bruto, o sensor de gás gera um valor de resistência correspondente à concentração de poluentes existentes no ar, incluindo gases provenientes de pinturas e lixo, além de VOCs eliminados pelo cozimento e consumo de alimentos, pela respiração e pela sudorese. A partir dos valores da resistência, o algoritmo do BSEC se calibra automaticamente considerando medições recentes, o que garante eficientes e consistentes medições de IAQ (BOSCH SENSORTEC, 2019).

O IAQ calculado pelo BSEC varia de 0 a 500. Nessa escala, maiores valores representam piores condições de poluição do ambiente. Após a calibração do BSEC, que geralmente ocorre em até 4 dias após o início de operação, o valor de IAQ tipicamente associado à boa qualidade de ar fica em torno de 25, enquanto IAQs com valores próximos de 250 caracterizam ar poluído (BOSCH SENSORTEC, 2019). No Quadro 3 apresentam-se os níveis de poluição associados a diferentes faixas de valores IAQ e as ações sugeridas para amenizar a ação dos poluentes.

Quadro 3 – Classificação do IAQ calculado pelo BSEC

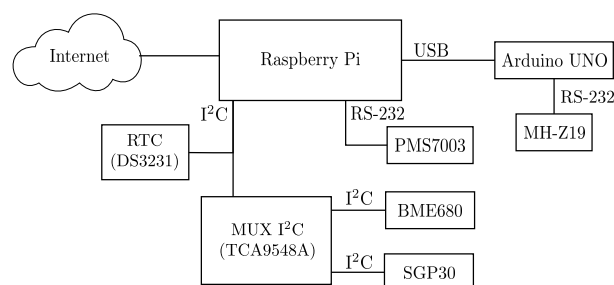
IAQ	Qualidade do ar	Impacto de exposição em longos períodos	Ação sugerida
0-50	Excelente	Ar puro, indicado para o bem-estar	Nenhuma ação necessária
51-100	Boa	Sem irritação ou impacto ao bem-estar	Nenhuma ação necessária
101-150	Levemente poluído	Possível redução do bem-estar	Ventilação sugerida
151-200	Moderadamente poluído	Possibilidade de irritações mais significantes	Aumentar a ventilação com ar puro
201-250	Fortemente poluído	A exposição pode levar a efeitos como dor de cabeça, dependendo do tipo de VOCs	Otimizar a ventilação
251-350	Severamente poluído	Problema de saúde mais grave possível se houver a presença de VOCs prejudiciais	Contaminação deve ser identificada se o nível for atingido mesmo sem a presença de pessoas. Maximizar a ventilação e reduzir a ocupação do ambiente
>351	Extremamente poluído	Dores de cabeça e possíveis problemas neurotóxicos	Contaminação precisa ser verificada. Evitar a ocupação do ambiente e maximizar a ventilação

Fonte: Adaptado de Bosch Sensortec (2019)

4.1.1 Amostragem de dados

O sistema embarcado para amostrar e armazenar os dados da qualidade do ar foi implementado utilizando um Raspberry Pi 3 modelo B. As conexões dos sensores com o minicomputador estão apresentadas na Figura 13. O relógio do sistema foi sincronizado com um relógio de tempo real modelo DS3231 conectado ao barramento da comunicação I²C. Para respeitar o limite de corrente da comunicação I²C, o BME680 e o SGP30 foram conectados através de um multiplexador TCA9548A.

O Raspberry Pi 3 possui apenas uma interface RS-232, que foi utilizada na comunicação do PMS7003. Utilizou-se um Arduino Uno conectado à porta USB do Raspberry Pi 3 para obter-se uma interface RS-232 adicional, sendo o MH-Z19B conectado ao microcontrolador. Para aumentar a acessibilidade os dados amostrados foram armazenados na nuvem por meio da conexão do Raspberry Pi 3 à Internet.

Figura 13 – Diagrama de ligação dos sensores ao Raspberry Pi 3

Fonte: Elaboração do próprio autor

O ambiente considerado nesta avaliação foi um quarto onde reside uma pessoa. Devido

aos trabalhos remotos desenvolvidos em ambientes domésticos durante a pandemia, esse quarto também foi utilizado como escritório durante o período de amostragem dos dados. Trata-se de um cômodo, com aproximadamente 11 metros quadrados, situado em um apartamento na cidade de Ilha Solteira, São Paulo. O conjunto de hardware foi instalado a cerca de 50 centímetros do chão. As taxas de amostragem dos sensores foram definidas como segue:

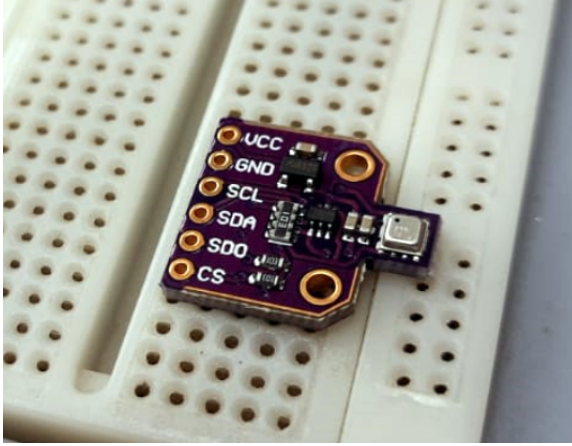
- BME680 e SGP30: o BME680 pode operar em três modos utilizando-se a biblioteca BSEC. No modo contínuo, a amostragem é de 1 Hz, sendo recomendada apenas para testes rápidos. Em aplicações a longo prazo, recomenda-se a utilização dos modos baixo consumo (*low power*) ou ultra baixo consumo (*ultra low-power*), que respectivamente empregam taxas de amostragem de 0,33 Hz e 3,3 mHz. Nesta aplicação escolheu-se o modo de operação de baixo consumo, obtendo-se amostras a cada 3 segundos. Utilizou-se a mesma taxa de amostragem no SGP30;
- MH-Z19B: o tempo de resposta da leitura de 90% do CO₂ (T₉₀) captado pelo MH-Z19B é no máximo 120 segundos. Dessa forma, verificou-se que repentinas leituras não geraram variações expressivas nos valores, sendo definido uma taxa de amostragem de 0,25 Hz para obter leituras em intervalos de 15 segundos;
- PMS7003: no modo contínuo de operação, o PMS7003 realiza leituras na frequência aproximada de 1 Hz, sendo a adotada nesta avaliação.

É importante ressaltar que o condicionamento dos sinais dos sensores não foi foco dessa pesquisa. Considerando os sensores de gás, foram adquiridas placas de circuito impresso já implementadas pela fabricante CJMCU, que contém o BME680 e o SGP30 e de outros componentes elétricos. A aquisição de dados do BME680 foi realizada utilizando-se as bibliotecas disponibilizadas pela Bosch, e no SGP30 as instruções da comunicação I²C foram implementadas. Em relação aos dispositivos com comunicação RS-232, isto é, o PMS7003 e o MH-Z19B, os dados recebem tratamentos definidos pelos fabricantes e são disponibilizados por meio de protocolos seriais. Os sensores utilizados nessa pesquisa estão apresentados na Figura 14. As diferentes quantidades de dados produzidas pelas taxas de amostragens distintas foram padronizadas, calculando-se a média no período de uma hora, o que resultou em 24 valores de cada variável por dia de amostragem.

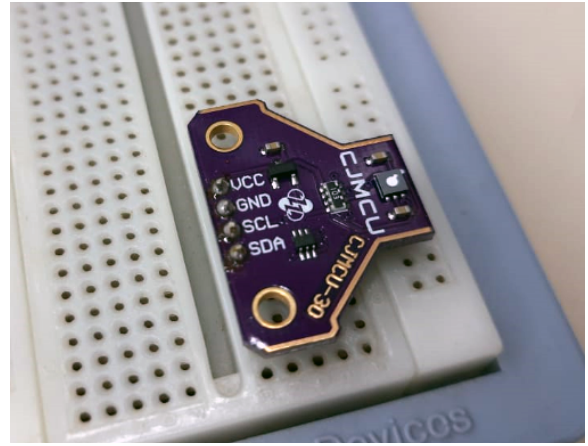
Para possibilitar o monitoramento da qualidade do ar do ambiente assistido em tempo real, implementou-se um servidor de computação de borda no Raspberry Pi 3. Esse módulo do sistema embarcado foi configurando aplicando-se o Node-Red, que é uma ferramenta utilizada com frequência no gerenciamento do fluxo de dados em aplicações

IoT. Um exemplo de leitura dos dados e os nós da configuração no Node-Red estão apresentadas no Apêndice A, respectivamente na Figura 37 e na Figura 38.

Figura 14 – Sensores utilizados nessa pesquisa



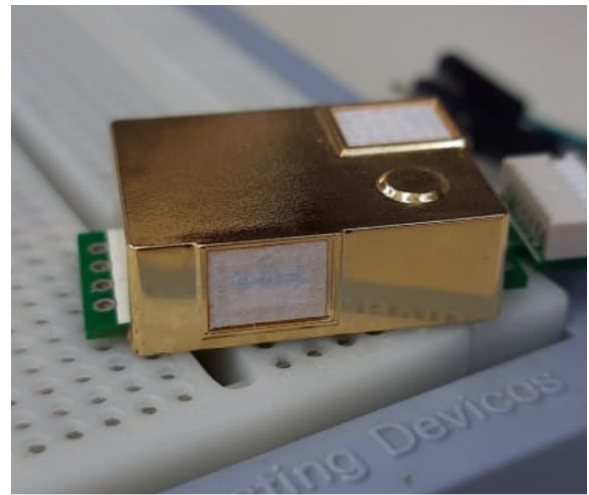
(a)BME680



(b)SGP30



(c)PMS7003



(d)MH-Z19B

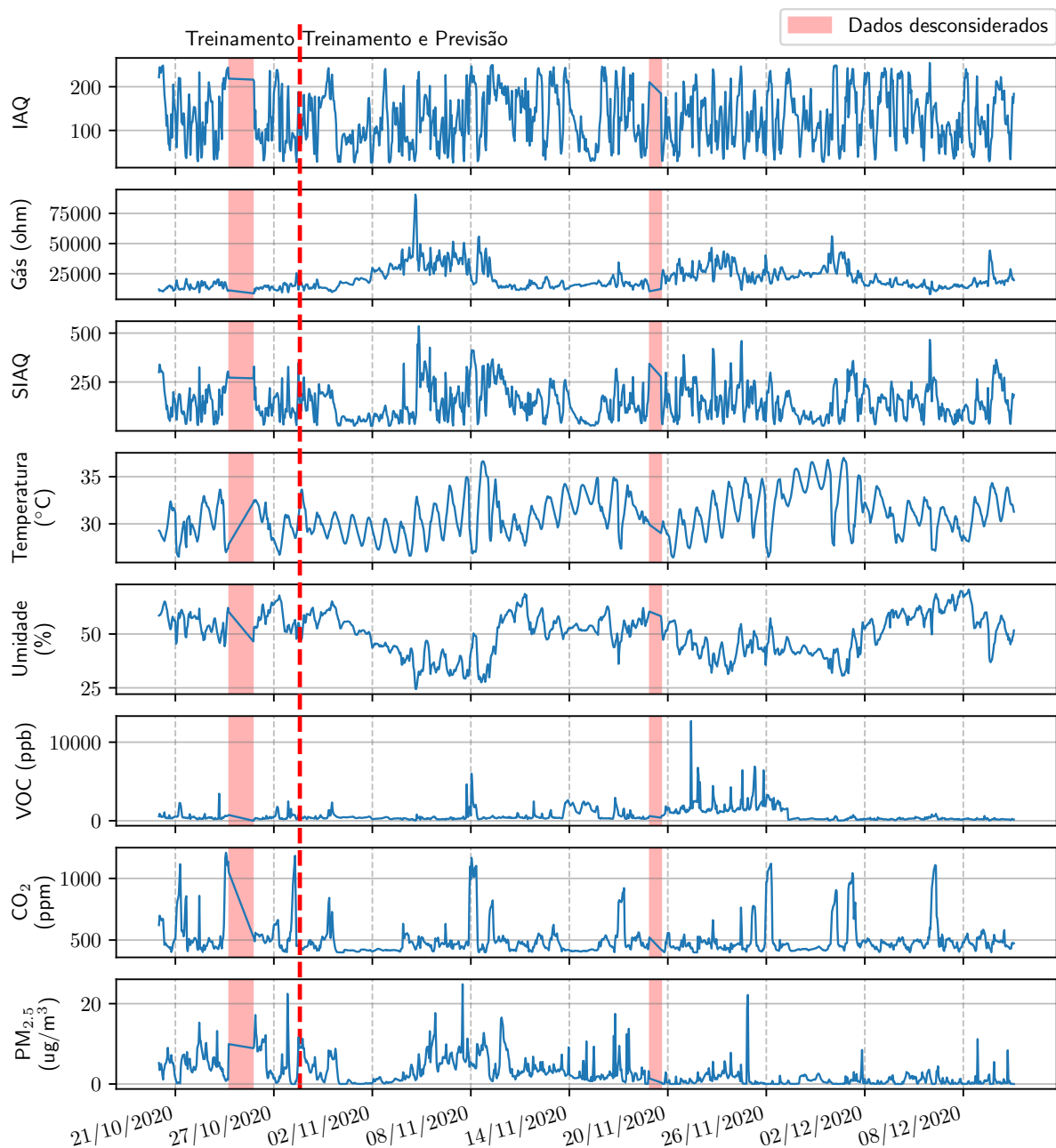
Fonte: Elaboração do próprio autor

A amostragem ocorreu de 20 de outubro a 11 de dezembro de 2020. Durante esse período, houve alguns erros de leitura e, conseqüentemente, alguns dados dos sensores não foram armazenados. Contudo, a amostragem descontinuada não prejudicou o treinamento e a predição da rede neural ARTMAP *fuzzy* devido à modelagem dos exemplos de treinamento utilizada, a qual é apresentada na próxima seção. Os valores das variáveis presentes nesta avaliação estão plotados nos gráficos da Figura 15. Os períodos em que houve erros de leituras estão destacados em vermelho.

Na metodologia de treinamento utilizada, a rede ARTMAP *fuzzy* é inicializada sem categoria. Como o conhecimento da rede neural é incrementado à medida que recebe novas entradas, é interessante que na inicialização do sistema seja efetuado apenas o

treinamento da RNA durante um certo período, sem realizar o cálculo de predição. Dessa forma, a previsão é liberada apenas quando houver conhecimento suficiente armazenado nas categorias da ARTMAP *fuzzy*. Nesse contexto, definiu-se que durante a primeira semana a RNA apenas seria treinada, período compreendido por 168 padrões de entrada, o qual está indicado pela linha vertical tracejada vermelha na Figura 15. No recebimento dos 1008 padrões de entrada subsequentes, que estão à direita da linha vermelha, manteve-se o treinamento e iniciaram-se os cálculos das predições, caracterizando uma abordagem de treinamento on-line capaz de realizar o cálculo de previsões em tempo real.

Figura 15 – Séries temporais dos dados amostrados



Fonte: Elaboração do próprio autor

Além do parâmetro IAQ calculado pelo BSEC, considerou-se também a variante IAQ estático, ou SIAQ (*static indoor air quality*), que é indicado para situações que o sensor permanece imóvel durante a operação (BOSCH SENSORTEC, 2019). Outras variáveis do BME680 analisadas foram o valor bruto da resistência do sensor de gás, a temperatura e a umidade. O VOC é proveniente do SPG30 e o CO₂ do MH-Z19B. O material particulado amostrado pelo PMS7003 considerado foi o de partículas com até 2,5 micrômetros, isto é, PM_{2.5}.

Na próxima seção apresentam-se as metodologias de treinamento e de predição envolvendo a rede neural ARTMAP *fuzzy* visando a previsão da qualidade de ar em ambientes internos modelada em séries temporais.

4.2 REDE NEURAL ARTMAP *FUZZY* APLICADA NA PREVISÃO DE SÉRIES TEMPORAIS

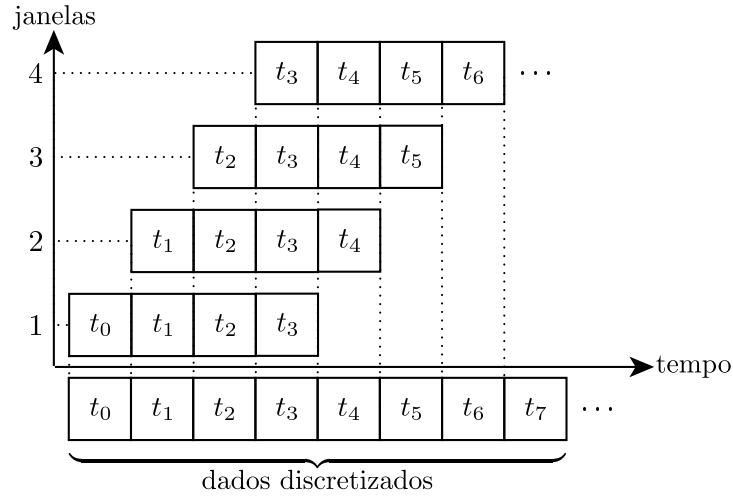
Com o objetivo de utilizar a rede ARTMAP *fuzzy* para calcular a previsão dos compostos da qualidade de ar, é necessário obter um modelo de treinamento supervisionado a partir da amostragem contínua dos sensores. Uma possibilidade é a organização das informações em janelas deslizantes, que agrupam os dados temporais discretizados para gerar os padrões de entrada da RNA.

Definiu-se a utilização de janelas deslizantes sobrepostas de tamanho fixo. Para gerar uma janela, a cada instantes de tempo t da série temporal discretizada, consideram-se n valores precedentes, os quais denominam-se *lags*, para formarem uma janela deslissante. A quantidade de *lags* define o tamanho da sobreposição de janelas adjacentes. Dessa forma, os componentes de uma janela deslissante da série temporal são organizados no vetor v , como apresentado na equação 14:

$$v = [v(t - n), \dots, v(t - 1), v(t)] \quad (14)$$

Um exemplo desse modelo de treinamento com sobreposição de 3 *lags* está apresentado na Figura 16. O treinamento se inicia no instante t_3 , já que são necessárias 3 amostragens anteriores para se obter a primeira janela. A cada nova amostragem da série temporal, gera-se um padrão de entrada, o qual é apresentado à rede neural ARTMAP *fuzzy*.

Além dos valores discretizados da série temporal organizados no vetor v , é necessário adicionar informações históricas aos padrões de entrada, de modo que a rede neural ARTMAP *fuzzy* realize a classificação baseando-se nos valores da variável e em informações

Figura 16 – Série temporal modelada em janelas deslizantes sobrepostas**Fonte:** Elaboração do próprio autor

temporais correspondentes. Assim como utilizado por Lopes (2005), os dados temporais considerados nesse trabalho para identificar o histórico dos padrões de entrada foram o dia da semana e a hora do dia, respectivamente representados pelas variáveis d e h que compõem o vetor t descrito na equação 15:

$$t = [d, h] \quad (15)$$

Dessa forma, os padrões de entrada da ARTMAP *fuzzy* são definidos pela equação 16:

$$a = [t, v] \quad (16)$$

A saída b da rede neural consiste no valor da variável v no instante de tempo subsequente ao da entrada, sendo representada pela equação 17:

$$b = [v(t + 1)] \quad (17)$$

Conforme descrito no Capítulo 3, durante a fase de aprendizagem da ARTMAP *fuzzy*, é necessário apresentar à rede os vetores de entrada (a) e de saída (b) para que seja realizado o mapeamento entre o padrão de entrada e a saída correspondente. O conhecimento é armazenado nas matrizes de pesos definidas internamente na arquitetura da rede (w_j^a , w_k^b , w^{ab}).

Neste trabalho, para realizar o cálculo de uma predição com a rede ARTMAP *fuzzy*, não foi aplicado o teste de vigilância do módulo ART_a *fuzzy* como apresentado no fluxograma da Figura 12, pois desejou-se garantir uma resposta na saída da rede. Dessa

forma, a partir do cálculo da função de escolha das categorias temporárias e definitivas do módulo $ART_a\ fuzzy$, selecionou-se a categoria J com maior função de escolha, já que é a melhor candidata para representar o padrão de entrada apresentado à rede durante o cálculo da predição.

A abordagem de treinamento apresentada permite o cálculo de previsões um instante à frente de cada amostragem, como definida na equação 17. Contudo, desejou-se obter previsões mais adiante. Para isso, realizaram-se predições utilizando-se a saída da rede neural como sua própria entrada para construir padrões de entrada subsequentes à última amostragem de dados.

Para verificar o desempenho da rede neural ARTMAP *fuzzy* na predição dos compostos da qualidade do ar, nas avaliações foram incluídos os seguintes itens:

- Normalização: uma das etapas do pré-processamento da rede neural ARTMAP *fuzzy* é a normalização dos dados, que tem o objetivo de transformar as informações de entrada em valores entre 0 e 1. É comum organizar os dados que alimentam a rede ARTMAP *fuzzy* na forma matricial. Nessa abordagem, os padrões de entrada são dispostos nas linhas, que por sua vez são formados por variáveis dispostas nas colunas, como pode ser visualizado na Figura 17. Por exemplo, o primeiro padrão de entrada está na linha $a0$, e na primeira coluna estão todos os valores da variável dia . É possível aplicar diferentes cálculos para normalizar os valores dessas matrizes, sendo que nessa pesquisa foram consideradas a normalização de dados por linha e por colunas, como segue:

- Linha: tradicionalmente, cada exemplo de treinamento é normalizado separadamente. Por exemplo, considerando a normalização do padrão de entrada $a0$ apresentado na Figura 17(a), calcula-se o somatório dos $a0_i$ valores dessa linha para normalizá-los aplicando-se equação 18:

$$a0_i = \frac{a0_i}{\sum_{i=1}^{nA} a0_i} \quad (18)$$

- Coluna: também foi analisada a normalização exclusiva de cada variável. Para isso, todos os valores de uma coluna são divididos pelo maior valor dela mesma. Como exemplo, apresenta-se na Figura 17(b) a representação da normalização dos dia_j valores da variável dia aplicando-se a equação 19:

$$dia_j = \frac{dia_j}{\text{argmax}(dia)} \quad (19)$$

Figura 17 – Diferentes metodologias de normalização dos padrões de entrada da rede neural ARTMAP *fuzzy*

a0	dia	hora	v(t-3)	v(t-2)	v(t-1)	v(t)
a1	dia	hora	v(t-3)	v(t-2)	v(t-1)	v(t)
a2	dia	hora	v(t-3)	v(t-2)	v(t-1)	v(t)
a3	dia	hora	v(t-3)	v(t-2)	v(t-1)	v(t)

(a) Linha

a0	dia	hora	v(t-3)	v(t-2)	v(t-1)	v(t)
a1	dia	hora	v(t-3)	v(t-2)	v(t-1)	v(t)
a2	dia	hora	v(t-3)	v(t-2)	v(t-1)	v(t)
a3	dia	hora	v(t-3)	v(t-2)	v(t-1)	v(t)

(b) Coluna

Fonte: Elaboração do próprio autor

- Tamanho das janelas deslizantes: para evitar a definição empírica do tamanho da sobreposição das janelas deslizantes, os treinamentos e as previsões foram executados considerando diferentes quantidades de *lags*;
- Análise univariada e multivariada: foram estudadas séries temporais univariadas, isto é, apenas uma variável no vetor v para prever essa mesma informação no instante de tempo subsequente. Adicionalmente, com foco na obtenção de melhores resultados na previsão do IAQ, verificaram-se algumas combinações de múltiplas variáveis na entrada da rede neural.

Resumindo, o objetivo das avaliações em software foi principalmente encontrar a configuração dos parâmetros estudados que ofereceram a melhor exatidão na previsão do IAQ. Assim, verificaram-se simultaneamente diferentes técnicas de normalização dos dados de entrada da rede ARTMAP *fuzzy* e tamanhos de sobreposição das janelas deslizantes, aplicadas em várias combinações de variáveis de entrada.

A rede neural ARTMAP *fuzzy* foi implementada na linguagem de programação Python, executada em um laptop com processador Intel Core i5-8250U e 8 GB de memória RAM. Ressalta-se que as avaliações apresentadas nesse capítulo não foram executadas em um sistema embarcado, nem em tempo real. Contudo, executou-se o treinamento on-line da rede neural, reproduzindo com fidelidade os acontecimentos que ocorreriam em uma aplicação do sistema com amostragem de dados em tempo real.

4.2.1 Avaliações de séries temporais univariadas

Os componentes do ar foram aplicados em processamentos distintos da rede ARTMAP *fuzzy*. O período de previsão foi definido em 24 horas à frente, sempre calculado no recebimento de cada padrão de entrada. Os parâmetros da ARTMAP *fuzzy* utilizados

com normalização dos dados por linha estão apresentados no Quadro 4. O único valor alterado na normalização por coluna foi o parâmetro de vigilância da ART_b *fuzzy*, que foi $\rho_b = 0,96$. Em relação aos parâmetros do treinamento on-line, definiu-se que as categorias temporárias deveriam ser ativadas 2 vezes para se tornarem definitivas ($N_{max} = 2$), e excluem-se as que ficassem por duas semanas inativas, isto é, aquelas que não classificassem nenhuma entrada após a RNA receber 336 padrões ($N_{min} = 336$).

Quadro 4 – Parâmetros da ARTMAP *fuzzy* com normalização por linha

Parâmetros	Valor
Parâmetro de escolha (α)	1,0
Taxa de treinamento (β)	1,0
Parâmetro de vigilância ART _a base ($\overline{\rho_a}$)	0,99
Parâmetro de vigilância ART _b (ρ_b)	0,997
Parâmetro de vigilância Inter-ART (ρ_{ab})	1,0
Incremento do ρ_a (ε)	0,0001
Parâmetros exclusivos do treinamento on-line	Valor
N_{max}	2
N_{min}	336

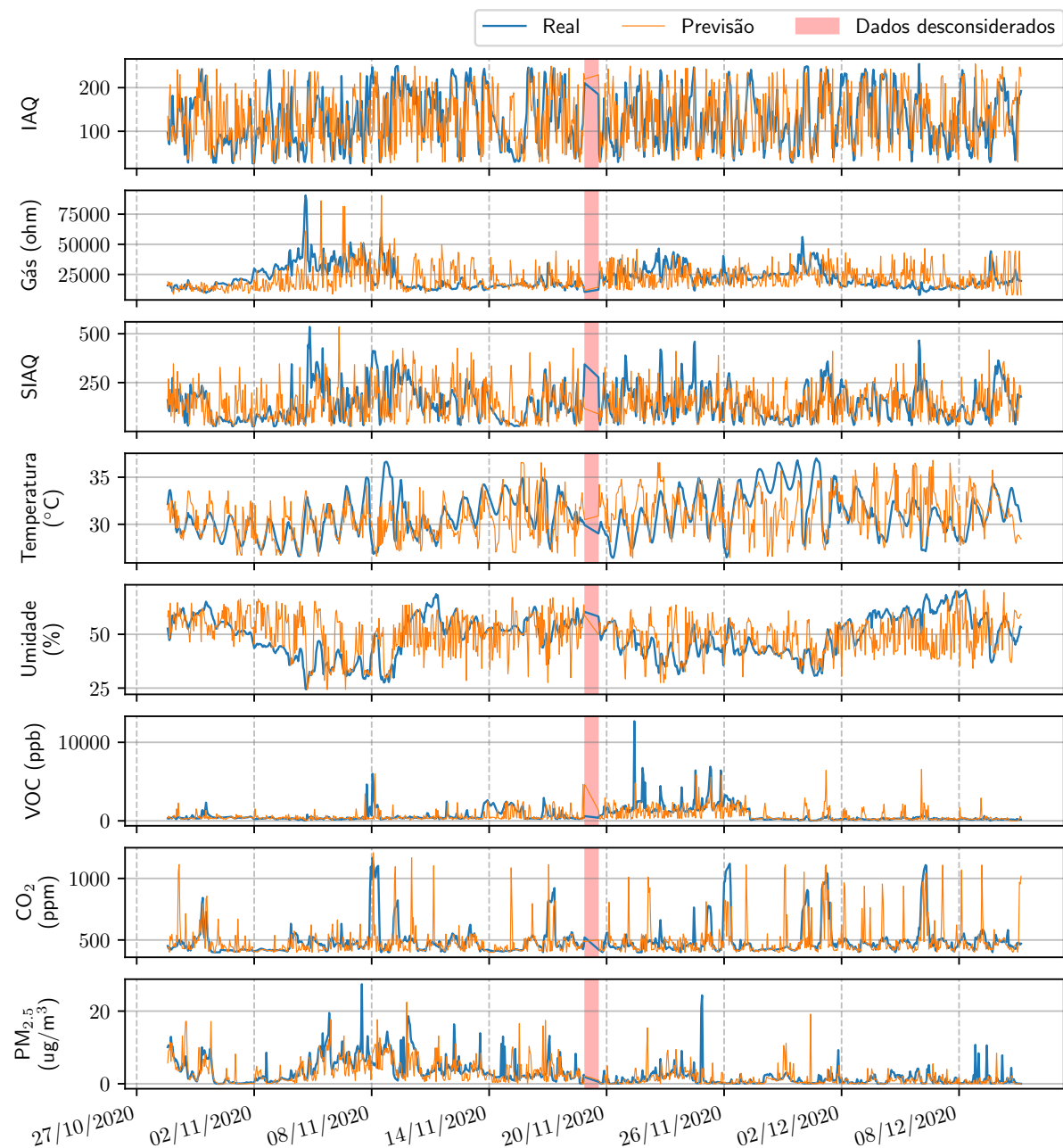
Fonte: Elaboração do próprio autor

Para realizar o processamento dos dados, as séries temporais foram modeladas em janelas deslizantes sobrepostas, conforme descrito anteriormente. Foram consideradas sobreposições de 1 a 3 *lags*, combinadas com normalização das entradas por linha e por coluna. A avaliação consistiu em utilizar dados referentes a uma semana (168 entradas) para iniciar o treinamento e, após esse período, iniciar o cálculo das previsões nas 6 semanas subsequentes (1008 entradas). O treinamento da ARTMAP *fuzzy* foi executado em todos os padrões de entrada, caracterizando na aprendizagem on-line da rede neural.

Considerou-se apenas a plotagem dos resultados referentes à utilização de janelas deslizantes com sobreposição de 1 *lag*, devido à similaridade dos gráficos das respostas obtidas com os demais tamanho de sobreposição. As previsões calculadas pela ARTMAP *fuzzy* aplicando-se normalização por linha estão representadas nos gráficos da Figura 18. Nesse caso, a normalização por linha foi aplicada apenas nas entradas da ART_a *fuzzy*, que possui entre 4 e 6 componentes as quais variam de acordo com o tamanho da sobreposição das janelas deslizantes. As entradas da ART_b *fuzzy* foram normalizadas por coluna, pois apresentam apenas um componente referente ao instante subsequente da série temporal. Ressalta-se que, para se obter o valor correto da predição é necessário desfazer a normalização. Logo, para cada linha normalizada é necessário armazenar o somatório calculado, assim como na normalização por coluna é preciso guardar o maior valor existente.

No geral, a normalização por linha gerou previsões ruidosas, principalmente quando os dados reais apresentaram mais variações. Esse comportamento fica mais visível no componente VOC, sendo que até aproximadamente o dia 14 de novembro, a concentração ficou constantemente próximo de 0 ppb e com previsões próximas aos valores amostrados. Após o início da oscilação da concentração do VOC, a previsão tornou-se mais ruidosa, e mesmo com a estabilidade dos dados reais após o dia 28 de novembro, as saídas da ARTMAP *fuzzy* continuaram a apresentar picos, prejudicando a exatidão das previsões.

Figura 18 – Previsões univariadas com normalização por linha



Fonte: Elaboração do próprio autor

Nota-se também que a normalização por linha nem sempre garantiu a proximidade

das previsões com os dados reais, considerando a tendência ou a sazonalidade dos sinais. Um exemplo da perda da tendência das previsões pode ser visualizado no gás a partir de um pouco antes do dia 2 de novembro, quando a concentração começa a aumentar, mas as previsões permanecem abaixo dos valores reais. Já a perda de sazonalidade de algumas previsões pode ser visualizada, por exemplo, na umidade, aproximadamente entre os dias 11 e 14 de novembro. Nesse intervalo as previsões oscilaram sem acompanhar a característica cíclica dos dados reais.

Como descrito, verificou-se também o comportamento da rede ARTMAP *fuzzy* aplicando-se a normalização de dados por coluna. Apresentam-se, na Figura 19, os resultados das previsões realizando a normalização das variáveis dos padrões de entrada separadamente, ou seja, dividindo cada coluna da matriz de dados pelo maior valor presente em cada coluna.

A normalização exclusiva de cada variável que compõe os padrões de entrada gerou previsões mais suavizadas e com menos variações em comparação à normalização por linha. Além disso, em todos os componentes do ar, as previsões estão mais próximas dos dados reais, incluindo as tendências e sazonalidades.

Além da avaliação gráfica, é possível medir a exatidão das previsões por meio de métricas que quantificam os erros dos cálculos. Nesse trabalho foram considerados os seguintes cálculos para validação das previsões, sendo n o número de predições calculadas, y_t o valor real da série temporal no instante de tempo t e $\hat{y}_t$ o valor da saída da ARTMAP *fuzzy*:

- Porcentagem de erro médio absoluto, ou MAPE (*mean absolute percentage error*), calculado pela equação 20:

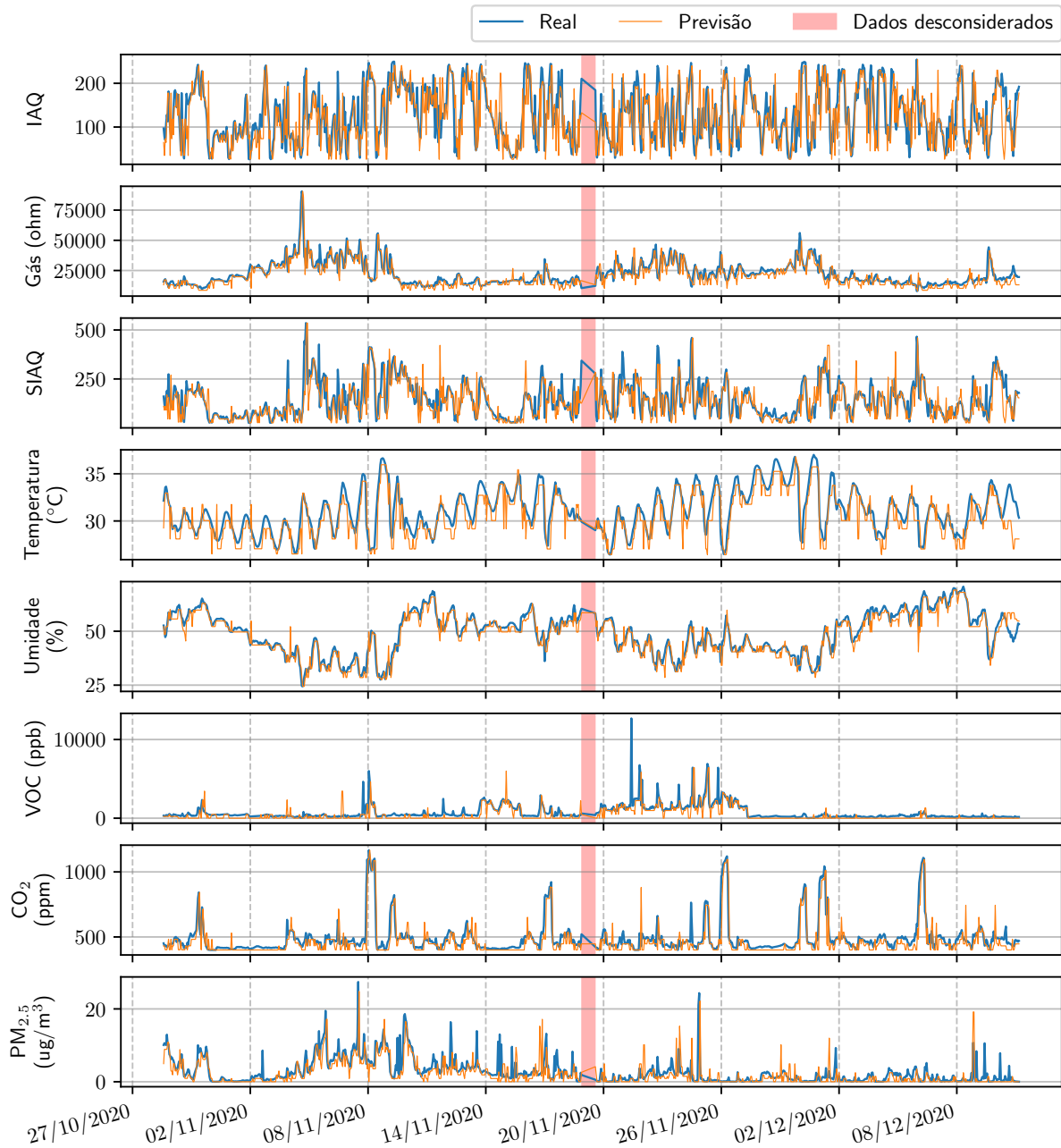
$$\text{MAPE} = \frac{100}{n} \sum_{t=1}^n \left| \frac{y_t - \hat{y}_t}{y_t} \right| \quad (20)$$

- Erro médio absoluto, ou MAE (*mean absolute error*), calculado pela equação 21:

$$\text{MAE} = \frac{1}{n} \sum_{t=1}^n |y_t - \hat{y}_t| \quad (21)$$

- Erro quadrático médio, ou RMSE (*root mean squared error*), calculado aplicando-se a equação 22:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{t=1}^n (y_t - \hat{y}_t)^2} \quad (22)$$

Figura 19 – Previsões univariadas com normalização por coluna

Fonte: Elaboração do próprio autor

Apresentam-se no Quadro 5 os MAPEs, MAEs e RMSEs das previsões dos componentes do ar modelados em séries temporais univariadas aplicando a normalização por linha e por coluna nos diferentes tamanhos de sobreposição das janelas deslizantes. Não foi possível calcular o MAPE para os compostos VOC e PM_{2.5} devido o cálculo de valores de predições nulos.

Como esperado após a interpretação dos gráficos das previsões, os valores do MAPE, do MAE e do RMSE de todas as avaliações aplicando-se normalização por coluna foram

Quadro 5 – Avaliação das previsões univariadas

Entrada / Saída	Normalização	<i>Lag</i>								
		1			2			3		
		MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE
IAQ / IAQ	Linha	52,4	52,7	67,6	54,9	53,4	68,5	55,1	52,7	67,3
	Coluna	36,9	37,1	50,1	38,9	37,4	50,7	39,3	36,8	49,8
Gás / Gás	Linha	37,6	8280,3	11627,8	39,5	8593,1	12032,3	36,1	8059,1	11193,3
	Coluna	15,7	3564,3	5327,3	15,4	3507,9	5110,8	15,2	3463,7	5046,4
SIAQ / SIAQ	Linha	68,6	75,3	102,3	67,6	75,1	101,4	64,1	72,0	98,4
	Coluna	40,0	46,9	69,4	39,4	46,5	69,2	37,5	44,8	67,0
Temperatura / Temperatura	Linha	5,9	1,9	2,5	5,9	1,9	2,5	5,7	1,8	2,4
	Coluna	3,0	0,9	1,3	2,9	0,9	1,2	2,8	0,9	1,2
Umidade / Umidade	Linha	16,4	7,8	10,2	16,2	7,6	10,0	16,6	7,8	10,2
	Coluna	4,8	2,3	3,1	4,8	2,2	3,1	4,7	2,2	3,0
CO ₂ / CO ₂	Linha	12,8	71,2	138,9	12,6	70,1	135,8	12,5	69,5	132,6
	Coluna	8,4	43,8	72,6	8,3	43,4	72,4	8,1	42,7	70,3
VOC / VOC	Linha	-	483,8	978,2	-	494,9	1079,8	-	470,1	1047,8
	Coluna	-	426,4	820,8	-	424,3	843,1	-	418,0	829,9
PM _{2.5} / PM _{2.5}	Linha	-	1,6	2,9	-	1,5	2,8	-	1,6	3,0
	Coluna	-	1,2	2,5	-	1,3	2,6	-	1,2	2,6

Fonte: Elaboração do próprio autor

menores em relação aos respectivos resultados com normalização por linha. Logo, o erro das previsões aplicando-se a normalização de cada variável dos padrões de entrada separadamente foi menor do que padronizar o intervalo dessas diferentes grandezas simultaneamente na normalização de cada padrão entrada da ARTMAP *fuzzy*.

Em relação às avaliações dos diferentes tamanhos de sobreposição das janelas deslizantes, o objetivo foi estudar a dependência dos dados discretizados e a influência disso no cálculo das previsões. Como a modelagem de uma série temporal em janelas deslizantes realiza a segmentação dos dados considerando cada instante de tempo precedido de um determinado número de *lags*, é interessante verificar a relação das informações contidas em cada janela.

Uma forma de mensurar a dependência entre duas variáveis denominadas X e Y , compostas por n componentes cada, é por meio do cálculo do coeficiente de correlação aplicando-se a equação 23:

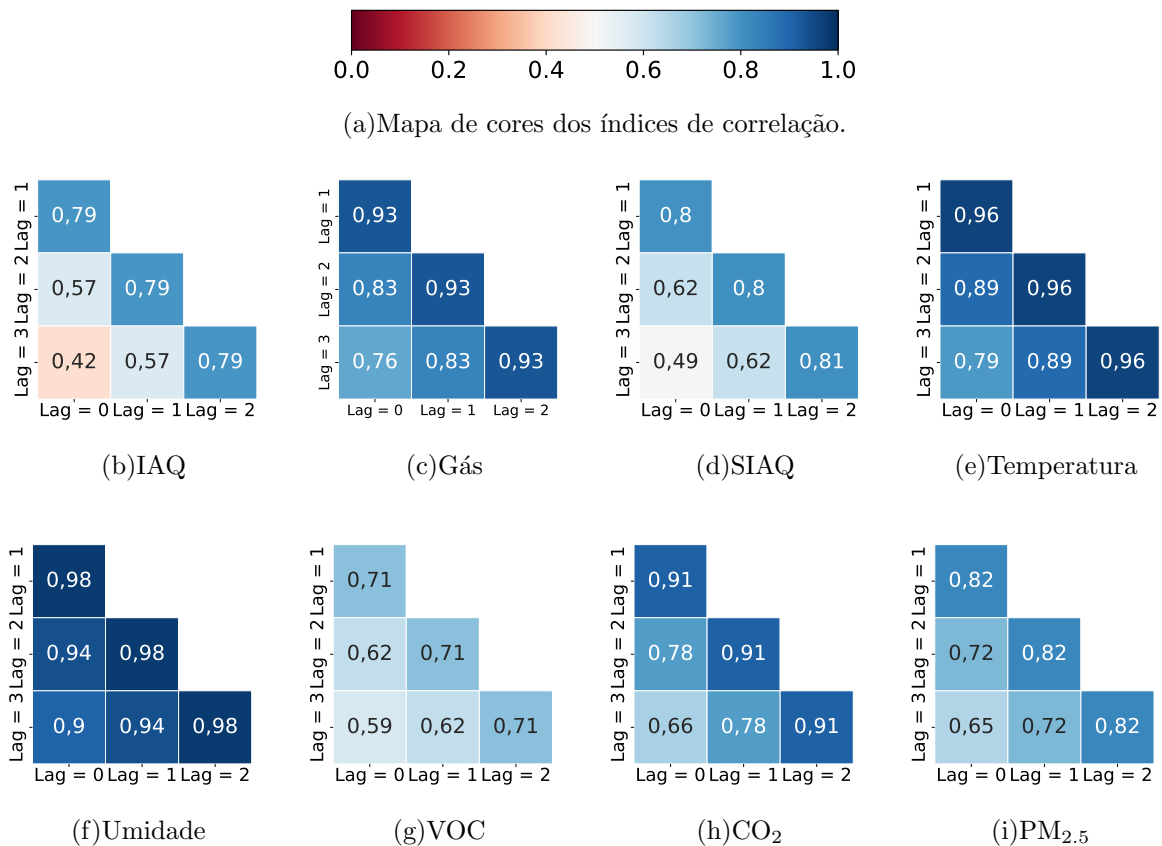
$$\text{Correlação}(X, Y) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 (y_i - \bar{y})^2}} \quad (23)$$

sendo $\bar{x}$ e $\bar{y}$ as médias dos valores das variáveis X e Y , respectivamente. Também conhecido como correlação de Person, o coeficiente de correlação, que pertence ao intervalo de -1 a 1, mede a dependência linear entre duas variáveis. Quanto mais próximos dos limites desse intervalo, maior será a dependência dos dados consideradas. Outra

informação proveniente do coeficiente de correlação é que valores negativos identificam variáveis inversamente proporcionais, e vice-versa.

Com o intuito de calcular as correlações de uma série temporal discretamente amostrada considerando até 3 *lags* anteriores, criaram-se séries auxiliares de 1 a 3 *lags* de defasagem. Essa operação foi realizada para todos os compostos do ar monitorados. A organização das correlações em formato matricial corresponde à matriz de correlação. Apresentam-se, na Figura 20, as matrizes de correlação dos compostos da qualidade do ar com suas respectivas séries temporais defasadas. A interpretação das correlações fica mais visível com a utilização da escala de cores representada na Figura 20(a). As informações essenciais encontram-se nas primeiras colunas das matrizes, sendo que nas linhas estão os coeficiente de correlação considerando-se as séries temporais defasadas de 1 a 3 *lags* em relação à série temporal original ($Lag = 0$).

Figura 20 – Matrizes de correlação das séries temporais com atraso de até 3 *lags*



Fonte: Elaboração do próprio autor

Por meio das análises das matrizes de correlação, conclui-se que para todos compostos do ar amostrados, à medida que a defasagem dos dados aumenta, isto é, quanto mais distante estão as amostragens antecessores considerando um certo instante, menor é a dependência dos dados. Contudo, o treinamento supervisionado da rede ARTMAP *fuzzy*

aplicando janelas deslizantes com mais dados pode garantir o incremento da capacidade da rede neural em extrair informações temporais dos dados de entradas, resultando em predições mais exatas. Essa possibilidade foi estudada aplicando-se janelas deslizantes de 1 a 3 *lags*.

No geral, variáveis que possuem maiores coeficientes de correlação entre *lags* obtiveram melhores resultados com o aumento do tamanho da sobreposição das janelas. Foi o caso da temperatura, cujos MAPEs das previsões aplicando normalização por coluna com 1 e 3 *lags* foram, respectivamente, 3,0 e 2,8, como está apresentado no Quadro 5. Por outro lado, aumentar a sobreposição das janelas deslizantes em variáveis com índices de correlação entre *lags* menos significativos não garantiu melhores predições, como ocorreu no IAQ. Nessa situação, maiores janelas deslizantes geraram piores resultados, tendo em vista que foram obtidos MAPEs de 36,9 e 39,3 considerando, respectivamente, janelas com 1 e 3 *lags* e normalização dos padrões de entrada por coluna.

Uma consideração importante é verificar a quantidade de categorias criadas pela rede neural ARTMAP *fuzzy* com treinamento on-line, como está apresentado no Quadro 6. As colunas ART_a^t e ART_a^d são respectivamente os números de categorias temporárias e definitivas da ART_a *fuzzy*.

A normalização por coluna criou mais categorias ART_a definitivas do que a normalização por linha em todas as variáveis, exceto no IAQ e no SIAQ. Isso evidencia que o ajuste dos intervalos dos dados de cada variável que compõem os padrões de entrada oferece melhores informações para a rede identificar similaridades entre as categorias, já que para um grupo tornar-se definitivo, deve passar no teste de vigilância do módulo ART_a *fuzzy* por N_{min} vezes. Ressalta-se que o parâmetro de vigilância ρ_a foi definido em 0,96 para todas as avaliações, não favorecendo a criação de mais categorias no módulo ART_a *fuzzy* em nenhum caso.

Já as maiores quantidades de categorias ART_b na normalização por linha ocorreram devido ao maior valor do parâmetro de vigilância da ART_b *fuzzy* ($\rho_b = 0,997$) utilizado em comparação ao considerado nas avaliações de normalização por coluna ($\rho_b = 0,96$). Quanto à utilização de diferentes tamanhos de sobreposição das janelas deslizantes, essa variação não ocasionou alterações significativas na quantidade de categorias criadas pela ARTMAP *fuzzy*.

As respostas da rede ARTMAP *fuzzy* apresentadas nos gráficos da Figura 18 e da Figura 19 representam a simulação do treinamento on-line e do cálculo de previsões em tempo real. Na metodologia considerada, deseja-se obter a estimativa do composto do

Quadro 6 – Número de categorias criadas nas avaliações univariadas

Entrada / Saída	Normalização	<i>Lag</i>								
		1			2			3		
		ART_a^t	ART_a^d	ART_b	ART_a^t	ART_a^d	ART_b	ART_a^t	ART_a^d	ART_b
IAQ / IAQ	Linha	311	11	260	316	7	260	323	6	259
	Coluna	336	1	27	336	0	27	336	0	27
Gás / Gás	Linha	319	24	140	320	22	140	322	23	139
	Coluna	307	73	22	300	70	22	292	77	22
SIAQ / SIAQ	Linha	308	14	212	313	11	212	314	9	212
	Coluna	329	16	28	333	7	27	336	3	27
Temperatura / Temperatura	Linha	310	33	107	308	35	108	309	36	109
	Coluna	279	105	11	274	96	11	253	111	11
Umidade / Umidade	Linha	309	24	198	310	26	198	308	27	198
	Coluna	321	45	22	325	34	22	322	51	20
CO ₂ / CO ₂	Linha	300	50	136	299	49	136	304	46	136
	Coluna	285	162	20	275	163	20	261	165	20
VOC / VOC	Linha	296	53	101	306	46	101	309	47	102
	Coluna	241	177	15	187	218	15	157	250	15
PM _{2.5} / PM _{2.5}	Linha	271	74	158	286	59	160	277	68	160
	Coluna	210	98	26	230	90	26	235	91	26

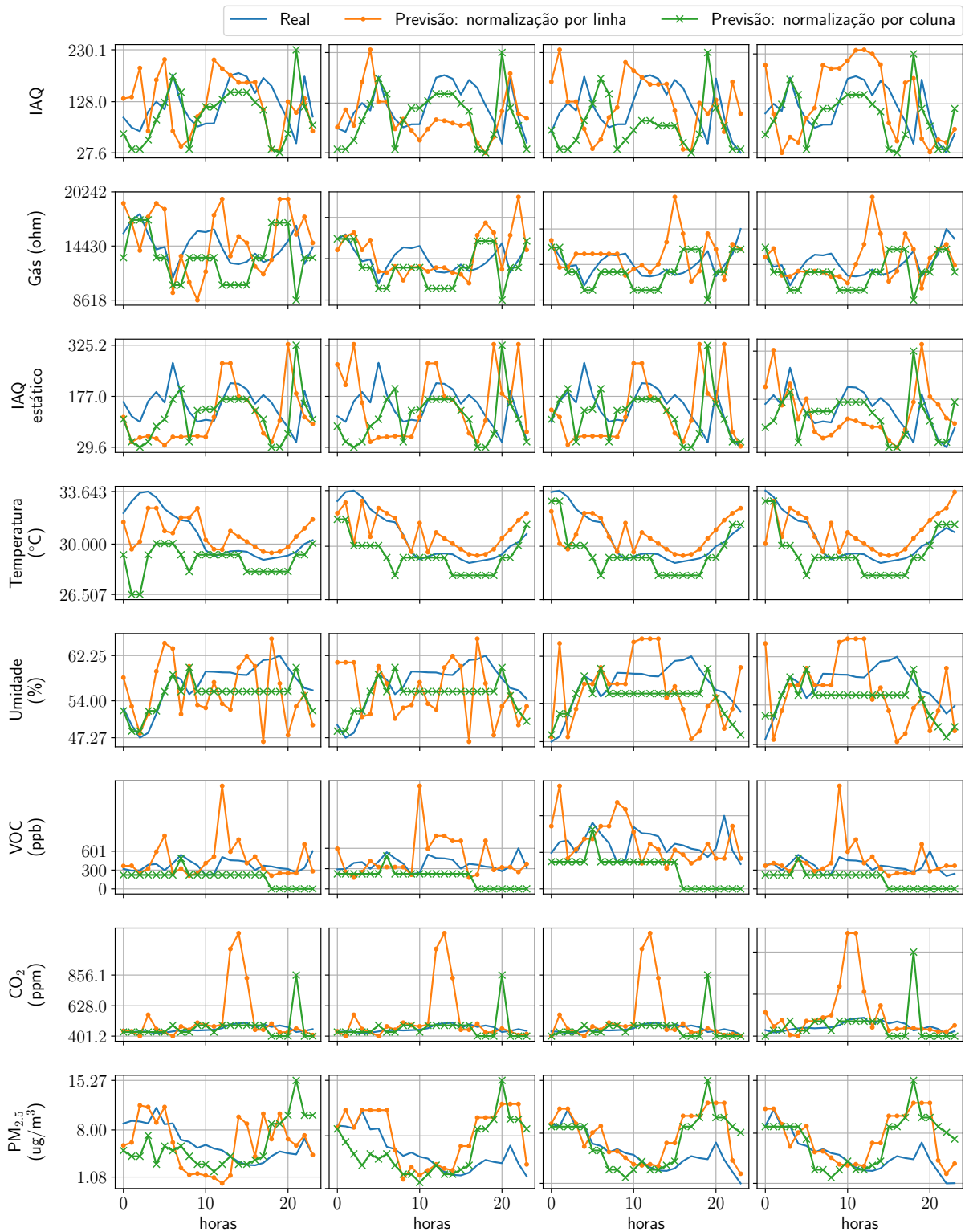
Fonte: Elaboração do próprio autor

ar analisado 24 horas à frente. Assim, a cada amostragem, realiza-se o treinamento da ARTMAP *fuzzy* e 24 previsões são calculadas para obter a previsão adiante constantemente atualizada. Ao concatenar os conjuntos de previsões 24 horas à frente em um gráfico, como foi realizado nas Figuras 18 e 19, mantém-se a previsão da primeira hora à frente do conjunto anterior. Logo, a organização dos sucessivos conjuntos de previsões 24 horas à frente equivalem à uma sequência de previsões 1 hora à frente.

Para demonstrar com maiores detalhes a operação do sistema em tempo real, apresentam-se na Figura 21 os quatro primeiros conjuntos de 24 previsões calculadas pela ARTMAP *fuzzy* aplicando normalização dos dados por linha e por coluna. À medida que uma nova amostragem é realizada, atualiza-se a resposta da rede neural calculando-se um novo conjunto de 24 previsões à frente, e o gráfico com os valores de saída é atualizado.

É possível avaliar a qualidade das previsões mais detalhadamente por meio do erro de cada conjunto de previsões 24 horas à frente, isto é, considerar o erro da previsão de cada subgráfico da Figura 21. Assim, a exatidão da resposta da ARTMAP *fuzzy* é representada com maior fidelidade analisando-se os conjuntos de previsões 24 horas adiante. Apresentam-se no Quadro 7 os valores médios dos MAPEs, MAEs e RMSEs dos 1008 conjuntos de previsões 24 horas à frente calculados de cada composto do ar amostrado.

Os valores do Quadro 7, que representam os erros médio das previsões 24 horas à frente,

Figura 21 – Previsões univariadas 24 horas à frente

Fonte: Elaboração do próprio autor

são superiores aos valores dos erros das previsões 1 hora à frente apresentados no Quadro 5. Logo, aumentar intervalo temporal no futuro das previsões calculadas pela rede ARTMAP *fuzzy* gera maiores taxas de erros, pois são realizadas mais previsões subsequentes. Como

Quadro 7 – Avaliação das previsões univariadas 24 horas à frente

Entrada / Saída	Normalização	<i>Lag</i>								
		1			2			3		
		MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE
IAQ / IAQ	Linha	70,5	68,2	82,3	68,6	66,6	80,8	69,9	66,3	80,2
	Coluna	69,4	64,5	78,0	67,7	64,2	77,9	68,4	64,5	78,1
Gás / Gás	Linha	55,4	11322,6	13697,0	58,5	11589,3	14132,4	58,3	11540,0	14027,2
	Coluna	35,8	8611,8	10262,5	31,5	7620,6	9169,5	31,5	7404,9	8950,4
SIAQ / SIAQ	Linha	94,4	96,7	120,1	96,6	96,7	121,2	91,2	93,1	116,2
	Coluna	79,5	87,1	106,9	80,4	87,5	107,0	77,6	85,5	105,2
Temperatura / Temperatura	Linha	6,6	2,0	2,3	6,4	2,0	2,3	6,4	2,0	2,3
	Coluna	6,8	2,1	2,4	6,6	2,1	2,4	6,4	2,0	2,3
Umidade / Umidade	Linha	22,0	10,3	12,0	22,0	10,2	11,8	22,7	10,5	12,1
	Coluna	15,0	7,2	8,3	13,6	6,6	7,6	13,2	6,4	7,5
CO ₂ / CO ₂	Linha	17,4	95,1	152,6	17,5	95,9	149,2	19,4	104,5	159,6
	Coluna	16,8	93,3	125,5	18,8	103,1	139,4	18,1	100,5	136,3
VOC / VOC	Linha	-	724,2	1134,0	-	854,9	1292,4	-	769,1	1137,5
	Coluna	-	688,7	862,8	-	662,5	833,8	-	660,3	830,5
PM _{2.5} / PM _{2.5}	Linha	-	2,6	3,6	-	2,6	3,5	-	2,7	3,6
	Coluna	-	2,3	3,2	-	2,3	3,2	-	2,2	3,1

Fonte: Elaboração do próprio autor

descrito, para obter previsões de vários instantes à frente, considera-se a resposta da rede neural como entrada dela mesma para criar os padrões de entrada futuros. Nesse processo, os erros calculados são propagados para as predições subsequentes, aumentando a perda de acurácia à medida que as predições avançam. O erro acumulado pode ser minimizado efetuando-se menos predições subsequentes, o que acarretaria na diminuição do alcance temporal das previsões.

Analisando os melhores resultados das previsões, nota-se que o RMSE da temperatura, da umidade e do PM_{2.5} foram iguais ou inferiores a 3,0. Dessa forma, a rede neural ARTMAP *fuzzy* conseguiu extrair com maior acurácia informações temporais desses compostos do ar com a modelagem de dados proposta, provavelmente devido aos padrões cíclicos desses dados, como pode ser visualizado na Figura 15, principalmente nos gráficos da temperatura e da umidade.

Em relação ao gás e ao VOC, os altos valores de RMSE foram resultantes devido à ocorrência de picos isolados de amostragem, com valores significativamente maiores do que a média desses compostos, sendo no gás o valor máximo superior a 75000 ohms, e o VOC teve uma leitura superior a 10000 ppm.

Comparando-se os gráficos do SIAQ e do IAQ com o da temperatura, nota-se que os índices automaticamente calculados pela biblioteca BSEC apresentaram dados mais ruidosos. Essas alterações provavelmente são resultantes da influência instantânea do sensor de gás no cálculo do IAQ. Por exemplo, ao manter a janela do quarto fechada e

dependendo da atividade dentro do apartamento, como cozinhar, o IAQ alcançava o valor acima de 200 facilmente, que indica a presença de algum tipo de poluente no ar. Ao abrir a janela para permitir a troca de ar entre o ambiente interno e externo, em poucos minutos o IAQ voltava para qualidade do ar tipicamente seguro, isto é, em torno de 25.

As alterações frequentes nos valores do IAQ e do SIAQ dificultaram a extração de padrões temporais pela rede ARTMAP *fuzzy* com a modelagem de dados utilizada em intervalos de uma hora. Uma mudança que pode amenizar o erro da previsão desses índices é considerar dados discretizadas em intervalos de tempos menores. Essa modelagem obteria mais características temporais dos índices monitorados, o que poderia aprimorar o conhecimento do sistema durante o treinamento. Contudo, diminuir o intervalo de discretização da série temporal também diminui o alcance de tempo de uma previsão, potencializando o aumento do erro acumulado no cálculo de um maior número de previsões subsequentes para prever 24 horas à frente.

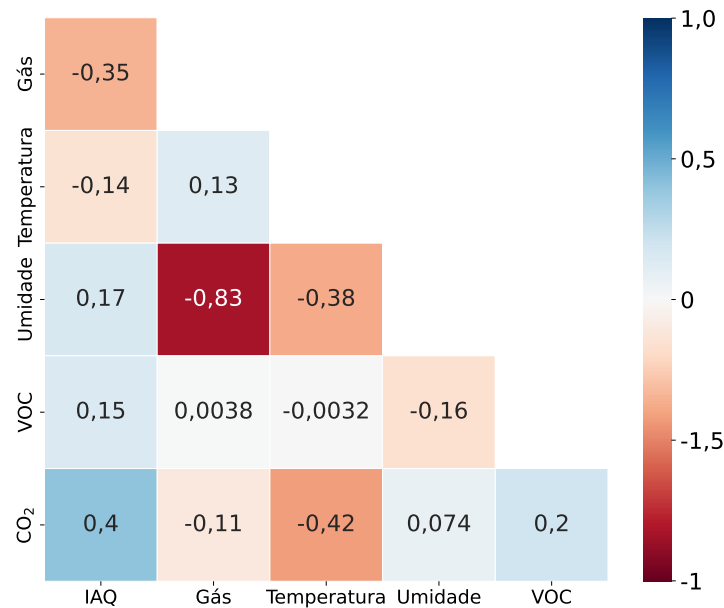
Visando obter melhores resultados da previsão do IAQ, na sequência apresenta-se um estudo considerando séries temporais multivariadas.

4.2.2 Previsão do IAQ aplicando séries temporais multivariadas

Quando se aplica algoritmos e ferramentas de IA para realizar a previsão de séries temporais, há a possibilidade de considerar simultaneamente mais de uma variável. Um dos objetivos dessa metodologia é explorar as relações entre diferentes tipos de dados para aprimorar a extração de características dos padrões de entrada, o que pode aumentar a acurácia das previsões.

Com a amostragem de vários compostos do ar nesse trabalho, foi possível realizar estudos com modelagens multivariadas da qualidade do ar em ambientes internos. As relações entre os dados dos sensores BME680, SGP30 e MH-Z19B foram analisadas por meio de coeficientes de correlação, que estão apresentados em forma matricial na Figura 22. Como o foco da utilização de séries temporais multivariadas foi a previsão do IAQ, o PM lido pelo PMS7003 não foi considerado, pois os parâmetros considerados pela biblioteca BSEC da Bosch no cálculo do IAQ consideram apenas compostos gasosos, temperatura e umidade.

Uma das saídas do BME680 é a resistência bruta do sensor de gás, que além de mensurar a concentração de VOC, também é influenciada por outros agentes, como a umidade (BOSCH SENSORTEC, 2019). Essa influência pôde ser notada com a correlação

Figura 22 – Matriz de correlação dos compostos do ar

Fonte: Elaboração do próprio autor

mais significativa de -0,83 entre o gás e a umidade.

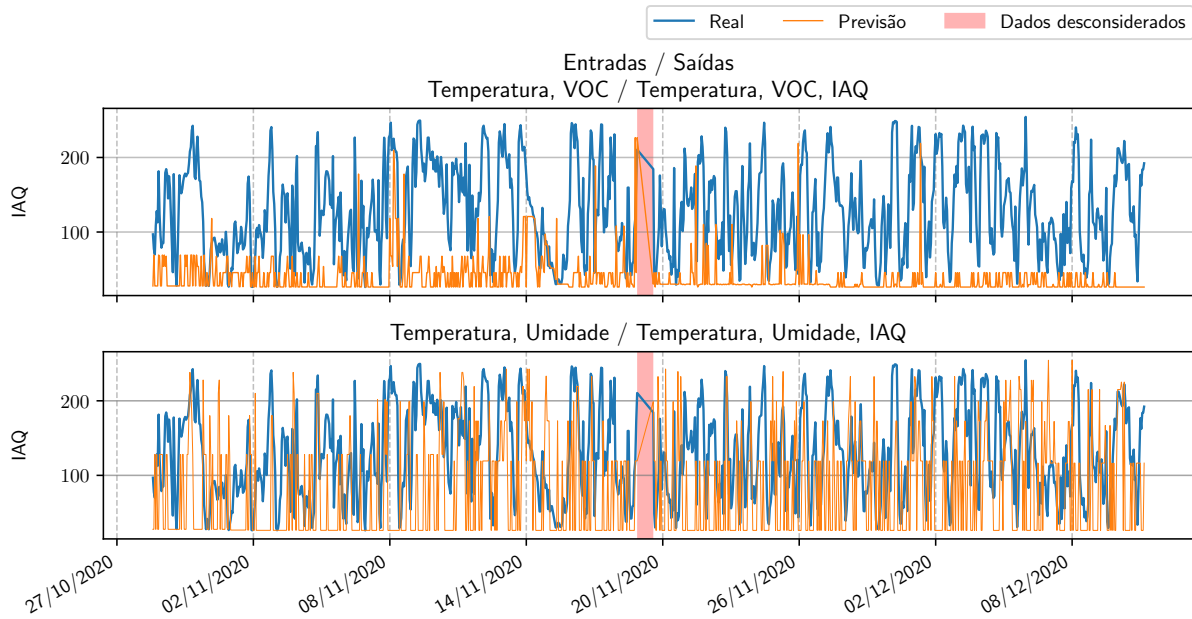
Outra dependência das saídas do BME680 ocorre entre o gás e o índice de IAQ, que é calculado pela biblioteca BSEC da Bosch. O valor da resistência é inversamente proporcional à concentração de gases (BOSCH SENSORTEC, 2019), isto é, à medida que há menores concentrações de VOC no ambiente monitorado, a saída da resistência bruta do sensor de gás aumenta. O incremento do valor da resistência é interpretado pelo BSEC, fornecendo menores valores de IAQ que indicam melhoria na qualidade do ar.

Considerando as relações entre as variáveis dos demais sensores com o IAQ, nota-se que os coeficientes de correlação do IAQ com o VOC e o CO₂ foram respectivamente 0,15 e 0,4. Com o intuito de verificar se todas essas relações são significantes para oferecerem à rede ARTMAP *fuzzy* informações relevantes para incrementar o aprendizado, e assim efetuar previsões do IAQ mais exatas, vários conjuntos de combinações dos compostos do ar foram avaliados. Os mesmos estudos apresentados com as séries temporais univariadas foram aplicados nas séries temporais multivariadas. Assim, foram avaliadas diferentes abordagens de normalização (linha e coluna) e distintos tamanhos de sobreposição das janelas deslizantes (com 1, 2, e 3 *lags* de atraso) considerando os mesmos dados amostrados.

Graficamente, as previsões aplicando-se normalização por linha resultaram basicamente nos dois padrões apresentados na Figura 23. A pouca abrangência do sinal da previsão do gráfico superior ocorreu devido à pequena quantidade de categorias criadas

na ART_b *fuzzy*. Mesmo com parâmetro de vigilância ρ_b máximo, o número de categorias não aumentou o suficiente para haver melhorias significativas nas respostas da rede neural. Já no gráfico inferior, nota-se o mesmo padrão com muitas oscilações ocorrido nas análises univariadas.

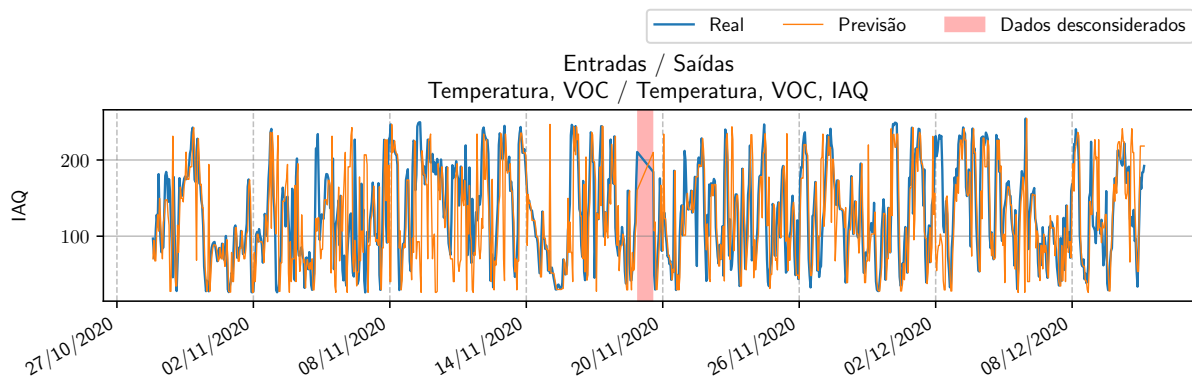
Figura 23 – Previsões do IAQ aplicando normalização por linha e séries temporais multivariadas



Fonte: Elaboração do próprio autor

Nos casos utilizando normalização por coluna, todos os gráficos foram semelhantes. Um exemplo está apresentado na Figura 24. Assim como nas avaliações univariadas, normalizar separadamente cada uma das variáveis que compõem os padrões de entrada forneceu melhores informações para a rede ARTMAP *fuzzy* extrair conhecimento das séries temporais para prever o IAQ.

Figura 24 – Previsão do IAQ aplicando normalização por coluna e séries temporais multivariadas



Fonte: Elaboração do próprio autor

Apresentam-se, no Quadro 8, os MAPEs, MAEs e RMSEs das previsões do IAQ

aplicando-se séries temporais multivariadas. As médias das métricas de erros dos conjuntos de previsões 24 à frente e a quantidade de categorias criadas nas avaliações multivariadas estão apresentadas respectivamente no Quadro 14 e no Quadro 15, no Apêndice B.

Quadro 8 – Avaliação das previsões do IAQ com séries temporais multivariadas

Entrada / Saída	Normalização	<i>Lag</i>								
		1			2			3		
		MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE
Temperatura, Gás / Temperatura, Gás, IAQ	Linha	63,7	89,4	107,5	61,6	86,1	104,7	60,5	85,9	104,7
	Coluna	33,2	35,2	49,9	31,9	33,2	46,6	32,0	33,0	45,9
Temperatura, CO ₂ / Temperatura, CO ₂ , IAQ	Linha	63,9	89,6	106,3	59,6	82,2	99,1	60,8	83,7	100,5
	Coluna	38,0	37,4	53,1	35,9	35,7	50,6	36,1	35,6	50,3
Temperatura, VOC / Temperatura, VOC, IAQ	Linha	65,3	92,5	110,1	65,9	90,3	108,0	63,7	90,9	109,3
	Coluna	40,2	40,0	57,1	38,2	38,9	55,2	36,8	37,7	54,0
Umidade, Gás / Umidade, Gás, IAQ	Linha	64,6	90,4	108,2	61,5	87,1	105,5	61,3	87,3	105,8
	Coluna	34,7	35,5	49,4	33,3	34,3	47,6	32,4	34,1	47,7
Umidade, CO ₂ / Umidade, CO ₂ , IAQ	Linha	63,7	89,3	106,3	63,2	88,6	105,8	62,6	86,6	103,5
	Coluna	34,7	34,1	47,8	34,7	34,0	47,6	33,4	32,7	45,5
Umidade, VOC / Umidade, VOC, IAQ	Linha	64,5	91,6	109,5	66,9	93,3	111,2	67,0	93,7	111,2
	Coluna	33,9	34,4	48,6	33,1	34,0	47,8	32,0	33,0	46,2
CO ₂ , Gás / CO ₂ , Gás, IAQ	Linha	64,9	90,7	107,6	63,8	90,0	106,8	62,7	88,5	105,1
	Coluna	35,4	36,7	51,2	35,1	35,5	49,1	34,2	34,3	47,7
CO ₂ , VOC / CO ₂ , VOC, IAQ	Linha	67,5	94,8	112,6	66,5	93,5	111,4	65,5	92,0	109,9
	Coluna	39,2	39,7	55,7	37,5	37,6	52,2	36,9	37,1	51,8
Temperatura, Umidade / Temperatura, Umidade, IAQ	Linha	66,9	78,3	96,0	65,7	78,5	96,6	68,0	81,6	99,6
	Coluna	31,3	32,7	45,9	30,9	32,2	45,0	30,7	31,9	44,6
Temperatura, Umidade, Gás / Temperatura, Umidade, Gás, IAQ	Linha	64,4	90,5	108,4	62,4	87,7	105,7	61,0	87,3	105,4
	Coluna	31,9	33,6	47,2	31,0	32,8	45,9	31,6	32,7	45,4

Fonte: Elaboração do próprio autor

Assim como na predição univariada do IAQ, a normalização por colunas gerou previsões com maior exatidão em todas as combinações multivariadas. Nota-se que, ao menos no período amostrado, variáveis que apresentaram maior dependência com o IAQ não garantiram melhores resultados. Avaliando os resultados da normalização por coluna e aplicando-se sobreposição de 1 *lag*, por exemplo, combinar o gás e o CO₂, que são as variáveis com maiores coeficientes de correlação com o IAQ, obteve 35,4 de MAPE. Nessa coluna da Tabela 8, a previsão mais exata ocorreu considerando a temperatura e a umidade nos padrões de entrada, que garantiram 31,3 de MAPE. Em comparação à previsão univariada do IAQ, cujo MAPE da previsão 1 hora à frente foi de 36,9, a série temporal multivariada com sobreposição de 1 *lag* de atraso composta pela temperatura e umidade gerou uma redução de 15,2% do MAPE em todo o período de previsão. Em relação às previsões 24 horas à frente, o MAPE médio dos conjuntos da previsão do IAQ a partir da temperatura e da umidade foi 9,8% menor em comparação ao MAPE da previsão univariada.

Em relação ao tamanho da sobreposição das janelas deslizantes, no geral, as séries multivariadas obtiveram melhores resultados a medida que o número de *lags* aumentou.

Logo, a rede neural ARTMAP *fuzzy* conseguiu extrair informações temporais dos padrões de entrada construídos com compostos que possuem maiores coeficientes de correlação entre instantes subsequentes das séries temporais, como temperatura e umidade, para realizar a previsão de outros compostos que não possuem tanta dependência entre os instantes de tempos, que foi o caso do IAQ.

É interessante ressaltar que a aplicação multivariada na metodologia de treinamento e predição utilizada exige que todas as entradas da ARTMAP *fuzzy* estejam obrigatoriamente nas saídas da rede neural. Isso porque, para obter previsão de instantes no futuro, é necessário montar os padrões de entrada empregando as respostas calculada em cada iteração. Logo, é possível realizar simultaneamente a previsão de todos os compostos presentes nos padrões de entrada, sendo necessário apenas desfazer a normalização dos pesos da categoria da ART_b *fuzzy* que contém as respostas da predição.

As avaliações apresentadas comprovaram a capacidade da rede neural ARTMAP *fuzzy* na previsão de séries temporais da qualidade do ar em ambiente interno. Foram demonstradas a combinação de várias possibilidades para realizar a modelagem dos dados de entrada, o que pode aprimorar a qualidade das predições. No geral, a normalização por coluna garantiu previsões mais exatas em comparação à normalização por linha. Em relação ao tamanho da sobreposição das janelas deslizantes, considerar mais *lags* por janela nem sempre garantiu melhores resultados, como foi o caso do IAQ na abordagem univariada.

O interesse na análise da qualidade do ar em ambientes internos torna-se mais visível devido às constantes pesquisas em desenvolvimento na atualidade. Além do IAQ calculado pela biblioteca BSEC da Bosch que interpreta os dados do BME680, recentemente surgiram outras opções de se realizar essas medições com algoritmos inteligentes. É o caso do sensor de VOC SGP40 integrado à uma biblioteca desenvolvida pela Sensirion (SENSIRION, 2020b). Assim como o IAQ calculado pelo BSEC, o software que interpreta os dados do SGP40 realiza estimativas da qualidade do ar interno por meio de um índice que varia de acordo com a concentração de VOC no ambiente monitorado.

Outro recente anúncio foi realizado pela Bosch na apresentação do BME688 (BOSCH SENSORTEC, 2021b). Além das características do BME680, o sensor de gás do BME688 também é capaz de detectar compostos de enxofre voláteis e, ainda, é possível calibrá-lo para detectar gases específicos. Para isso, realiza-se a amostragem de dados em diferentes situações, que são submetidos ao treinamento de uma RNA (BOSCH SENSORTEC, 2021a). Nesse treinamento supervisionado, geram-se parâmetros que são utilizados

pelas bibliotecas de código-fonte fechado da Bosch para o sensor realizar as leituras específicas dos gases, conseguindo assim classificar as classes aprendidas pela RNA durante o treinamento.

Analisando as características das redes neurais da família ART, um estudo interessante seria a aplicação dessas RNAs para classificar gases ou eventos específicos no monitoramento de ambientes internos, assim como a Bosch utilizou IA na interpretação de dados do BME688 para classificar gases. Não seria necessariamente obrigatório utilizar o BME688, basta realizar um estudo para verificar, a partir de dados amostrados por qualquer sensor, a possibilidade de extrair atributos dessas informações para realizar o treinamento das redes ARTs. A plasticidade e a estabilidade dessas redes dão suporte para a realização de treinamento on-line supervisionado e não supervisionado, possibilitando a implementação de sistemas de monitoramento e de predição que atuem em tempo real.

O foco desse capítulo foi apresentar o desenvolvimento da rede neural ARTMAP *fuzzy* em software implementada na linguagem de programação Python, que permitiu a realização de estudos para determinar a melhor configuração obtida na previsão do IAQ. Com o objetivo de aumentar a aplicabilidade da ARTMAP *fuzzy*, apresenta-se no próximo capítulo o desenvolvimento de uma arquitetura de hardware em FPGA dessa RNA.

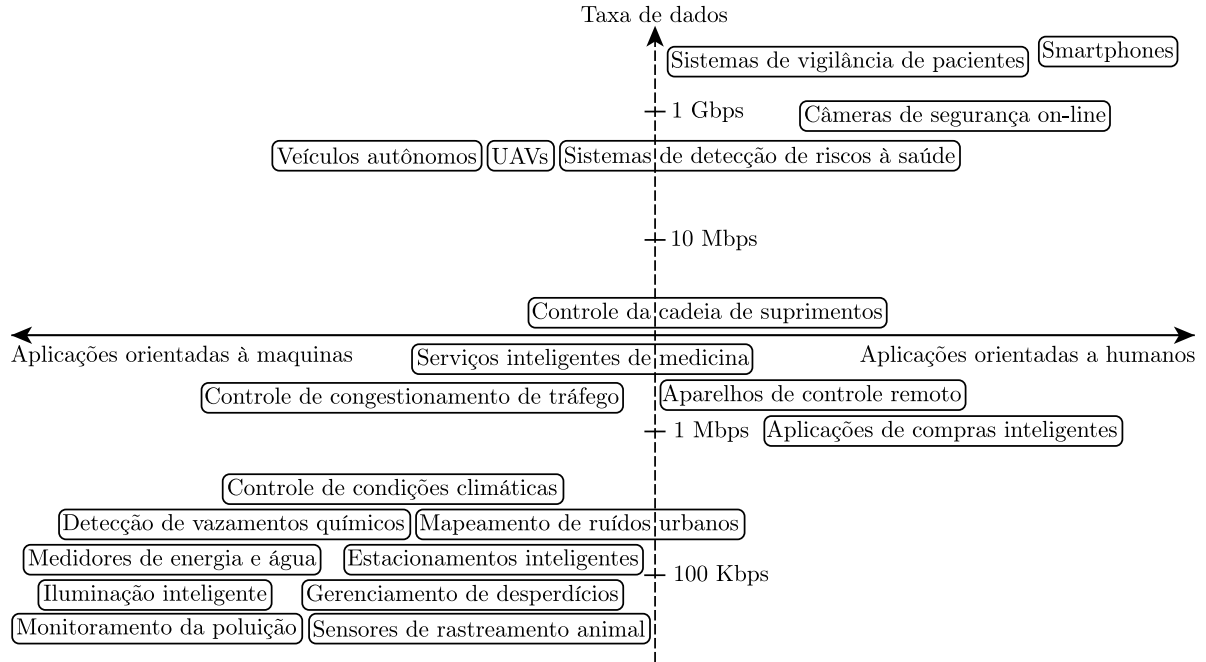
5 IMPLEMENTAÇÃO DA REDE NEURAL ARTMAP *FUZZY* EM HARDWARE

A facilidade de acesso a dispositivos de monitoramento tem criado uma crescente demanda no desenvolvimento de serviços computacionais. Além do armazenamento dos dados amostrados, é necessário interpretar e extrair informações relevantes para gerar as respostas esperadas pelos usuários desses sistemas. Considerando esses aspectos, intensificou-se a implementação de sistemas IoT, que passaram a integrar o cotidiano.

Diante da grande quantidade de aplicações assistidas pela IoT, existem várias configurações de dispositivos que são empregados na sua implementação. No geral, para se obter sistemas eficientes capazes de atender à demanda computacional exigida em cada contexto, deve-se considerar o limite de tempo de atuação estipulado. Há muitas variáveis que influenciam no tempo de processamento desses sistemas, como quantidade de informações produzidas, local de armazenamento e algoritmos aplicados na interpretação dos dados para se obter as respostas. Logo, o projeto de um sistema IoT deve considerar todos esses aspectos para ser bem-sucedido.

Especificamente em relação à quantidade de dados gerados por aplicações da IoT, pode haver diferenças significativa na taxa de informações amostradas, como está exemplificado na Figura 25. Há desde cenários simples, como a utilização de atuadores inteligentes que produzem poucos dados, até situações críticas com elevadas taxas de amostragem. Contextos que exigem respostas em intervalos de tempos reduzidos podem ser beneficiados se implementados em sistemas de computação de borda. Esses dispositivos localizam-se mais próximos das fontes dos dados, gerando menos latência e mais segurança no armazenamento de informações (CAO *et al.*, 2020). Mesmo se a aplicação exige atuação remota em relação aos espectadores, é possível empregar processamento local e enviar apenas as respostas obtidas.

O processamento local e centralizado geralmente é implementado em sistemas embarcados de tamanhos reduzidos, que executam tarefas específicas. A capacidade de processamento desses dispositivos miniaturizados pode ser insuficiente para realizar as tarefas de sistemas IoT com altas taxas de amostragens. A limitação computacional é ainda mais crítica quando se emprega algoritmos da IA para interpretar os dados,

Figura 25 – Taxa de dados produzidos por várias aplicações

Fonte: Adaptado de Ding *et al.* (2020).

pois tendem a aumentar a quantidade de cálculos, o que demanda mais tempo de processamento.

Com o objetivo de obter uma alternativa capaz de minimizar os gargalos de processamento existentes em plataformas baseadas em software, propõe-se o desenvolvimento de um sistema inteligente embarcado em hardware. Para isso, a rede neural ARTMAP *fuzzy* foi implementada em uma FPGA por meio da linguagem de descrição de hardware VHDL. Um dos principais diferenciais oferecidos pela arquitetura de hardware projetada foi a implementação do treinamento on-line da rede neural.

A avaliação da rede ARTMAP *fuzzy* desenvolvida em hardware foi realizada por meio da previsão do IAQ aplicando uma das séries temporais multivariada apresentadas. As variáveis de entrada foram a temperatura e a umidade, com 1 *lag* de atraso na sobreposição das janelas deslizantes. Logo, os padrões de entrada da ARTMAP *fuzzy* avaliada em hardware foram definidos por:

$$a = [t, T(t-1), U(t-1), T(t), U(t)] \quad (24)$$

sendo t o vetor de dados temporais (dia da semana e hora do dia) referente ao instante t da série temporal, T o valor da temperatura e U a umidade. A saída b corresponde aos valores da temperatura, umidade e IAQ no instante de tempo subsequente ao da entrada,

isto é:

$$b = [T(t + 1), U(t + 1), IAQ(t + 1)] \quad (25)$$

O resultado da previsão desejado é especificamente o IAQ. Porém, a temperatura e a umidade também são necessárias na saída para montar os padrões de entrada subsequentes, permitindo a previsão em instantes de tempo à frente. Tanto as saídas quanto as entradas foram normalizadas por colunas. As avaliações apresentadas para o desenvolvimento em hardware utilizaram os mesmos parâmetros da rede neural ARTMAP *fuzzy* nas implementações em software.

Ressalta-se que as taxas de amostragem de dados aplicadas em monitoramentos da qualidade do ar podem não ser elevadas, como ocorreu nessa pesquisa. Considerando as características do algoritmo da rede ARTMAP *fuzzy*, o tempo de processamento necessário para executar o treinamento on-line e o cálculo das previsões da qualidade do ar nessas condições não é um fator crítico, pois há dispositivos embarcados baseados em software com capacidade computacional suficiente. Um exemplo é o Raspberry Pi 3, que é capaz de prever, até determinadas taxas de amostragem, a qualidade do ar de ambientes internos em tempo real aplicando-se a rede ARTMAP *fuzzy* com treinamento on-line.

Na sequência apresenta-se um estudo realizado para obter um novo gerenciamento da quantidade de categorias da rede neural ARTMAP *fuzzy*, que foi ideal para a implementação em hardware.

5.1 PROPOSTA DE UM NOVO GERENCIAMENTO DE CATEGORIAS

Como a rede neural ARTMAP *fuzzy* agrega informações em forma de categorias à medida que incrementa seu conhecimento, o treinamento on-line realizado por longos períodos pode gerar grandes quantidades de pesos, principalmente se os padrões de entrada variarem frequentemente. Esse problema é minimizado aplicando-se a metodologia proposta por Marchiori *et al.* (2011), que define categorias definitivas e temporárias no módulo ART_a *fuzzy*. O objetivo é manter apenas as informações mais relevantes e descartar os dados com menos frequência de uso.

Contudo, ainda há possibilidade de ocorrer uma proliferação excessiva de categorias na ART_b *fuzzy*, pois não há nenhum controle da criação de categorias desse módulo. Esse problema é ainda mais crítico se a rede neural ARTMAP *fuzzy* for implementada em sistemas embarcados, que geralmente possuem capacidades de armazenamento de dados

mais limitadas.

Com o objetivo de minimizar o uso de memória para armazenar os pesos da neural ARTMAP *fuzzy* durante o treinamento on-line, propõe-se uma nova metodologia de gerenciamento desses dados. O objetivo foi limitar o número de categorias em ambos os módulos ART *fuzzy* seguindo a metodologia proposta por Marchiori *et al.* (2011), mas sem diferenciar as informações entre temporárias ou definitivas. Cada categoria possui um contador, sendo $sleep_a$ o da ART_a e $sleep_b$ o da ART_b, para armazenar a quantidade de ciclos que está ociosa, isto é, contabiliza o número de padrões de entrada apresentados à rede neural que a categoria não foi atualizada na ressonância. Ao criar ou atualizar uma categoria na ressonância, seu contador $sleep$ é zerado. A cada entrada recebida, as categorias que estiverem há uma certa quantidade de ciclos de treinamento inativas, sendo N_{max}^a e N_{max}^b respectivamente o limite permitido para a ART_a *fuzzy* e ART_b *fuzzy*, são deletadas. Consequentemente, o número máximo de padrões de entrada armazenados são os próprios parâmetros N_{max}^a e N_{max}^b .

Considerando o mapeamento que é realizado entre as informações de entrada e saída da ARTMAP *fuzzy* durante o treinamento supervisionado, a exclusão de uma categoria da ART_b pode tornar o cálculo de uma predição infactível, já que as categorias da ART_a que estavam mapeadas às informações deletadas não possuirão mais dados correspondentes de saída. Para solucionar esse problema, foram avaliadas duas possibilidades: retreinar as categorias da ART_a ou simplesmente excluí-las simultaneamente com as informações da ART_b. A atualização do mapeamento de uma categoria da ART_a consiste na execução do treinamento desse módulo ART *fuzzy* para encontrar a categoria mais similar existente, e então, aplica-se o mesmo mapeamento para os dados que estão sendo retreinados.

A premissa de manter e atualizar os dados de entrada que tiveram as saídas excluídas foi minimizar a perda de informações da ART_a, pois não há mais categorias definitivas. Assim, para verificar a influência da exclusão de diferentes quantidades de pesos na exatidão das previsões do IAQ, a rede ARTMAP *fuzzy* também foi treinada sem restrição no número de categorias. Esse estudo foi realizado em software implementado em Python executado em um laptop. Os resultados estão apresentados no Quadro 9. O número máximo de categorias nos casos com limitação foi de 336 (N_{max}^a) e de 168 (N_{max}^b), respectivamente para a ART_a *fuzzy* e ART_b *fuzzy*.

O MAPE da previsão do IAQ com memória ilimitada foi de 33,0. Com a restrição do número de categorias e aplicando-se a atualização do mapeamento Inter-ART após a exclusão de um padrão do módulo ART_b *fuzzy*, o erro percentual subiu para 34,3. Por

outro lado, a simples exclusão dos dados da ART_a mapeados às categorias excluídas da ART_b garantiu o melhor resultado das previsões, obtendo o MAPE de 30,5.

Quadro 9 – Avaliação da rede ARTMAP *fuzzy* com diferentes números de categorias

Categorias ilimitadas			Categorias limitadas Atualiza ART_a ao excluir ART_b			Categorias limitadas Exclui ART_a ao excluir ART_b		
N^a	N^b	MAPE	N^a	N^b	MAPE	N^a	N^b	MAPE
645	327	33,0	316	60	34,3	175	60	30,5

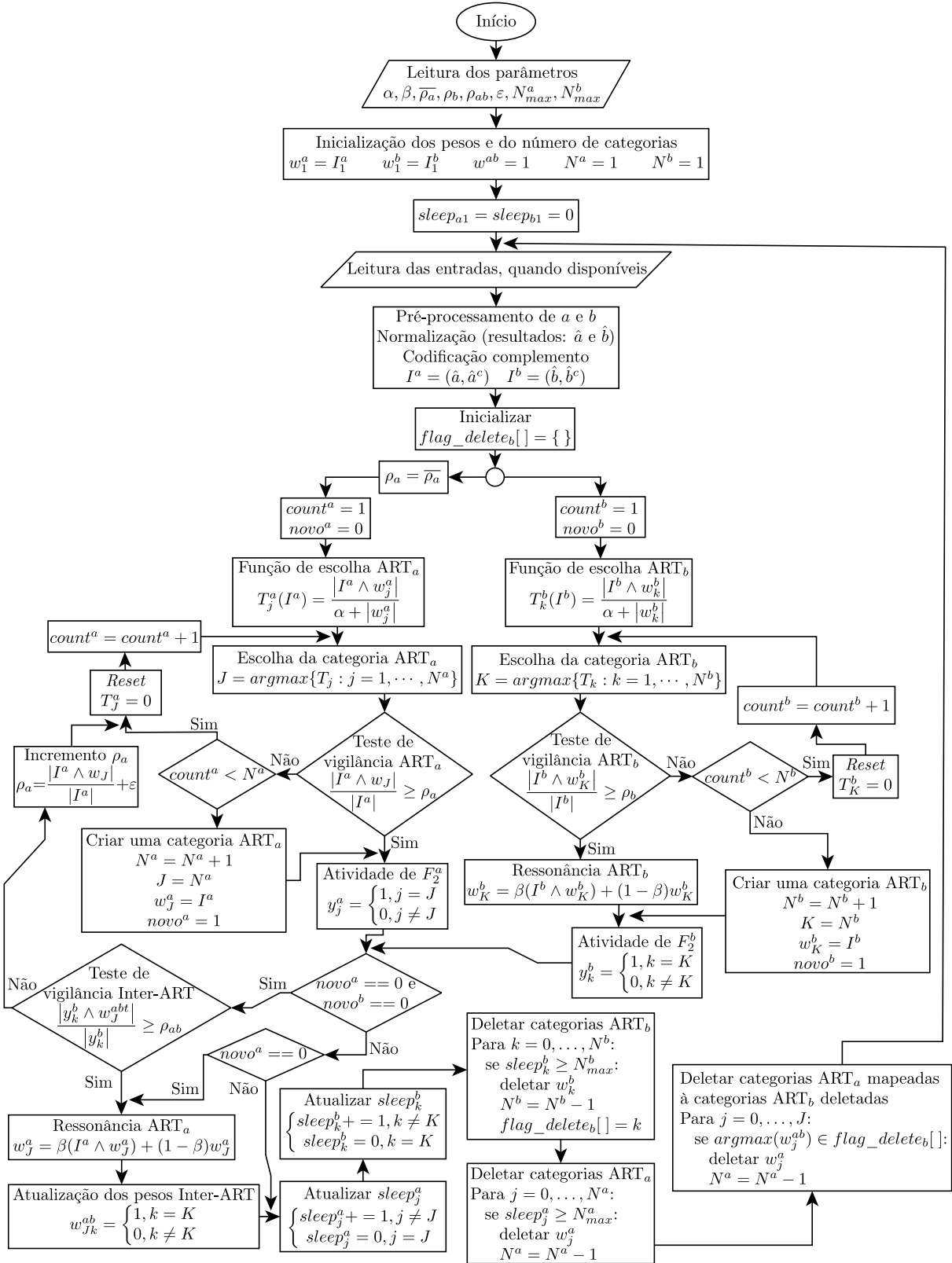
Fonte: Elaboração do próprio autor

Conclui-se que é possível manter e até mesmo aprimorar a exatidão das classificações aplicando-se um gerenciamento simples no número de categorias da rede ARTMAP *fuzzy*. Nesse caso em específico, a atuação dos parâmetros N_{max}^a e N_{max}^b foi o suficiente para minimizar em mais de 70% o consumo de memória por meio do armazenamento das informações recentes relevantes para a rede neural, compreendendo cerca de 1 a 2 semanas de dados. Caso a exatidão da previsão fosse afetada em consequência de poucos dados armazenados na ARTMAP *fuzzy*, poderia-se tentar relaxar as restrições do número de categorias, estendendo assim o período que os dados são mantidos.

Como a exclusão das categorias da ART_a associadas aos dados deletados da ART_b foi a técnica de gerenciamento que obteve a previsão mais exata, essa foi a metodologia aplicada no desenvolvimento embarcado na FPGA. A economia do uso de memória foi um fator importante para a descrição de uma arquitetura de hardware eficiente da rede neural ARTMAP *fuzzy* com treinamento on-line. Além de otimizar o consumo dos recursos de hardware, também garantiu respostas mais rápidas no cálculo das previsões.

O fluxograma da rede neural ARTMAP *fuzzy* aplicando-se a nova metodologia de exclusão de categorias está apresentado na Figura 26. Além dos novos parâmetros N_{max}^a e N_{max}^b , empregou-se um vetor denominado *flag_delete_b* para armazenar os índices das categorias excluídas da ART_b *fuzzy*. Esses valores são considerados na verificação do mapeamento Inter-ART para determinar se alguma informação da ART_a será descartada pela exclusão de dados de saída.

Figura 26 – Fluxograma da rede neural ARTMAP *fuzzy* com o novo gerenciamento de categorias



Fonte: Elaboração do próprio autor

5.2 OPERAÇÕES ARITMÉTICAS MODELADAS EM PONTO FIXO

A implementação de qualquer algoritmo em linguagens de descrição de hardware requer procedimentos que garantam a sintetização da arquitetura projetada. Um dos desafios é o cálculo de operações aritméticas. Focando-se nas FPGAs e na linguagem de descrição de hardware VHDL, as metodologias mais comuns para realizar cálculos aritméticos sintetizáveis são as operações em ponto flutuante e em ponto fixo.

Cálculos em ponto flutuante usualmente baseiam-se no padrão IEEE-754, que define a representação dos números com precisão simples, simples estendida e dupla. Em geral, as fabricantes de FPGA oferecem entidades de propriedade intelectual, ou IP (*intellectual property*) que geram hardware sintetizável para a execução de operações aritméticas em ponto flutuante nas unidades lógicas das FPGAs. Um exemplo é a unidade de ponto flutuante da Altera (ALTERA CORPORATION, 2016), que atualmente é propriedade da Intel.

Considerando operações aritméticas em ponto flutuante que utilizam a mesma padronização, como o IEEE-754, a acurácia de operações executadas em software e em hardware é a mesma. Contudo, modelar estes cálculos em hardware pode ser uma tarefa custosa, pois consomem consideráveis recursos físicos dos dispositivos baseados em hardware, como as FPGAs. Uma possibilidade de se otimizar essas operações nas FPGAs é utilizar dispositivos aprimorados com hardware específico para cálculos aritméticos, como o bloco DSP da Altera (ALTERA CORPORATION, 2014), que está presente nas FPGAs com maior capacidade de desempenho dessa fabricante.

Outra alternativa para otimizar a execução de operações aritméticas é a utilização de números em ponto fixo. Nessa representação, números reais são modelados como inteiros por meio de vetores binários compostos por duas componentes que armazenam a parte inteira e fracionária do número real. Calculam-se as operações com números inteiros, respeitando as restrições impostas pela posição do ponto fixo, e os resultados são convertidos em ponto flutuante. Essa metodologia consome menos hardware e os cálculos são executados mais rapidamente. Contudo, a utilização de ponto fixo limita a exatidão e o alcance dos números de acordo com a representação em valores inteiros utilizada. Para se obter resultados corretos, deve-se utilizar quantidades de bits suficientes para representar precisamente os valores existentes.

Com o objetivo de implementar o sistema de previsão da qualidade de ar em hardware com cálculos em ponto fixo, verificou-se o desempenho do algoritmo da rede ARTMAP

fuzzy com operações aritméticas modeladas em ponto fixo implementadas em Python. Essa metodologia gera os mesmos resultados que operações em ponto fixo calculadas em hardware sintetizado na FPGA, como foi obtido por Ferreira, Grout e Silva (2019) no desenvolvimento de uma regressão linear, o que auxilia na fase de projeto da descrição do hardware. Considerando todas as avaliações apresentadas no Capítulo 4 e os estudos em software específicos visando o desenvolvimento na FPGA, as etapas para se obter a configuração do hardware foram:

- Normalização dos dados de entrada: por coluna;
- Entradas / Saídas da RNA: Temperatura, umidade / Temperatura, umidade, IAQ;
- Tamanho da sobreposição das janelas deslizantes: sobreposição com 1 *lag* de atraso;
- Gerenciamento do número de categorias no treinamento on-line: além de restringir o número máximo de categorias dos módulos ART *fuzzy* por meio dos parâmetros N_{max}^a e N_{max}^b , ao excluir uma categoria ART_b , excluem-se as categorias ART_a associadas. No hardware, ambos N_{max}^a e N_{max}^b foram definidos em 200.
- Modelagem dos dados em ponto fixo: representação dos padrões de entrada e saída por meio de números inteiros de com bits, sendo 1 bit para a parte inteira e 15 bits para a parte fracionária. Nas operações que resultaram números maiores, utilizaram-se 5 bits na parte inteira para possibilitar a correta representação dos valores.

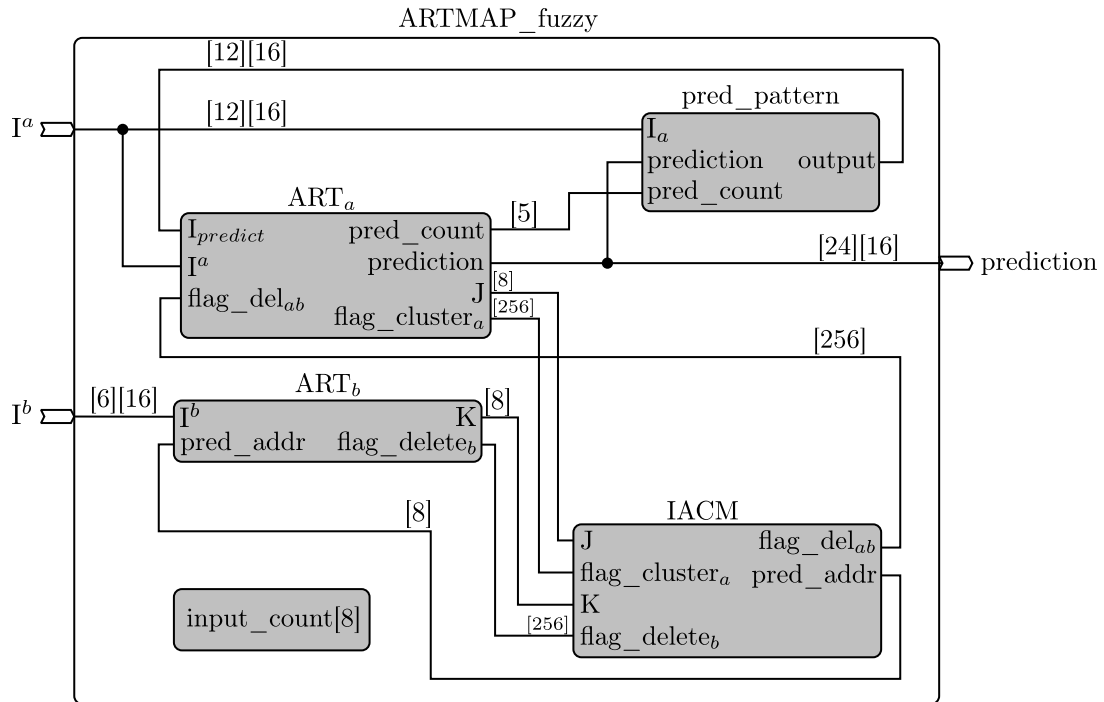
Utilizando cálculos em ponto flutuante padrão do Python, as demais configurações citadas e os dados utilizados no Capítulo 4, obteve-se a previsão do IAQ com MAPE de 30,38. Já esse mesmo caso, mas aplicando operações modeladas em ponto fixo, resultou no MAPE de 30,7. Assim, comprovou-se a eficácia da utilização de ponto fixo na realização dos cálculos em hardware. A implementação realizada na FPGA está apresentada na sequência.

5.3 ARQUITETURA DA REDE NEURAL ARTMAP *FUZZY* EM HARDWARE

Com o objetivo de se obter uma descrição em hardware customizável da rede ARTMAP *fuzzy* em FPGA capaz de atender outras aplicações além da previsão de séries temporais, projetou-se uma arquitetura parametrizável aplicando-se operadores da linguagem de descrição de hardware VHDL. Logo, é possível definir diferentes números de entradas e saídas da RNA para sintetizar a configuração necessária.

O diagrama de blocos simplificado da entidade topo da rede neural nomeada *ARTMAP_fuzzy* está apresentado na Figura 27. As entradas consistem nos vetores pré-formatados, isto é, normalizados, com codificação complemento e convertidos em ponto fixo. Logo, I^a e I^b possuem respectivamente 12 e 6 valores quantizados em 16 bits. Os módulos ART *fuzzy* foram desenvolvidas em entidades distintas nomeadas ART_a e ART_b . A entidade *pred_pattern* é responsável por criar os padrões de entrada utilizados nas previsões. Na entidade Inter-ART *cluster manager* (IACM), armazena-se o mapeamento Inter-ART e realiza-se o gerenciamento da exclusão de categorias da ART_a associadas às informações deletadas da ART_b durante o treinamento on-line. No vetor *input_count* armazena-se a quantidade de padrões de entradas que o sistema recebeu para controlar o início do cálculo das previsões. Utilizou-se um novo parâmetro na rede ARTMAP *fuzzy*, denominado P_{start} , cujo propósito é indicar o número do ciclo de treinamento no qual deve-se começar a calcular as previsões.

Figura 27 – Diagrama de blocos da entidade *ARTMAP_fuzzy*



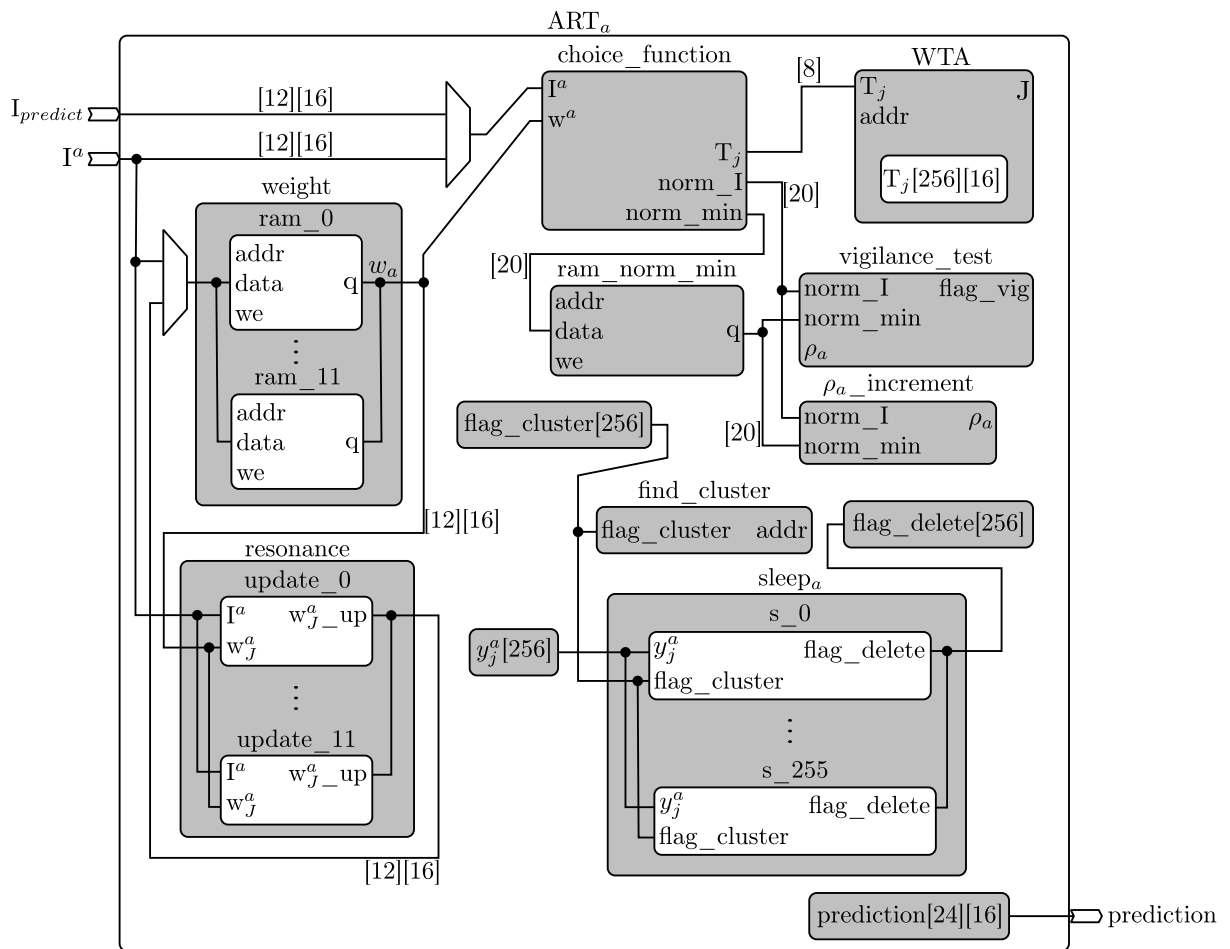
Fonte: Elaboração do próprio autor

Apresentam-se, na Figura 28, os sub-módulos da entidade ART_a , que são responsáveis pela execução das etapas específicas do algoritmo da rede neural ART *fuzzy*. O fluxo de dados e os sinais de controle dessas entidades são sincronizados por uma máquina de estado finitos, ou FSM (*finite state machine*).

Os pesos da ART_a *fuzzy*, isto é, as categorias criadas durante o treinamento, são armazenadas utilizando memórias *on-chip* da FPGA. Essas memórias são implementadas

no mesmo circuito integrado que contém a FPGA, garantindo menor latência de leitura e escrita de dados em comparação ao uso de memórias externas. Para otimizar os cálculos da função de escolha, instanciou-se um módulo de memória para armazenar cada um dos componentes de um padrão de entrada, como pode ser visto na entidade *weight*, sendo representado apenas o primeiro e o último módulo (*ram_0* e *ram_11*). Essa organização permite ler os 12 números de uma categoria paralelamente, que são considerados na entidade *choice_function* para calcular a função de escolha entre o padrão de entrada atual e cada categoria armazenada nos pesos da ART_a .

Figura 28 – Diagrama de blocos da entidade ART_a

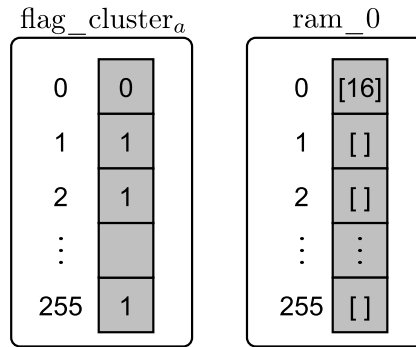


Fonte: Elaboração do próprio autor

Para possibilitar o gerenciamento do número de categorias durante o treinamento on-line, definiu-se que cada módulo *ART fuzzy* pode armazenar no máximo 256 categorias. Logo, cada módulo de memória *on-chip* alocado para guardar os pesos contém 256 posições de 16 bits em função do número de categorias e da quantidade de bits que as entradas foram modeladas em ponto fixo.

Quando uma categoria é criada, é necessário encontrar uma posição de memória livre para armazenar os novos dados. Complementarmente, a exclusão de informações torna espaços de memórias acessíveis. O controle da disponibilidade de memória foi realizado por meio de um vetor binário com 256 posições denominado $flag_cluster_a$, sendo que o valor de cada índice define se a posição de memória correspondente está livre (1) ou não (0). Um exemplo dessa operação está apresentado na Figura 29, que representa o resultado do armazenamento de uma categoria na primeira posição da memória. Ressalta-se que a exclusão de uma categoria não apaga os dados armazenados, apenas atualiza-se o índice associado do vetor $flag_cluster_a$ para 1, indicando que a posição está livre para receber o valor de uma eventual nova categoria.

Figura 29 – Exemplo de uso do vetor $flag_cluster_a$



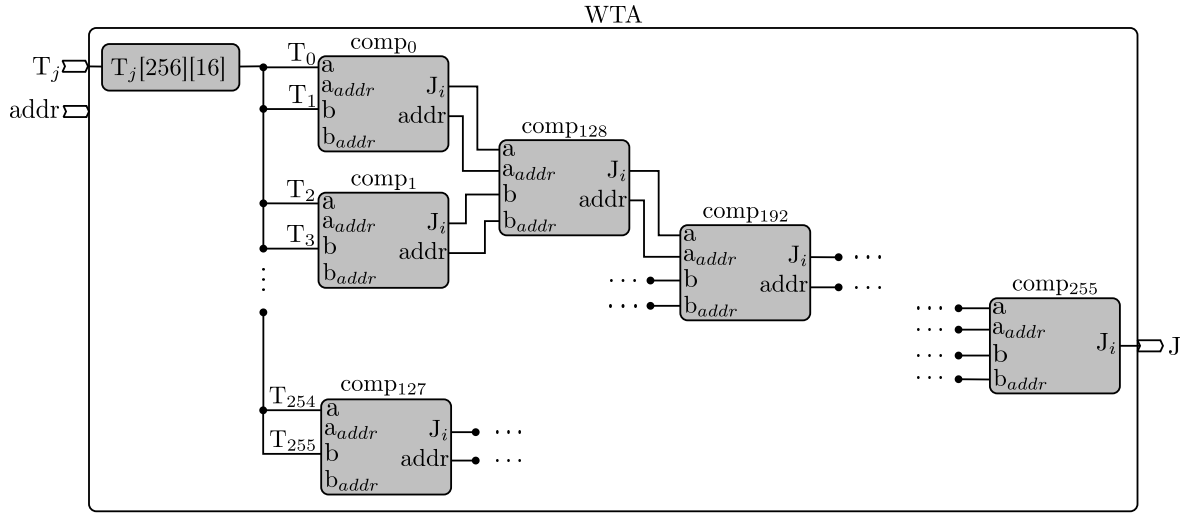
Fonte: Elaboração do próprio autor

O vetor $flag_cluster_a$ também é utilizado para encontrar as posições de memória que contém dados em uso durante os cálculos da função de escolha. Nesse processo, a entidade $find_cluster$ verifica sequencialmente os valores do vetor $flag_cluster_a$, e retorna o índice que coincide com o endereço das memórias que contém categorias armazenadas. O cálculo da função de escolha realizado na entidade $choice_function$ foi implementado em um processo sequencial do VHDL. A operação de divisão foi substituída pela multiplicação do inverso do denominador, calculado por meio do método iterativo de Newton-Raphson. Além dos T_j resultantes das funções de escolha, também foram mantidos os cálculos da norma do padrão de entrada ($norm_I$) e da norma do mínimo entre o padrão de entrada e os pesos ($norm_min$), pois são aplicados no teste de vigilância e no eventual incremento do p_a . Ao todo, são necessários 9 ciclos de $clock$ para realizar o cálculo de uma função de escolha.

Os resultados das funções de escolha são armazenados no vetor T_j com 256 posições dentro da entidade *winner takes all* (WTA). Para encontrar o índice J do maior valor de T_j , utilizou-se uma estrutura de comparadores em cascata que está apresentada na Figura 30. Cada comparador verifica qual é o maior entre 2 valores de T_j , e propaga o maior e o respectivo endereço para o nível subsequente, criando uma estrutura de *pipelining* que

necessita de 8 ciclos de *clock* para encontrar a saída.

Figura 30 – Diagrama de blocos da entidade *WTA*



Fonte: Elaboração do próprio autor

O teste de vigilância é verificado na entidade *vigilance_test* em um processo sequencial do VHDL. Como já mencionado, as normas de vetores aplicadas nessa operação são calculadas pela *choice_function*. O sinal *norm_I* é proveniente diretamente da *choice_function*, e considera-se o índice da categoria ativa, isto é, o valor *J* obtido na entidade *WTA*, como endereço de leitura para obter o valor *norm_min* armazenado no módulo de memória *ram_norm_min*. São necessários 3 ciclos de *clock* para calcular o teste de vigilância. Caso esse teste não seja satisfeito, aplica-se o reset na categoria ativa escrevendo o valor zero no índice *J* no vetor *J_t* e, se ainda houver categorias que não foram consideradas no teste de vigilância, a *WTA* é executada novamente para encontrar a próxima categoria candidata. Esse processo se repete até o teste de vigilância ser satisfeito ou criar uma categoria, que é adicionada na primeira posição de memória disponível localizada pela entidade *find_cluster*.

Na ressonância, as atualizações dos valores dos componentes de uma categoria ocorrem simultaneamente nas entidades *update* que operam em paralelo e pertencem à entidade *resonance* apresentada na Figura 28. No treinamento rápido ($\beta = 1$), a entidade *update* necessita de apenas um ciclo de *clock* para realizar as operações. Os novos valores atualizados substituem os correspondentes desatualizados armazenados nas memórias de pesos.

Basicamente, o hardware descrito na entidade *ART_a* foi duplicado para realizar o tratamento dos padrões de saída da rede neural ARTMAP *fuzzy* na entidade *ART_b*, que opera paralelamente à *ART_a*. A principal diferença entre as entidades dos módulos ART

fuzzy é que a ART_a contém as operações do módulo Inter-ART e a entidade $\rho_a_increment$.

Para realizar o mapeamento entre as categorias de entrada e saída, utilizou-se uma codificação da matriz w_{ab} diferente do padrão. Tradicionalmente, os pesos w_{ab} são armazenados em uma matriz binária com N^a linhas e N^b colunas, sendo que cada linha possui o valor 1 na coluna correspondente à categoria de saída mapeada. No hardware, o mapeamento foi armazenado em um vetor cujos índices representam as categorias da ART_a , e os valores inteiros armazenados correspondem aos índices das categorias ART_b mapeadas. Essa representação otimizou o hardware necessário para armazenar a matriz w_{ab} e o cálculo das predições. Um exemplo de mapeamento Inter-ART padrão e o correspondente utilizado em hardware está apresentado respectivamente nas Figuras 31(a) e 31(b).

Figura 31 – Diferentes metodologias de mapeamentos Inter-ART

	w_{b0}	w_{b1}	w_{b2}
w_{a0}	1	0	0
w_{a1}	0	0	1
w_{a2}	0	1	0
w_{a3}	1	0	0

(a) Mapeamento padrão.

	w_b
w_{a0}	0
w_{a1}	2
w_{a2}	1
w_{a3}	0

(b) Mapeamento aplicado em hardware.

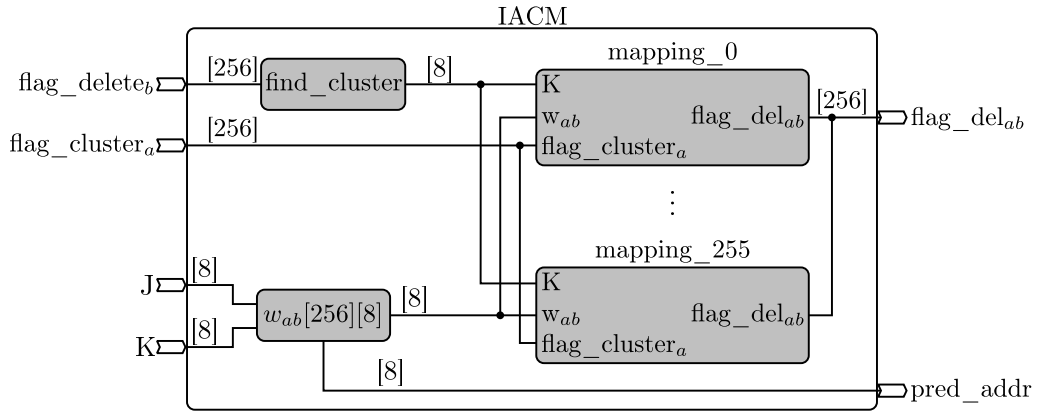
Fonte: Elaboração do próprio autor

Ao final de cada ciclo de treinamento, atualizam-se os contadores *sleep* que acumulam a quantidade de ciclos inativas das categorias. Esses contadores foram descritos em entidades com comportamento paralelo nomeadas *s* dentro da entidade *sleep*, como está apresentado na Figura 28. Com base nos valores do vetor y_j^a , que representa a atividade da camada F_2^a , e o vetor *flag_cluster*, incrementam-se apenas os contadores das posições onde há informações armazenadas, exceto no índice *J* que é zerado, pois foi atualizado na ressonância ou criou-se uma categoria.

Como foi proposto no novo método de gerenciamento da manutenção das informações da rede ARTMAP *fuzzy*, as categorias que permanecerem ociosas durante o número máximo de ciclos permitidos são excluídas. Nessa operação, as posições dos contadores *sleep* com valores iguais ou maiores ao limite definido são indicadas no vetor de saída *flag_delete*. Com base nos índices das categorias excluídas, as posições de memória que se tornaram disponíveis são atualizadas no vetor *flag_cluster*.

A exclusão das categorias na ART_b *fuzzy* é feita apenas a partir do vetor $flag_delete_b$. Contudo, na ART_a , além de basear-se no vetor $flag_delete_a$ é necessário verificar se há alguma informação de entrada mapeada à uma categoria excluída na ART_b . Essa operação foi adicionada ao processamento Inter-ART, sendo que os dados ficam armazenados na entidade *IACM* apresentada na Figura 32. Na *IACM* também localizam-se os mapeamentos Inter-ART no vetor w_{ab} .

Figura 32 – Diagrama de blocos da entidade *IACM*



Fonte: Elaboração do próprio autor

Na entidade *find_cluster*, procura-se sequencialmente no vetor $flag_delete_b$ por índices dos pesos das saídas deletadas. As entidades *mapping* recebem o índice da categoria ART_b excluída (K) e os mapeamentos armazenados em w_{ab} para verificarem, paralelamente em todas as categorias ART_a , se há alguma associada aos dados da ART_b excluídos. Caso haja, as ocorrências são reunidas no vetor de saída $flag_del_{ab}$ para ser associado ao vetor $flag_delete_a$ na exclusão das categorias da ART_a . Na entidade *IACM* também realiza-se o mapeamento entre as categorias ativas da ART_a e ART_b , cujos índices são respectivamente indicadas pelas entradas J e K .

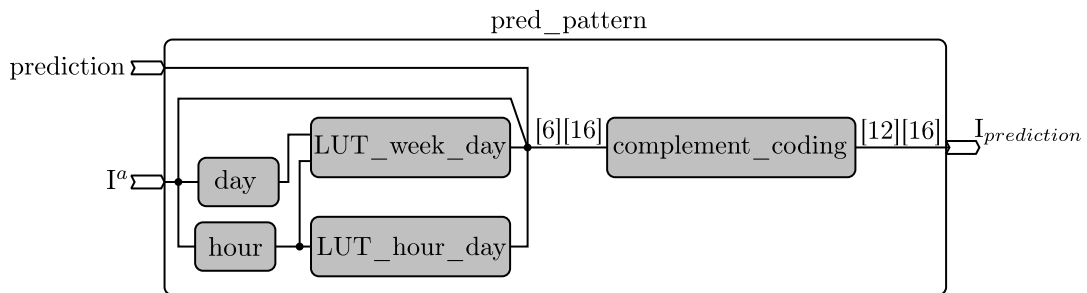
Assim que um ciclo de treinamento é finalizado, incrementa-se em uma unidade o contador *input_count*, que acumula a quantidade de padrões de entradas apresentados à rede neural. A finalidade dessa operação é controlar o início do cálculo das predições, que ocorrem a partir da iteração de treinamento P_{start} , já que no início da operação do sistema a rede neural ARTAMP *fuzzy* é apenas treinada. Com a obtenção do conhecimento inicial estabelecido, nas entradas subsequentes realiza-se tanto o treinamento quanto o cálculo das predições.

A primeira predição considera como entrada o padrão recém apresentado na ART_a *fuzzy*. Calculam-se as funções de escolha e obtém-se o maior T_J respectivamente nas entidades *choice_function* e *WTA* da entidade ART_a , que é apresentado à entidade

IACM para obter-se o mapeamento Inter-ART. De posse do índice da categoria de saída correspondente, a predição é obtida na entidade *weight* da ART_b , e é armazenada no vetor *prediction* localizado dentro da entidade ART_a .

Para criar padrões de entrada subsequentes ao atual e possibilitar predições de instantes futuros, é necessário combinar informações da última resposta da rede neural ARTMAP *fuzzy* e da entrada atual I^a . O processamento dessa etapa é realizado na entidade *pred_pattern* apresentada na Figura 33. Os dados temporais do dia da semana e da hora do dia são obtidos a partir de *look-up tables* (LUT), cujo objetivo é fornecer instantes subsequentes de tempo com base nos valores atuais contidos na I^a . Os compostos do ar são adicionados ao vetor do padrão de entrada selecionando-se os valores dos índices dos vetores de entrada adequados, e o cálculo da codificação complemento gera os dados necessários na entrada da entidade ART_a para calcular a predição. Esse processo se repete até que se obtenha a previsão um dia à frente.

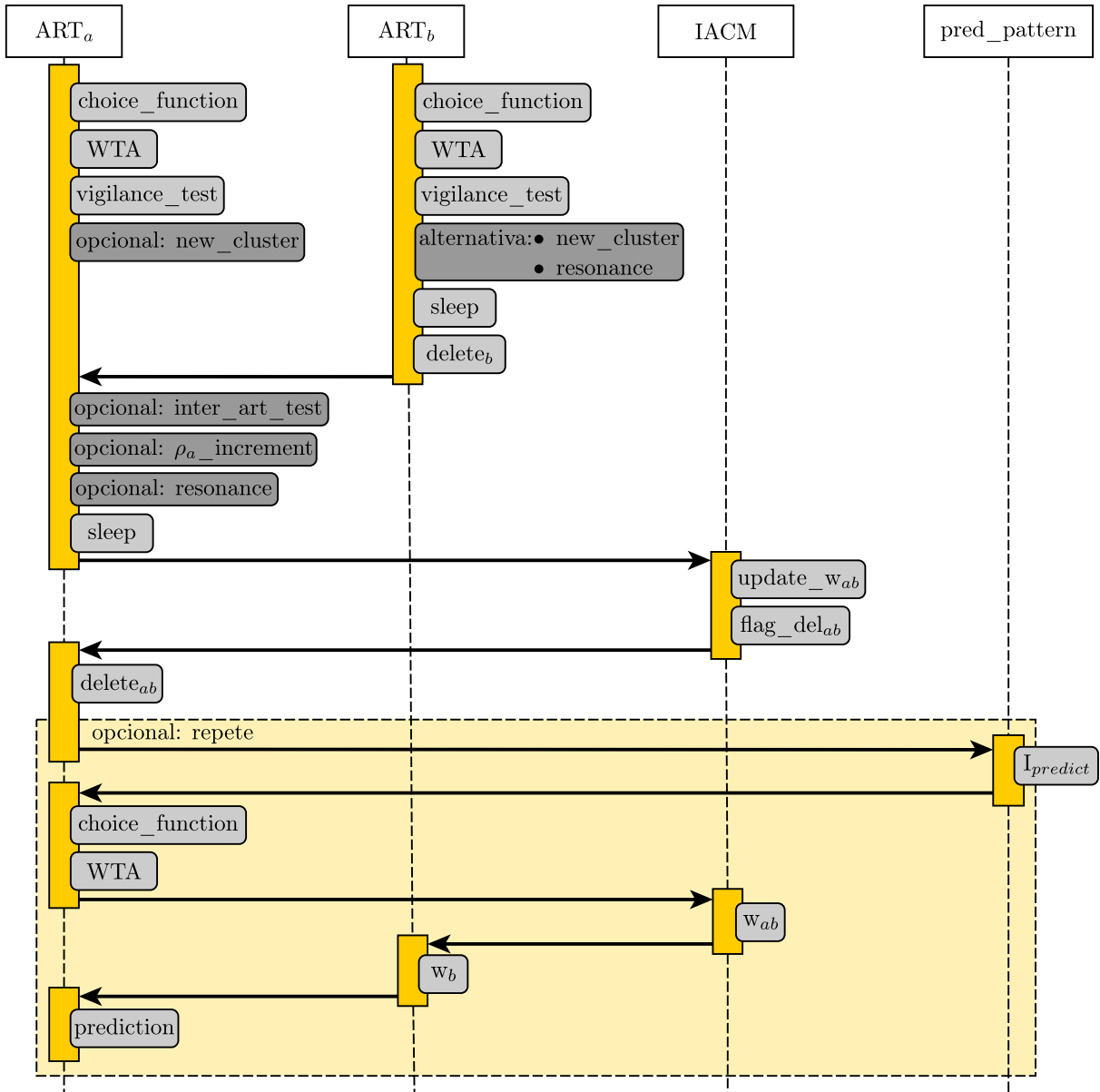
Figura 33 – Diagrama de blocos da entidade *pred_pattern*



Fonte: Elaboração do próprio autor

Apresenta-se, na Figura 34, o diagrama de sequência do fluxo de dados que ocorre na arquitetura da rede neural ARTMAP *fuzzy* em hardware. Os módulos ART_a *fuzzy* e ART_b *fuzzy* operam em paralelo, estando sincronizada a ativação de categorias em ambas as entidades devido à satisfação do teste de vigilância ou pela criação de categorias. Realiza-se o eventual teste de vigilância Inter-ART, e as informações de mapeamento são atualizadas na entidade *IACM*. Aplica-se o gerenciamento da manutenção de categorias, e se as predições estiverem liberadas, calculam-se sucessivas respostas para obter a previsão 24 horas à frente.

A avaliação da execução da rede ARTMAP *fuzzy* na FPGA está apresentada na sequência.

Figura 34 – Diagrama de sequência da rede ARTMAP *fuzzy* em hardware**Fonte:** Elaboração do próprio autor

5.4 AVALIAÇÃO DA DESCRIÇÃO EM HARDWARE

A descrição do hardware foi realizada em VHDL no software gratuito Quartus II 12.0 Web Edition e sintetizada na placa de desenvolvimento DE2-115, que possui uma FPGA Altera Cyclone IV E EP4CE115F29C7. Apresenta-se, no Quadro 10, o consumo dos recursos da FPGA necessários para implementar o hardware. Bits de memória correspondem à memória *on-chip* da FPGA, e os multiplicadores embarcados são hardware específico aplicado em operações DSP. Não houve nenhuma quantidade crítica de componentes para sintetizar a descrição proposta nesse modelo de FPGA.

Quadro 10 – Hardware da FPGA utilizado na descrição em VHDL

Componente	Total	Usado	Porcentagem de uso
Elementos lógicos	114480	50862	44%
Pinos de entrada e saída	529	59	11%
Bits de memória	3981312	83968	2%
Multiplicadores embarcados	532	88	17%

Fonte: Elaboração do próprio autor

Com mais da metade dos recursos dessa FPGA sem uso, é possível obter uma arquitetura mais otimizada da rede ARTMAP *fuzzy* seguindo a estrutura definida para configurar as entidades. Por exemplo, pode-se implementar entidades *choice_function* adicionais e fragmentar os pesos da RNA em mais unidades de memória *on-chip*, o que permite realizar cálculos paralelos das funções de escolha. Contudo, alterações na arquitetura devem considerar o consumo de partes específicas de recursos da FPGA. Por exemplo, com a quantidade de componentes de entradas consideradas nessa aplicação, uma entidade *choice_function* da ART_a *fuzzy* usa 26 multiplicadores embarcados, que pode ser um dos recursos mais limitadores desse modelo de FPGA. Logo, deve-se encontrar um equilíbrio entre a otimização do algoritmo e o uso dos componentes de hardware.

Para avaliar o hardware sintetizado, utilizou-se um laptop auxiliar que executou as operações de pré-processamento dos dados da qualidade do ar amostrados anteriormente. O pré-processamento consiste na modelagem da série temporal em janelas deslizantes, normalização e conversão em ponto fixo, que foi transmitida para a FPGA através de uma conexão serial. Na FPGA, executou-se o algoritmo da rede neural ARTMAP *fuzzy* e os resultados foram transmitidos de volta para o laptop. As previsões em ponto fixo foram convertidas em ponto flutuante com a finalidade de comparar os resultados do hardware com os dados reais e com as previsões calculadas em software.

Apresentam-se no Quadro 11 o MAPE, MAE e RMSE das previsões calculadas em hardware e em software, sendo o mesmo caso processado em Python para verificação da acurácia obtida na FPGA. Em hardware, o MAPE geral obtido foi de 31,0, e as previsões calculada pela execução em Python resultaram no MAPE de 30,7. Esse resultado comprova que a arquitetura proposta em hardware aplicando operações em ponto fixo obteve o resultado esperado.

A atuação da limitação de categorias durante a execução da rede ARTMAP *fuzzy* em hardware está apresentada na Figura 35. O maior número de dados no módulo ART_a *fuzzy* foi devido ao maior valor do parâmetro de vigilância comparado ao da ART_b *fuzzy*. Com

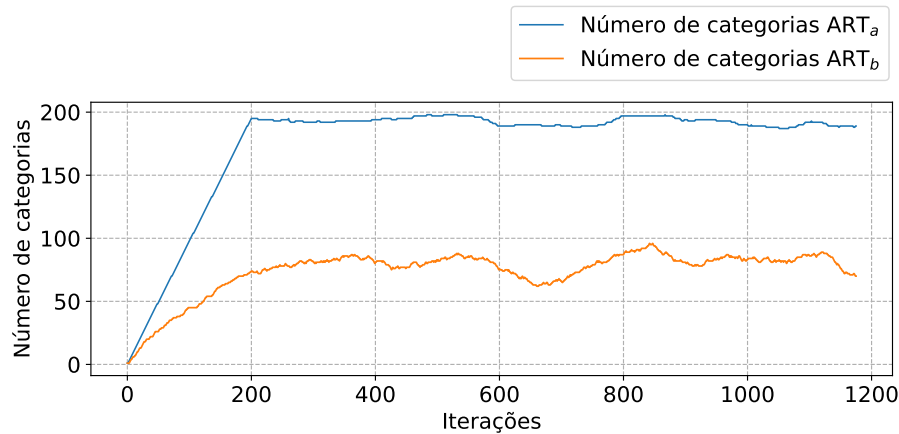
Quadro 11 – Comparação da acurácia da rede neural ARTMAP *fuzzy* implementada em software e em hardware

Dispositivo	MAPE	MAE	RMSE
Software (Python)	30,7	31,0	43,5
Hardware (FPGA)	31,0	31,3	44,1

Fonte: Elaboração do próprio autor

a exclusão simultânea de categorias da ART_b e dos dados da ART_a associadas, nota-se uma semelhança na tendência de ambos os gráficos durante alguns ciclos da RNA, como entre 400 e 800.

Figura 35 – Número de categorias criadas durante o treinamento on-line da rede neural ARTMAP *fuzzy* em hardware



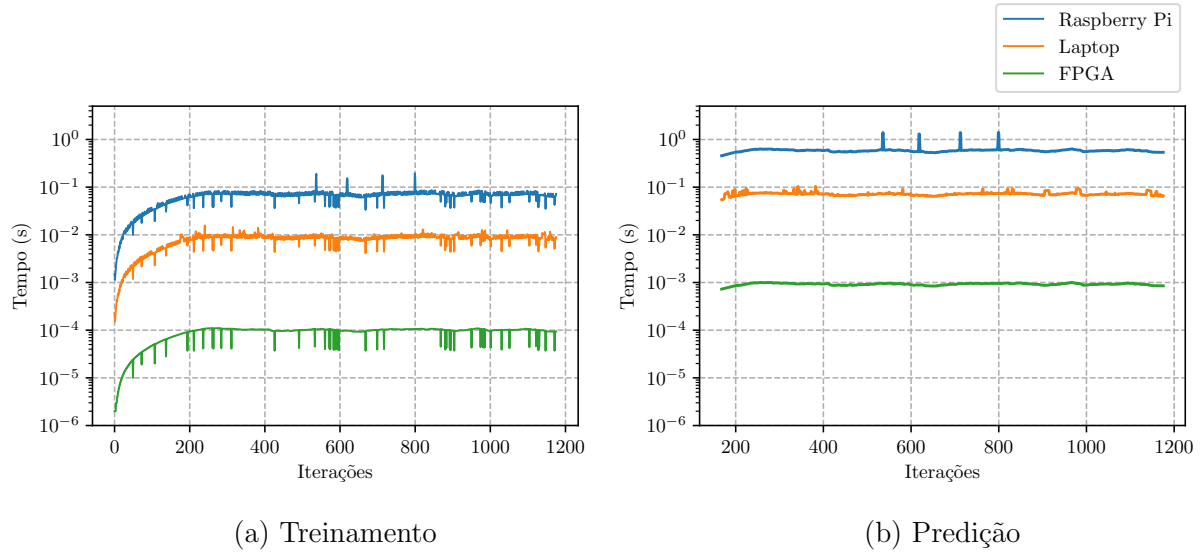
Fonte: Elaboração do próprio autor

Um dos aprimoramentos mais evidentes da descrição da rede neural ARTMAP *fuzzy* embarcada em hardware foi a obtenção de respostas mais rápidas em comparação à implementação em software. Com a finalidade de comparar o desempenho dessas diferentes abordagens, foram calculados os tempos necessários para realizar estritamente as operações de treinamento e de predição, sendo desconsiderados outros processos, como pré-processamento e transmissão de dados.

Os gráficos dos tempos em escala logarítmica gastos no treinamento e na previsão estão respectivamente apresentados nas Figuras 36(a) e 36(b). Realizou-se o monitoramento de cada ciclo de processamento da RNA, sendo que as predições ocorreram a partir da iteração 168. Para ter uma estimativa do comportamento do sistema implementado em uma versão embarcada com processamento em software, o mesmo código-fonte em Python avaliado no laptop foi executado no Raspberry Pi 3. Os maiores tempos foram do Raspberry Pi 3, seguidos pelo laptop e pelo hardware. Os tempos obtidos em hardware foram medidos

com a FPGA operando na frequência de 50 Mhz, e a arquitetura obtida pode operar até a frequência de 64 MHz. Logo, é possível otimizar os tempos do hardware em até 28%.

Figura 36 – Tempos de treinamento e predição executados em diferentes dispositivos



Fonte: Elaboração do próprio autor

Em software ocorrem picos positivos no tempo de execução decorrentes de fatores externos à execução do Python, como gerenciamento de processos do sistema operacional, o que gera latências. Esse problema não ocorreu no hardware descrito em VHDL, pois obteve-se uma arquitetura especializada que executa apenas o algoritmo projetado. Nos gráficos de treinamento há picos negativos, que representam iterações em que houve nenhuma ou baixa quantidade de reset no teste de vigilância da ART_a fuzzy.

Os valores extremos e da média dos tempos plotados nos gráficos da Figura 36 estão apresentados no Quadro 12. Os maiores tempos são importantes para identificar possíveis gargalos em aplicações que necessitam de maior taxa de amostragem. Por exemplo, se fosse considerado qualquer um dos sensores utilizados nesse trabalho com taxa de amostragem de 1Hz e o sistema proposto implementado no Raspberry Pi 3 para operar em tempo real, algumas leituras de dados seriam perdidas, pois houve cálculos de previsões que consumiram de 1 segundo de processamento.

Quadro 12 – Tempos de execução em segundos de diferentes dispositivos

Dispositivo	Treinamento			Predição		
	Menor	Maior	Média	Menor	Maior	Média
Raspiberry Pi	$1,14 \times 10^{-3}$	$1,95 \times 10^{-1}$	$6,50 \times 10^{-2}$	$4,57 \times 10^{-1}$	1,40	$5,88 \times 10^{-1}$
Laptop	$1,52 \times 10^{-4}$	$1,57 \times 10^{-2}$	$8,09 \times 10^{-3}$	$5,45 \times 10^{-2}$	$1,04 \times 10^{-1}$	$7,20 \times 10^{-2}$
FPGA	$2,00 \times 10^{-6}$	$1,11 \times 10^{-4}$	$9,04 \times 10^{-5}$	$7,27 \times 10^{-4}$	$1,01 \times 10^{-3}$	$9,23 \times 10^{-4}$

Fonte: Elaboração do próprio autor

Para facilitar a visualização do desempenho superior da FPGA em relação aos dispositivos baseados em software, apresentam-se no Quadro 13 as quantidades de vezes que o hardware foi mais rápido em comparação às execuções em software.

Quadro 13 – Relação de tempos de execução da FPGA com os dos dispositivos baseados em software

Dispositivo	Treinamento			Predição		
	Menor	Maior	Média	Menor	Maior	Média
Raspiberry Pi	570	1844	729	625	1477	637
Laptop	76	175	91	75	112	78

Fonte: Elaboração do próprio autor

Em hardware, considerando o pior caso, isto é, quando há o número máximo de categorias na ART_a *fuzzy*, são necessários 47 microssegundos para realizar o cálculo de uma predição. Especificamente na presente aplicação em que são calculadas 24 predições subsequentes para se obter a previsão do IAQ 24 horas à frente, a rede neural ARTMAP *fuzzy* em hardware gasta 1,13 milissegundos para gerar a resposta.

Em busca de outras implementações da rede ARTMAP *fuzzy* em hardware considerando especificamente descrições sintetizadas em FPGA, foram encontrados outros trabalhos na literatura. Yahiaoui, Alilat e Loumi (2017) realizaram a classificação de imagens de satélite, e Lomas-Barrie *et al.* (2021) implementaram um classificador de objetos para robôs por meio da análise de vídeo. No primeiro trabalho foi empregado apenas hardware, enquanto no segundo utilizou-se uma FPGA com microprocessador integrado que executou software em algumas operações auxiliares. Nesses trabalhos também foram realizados cálculos matemáticos modelados em ponto fixo quantizados em diferentes quantidades de bits suficientes para as aplicações específicas.

As demais implementações da rede neural ARTMAP *fuzzy* em hardware sintetizadas em FPGA obtiveram baixa latência para gerar os resultados, tal qual ocorreu no presente trabalho. Contudo, um diferencial da descrição de hardware proposta nessa pesquisa foi a implementação do treinamento on-line da rede neural ARTMAP *fuzzy*. Nas demais implementações em FPGA, aplicou-se treinamento off-line, sendo que a etapa de aprendizagem da RNA ocorreu em software processado em um microcomputador considerando informações amostradas anteriormente. Com a finalização do treinamento, os pesos armazenados nas categorias são transferidos para o hardware, que realizou apenas predições.

Como foi apresentada no Capítulo 3, a rede neural ARTMAP *fuzzy* com operações otimizadas para realização do treinamento on-line também é compatível com a

inicialização de pesos obtidos a priori. Assim, a arquitetura de hardware proposta nesse trabalho pode ser adaptada para contemplar apenas treinamento off-line, ou ainda, ser inicializada com pesos e continuar incrementando seu conhecimento durante a operação do sistema.

Com foco na previsão de séries temporais, há outras arquiteturas de RNAs implementadas em hardware sintetizado em FPGAs que atendem essa necessidade. É o caso da RNA com múltiplas camadas implementada em FPGA para prever séries temporais caóticas proposta por Pano-Azucena *et al.* (2018). Novamente, nesse trabalho considerou-se treinamento off-line em microcomputador, sendo descrito na FPGA apenas a arquitetura da RNA para calcular previsões.

É possível utilizar outros algoritmos de IA com treinamento on-line implementados em hardware. É o caso da rede neural *On-line Sequential Extreme Learning Machine* (OS-ELM), desenvolvida por Frances-Villora *et al.* (2018), que mostrou como a aprendizagem incremental realizada a cada novo padrão de entrada pode ser aplicada na classificação de imagens. A sintetização do hardware foi realizada automaticamente utilizando uma ferramenta de HLS (*high-level synthesis*), que gera hardware a partir de códigos-fonte escritos em linguagens de alto nível. Essa abordagem geralmente requer menos tempo na fase de projeto da arquitetura, contudo pode não garantir a melhor otimização possível. Outra configuração utilizada que gasta recursos consideravelmente foi a utilização de ponto flutuante de precisão dupla.

Particularmente no sistema desenvolvido por Frances-Villora *et al.* (2018), a quantidade de recursos físicos não foi crítica, pois utilizou-se uma FPGA de alto desempenho. Mesmo quando não se tem acesso à FPGAs com densa disponibilidade de hardware, ainda é possível desenvolver aplicações de IA com treinamento on-line, como comprovou-se no presente trabalho. Aplicando a linguagem de descrição de hardware VHDL e operações matemáticas modeladas em ponto fixo, obteve-se uma arquitetura específica e eficiente da rede neural ARTMAP *fuzzy* com treinamento on-line. Outra contribuição foi a proposta de uma nova metodologia do gerenciamento das categorias de ambos os módulos ART *fuzzy*, o que possibilitou controlar o uso de memória durante a aprendizagem contínua que ocorre a cada padrão de entrada recebido. Dessa forma, gerou-se um sistema que pode ser uma boa alternativa de processamento em computação de borda para ser empregado em aplicações IoT de tempo real com elevadas taxas de amostragens.

6 CONCLUSÕES

Nessa pesquisa desenvolveu-se a implementação de uma rede neural ARTMAP *fuzzy* com treinamento on-line para realizar a previsão de séries temporais da qualidade do ar em um ambiente doméstico interno, utilizado como quarto e escritório. Aplicando-se sensores de baixo custo, os seguintes compostos do ar foram amostrados: IAQ, temperatura, umidade, CO₂, VOC e PM.

A codificação da RNA em software foi realizada na linguagem de programação Python, que simulou o cálculo de previsões em tempo real. Foram realizadas previsões de cada um dos compostos amostrados modelados em séries temporais univariadas. No geral, normalizar separadamente as colunas da matriz que organizou os dados garantiu as previsões mais exatas, sendo o melhor resultado obtido na previsão da temperatura, com MAPE de 2,8 e 6,4, respectivamente na previsão de 1 hora à frente e na média dos conjuntos de previsões 24 horas à frente. Para obter melhores previsões do IAQ, os compostos do ar foram combinados para criar padrões de entradas multivariados. A melhor combinação foi a temperatura e a umidade, com a obtenção de 31,3 de MAPE, que foi 15,2% menor em comparação à previsão univariada do IAQ 1 hora à frente.

Outra contribuição foi a implementação da rede neural ARTMAP *fuzzy* em hardware, descrito em VHDL e sintetizada em uma FPGA. Para possibilitar o treinamento on-line embarcado, desenvolveu-se um novo gerenciamento da manutenção de categorias da ARTMAP *fuzzy*, que garantiu uma redução de 70% do uso de memória para armazenar o conhecimento da RNA, sem afetar a exatidão da previsão no IAQ. Considerando as médias dos tempos gastos no treinamento e na predição, a FPGA foi, respectivamente, 84 e 683 vezes mais rápida que um laptop equipado com um processador Intel Core-i5 8250U e que um Raspberry Pi 3, que executaram implementações do sistema em Python.

O tempo de resposta da previsão da qualidade do ar realizada nesse trabalho não foi um fator crítico para haver a necessidade da implementação em hardware. O principal objetivo foi oferecer uma inovação em algoritmos de ML baseados em hardware com treinamento on-line para atender aplicações IoT que precisam de respostas mais imediatas. Por meio da previsão do IAQ na FPGA, comprovou-se que é possível implementar a rede neural ARTMAP *fuzzy* com treinamento on-line em hardware

6.1 TRABALHOS FUTUROS

Como proposta de continuidade dessa pesquisa, provavelmente uma das ações mais imediatas é a implementação do sistema para operar em tempo real. Essa configuração pode ser exclusivamente embarcada em software, como Python processado em um Raspberry Pi 3, ou ainda aplicar o *co-design*, mantendo o pré-processamento em software e migrar a operação da rede neural ARTMAP *fuzzy* para hardware sintetizado em FPGA. Ressalta-se que, caso seja considerado o IAQ do BME680 nas previsões, é necessário utilizar um dispositivo compatível com a biblioteca BSEC da Bosch, nesse caso, o Raspberry Pi 3. Uma possibilidade para não depender de biblioteca de código fechado e ainda ter um índice da qualidade de ar, é utilizar outras metodologias de cálculo, o que incrementa o grau de liberdade para configurar o sistema embarcado. Um exemplo de equacionamento da combinação de vários gases para calcular um índice de IAQ foi realizado por Ha, Metia e Phung (2020).

Outro estudo possível é o monitoramento de outros ambientes internos onde há maior quantidade de pessoas. Atuando em tempo real, o sistema poderia manter os usuários do espaço assistido sempre informados a respeito da qualidade do ar, e o cálculo das previsões possibilitaria a tomada de ações preventivas para garantir condições saudáveis e confortáveis de ambientes fechados.

Especificamente considerando a implementação da rede neural ARTMAP *fuzzy*, há ainda outros estudos que podem ser investigados visando aprimorar o desempenho das classificações de séries temporais. Algumas dessas possibilidades são:

- tratamento de dados: não foi utilizada nenhuma técnica de tratamento nos dados amostrados nesse trabalho. Contudo, é possível aplicar um condicionamento desses sinais para obter informações com mais qualidade. Um exemplo é a aplicação de filtros, como a transformada *wavelet*;
- minimizar erro das previsões subsequentes: como apresentado, o uso de respostas da RNA na construção de padrões de entrada subsequentes para calcular a previsão de vários instantes à frente gerou maiores taxas de erro. Uma possibilidade para minimizar esse acúmulo de erro é a utilização de conceitos da estatística, como variância e desvio padrão, para comparar os valores das previsões com os dados reais da série temporal, e assim, realizar alguma atuação para aumentar a exatidão das previsões.

Seria interessante também comparar o comportamento e o desempenho das previsões da rede neural ARTMAP *fuzzy* com as de outros algoritmos de ML. Considerando especificamente algoritmos de previsões de séries temporais, alguns exemplos são os modelos auto-regressivos de média móvel, ou ARMA (*Autoregressive Moving Average*), e outras arquiteturas de redes neurais, como a LSTM e ELM, além de suas variantes.

Considerando a implementação em hardware, a arquitetura da ARTMAP *fuzzy* sintetizada em FPGA pode ser explorada em outros cenários. Como foi realizada uma descrição de hardware parametrizável, é possível sintetizar arquiteturas com diferentes números de componentes de entrada e saída apenas definindo constantes numéricas, o que facilita configurar a rede ARTMAP *fuzzy* de acordo com a necessidade da aplicação. Uma situação na qual esse sistema pode ser empregado é na predição de eventos envolvendo monitoramento de pessoas por meio de sensores corporais em tempo real, pois podem aplicar altas taxas de amostragem de dados.

REFERÊNCIAS

- ABDUL-WAHAB, S. A.; EN, S. C. F.; ELKAMEL, A.; AHMADI, L.; YETILMEZSOY, K. A review of standards and guidelines set by international bodies for the parameters of indoor air quality. **Atmospheric Pollution Research**, New York, v. 6, n. 5, p. 751 – 767, 2015.
- ALAM, F.; MEHMOOD, R.; KATIB, I.; ALBOGAMI, N. N.; ALBESHRI, A. Data fusion and IoT for smart ubiquitous environments: a survey. **IEEE Access**, New York, v. 5, p. 9533–9554, 2017.
- ALTERA CORPORATION. **The Industry’s First Floating-Point FPGA**. [S.l.], 2014.
- ALTERA CORPORATION. **Floating-Point IP cores user guide**. [S.l.], 2016. Ver. 1.0.
- AMATO, G.; CARRARA, F.; FALCHI, F.; GENNARO, C.; MEGHINI, C.; VAIRO, C. Deep learning for decentralized parking IoT occupancy detection. **Expert Systems with Applications**, Oxford, v. 72, p. 327 – 334, 2017.
- ASHTON, K. That “internet of things” thing. **RFID Journal**, Alpharetta, v. 22, n. 7, p. 97–114, 2009.
- AYYALASOMAYAJULA, H.; GABRIEL, E.; LINDNER, P.; PRICE, D. Air quality simulations using big data programming models. *In: IEEE INTERNATIONAL CONFERENCE ON BIG DATA COMPUTING SERVICE AND APPLICATIONS (BIGDATASERVICE)*, 2., 2016, Oxford. **Anais [...]**. Oxford, 2016. p. 182–184.
- BADURA, M.; BATOG, P.; DRZENIECKA-OSIADACZ, A.; MODZEL, P. Evaluation of low-cost sensors for ambient PM_{2.5} monitoring. **Journal of Sensors**, Hindawi, New York, v. 2018, 2018.
- BAHNFLETH, W. P. Defining indoor air quality. **ASHRAE Journal**, American Society of Heating, Refrigeration and Air Conditioning Engineers, Atlanta, v. 59, n. 5, p. 84, 2017.
- BARROS, A. C. **Detecção e classificação de distúrbios de tensão usando redes neurais ARTMAP euclidiana modificada com treinamento continuado**. 109 f. Tese (Doutorado em Engenharia Elétrica) — Faculdade de Engenharia, Universidade Estadual Paulista, Ilha Solteira, 2013.
- BOSCH SENSORTEC. **BME680 Low power gas, pressure, temperature & humidity sensor**. [S.l.], 2019.
- BOSCH SENSORTEC. **BME AI-Studio Documentation**. [S.l.], 2021.

- BOSCH SENSORTEC. **BME688 Digital low power gas, pressure, temperature & humidity sensor with AI**. [S.l.], 2021.
- CAO, K.; LIU, Y.; MENG, G.; SUN, Q. An overview on edge computing research. **IEEE Access**, Piscataway, v. 8, p. 85714–85728, 2020.
- CARON, A.; REDON, N.; CODDEVILLE, P.; HANOUNE, B. Identification of indoor air quality events using a K-means clustering analysis of gas sensors data. **Sensors and Actuators B: Chemical**, Lausanne, v. 297, p. 126709, 2019.
- CARPENTER, G. A.; GROSSBERG, S. ART 2: self-organization of stable category recognition codes for analog input patterns. **Applied Optics**, Washington, v. 26, n. 23, p. 4919–4930, 1987.
- CARPENTER, G. A.; GROSSBERG, S. A massively parallel architecture for a self-organizing neural pattern recognition machine. **Computer Vision, Graphics, and Image Processing**, Amsterdam, v. 37, n. 1, p. 54 – 115, 1987.
- CARPENTER, G. A.; GROSSBERG, S.; MARKUZON, N.; REYNOLDS, J. H.; ROSEN, D. B. Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. **IEEE Transactions on Neural Networks**, Piscataway, v. 3, n. 5, p. 698–713, 1992.
- CARPENTER, G. A.; GROSSBERG, S.; REYNOLDS, J. H. ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. **Neural Networks**, Oxford, v. 4, n. 5, p. 565–588, 1991.
- CARPENTER, G. A.; GROSSBERG, S.; ROSEN, D. B. Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. **Neural Networks**, Oxford, v. 4, n. 6, p. 759 – 771, 1991.
- CHEN, Q.; WANG, W.; WU, F.; DE, S.; WANG, R.; ZHANG, B.; HUANG, X. A survey on an emerging area: deep learning for smart city data. **IEEE Transactions on Emerging Topics in Computational Intelligence**, New York, v. 3, n. 5, p. 392–410, 2019.
- CHEN, S.; MIHARA, K.; WEN, J. Time series prediction of CO₂, TVOC and HCHO based on machine learning at different sampling points. **Building and Environment**, Oxford, v. 146, p. 238 – 246, 2018.
- CHOJER, H.; BRANCO, P.; MARTINS, F.; ALVIM-FERRAZ, M.; SOUSA, S. Development of low-cost indoor air quality monitoring devices: Recent advancements. **Science of The Total Environment**, Amsterdam, v. 727, p. 138385, 2020.
- CHOU, J. **Hazardous gas monitors: a practical guide to selection, operation and applications**. New York: McGraw-Hill Professional Publishing, 2000.
- DAMLJANOVIC, A.; LANZA-GUTIERREZ, J. M. An embedded cascade SVM approach for face detection in the IoT edge layer. *In: ANNUAL CONFERENCE OF THE IEEE INDUSTRIAL ELECTRONICS SOCIETY (IECON)*, 44., 2018, Washington. **Anais** [...]. Washington, 2018. p. 2809–2814.

DING, J.; NEMATI, M.; RANAWEERA, C.; CHOI, J. IoT connectivity technologies and applications: a survey. **IEEE Access**, New York, v. 8, p. 67646–67673, 2020.

DINIZ, W. F.; FREMONT, V.; FANTONI, I.; NÓBREGA, E. G. An FPGA-based architecture for embedded systems performance acceleration applied to optimum-path forest classifier. **Microprocessors and Microsystems**, Amsterdam, v. 52, p. 261 – 271, 2017.

DU, R.; SANTI, P.; XIAO, M.; VASILAKOS, A. V.; FISCHIONE, C. The sensible city: a survey on the deployment and management for smart city monitoring. **IEEE Communications Surveys Tutorials**, New York, v. 21, n. 2, p. 1533–1560, 2019.

ENVIRONMENTAL PROTECTION AGENCY. **Indoor Air Quality (IAQ)**. 2017. Disponível em: <<https://www.epa.gov/indoor-air-quality-iaq/introduction-indoor-air-quality>>. Acesso em: 9 Julho 2020.

EUROPEAN ENVIRONMENT AGENCY. **Air quality in Europe — 2018 report**. [S.l.], 2018.

FERREIRA, W. de A. P.; GROUT, I.; SILVA, A. C. R. da. FPGA hardware linear regression implementation using fixed-point arithmetic. *In*: SYMPOSIUM ON INTEGRATED CIRCUITS AND SYSTEMS DESIGN (SBCCI), 32., 2019, São Paulo. **Anais** [...]. São Paulo, 2019. p. 1–6.

FONSECA, A.; ABREU, I.; GUERREIRO, M. J.; ABREU, C.; SILVA, R.; BARROS, N. Indoor air quality and sustainability management - case study in three portuguese healthcare units. **Sustainability**, Amsterdam, v. 11, n. 1, 2018.

FRANCES-VILLORA, J.; ROSADO-MUÑOZ, A.; BATALLER-MOMPEAN, M.; BARRIOS-AVILES, J.; GUERRERO-MARTINEZ, J. Moving learning machine towards fast real-time applications: A high-speed FPGA-based implementation of the OS-ELM training algorithm. **Electronics**, Amsterdam, v. 7, n. 11, 2018.

GUO, K.; SUI, L.; QIU, J.; YU, J.; WANG, J.; YAO, S.; HAN, S.; WANG, Y.; YANG, H. Angel-eye: a complete design flow for mapping CNN onto embedded FPGA. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, Piscataway, v. 37, n. 1, p. 35–47, 2018.

HA, Q. P.; METIA, S.; PHUNG, M. D. Sensing data fusion for enhanced indoor air quality monitoring. **IEEE Sensors Journal**, Piscataway, v. 20, n. 8, p. 4430–4441, 2020.

IBRAHIM, A.; VALLE, M. Real-time embedded machine learning for tensorial tactile data processing. **IEEE Transactions on Circuits and Systems I: Regular Papers**, New York, v. 65, n. 11, p. 3897–3906, 2018.

KIRIMTAT, A.; KREJCAR, O.; KERTESZ, A.; TASGETIREN, M. F. Future trends and current state of smart city concepts: a survey. **IEEE Access**, New York, v. 8, p. 86448–86467, 2020.

KLEPEIS, N. E.; NELSON, W. C.; OTT, W. R.; ROBINSON, J. P.; TSANG, A. M.; SWITZER, P.; BEHAR, J. V.; HERN, S. C.; ENGELMANN, W. H. The national human

activity pattern survey (NHAPS): a resource for assessing exposure to environmental pollutants. **Journal of Exposure Science and Environmental Epidemiology**, Nature Publishing Group, New York, v. 11, n. 3, p. 231, 2001.

KUMAR, P.; MORAWSKA, L.; MARTANI, C.; BISKOS, G.; NEOPHYTOU, M.; SABATINO, S. D.; BELL, M.; NORFORD, L.; BRITTER, R. The rise of low-cost sensing for managing air pollution in cities. **Environment International**, Oxford, v. 75, p. 199 – 205, 2015.

LACEY, G.; TAYLOR, G. W.; AREIBI, S. Deep learning on FPGAs: Past, present, and future. **ArXiv**, New York, abs/1602.04283, p. 1–8, 2016.

LEE, J.; SHIN, D.; YOO, H. A 21mW low-power recurrent neural network accelerator with quantization tables for embedded deep learning applications. *In*: IEEE ASIAN SOLID-STATE CIRCUITS CONFERENCE (A-SSCC), 14., 2017, Seoul. **Anais** [...]. Seoul, 2017. p. 237–240.

LIN, B.; HUANGFU, Y.; LIMA, N.; JOBSON, B.; KIRK, M.; O'KEEFFE, P.; PRESSLEY, S. N.; WALDEN, V.; LAMB, B.; COOK, D. J. Analyzing the relationship between human behavior and indoor air quality. **Journal of Sensor and Actuator Networks**, Basel, v. 6, n. 3, 2017.

LINDWER, M.; MARCULESCU, D.; BASTEN, T.; ZIMMENNANN, R.; MARCULESCU, R.; JUNG, S.; CANTATORE, E. Ambient intelligence visions and achievements: linking abstract ideas to real-world concepts. *In*: DESIGN, AUTOMATION AND TEST IN EUROPE CONFERENCE AND EXHIBITION (DATE), 6., 2003, Munich. **Anais** [...]. Munich, 2003. p. 10–15.

LIU, Y.; MA, X.; SHU, L.; HANCKE, G. P.; ABU-MAHFOUZ, A. M. From industry 4.0 to agriculture 4.0: current status, enabling technologies, and research challenges. **IEEE Transactions on Industrial Informatics**, Piscataway, p. 1–13, 2020.

LOMAS-BARRIE, V.; PENA-CABRERA, M.; LOPEZ-JUAREZ, I.; NAVARRO-GONZALEZ, J. L. Fuzzy ARTMAP-based fast object recognition for robots using FPGA. **Electronics**, Amsterdam, v. 10, n. 3, p. 1–16, 2021.

LOPES, M. L. M. **Desenvolvimento de redes neurais para previsão de cargas elétricas de sistemas de energia elétrica**. 148 f. Tese (Doutorado em Engenharia Elétrica) — Faculdade de Engenharia, Universidade Estadual Paulista, Ilha Solteira, 2005.

LOY-BENITEZ, J.; VILELA, P.; LI, Q.; YOO, C. Sequential prediction of quantitative health risk assessment for the fine particulate matter in an underground facility using deep recurrent neural networks. **Ecotoxicology and Environmental Safety**, San Diego, v. 169, p. 316–324, 2019.

MARCHIORI, S. C.; G., M. do C.; LOTUFO, A. D. P.; MINUSSI, C. R.; LOPES, M. L. M. Neural network based on adaptive resonance theory with continuous training for multi-configuration transient stability analysis of electric power systems. **Applied Soft Computing**, Amsterdam, v. 11, n. 1, p. 706 – 715, 2011.

MARQUES, G.; PITARMA, R. mHealth: indoor environmental quality measuring system for enhanced health and well-being based on internet of things. **Journal of Sensor and Actuator Networks**, Basel, v. 8, n. 3, 2019.

MOHAMMADI, M.; AL-FUQAHA, A.; SOROUR, S.; GUIZANI, M. Deep learning for IoT big data and streaming analytics: a survey. **IEEE Communications Surveys & Tutorials**, Piscataway, v. 20, n. 4, p. 2923–2960, 2018.

MOLANES, R. F.; AMARASINGHE, K.; RODRIGUEZ-ANDINA, J.; MANIC, M. Deep learning and reconfigurable platforms in the Internet of Things: Challenges and opportunities in algorithms and hardware. **IEEE Industrial Electronics Magazine**, Piscataway, v. 12, n. 2, p. 36–49, 2018.

MUHURI, P. K.; SHUKLA, A. K.; ABRAHAM, A. Industry 4.0: A bibliometric analysis and detailed overview. **Engineering Applications of Artificial Intelligence**, Oxford, v. 78, p. 218 – 235, 2019.

OCCUPATIONAL SAFETY AND HEALTH ADMINISTRATION. Indoor air quality investigation. *In*: _____. **OSHA Technical Manual**. 1999. cap. 2. Disponível em: <https://www.osha.gov/dts/osta/otm/otm_iii/otm_iii_2.html>. Acesso em: 10 Julho 2020.

OSTA, M.; IBRAHIM, A.; CHIBLE, H.; VALLE, M. Inexact arithmetic circuits for energy efficient IoT sensors data processing. *In*: IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS (ISCAS), 51., 2018, Florence. **Anais** [...]. Florence, 2018. p. 1–4.

PACHECO, A.; CANO, P.; FLORES, E.; TRUJILLO, E.; MARQUEZ, P. A smart classroom based on deep learning and osmotic IoT computing. *In*: CONGRESO INTERNACIONAL DE INNOVACIÓN Y TENDENCIAS EN INGENIERÍA (CONIITI), 4., 2018, Bogota. **Anais** [...]. Bogota, 2018. p. 1–5.

PANO-AZUCENA, A. D.; TLELO-CUAUTLE, E.; TAN, S. X. D.; OVILLA-MARTINEZ, B.; FRAGA, L. G. De la. FPGA-based implementation of a multilayer perceptron suitable for chaotic time series prediction. **Technologies**, Amsterdam, v. 6, n. 4, 2018.

PÉREZ-SÁNCHEZ, B.; FONTENLA-ROMERO, O.; GUIJARRO-BERDIÑAS, B. A review of adaptive online learning for artificial neural networks. **Artificial Intelligence Review**, Dordrecht, v. 49, n. 2, p. 281–299, 2018.

PIERPAOLI, M.; RUELLO, M. L. Indoor air quality: A bibliometric study. **Sustainability**, Amsterdam, v. 10, n. 11, 2018.

PLANTOWER. **Digital universal particle concentration sensor. PMS7003 series**. [S.l.], 2016.

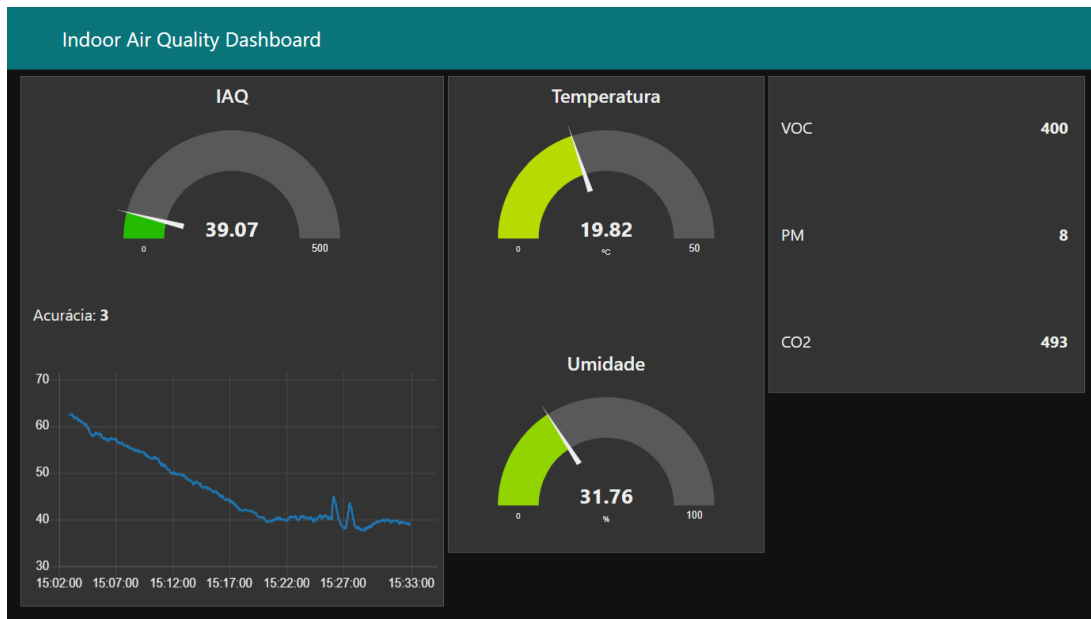
SAINI, J.; DUTTA, M.; MARQUES, G. Sensors for indoor air quality monitoring and assessment through internet of things: a systematic review. **Environmental Monitoring and Assessment**, Dordrecht, v. 193, n. 2, p. 1–32, 2021.

- SAMIE, F.; BAUER, L.; HENKEL, J. From cloud down to things: an overview of machine learning in internet of things. **IEEE Internet of Things Journal**, New York, v. 6, n. 3, p. 4921–4934, 2019.
- SEGUEL, J. M.; MERRILL, R.; SEGUEL, D.; CAMPAGNA, A. C. Indoor air quality. **American Journal of Lifestyle Medicine**, Los Angeles, v. 11, n. 4, p. 284–295, 2017.
- SENSIRION. **Datasheet SGP30. Indoor air quality sensor for TVOC and CO₂eq measurements**. [S.l.], 2020. Ver. 1.0.
- SENSIRION. **SGP40 - VOC Index for experts**. [S.l.], 2020.
- SEZER, O. B.; DOGDU, E.; OZBAYOGLU, A. M. Context-aware computing, learning, and big data in Internet of Things: a survey. **IEEE Internet of Things Journal**, New York, v. 5, n. 1, p. 1–27, 2018.
- SHAFIQUE, M.; THEOCHARIDES, T.; BOUGANIS, C.; HANIF, M. A.; KHALID, F.; HAFIZ, R.; REHMAN, S. An overview of next-generation architectures for machine learning: roadmap, opportunities and challenges in the IoT era. *In*: DESIGN, AUTOMATION TEST IN EUROPE CONFERENCE EXHIBITION (DATE), 21., 2018, Dresden. **Anais [...]**. Dresden, 2018. p. 827–832.
- SILVA, L. E. B. da; ELNABARAWY, I.; WUNSCH, D. C. A survey of adaptive resonance theory neural network models for engineering applications. **Neural Networks**, Oxford, v. 120, p. 167 – 203, 2019. Special Issue in Honor of the 80th Birthday of Stephen Grossberg.
- STORNAIUOLO, L.; SANTAMBROGIO, M.; SCIUTO, D. On how to efficiently implement deep learning algorithms on PYNQ platform. *In*: IEEE COMPUTER SOCIETY ANNUAL SYMPOSIUM ON VLSI (ISVLSI), 2018, Hong Kong. **Anais [...]**. Hong Kong, 2018. p. 587–590.
- TANABE, S.-I. International standards for better indoor air quality. **ASHRAE Journal**, American Society of Heating, Refrigerating, and Air-Conditioning Engineers, Atlanta, v. 59, n. 9, p. 86–89, 2017.
- VALLADARES, W.; GALINDO, M.; GUTIÉRREZ, J.; WU, W.-C.; LIAO, K.-K.; LIAO, J.-C.; LU, K.-C.; WANG, C.-C. Energy optimization associated with thermal comfort and indoor air control via a deep reinforcement learning algorithm. **Building and Environment**, Oxford, v. 155, p. 105 – 117, 2019.
- VERHELST, M.; MOONS, B. Embedded deep neural network processing: algorithmic and processor techniques bring deep learning to IoT and edge devices. **IEEE Solid-State Circuits Magazine**, New York, v. 9, n. 4, p. 55–65, 2017.
- VIEL, F.; SILVA, L. A.; LEITHARDT, R. Q. V.; ZEFERINO, C. A. Internet of things: Concepts, architectures and technologies. *In*: IEEE INTERNATIONAL CONFERENCE ON INDUSTRY APPLICATIONS (INDUSCON), 13., 2018, São Paulo. **Anais [...]**. São Paulo, 2018. p. 909–916.
- WANG, J.; WANG, Y.; ZHANG, D.; LV, Q.; CHEN, C. Crowd-powered sensing and actuation in smart cities: current issues and future directions. **IEEE Wireless Communications**, Piscataway, v. 26, n. 2, p. 86–92, 2019.

- WEI, W.; RAMALHO, O.; MALINGRE, L.; SIVANANTHAM, S.; LITTLE, J. C.; MANDIN, C. Machine learning and statistical models for predicting indoor air quality. **Indoor Air**, New Jersey, v. 29, n. 5, p. 704–726, 2019.
- WEISER, M. The computer for the 21st century. **Scientific American**, Nature Publishing Group, New York, v. 265, n. 3, p. 94–104, 1991.
- WEISER, M.; GOLD, R.; BROWN, J. S. The origins of ubiquitous computing research at parc in the late 1980s. **IBM Systems Journal**, Riverton, v. 38, n. 4, p. 693–696, 1999.
- WILLIAMS, R.; KILARU, V.; SNYDER, E.; KAUFMAN, A.; DYE, T.; RUTTER, A.; RUSSELL, A.; HAFNER, H. **Air sensor guidebook**. Washington: US Environmental Protection Agency, 2014.
- WINSEN. **Intelligent infrared CO₂ module. MH-19B**. [S.l.], 2016.
- WOEHRLE, H.; KIRCHNER, F. CAEMO - a flexible and scalable high performance matrix algebra coprocessor for embedded reconfigurable computing systems. **Microprocessors and Microsystems**, Amsterdam, v. 56, p. 47 – 63, 2018.
- WORLD HEALTH ORGANIZATION. **WHO guidelines for indoor air quality: selected pollutants**. Copenhagen: World Health Organization Regional Office for Europe, 2010.
- YAHIAOUI, R.; ALILAT, F.; LOUMI, S. Parallelization of fuzzy ARTMAP architecture on FPGA: multispectral classification of ALSAT-2A images. **IEEE Transactions on Industrial Electronics**, Piscataway, v. 64, n. 12, p. 9487–9495, 2017.
- YI, W. Y.; LO, K. M.; MAK, T.; LEUNG, K. S.; LEUNG, Y.; MENG, M. L. A survey of wireless sensor network based air pollution monitoring systems. **Sensors**, Basel, v. 15, n. 12, p. 31392–31427, 2015.
- ZADEH, L. Fuzzy sets. **Information and Control**, Amsterdam, v. 8, n. 3, p. 338 – 353, 1965.
- ZEBIN, T.; BALABAN, E.; OZANYAN, K. B.; CASSON, A. J.; PEEK, N. Implementation of a batch normalized deep LSTM recurrent network on a smartphone for human activity recognition. *In: IEEE EMBS INTERNATIONAL CONFERENCE ON BIOMEDICAL HEALTH INFORMATICS (BHI)*, 6., 2019, Chicago. **Anais [...]**. Chicago, 2019. p. 1–4.

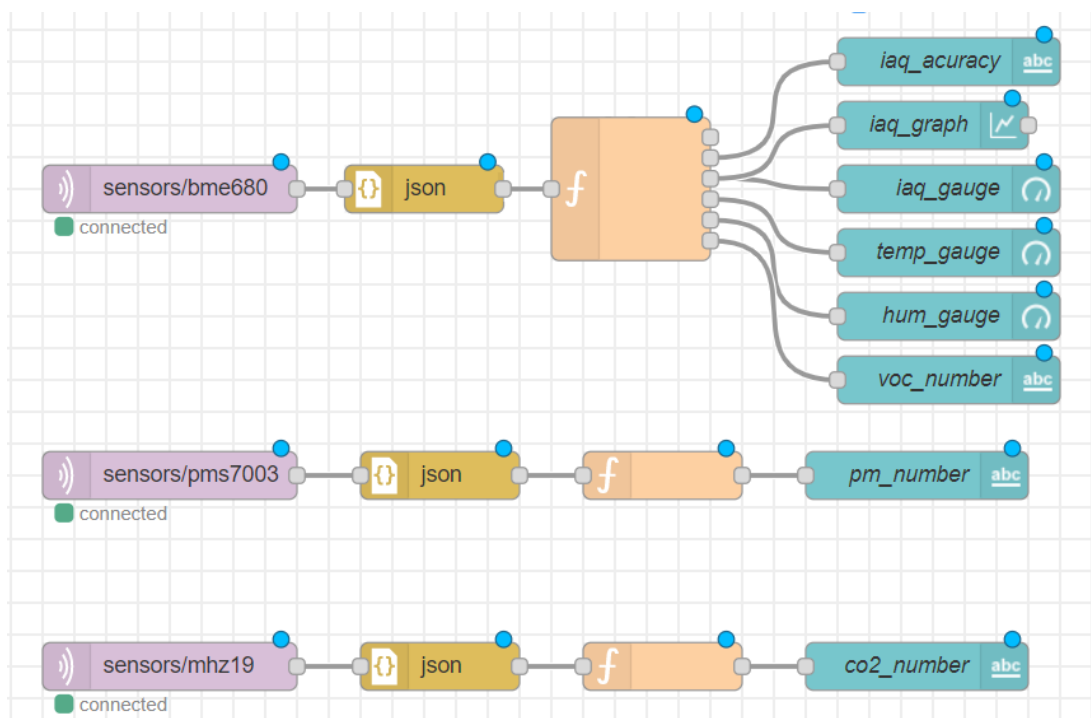
APÊNDICE A - *DASHBOARD* PARA VISUALIZAÇÃO DOS DADOS EM TEMPO REAL

Figura 37 – Dados da qualidade do ar atualizados em tempo real



Fonte: Elaboração do próprio autor

Figura 38 – Configuração do servidor Node-RED



Fonte: Elaboração do próprio autor

APÊNDICE B - AVALIAÇÃO DAS PREVISÕES DO IAQ APLICANDO SÉRIES TEMPORAIS MULTIVARIADAS

Quadro 14 – Avaliação das previsões do IAQ 24 horas à frente com séries temporais multivariadas

Entrada / Saída	Normalização	<i>Lag</i>								
		1			2			3		
		MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE
Temperatura, Gás / Temperatura, Gás, IAQ	Linha	66,1	91,8	106,2	66,2	91,9	106,2	66,4	92,3	106,5
	Coluna	63,2	62,3	75,9	65,5	64,2	77,7	65,8	64,1	77,5
Temperatura, CO2 / Temperatura, CO2, IAQ	Linha	65,6	92,8	107,3	64,4	89,8	104,1	60,8	84,3	98,2
	Coluna	65,3	63,3	76,6	63,9	62,9	76,1	64,5	63,7	77,13
Temperatura, VOC / Temperatura, VOC, IAQ	Linha	70,1	99,5	113,0	68,2	92,9	107,0	66,8	94,2	108,3
	Coluna	66,1	63,9	78,0	65,6	64,4	78,2	66,7	66,5	80,8
Umidade, Gás / Umidade, Gás, IAQ	Linha	66,7	92,4	106,7	65,8	92,2	106,5	66,5	92,5	106,6
	Coluna	65,2	63,1	76,7	62,6	61,2	74,4	65,3	62,4	75,7
Umidade, CO2 / Umidade, CO2, IAQ	Linha	63,6	86,9	100,9	62,2	88,0	102,1	64,8	91,8	106,1
	Coluna	65,2	63,0	76,2	65,5	62,9	75,8	65,4	62,5	75,2
Umidade, VOC / Umidade, VOC, IAQ	Linha	68,6	96,5	110,0	69,1	96,3	110,2	68,6	95,8	109,8
	Coluna	64,6	62,1	75,5	62,5	61,1	74,6	62,6	61,6	75,3
CO2, Gás / CO2, Gás, IAQ	Linha	67,7	94,4	108,2	67,0	94,0	107,8	65,9	92,4	106,4
	Coluna	61,7	61,9	74,9	60,3	61,4	74,2	63,1	63,1	76,1
CO2, VOC / Co2, VOC, IAQ	Linha	70,9	99,5	112,7	70,7	99,9	113,2	70,3	98,5	111,7
	Coluna	65,2	63,1	76,8	64,5	63,7	77,5	62,6	62,7	76,0
Temperatura, Umidade / Temperatura, Umidade, IAQ	Linha	67,7	78,7	94,1	68,9	81,5	97,17	73,3	90,1	104,8
	Coluna	62,6	61,7	74,7	63,2	62,2	75,2	62,9	61,5	74,4
Temperatura, Umidade, Gás / Temperatura, Umidade, ,Gás, IAQ	Linha	66,9	93,2	107,3	66,2	92,6	106,7	66,9	93,6	107,6
	Coluna	62,0	61,8	74,7	60,8	61,2	74,44	62,1	61,7	74,3

Fonte: Elaboração do próprio autor

Quadro 15 – Número de categorias criadas nas avaliações de séries temporais multivariadas

Entrada / Saída	Normalização	<i>Lag</i>								
		1			2			3		
		ART_a^t	ART_a^d	ART_b	ART_a^t	ART_a^d	ART_b	ART_a^t	ART_a^d	ART_b
Temperatura, Gás / Temperatura, Gás, IAQ	Linha	295	99	29	282	103	29	295	99	29
	Coluna	313	27	293	298	48	293	313	27	293
Temperatura, CO2 / Temperatura, CO2, IAQ	Linha	152	163	10	150	182	11	152	163	10
	Coluna	315	26	258	295	50	258	315	26	258
Temperatura, VOC / Temperatura, VOC, IAQ	Linha	159	266	9	211	223	10	159	266	9
	Coluna	295	36	233	271	68	232	295	36	233
Umidade, Gás / Umidade, Gás, IAQ	Linha	287	112	29	279	113	29	287	112	29
	Coluna	325	29	294	314	44	297	325	29	294
Umidade, CO2 / Umidade, CO2, IAQ	Linha	171	203	9	133	203	9	171	203	9
	Coluna	319	31	328	314	41	328	319	31	328
Umidade, VOC / Umidade, VOC, IAQ	Linha	161	267	9	196	229	10	161	267	9
	Coluna	315	25	299	296	43	298	315	25	299
CO2, Gás / CO2, Gás, IAQ	Linha	258	155	28	256	137	28	258	155	28
	Coluna	313	41	266	304	53	266	313	41	266
CO2, VOC / Co2, VOC, IAQ	Linha	180	261	10	163	272	11	180	261	10
	Coluna	293	58	225	285	67	223	293	58	225
Temperatura, Umidade / Temperatura, Umidade, IAQ	Linha	214	191	10	206	191	10	214	191	10
	Coluna	323	19	327	305	38	327	323	19	327
Temperatura, Umidade, Gás / Temperatura, Umidade, ,Gás, IAQ	Linha	282	110	29	279	119	29	282	110	29
	Coluna	300	52	348	292	66	352	300	52	348

Fonte: Elaboração do próprio autor