UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

PAULO FILIPE CÂNDIDO BARBOSA

UM MÉTODO DE OTIMIZAÇÃO DE ENERGIA MULTI-PROTOCOLO MAC E ADAPTÁVEL POR MEIO DE APRENDIZADO DE MÁQUINA PARA REDES IOT E DE SENSORES

Recife
2023

PAULO FILIPE CÂNDIDO BARBOSA

UM MÉTODO DE OTIMIZAÇÃO DE ENERGIA MULTI-PROTOCOLO MAC E ADAPTÁVEL POR MEIO DE APRENDIZADO DE MÁQUINA PARA REDES IOT E DE SENSORES

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação. Área de Concentração: Redes de Computadores e Sistemas Distribuídos.

Área de Concentração: Redes de Computadores e Sistemas Distribuídos

Orientador: Renato Mariz de Moraes

Coorientador: Cleber Zanchettin

Recife
2023

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

B238m Barbosa, Paulo Filipe Cândido
Um método de otimização de energia multi-protocolo MAC adaptável por meio de aprendizado de máquina para redes IOT e de sensores / Paulo Filipe Cândido Barbosa. – 2023.
142 f.: il., fig, tab.

Orientador: Renato Mariz de Moraes.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2023.
Inclui referências.

1. Redes de computadores. 2. Aprendizado de máquina. 3. Redes de sensores. I. Moraes, Renato Mariz de (orientador). II. Título.

025.04 CDD (23. ed.) UFPE - CCEN 2023-71

PAULO FILIPE CÂNDIDO BARBOSA

“ UM MÉTODO DE OTIMIZAÇÃO DE ENERGIA MULTI-PROTOCOLO MAC E ADAPTÁVEL POR MEIO DE APRENDIZADO DE MÁQUINA PARA REDES IOT E DE SENSORES ”

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação. Área de Concentração: Redes de Computadores e Sistemas Distribuídos.

Aprovado em: 21/03/2023.

Orientador: Renato Mariz de Moraes

BANCA EXAMINADORA

Prof. Dr. Adriano Lorena Inacio de Oliveira
Centro de Informática / UFPE

Prof. Dr. Leandro Maciel Almeida
Centro de Informática / UFPE

Prof. Dr. Paulo Roberto de Lira Gondim
Departamento de Engenharia Elétrica / UnB

Prof. Dr. Francisco Madeiro Bernardino Júnior
Escola UNICAP ICAM-TECH / UNICAP

Prof. Dr. Daniel Pedro Bezerra Chaves
Departamento de Eletrônica e Sistemas / UFPE

Dedico este trabalho a Deus, a minha esposa Amanda, a meu filho Isaque e minha família que foram porto seguro perante as dificuldades durante este percurso.

AGRADECIMENTOS

Agradeço primeiro a meu Deus, por me conceber mais essa vitória mesmo em meio aos problemas, adversidade e doenças que tive durante essa fase de minha vida, Ele se manteve fiel e justo me dando Inteligência, Sabedoria e Amigos. Em segundo agradeço a meu filho Isaque Góis Cândido Barbosa minha maior motivação para conclusão e a minha esposa Amanda Regina da Silva Góis que sempre me ajudou na minha trilha acadêmica. Agradeço a meu Orientador Prof. Renato Mariz, pela compreensão nas adversidades ocorridas, paciência e serenidade nas correções e ajuda, se fazendo presente sempre que precisei. Agradeço a meu Coorientador Prof. Cleber Zanchettin por ter aceitado o desafio de contribuir com seu conhecimento em IA para um trabalho na Área de Redes de Computadores. Agradeço também à Fundação de Amparo a Ciência e Tecnologia do Estado de Pernambuco, FACEPE, e à Nvidia *Corporation* financiadores dessas pesquisas. Dedico essa conquista a toda minha família que me ajudou direta e indiretamente na conclusão desse trabalho, principalmente meu pai Paulo Cândido, minha mãe Selma Maria, tia Severina Cândido, vó Maria das Dores e avô Luiz Cândido (em memória). A esses a minha eterna gratidão...

"E conhecereis a Verdade e a Verdade vos libertará"(Jo 3, 16).

RESUMO

Em redes IoT e redes de sensores, estudos têm sido realizados para definir o melhor esquema de gerenciamento para controle eficiente de energia, especialmente nos chamados dispositivos de baixa energia. Durante décadas, essa otimização foi realizada por meio de modelos e algoritmos que, apesar de apresentarem bons resultados, são incapazes de trabalhar sob diferentes protocolos, aprender e auto-adaptar as especificações do dispositivo sem a necessidade de interferência humana, tornando sua implementação em sistemas reais impraticável. Diante disto esta tese propõe duas contribuições: a primeira é a criação de um método baseado em aprendizado de maquina que combina as informações operacionais de dois algoritmos de controle de consumo de energia para os protocolos CSMA/CA, *slotted* ALOHA (uma variante de ALOHA), protocolos de referência usados na camada de enlace nas importantes tecnologias Wi-Fi e ZigBee. O método proposto aprende o comportamento de operacionalização de múltiplas variáveis de camada física (PHY) e de enlace (MAC) para diferentes dispositivos e protocolos, contribuindo para uma maior eficiência energética durante as transmissões. A segunda contribuição é uma Rede Neural Artificial (RNA) codificadas dentro do sensor agregando inteligência ao processo de transmissão de dados e tornando-o capaz de entender dinamicamente o estado atual da rede e manipular de forma autônoma variáveis das camadas PHY e MAC na busca de minimizar o custo de energia durante cada transmissão. Entre os resultados alcançados estão a redução de energia aproximando ou melhorando o desempenho das técnicas, economizando 97,6% na computação da CPU e 113.322.733% do tempo de processamento na busca pelas mesmas soluções em relação ao que o modelo de eficiência energética concorrente, obtidos na fase 1 da pesquisa. E na fase 2, a inserção de um sistema de otimização energética baseado em RNA, definido na literatura como um método de alto custo computacional, em um dispositivo sensor com 10 kB de RAM, 48 kB de ROM, e processador de 8 MHz operando sob o padrão IEEE 802.15.4. Para isso esta rede foi configurada com notação assintótica de magnitude $O(n)$, consumo operacional de energia entre 30 e 26 fJ e tamanho de 3,59 kB de memória ROM, sem a necessidade de importar bibliotecas adicionais do sistema.

Palavras-chaves: gerenciamento de energia; redes IoT; aprendizado de máquina; protocolos MAC; redes de sensores.

ABSTRACT

In IoT networks and sensor networks, studies have been carried out to define the best management scheme for efficient energy control, especially in so-called low-energy devices. For decades, this optimization was carried out through models and algorithms that, despite showing good results, are unable to work under different protocols, learn and self-adapt device specifications without the need for human interference, making their implementation in systems real impractical. In view of this, this thesis proposes two contributions: the first is the creation of a method based on machine learning that combines operational information from two energy consumption control algorithms for CSMA/CA protocols, *slotted* ALOHA (a variant of ALOHA), reference protocols used at the data link layer in leading Wi-Fi and ZigBee technologies. The proposed method learns the operationalization behavior of multiple physical layer (PHY) and data link (MAC) variables for different devices and protocols, contributing to greater energy efficiency during transmissions. The second contribution is an Artificial Neural Network (ANN) encoded within the sensor, adding intelligence to the data transmission process and making it capable of dynamically understanding the current state of the network and autonomously manipulating variables from the PHY and MAC layers in the search for minimize energy cost during each transmission. Among the results achieved are the energy reduction approaching or improving the performance of the techniques, saving 97.6% in the CPU computation and 113,322,733% of the processing time in the search for the same solutions in relation to what the efficiency model competitor energy, obtained in phase 1 of the research. And in phase 2, the insertion of an energy optimization system based on ANN, defined in the literature as a method of high computational cost, in a sensor device with 10 kB of RAM, 48 kB of ROM, and 8 MHz processor operating under the IEEE 802.15.4 standard. For this, this network was configured with asymptotic notation of magnitude $O(n)$, operational energy consumption between 30 and 26 fJ and size of 3.59 kB of ROM memory, without the need to import additional libraries of the system.

Keywords: energy consumption; IoT networks; machine learning; MAC protocols; sensor networks;

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

6LoWPAN	Redes de área pessoal sem fio IPv6 sobre baixa potência
AdaDelta	*Adaptive learning rate*
Adagrad	*Adaptive Gradient Algorithm*
Adam	*Adaptive Moment Estimation*
AF	Funções de Ativação
BLE	Bluetooth *Low Energy*
CARP	Protocolo de roteamento com reconhecimento de canal
CoAP	Protocolo de Aplicação Restrita
CTS	*Clear to Send*
CSMA/CA	*Carrier Sense Multiple Access with Collision Avoidance*
CW	*Contention Window*
DODAG	Grafo Direcionado Acíclico Orientado ao Destino
DTLS	*Datagram Transport Layer Security*
ELiSH	AF de *Squashing Exponencial linear*
GPS	*Global Positioning System*
GTS	Compartimentos de Tempo Garantido
HART	*Highway Addressable Remote Transducer*
HTTP	*Hypertext Transfer Protocol*
IP	*Internet Protocol*
IoT	Internet das Coisas
L2CAP	protocolo de adaptação de controle de link lógico
LoRaWAN	*Long Range* WAN
LR-WPAN	*Low Rate Wireless Personal Area Networks*
LTE	*Long Term Evolution*

LTE-A	*Long Term Evolution Advanced*
M2M	*Machine-to-Machine*
MAC	*Medium Access Control*
MCE	Modelo de Consumo de Energia
MIWI	*Embedded Wireless Networking Protocol*
ML	*Machine Learning*
MLP	*MultiLayer Perceptron*
MP2P	Multiponto-a-ponto
MQTT	Transporte de Filas de Mensagem de Telemetria
NesC	*Network Embedded Systems* C
OFDMA	Acesso Múltiplo por Divisão de Frequências Ortogonais
P2P	*Peer-to-peer*
PHY	Camadas Físicas
QoS	Qualidade de Serviço
RNA	Redes Neurais Artificiais
ReLU	Função de Unidade Linear Retificada
REST	*Representational State Transfer*
RPL	IPv6 *Routing Protocol for Low Power and Lossy Networks*
RTS	*Request to Send*
SO	Sistema Operacional
SDK	Kit de Desenvolvimento de Software
SDN	Rede Definida por Software
SGD	*Stochastic Gradient Descent*
TCB	Sistema de Controle de Tarefas
TCP	Protocolo de Controle de Transmissão
UDP	*User Datagram Protocol*

WI-Fi	*Wireless fidelity*
VANETs	*Vehicular Ad hoc Networks*
WAN	Rede de longa distância
WLAN	*Wireless Local Area Network*
WPAN	*Wireless Personal Area Networks*
WSN	Redes de Sensores sem Fio
XMPP	Protocolo extensível de mensagens e presença

LISTA DE SÍMBOLOS

α	Coeficiente de atenuação do sinal com a distância
α_m	Constante proveniente do tipo de modulação empregada
α_{amp}	Nível de potência do circuito de amplificação
β_m	Constante proveniente do tipo de modulação empregada
β_{amp}	Constante de proporcionalidade do circuito de amplificação
γ	Relação sinal-ruído
σ	Probabilidade de apenas uma estação transmitir com sucesso (*slotted* Aloha)
ζ	Probabilidade de apenas uma estação transmitir com sucesso (CSMA/CA)
λ	Comprimento de onda do sinal transmitido
μ	Relação inversa de σ
τ	Probabilidade que uma estação transmita no slot a ela destinado (CSMA/CA)
Δ	Combinação de intervalos de valores
B	Bytes

SUMÁRIO

1 INTRODUÇÃO

> *A ciência sem a religião é manca, a religião sem a ciência é cega.*
>
> Albert Einstein,

Este capítulo introduz as motivações que resultaram na elaboração de um modelo inteligente baseado em aprendizado de máquina que visa a redução do consumo de energia em redes de sensores sem fio (WSN, do inglês *Wireless Sensor Network*). Para tanto, são introduzidos problemas e soluções existentes na literatura que contribuem para a resolução das dificuldades ainda enfrentadas no acesso ao canal destas redes. Ao fim, este capítulo apresentará a metodologia utilizada na pesquisa e a organização estrutural desta tese de doutorado.

1.1 CONTEXTUALIZAÇÃO

Para dar suporte a um crescimento explosivo no tráfego de dados gerado por tecnologias emergentes, foram necessárias mudanças radicais na teoria das comunicações e a implementação de novas arquiteturas de rede (Khan et al., 2017).

Sistemas embarcados, internet móvel, *Big Data*, computação em nuvem e a inteligência artificial fizeram com que, principalmente, as comunicações sem fio e redes sem fio enfrentassem um acelerado desenvolvimento (Zhu et al., 2018).

Espera-se ainda que trilhões de máquinas, dispositivos, produtos e pessoas se conectem em tempo real em um sistema de informação único que englobe as áreas de controle industrial, biomédica, agricultura, cidades inteligentes, monitoramento ambiental e outros campos. Esses sistemas funcionarão sem interação humana direta usando apenas redes IoT (do inglês *Internet of Things*) para regulação e controle (Santos; Volante, 2018).

Porém o número crescente de redes IoT trouxe desafios na infraestrutura, gerenciamento, segurança e resiliência da rede, problemas que estão diretamente ligados a limitações na capacidade de armazenamento de dados, poder computacional e, principalmente, recursos energéticos limitados da grande maioria dos dispositivos que compõem essa rede (Neshenko et al., 2019).

1.2 MOTIVAÇÃO

Em redes IoT e redes de sensores, estudos vêm sendo realizados para definir o melhor esquema de gerenciamento de variáveis críticas para um controle energético eficiente, principalmente em dispositivos com recursos limitados de energia, uma vez que o tempo de vida desses dependem da capacidade de suas baterias (Da Silva; De Moraes, 2018).

Entre as soluções mais difundidas para gerenciamento de recursos em redes IoT e de sensores estão as modificações de tecnologias aplicadas a protocolos tradicionais de comunicação como Ethernet, Bluetooth e Wi-Fi. Muitas dessas tecnologias, inicialmente, não foram desenhadas para lidar com esse tipo de rede mas estão sendo adotadas por possuírem menores custos de seus equipamentos, baixo custo de implantação e um amplo conhecimento difundido, tornando-as de fácil modelagem na obtenção dos requisitos de novas aplicações e otimização dessas tecnologias (Sousa, 2018).

Especificamente os protocolos *slotted* ALOHA, CSMA/CA e IEEE 802.15.4 vêm recorrentemente sendo adaptados para otimizar o custo energético em dispositivos de baixa bateria ou *low energy*, através da busca de parâmetros ótimos para taxa de transmissão de dados, numero de bits por quadros, número de saltos, potência gasta nas transmissões, perdas de pacotes e detecção de sinais entre os dispositivos (Munari; Clazzer; Liva, 2015), (Mbacke; Mitton; Rivano, 2018), (Kumar; Bhuvana; Anushya, 2017).

Por décadas a otimização dessas variáveis foi realizada através de modelos e algoritmos que proporcionaram um melhor desempenho em redes sem fio (Sun et al., Jul. 2017). Um número significativo destas técnicas pertence à classe de algoritmos iterativos que trabalham com um determinado conjunto de parâmetros de rede em tempo real (ex: realizações de canal e especificações de relação sinal-ruído) como entradas e executam uma série de iterações (potencialmente complexas) para a otimização de recursos a partir de uma métrica de desempenho predefinida pela aplicação (Liu; Zhang; Gooi, 2020).

Portanto, apesar desses algoritmos apresentarem resultados satisfatórios, estes precisam de operações de alto processamento que encontrem as melhores configurações para a composição de estratégias ideais em cada cenário de transmissão. Para isso, essas técnicas usam esquemas de formulação complexa para ponderação de pesos das variáveis envolvidas na otimização dos recursos durante as transmissões, em sua maioria, desconsiderando uma otimização multiobjetivo da rede, gerando no decorrer de sua execução sobrecarga de banda em canais de transmissão, problemas de desempenho em redes grandes e consumo exagerado de energia dos dispositivos envolvidos (Patel; Jhaveri, 2016).

O uso de técnicas meta-heurísticas resolve alguns desses problemas, obtendo rapidamente grande diversidade de soluções, avaliando simultaneamente muitos parâmetros. Porém, mesmo encontrando várias soluções, essas técnicas não garantem soluções ótimas para o sistema, além de não avaliarem as consequências do uso prolongado dessas aplicações em uma rede real, mantendo assim os mesmos problemas de desempenho e consumo energético excessivo das técnicas tradicionais (Barbosa; Menezes, 2017).

Esses problemas ainda são vistos porque parte das técnicas tradicionais e heurísticas são simuladas e observadas somente através de computações numéricas e análises teóricas, tornando sua implementação em sistemas reais ainda um obstáculo, principalmente em dispositivos de baixo desempenho e com limitado recurso energético (Mao; Hu; Hao, 2018).

A natureza das soluções de gerenciamento de recursos exigem um alto custo com-

putacional tornando suas aplicações em dispositivos, como IoT, um processo desafiador. Isso porque as tarefas de gerenciamento são executadas em um período de milissegundos, precisando de respostas em tempo real devido a parâmetros sensíveis como condições do canal, número de dispositivos e usuários conectados, entre outros, os quais mudam rapidamente. Essa heterogeneidade de dispositivos e usuários em redes IoT trazem consigo outros problemas (Derhamy; Eliasson; Delsing, 2017).

Através de diferentes desenvolvedores, proprietários, fabricantes e partes interessadas diversos protocolos foram desenvolvidos resultados da evolução dos requisitos e da tecnologia, levando a um ecossistema altamente dinâmico de protocolos coexistentes, e muitas vezes incapazes de trabalhar um com o outro (Abdelouahid; Oqaidi; Marzak, 2018).

A interoperabilidade de protocolos em um ecossistema é um desafio crucial para a comunicação, principalmente em redes de vários dispositivos como a IoT. Pesquisas nessa área não são novas, embora a interoperabilidade via camada de aplicação só tenha sido abordada recentemente (Derhamy; Eliasson; Delsing, 2017).

Uma alternativa natural para a interoperabilidade é a convergência para um protocolo único que possibilite os dispositivos IoT darem um próximo passo de comunicação resultando em ambientes de redes inteligentes. Embora a criação da Internet tenha sido uma proposta de unificação desses protocolos, esta teve que manter ainda inúmeros outros protocolos baseados em TCP e UDP.

Este fato é explicado porque a realização de uma interoperabilidade ampla pode ser uma tarefa demorada associada a um esforço de desenvolvimento manual significativo. Considerando que grandes ambientes de IoT são compostos por inúmeros componentes diferentes, tornando a adaptação manual de diferentes protocolos e formatos de dados pouco prática (Kotstein; Decker, 2019).

Esse problema se mostra mais evidente nas soluções implementadas em software que, em sua maioria, tratam as camadas inferiores da rede como um bloco fortemente acoplado ou único não modificável. Desconsideram assim, a heterogeneidade de hardwares modelados na camada física (PHY) e as mudanças nas topologias da rede controladas pela camada de enlace (MAC) (Isolani et al., 2019).

1.3 PROBLEMA E HIPÓTESES DA PESQUISA

Conforme destacado anteriormente, problemas como otimização energética requerem das redes de computadores mais autonomia e mecanismos que incorporem inteligência em sistemas de controles garantindo assim vários requisitos de QoS (Fadlullah et al., 2017).

Nas últimas décadas, o aprendizado de máquina (ML, do inglês *machine learning*) foi muito explorado em sistemas IoT em áreas como Projetos de cidades inteligentes, monitoramento de saúde, segurança em rede, fabricação, automóveis autônomos, vigilância, controle inteligente de fronteiras; cada aplicativo tem suas idiossincrasias e requer recursos

personalizados, aprendizado adaptativo e fusão de dados, utilizado principalmente para tomadas de decisões e controle autônomo de sistemas (SHANTHAMALLU et al., 2017).

Diante disto propõem-se uma nova abordagem baseada em aprendizado de máquina que atue no gerenciamento autônomo de recursos energético otimizando o consumo durante trocas de informações em dispositivos sensores e IoT que utilizem protocolos *ad hoc slotted* Aloha, *Carrier Sense Multiple Access with Collision Avoidance* (CSMA/CA) ou IEEE 802.15.4.

A hipótese defendida e evidenciada por este trabalho foi que um dispositivo sensor é capaz de aprender simultaneamente, os critérios de otimização energética utilizados por diferentes modelos de consumo de energia advindas de diferentes protocolos, ainda que com eventual erro residual, sendo possível sua aplicação em sistemas reais de comunicação sem fio.

1.4 OBJETIVOS

O principal objetivo deste trabalho foi propor um novo modelo de acesso ao meio de comunicação (MAC) inteligente, baseado em aprendizado de máquina, capaz de se adaptar automaticamente a diferentes dispositivos que usem os protocolos *slotted* ALOHA, CSMA/CA ou IEEE 802.15.4 otimizando sua eficiência energética. Para alcançar esse objetivo foram definidos os seguintes objetivos específicos divididos em duas fases do projeto. Na primeira fase do projeto:

1. Modelar algoritmos atuais de gerenciamento de recursos energéticos utilizados pelos protocolos *slotted* ALOHA e CSMA/CA.

2. Modelar, desenvolver e treinar uma RNA de forma a otimizar os recursos energéticos em redes IoT e de sensores.

3. Realizar simulações numéricas implementando matematicamente hardwares de rádio para avaliar o comportamento da técnica proposta em diferentes cenários de transmissão.

Como objetivos específicos da segunda fase da pesquisa definiu-se:

1. Modelar, desenvolver e treinar uma RNA, compatível com sistemas sensores de baixo processamento, de forma a otimizar os recursos energéticos desse sensor.

2. Simular cenários reais que avaliem o comportamento da técnica proposta em diferentes transmissões.

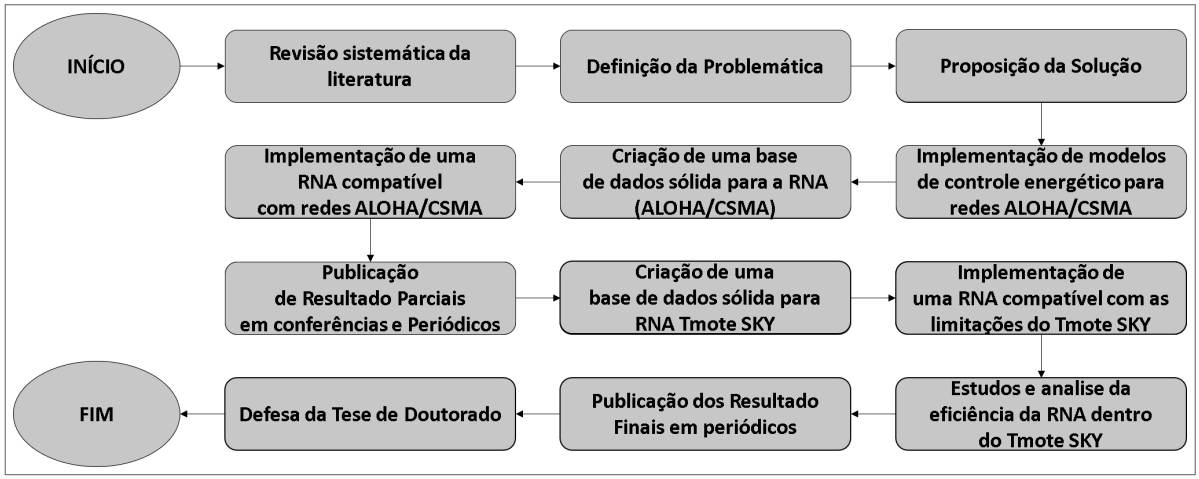
3. Realizar testes de eficiência energética nesses dispositivos.

1.5 METODOLOGIA DE PESQUISA

Trata-se de uma pesquisa científica de natureza tecnológica com objetivo explicativo através de procedimentos experimentais executados em laboratório com duração de 48 meses (JUNG, 2004).

A fim de alcançar os objetivos proposto nesse trabalho, definiu-se o percurso metodológico ilustrado na Figura 1, onde são exibidas, passo-a-passo, das atividades realizadas durante a pesquisa.

Figura 1 – Fluxograma de atividades do doutorado



Fonte: O autor

O percurso do estudo iniciou-se com uma extensa revisão sistemática da literatura em artigos científicos disponíveis nas plataformas IEEExplore, Google Scholar, arXiv, Elsevier e Portal de Periódicos Capes utilizando os descritores: "*iot survey*", "*iot network protocols*", "*iot multiprotocol*", "*iot Interoperability*", "*iot energy optimization*", "*machine learning iot interoperability*", "*machine learning control consumpsion*", "*sensor survey*", "*machine learning control consumpsion to WSN*" e "*RNA to WSN*".

Foi feita leitura dos títulos de cada trabalho relacionando-o à temática do estudo, seguida da leitura do resumo, tendo sido utilizados artigos em inglês ou português, publicados entre os anos de 2017 a 2023, ou um pouco mais antigos de alta relevância ao tema. Após a coleta dos materiais realizou-se uma leitura crítica e identificação das problemáticas.

Através da revisão literária do estado-da-arte foi possível entender que os protocolos PHY e MAC impactam significativamente variáveis de redes sem fio como taxa de transferência, consumo de energia, confiabilidade e interoperabilidade, formulou-se uma proposta de solução baseada na criação de um protocolo autônomo baseado em aprendizado de máquina capaz de atuar nas camadas inferiores da rede, considerando assim as especificações de hardwares dos dispositivos e as topologias de rede formadas durante as transmissões.

Este novo método de codificação em hardware e software para dispositivos IoT/WSN otimiza funções estáveis das camadas PHY e MAC inferiores implementadas em hardware como definição da potencia de transmissão, número de bits por quadro e taxa de transmissão, enquanto a camada MAC superior e de aplicação permanece no lado do kernel lidando com parâmetros variáveis da rede como distância entre os nós, números de saltos ideais do remetente ao destino e verificação/detecção de protocolos utilizados na transmissão.

O sistema, baseado em Redes Neurais Artificiais, coletou informações de algoritmos tradicionais de otimização energética em redes *slotted* ALOHA e CSMA/CA e dados reais aferidos em simuladores para o protocolo IEEE 802.15.4, buscando entender as regras do consumo energético em redes de sensores, inicialmente utiliando esses dados como *black box* para aprender suas relações de entrada/saída de forma a proporcionar um sistema único de gerenciamento de recursos que se adapte aos dispositivos IoT, *low memory*, independentemente do protocolo utilizado por eles (Barbosa et al., 2019).

Esse sistema RNA funcionará, em tempo de execução, em redes IoT e sensores garantindo um certo nível de adaptabilidade e autonomia aos dispositivos mesmo sem intervenção externa de outros hardwares de controle.

Para validação do sistema proposto serão realizados testes utilizando cenários de simulações distintos na tentativa de eliminar a criação de tendências nos resultados.

Pretende-se ainda submeter o protocolo a testes em sistemas reais em redes de sensores compostos por dispositivos do tipo MICAZ (Farshchi et al., 2006) ou Tmote Sky (VELINOV; MILEVA, 2016b). A partir dessa modelagem serão obtidas amostras do comportamento do protocolo que permitirão avaliar métricas de desempenho da rede neural dentro desses dispositivos.

Os resultados serão comparados a outras técnicas que visam otimização energética e as conclusões desses experimentos serão extraídos por meio de análises estatísticas dos dados de forma a obter conclusões que fundamentem o estudo.

1.6 CONTRIBUIÇÕES

Como consequência do percurso metodológico realizado esta tese traz duas contribuições principais:

1. A criação de um método baseado em RNA que combina as informações operacionais de dois algoritmos de controle de consumo de energia para os protocolos *slotted* ALOHA e CSMA/CA, aprendendo o comportamento de operacionalização de múltiplas variáveis de camada PHY e MAC que contribuem para uma maior eficiência energética em dispositivos que se utilizam desses protocolos, garantindo uma redução de energia de até 97,6% na computação da CPU e 113.322.733% do tempo de

processamento na busca pelas mesmas soluções que os modelos de eficiência energética concorrente.

2. Uma RNA de tamanho de 3,59 kB e consumo de operação entre 30 e 26 fJ, codificadas dentro de um dispositivo sensor com 10 kB de RAM, 48 kB de ROM, e processador de 8 MHz operando sob o padrão IEEE 802.15.4; utilizada para gerenciamento inteligente do consumo de energia através da otimização, de forma autônoma, de variáveis como o números de saltos do remetente ao destino, o número de bits por quadro e a potência de transmissão.

Outras contribuições resultantes da pesquisa como artigos científicos, base de dados registradas e patentes podem ser vista na Seção 9.4.

1.7 ORGANIZAÇÃO DA TESE

Este trabalho está organizado da seguinte maneira:

O Capitulo 2 relata a fundamentação teórica de Redes IoT, alguns protocolos de gerenciamento utilizados e os problemas enfrentados em transmissões em redes IoT/WSN reais;

O Capitulo 3 fundamenta modelos baseados em aprendizado de máquina, com ênfase em redes neurais artificiais, e o uso dessas técnicas para regressão de funções;

O Capítulo 4 mostra técnicas exatas e heurísticas utilizadas atualmente na resolução de problemas na área de interoperabilidade e otimização energética de dispositivos em redes IoT;

O Capítulo 5 descreve os detalhes da arquitetura, modelagem e implementação envolvidas na criação da rede neural para otimização energética de dispositivos que atuam sob protocolos *slotted* ALOHA e CSMA/CA, com o Capítulo 6 relatando os resultados e conclusões dos experimentos realizados;

O Capítulo 7 descreve os detalhes da arquitetura, modelagem, implementação e as modificações necessárias para a criação e inserção de uma rede neural dentro de um dispositivo do baixo de processamento Tmote Sky.

O Capítulo 8 relata os resultados, impactos e conclusões dos experimentos realizados no sensor. E o Capítulo 9 traz as conclusões e discute sobre possíveis desdobramentos futuros decorrentes da pesquisa.

2 REDES IOTS E DE SENSORES

> *Sabei escutar, e podeis ter certeza de que o silêncio produz, muitas vezes, o mesmo efeito que a ciência.*
>
> Napoleão Bonaparte,

Este capítulo fundamenta os conceitos básicos de redes IoTs, os principais protocolos destinados a esses dispositivos e os problemas enfrentados por essa rede. Na Seção 2.1 são explicadas características dos dispositivos IoTs junto com uma breve descrição das propriedades e problemas enfrentados pelas redes compostas por esses aparelhos. Na Seção 2.2 é feita uma revisão bibliográfica sobre os principais protocolos destinados a gerenciamento de conexão da rede IoT culminando nos mais difundidos na literatura como CSMA/CA, *slotted* ALOHA e ZIGBEE. Já na Seção 2.3 são expostas as principais dificuldades enfrentadas por esses protocolos. Na Seção 2.4 são detalhados com especificidades os problemas que este trabalho se propõe a resolver como o consumo energético e a interoperabilidade de comunicação entre dispositivos. Por fim, a Seção 2.5 apresenta uma breve discussão sobre o capítulo.

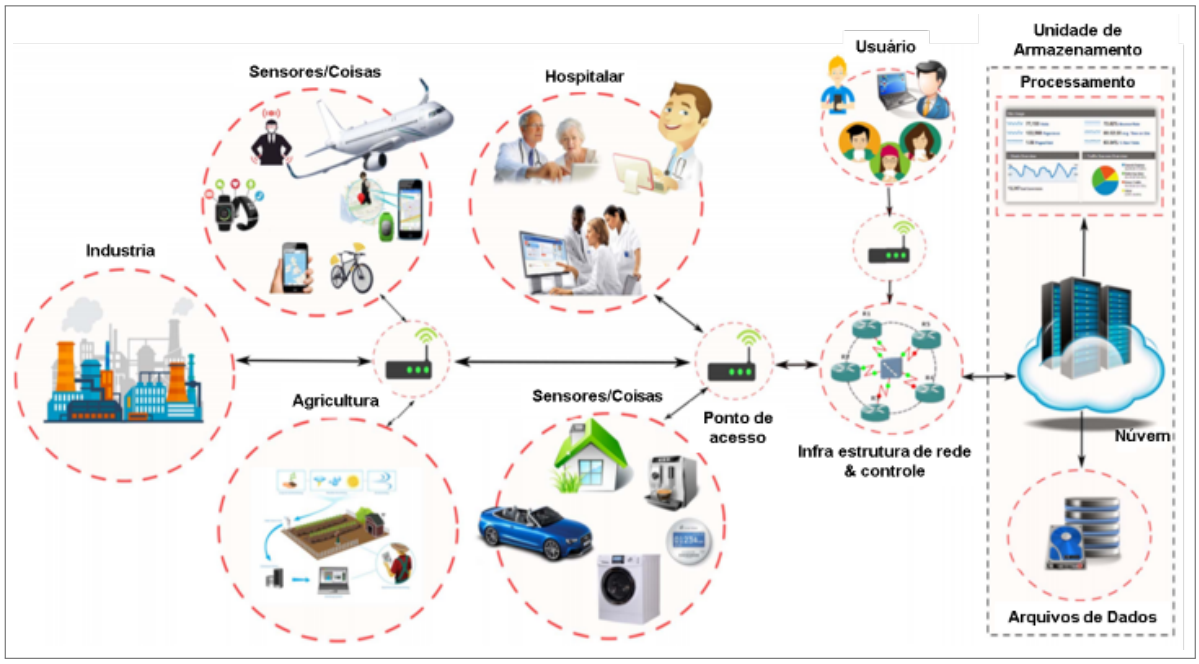
2.1 *INTERNET DAS COISAS* (IOT)

O conceito de IoT foi inicialmente ligado à idéia de usar identificação por radiofrequência nas cadeias de suprimentos (CARRION; QUARESMA, 2019). Logo depois, isso tornou-se popular e conhecido como uma nova Tecnologias da Informação e Comunicação (TICs), onde a internet está conectada ao mundo físico via redes de sensores sem fio (do inglês *Wireless sensor network*) (Liu et al., 2017).

Atualmente IoT são descritos como *internetworkings* de dispositivos físicos que usam arquiteturas de comunicação padrão para fornecer serviços para usuários finais por meio de processamento, transferência e visualização de dados em tempo real. Esse sistema possibilita que vários dispositivos de diferentes tecnologias se comuniquem entre si de forma autônoma, sem precisar da intervenção humana, definindo as tarefas e as funções a serem desempenhadas para o correto funcionamento de outros sistemas (Ghazanfar et al., 2020).

Durante a última década, IoT atraíram a atenção intensiva devido a aplicações em observação industrial, biomédica, agricultura, cidades inteligentes, monitoramento ambiental e outros campos (Farhan et al., 2017). Hoje, IoTs compõem uma ampla variedade de dispositivos da nossa vida cotidiana, como smartphones, eletrodomésticos, máquinas industriais, robôs, drones, sistemas de geração de energia e muitos outros (Figura 2).

Figura 2 – Arquitetura da Internet das Coisas (IoT)



Fonte:(Farhan et al., 2017)

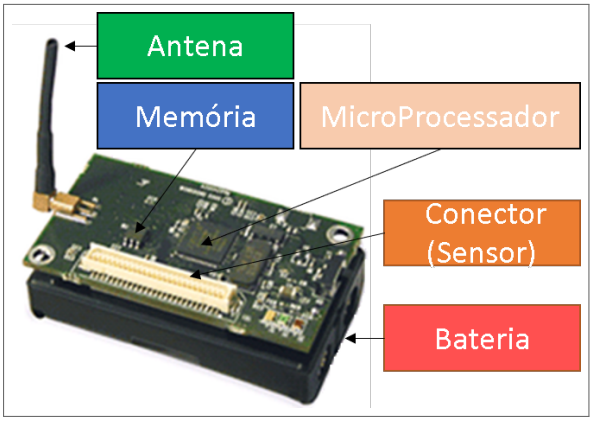
Num futuro próximo, espera-se que bilhões de humanos sejam conectados através de trilhões de dispositivos menores regulados sem interação humana direta e usando apenas o IoT (Internet das Coisas) (Khan et al., 2017). De acordo com o relatório da Ericsson Mobility Report (ERICSSON, 2023), a conectividade IoT apenas de dispositivos móveis celulares deve crescer cerca de 27% ao ano trazendo cerca de 4,1 bilhões de conexões IoT em 2024 (Chung; Jeong, 2020). Do ponto de vista conceitual esse crescimento exponencial está ligado à grande quantidade de dispositivos que são cotidianamente inseridos na rede, esses elementos IoT podem ser resumidos na seguinte expressão:

$$IoT = Pessoas + Internet + Coisas\,(sensores, controladores, dispositivos, unidades\ de\ armazenamentos). \tag{2.1}$$

2.2 REDES DE SENSORES SEM FIO

Essas WSNs são redes formadas por uma ampla gama de dispositivos sensores sem fio inteligentes, pequeno, autônomos e autoconfiguráveis, compostos por um transmissor-receptor (transceptor) de rádio frequência, microcontrolador, memória, fonte de energia (bateria), e um sensor como exposto na Figura 3.

Figura 3 – Componentes de um dispositivo Sensor



Fonte: O autor

Funcionalidades como fácil implantação, confiabilidade e mobilidade faz com que esses dispositivos sejam utilizados em larga escala por diferentes áreas de aplicação, monitorando e medindo condições físicas e ambientais (MEDINA, 2017).

Grande parte das WSN são compostas por numerosos dispositivos sensores minúsculos que faz com que esses sistemas apresentem limitações como:

- **Interoperabilidade**: A rede IoT combina uma infinidade de dispositivos, objetos e sistemas, cada um desses operando sob suas próprias arquiteturas, protocolos e formatação de dados, criando assim, um desafio na formação de uma comunicação homogênea transparente entre esses dispositivos com outros sistemas (semelhante ou não). Esses problemas requisitam que protocolos operem de forma adaptável para permitir conectividade dinâmica e suporte à natureza descentralizada de trilhões de objetos conectados pela Internet (Derhamy; Eliasson; Delsing, 2017).

- **Escalabilidade**: Devido à enorme quantidade de dados gerados nos sistemas de IoT, a escalabilidade, capacidade da rede crescer sem perda de desempenho, se torna uma preocupação significativa, devido à formação de gargalo ocasionado por uma distribuição irregular da carga de dados por múltiplos nós existentes na rede. Uma abordagem descentralizada de troca de dados seria a solução mais eficiente para enfrentar o crescimento iminente das redes IoTs, pois otimizaria o tempo de resposta e rendimento, minimizando assim os saltos necessários para a transmissão das informações. No entanto, essa troca de dados entre serviços de IoT fracamente acopladas é um desafio, especialmente em ambientes com recursos limitados, onde os sistemas tem uma conexão de rede ruim, pouco espaço em disco, baixo poder de processamento e pouca bateria (Arellanes; Lau, 2019).

- **Energia**: redes IoTs devem ser capazes de lidar bem com o consumo de energia, pois grande parte dos dispositivos que compõem esses sistemas são alimentados por

baterias. Isso significa que não é razoável o desperdício de energia com a transmissão de dados desnecessários, sobrecargas geradas por protocolos ou execução irregular do rádio durante o tempo (ouvindo ou transmitindo). Os protocolos destinados a essas redes devem permitir uma comunicação autossustentável eficiente através do entendimento dos impactos dos algoritmos executados no recurso energético de cada dispositivo especificamente (Barbosa et al., 2019).

- **Segurança**: IoTs não são usados apenas para aplicativos domésticos inteligentes, mas também estão profundamente envolvidos em aplicações comerciais e confidenciais como banco, saúde e defesa. Estar conectado à internet através do canal sem fio aumenta a ameaça de espionagem, requisitando desses dispositivos a segurança de sistemas robustos. Essa demanda tem sido um grande desafio devido limitações de hardware dos sistemas IoTs pois técnicas criptográficas convencionais usadas em redes complexas não são adequadas, sendo necessário que os protocolos destinados a esse tipo de rede utilizem técnicas simplificadas que não degradam o desempenho dos dispositivos, não aumente a taxa de erros de bits e garanta o sigilo dos dados de forma correta (Pechetti; Jindal; Bose, 2017).

- **Tolerância a erros**: O crescimento explosivo dos dispositivos conectados a redes sem fios exigem maior confiabilidade e desempenho, tornando tolerância a erros uma das questões mais importantes em WSN e aplicativos IoTs. Isso se dá porque ambas as tecnologias envolvem grande número de nós sensores executando tarefas específicas por uma grande área geográfica. Protocolos IoTs devem ser capazes de lidar com erros ocasionados muitas vezes por nós sensores bloqueados, falhas devido interferências no ambiente, dano físico e falta de bateria durante uma transmissão. Essas falhas dos nós não deve afetar as tarefas gerais dos sensores, a rede deve ser capaz de tratar esses problemas e relatar quando um nó sensor não pode receber mensagens do vizinho por um período específico, excluindo os nós mortos do caminho de roteamento ainda que morram dois ou mais nós sensores na mesma área (Farhan et al., 2017).

- **Softwares de baixo custo**: Manipulação de grandes volumes de dados, velocidade de transmissão e confiabilidade são problemas comuns em todas redes de transmissões de dados. Em IoT a simples tarefa de analisar, processar ou armazenar dados se torna algo difícil, se pensado através das tecnologias tradicionais. Assim, softwares de novos protocolos destinados a WSN devem entender esses problemas e implementar algoritmos capazes de manipular essa vasta quantidade de dados em sistemas que operam em ambientes instáveis com a mínima intervenção humana, enquanto lida com os demais problemas intrínsecos dos dispositivos (Aljarrah; Yassein; Aljawarneh, 2016).

- **Interferência humana**: Muitas aplicações inteligentes são usadas por seres humanos havendo interação entre pessoas, sensores, controladores, dispositivos e unidades de armazenamentos (Eq. 2.1), oferecidos em uma ampla gama de aplicações como assistência biomédica, agricultura, casas e cidades inteligentes, transporte entre outros (Santos; Volante, 2018). Poucos trabalhos consideram a interferência humana em sistemas IoT, no entanto, quando situações anormais acontecem há, em sua maioria, uma interferência humana de forma negativa, quando este é o ocasionador do erro, ou de forma positiva, com reparos realizados ao sistema. Os novos protocolos IoT precisam ser capazes de gerir essas questões, principalmente em sistemas de iteração homem-máquina em loop.

Grande parte desses problemas começaram a atrair a atenção de acadêmicos, pesquisadores e empresas, que propõe novos protocolos que resolvam simultaneamente parte desses desafios e preocupações em redes IoTs.

2.3 PROTOCOLOS IOTS/WSN

Protocolos, em redes de computadores, podem ser definidos como convenções que garantem a comunicação entre diferentes dispositivos em redes de transferência de dados (Forouzan, 2009).

Em redes IoTs essas convenções se tornam mais complexas, pois diferentes tipos de dispositivos são aplicados à diferentes áreas, levando ao desenvolvimento de vários protocolos direcionados à atender especificação de cada um desses sistemas (ROTTA; CHARÃO; DANTAS, 2017).

Para viabilizar uma possível comunicação entre esses dispositivos, a maioria dos protocolos seguem uma arquitetura única consensualmente dividida em três camadas a camada de percepção, composta pela camada física e a camada de enlace, a camada de rede e a camada de aplicação.(Srinidhi; Kumar; Banu, 2017)

1. **Camada física**: Esta camada é responsável por transportar informações entre duas máquinas, transformando dados em sinais físicos (elétricos, eletromagnéticos ou ópticos) adequados à propagação no meio físico (canal). Para tanto os hardwares da camada física convertem esses dados em bits na forma elétrica, óptica ou eletromagnética, pois essa camada é a única a possuir acesso físico ao meio de transmissão da rede (PANAHI; BAYILMIŞ, 2023). Em WSN essa camada também é responsável pela obtenção de dados, estáticos ou dinâmicos, de fontes materiais. Essa coleta de informações é obtida de dispositivos físicos ou sensores que aferem parâmetros como temperatura, umidade, GPS, vibração entre outros.(Tukur; Ali, 2019)

2. **Camada de enlace e controle de acesso ao meio (MAC)**: Essa camada é responsável por gerenciar o acesso aos canal de comunicação de um dispositivo, criando endereços físicos que serão usados na identificação deste na rede (Alves; da Silva Filho, 2020). Em redes de sensores a camada de enlace e a camada física juntas compõem a camada de percepção, responsável pelo controle de dispositivos, coleta de dados e pelo transporte seguro das informações direcionando-as a camada de rede, para isso os dispositivos podem usar diferentes padrões ou protocolos como IEEE 802.15.4 (ZigBee), RFID, Bluetooth, WiFi para WSN, etc. (Tukur; Ali, 2019).

3. **Camada de rede**: É uma camada de transição responsável por tratar os dados coletados pela camada MAC e endereça-los à camada de aplicação. Essa camada usa diferentes técnicas de roteamento que pode variar dependendo do tipo de tecnologia usada a exemplo de smart phone, Bluetooth, ZigBee etc. A camada de rede compreende um dispositivo de gateway que recebe as informações coletadas e transferida entre os dispositivos sensores e envia esses dados para uma Ethernet ou Internet que posteriormente armazenam essas informações obtidas em base de dados do sistema destino ou na nuvem (Owada et al., 2019).

4. **Camada de aplicação**: A camada de aplicação é responsável por fornecer vários tipos de serviços específicos requeridos pelas camadas subjacentes. Essa camada também fornece a estrutura para o desenvolvimento de vários aplicativos tornando os dispositivos IoTs versáteis para uso em diferentes áreas como casas, agricultura, vigilância, médica e outra (Ejaz et al., 2020).

Além das arquiteturas, a topologia da rede desempenha um papel fundamental na comunicação sem fio, principalmente em dispositivos IoTs que podem ter seus protocolos comprometidos sem um planejamento de sua disposição espacial (Dey; Stuart; Tolentino, 2018).

Portanto, para estabelecer a conectividade de rede, vários protocolos IoT como ZigBee, Aloha, Bluetooth e Wi-Fi definem seu sistema topológico classificando-os em três tipos principais: *point-to-point*, *star* e *mesh*.

- **Topologia *Point-to-Point* (P2P)**: Nessa topologia as conexões são estabelecidas de forma direta entre dois nós da rede, não oferecendo suporte a inserções de múltiplos nós á comunicação. O protocolo Bluetooth é o que mais se apropria dessa topologia principalmente para conexão entre dispositivos móveis e seus aparelhos (Tracey; Sreenan, 2019).

- **Topologia *Star***: Essa topologia de rede consiste em um nó central denominado gateway gerenciando a camada de rede dos nós sensores restantes. O gateway atua como o ponto de interseção comum para os nós restantes, gerenciando os nós periféricos que enviam ou recebem os dados na rede por seu intermédio. Um exemplo

de protocolo que usa esse tipo de topologia é o *hub* Wi-Fi da rede doméstica à qual esse *hub* é o único dispositivo que se conecta a redes externas (dos Santos et al., 2019).

- **Topologia *Mesh***: denominada de topologia em malha ela compreende três tipos de nós, um nó emissor, um receptor e um nó de gateway, com função equivalente ao gateway da topologia *Star*. Nesta topologia um nó sensor (origem) encaminha as informações geradas a um nó central gatway que, por sua vez, roteia, endereça e envia os dados a outro gateway ou diretamente ao nó receptor (destino).

  Para que essa topologia seja implantada os nós sensores devem ser inseridos dentro do raio de alcance da antena de um nó gateway, viabilizando a comunicação com o destino. A topologia de rede em malha é usada principalmente em redes de sensores que visam cobrir ampla áreas e não é possível uma conexão direta desses dispositivos com o sistema de base de dados ou nuvem, como por exemplo, em sistemas de gerenciamento de frota, rastreamento de ativos, automação industrial, *smart grid* entre outros (dos Santos et al., 2019).

Para uma melhor visualização das principais características de cada uma dessas topologias usadas em redes IoTs a TABELA 1 mostra uma comparação detalhada de cada um de seus atributos específicos.

Tabela 1 – Comparação das diferentes topologias usadas em IoT

Atributos	***P2P***	***Star***	***Mesh***
Escalabilidade	Dois nós	10-15 dispositivos por ponto de acesso	Centenas por rede
Complexidade	Baixa	Média	Alta
Taxa de transmissão	1-3 Mbps	100 Mbps	250 Kbps (em média)
Interoperabilidade	Alta	Alta	Alto (entre semelhantes) Médio (entre diferentes)
Alcance médio	10 metros	32 metros (ambientes fechados) 95 metros (ao ar livre)	>300 metros
Latência	Baixa	Média	Alto devido aos vários saltos entre os nós origem e destino
Protocolo Exemplo	Bluetooth ALOHA	Wi-Fi	ZigBee

Fonte: adaptado de (Srinidhi; Kumar; Banu, 2017)

Diante desses cenários complexos, dinâmicos e restritivos novos protocolos de comunicação tentam gerenciar a conexão entre os dispositivos IoTs, criando conjuntos específicos de regras e regulamentos que priorizem uma comunicação eficiente entre dispositivos de mesma arquitetura e topologia, ou que garantam de forma abrangente uma interoperabilidade entre dispositivos distintos. (Datta; Sharma, 2017)

2.4 EXEMPLOS DE PROTOCOLOS IOTS/WSN

Protocolos destinados a redes IoTs visam resolver os problemas apresentados nas seções anteriores, subdividindo-os e atacando o problema na camada em que ela se apresenta. Nesta seção, será discutido os principais protocolos encontrados no estado da arte utilizado para resolver problemas nas camadas de percepção, rede e aplicação (Srinidhi; Kumar; Banu, 2017).

2.4.1 Protocolos para Camada de Percepção

A grande parte dos dispositivos IoTs possui algum tipo de limitação de hardware, pois são compostos por microcontroladores que contam com Memória de acesso randômico (RAM) e memória somente leitura (ROM) de baixo custo, destinados à realização de um pequeno número de instruções simples (Hamad et al., 2020).

Essas limitações fazem com que os protocolos responsáveis pela obtenção de dados criem padrões que garantam a segurança das informações que serão passadas (Pechetti; Jindal; Bose, 2017).

A segurança do IoT via seu Sistema Operacional (SO) é muito importante pois esse deve oferecer suporte a serviços de segurança e privacidade. No entanto, a maioria destes sistemas são incapazes de atender às necessidades e requisitos de segurança, tornando esse gerenciamento um grande desafio (Datta; Sharma, 2017).

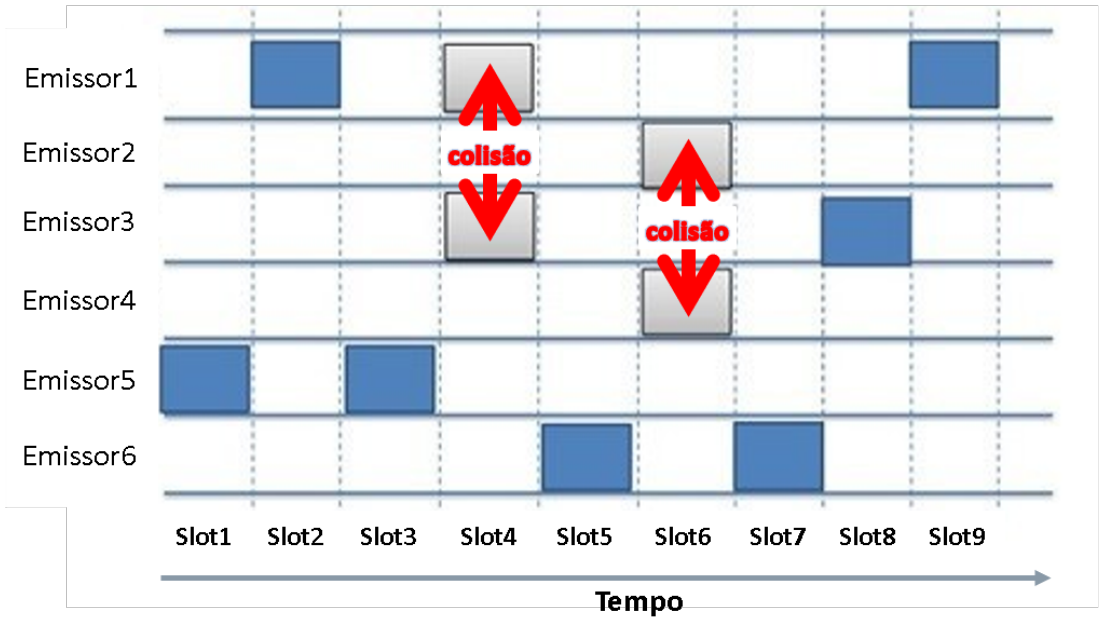
Nesta seção será discutido alguns protocolos específicos das camadas MAC que tentam gerir recursos resolvendo partes dos problemas apresentados.

- ***Slotted* ALOHA**: Redes baseadas no protocolo ALOHA garatem que qualquer terminal que tenha um pacote pronto para transmitir simplesmente transmita independentemente de outros terminais que já estejam transmitindo (CHOUDHURY; RAPPAPORT, 1983). Quando uma transmissão é mal sucedida será detectada pelo terminal se, após um certo tempo, nenhuma confirmação for recebida do destinatário, o dispositivo emissor determinará de forma aleatória um certo tempo de atraso e, após esse tempo, retransmitirá o pacote.

  Para melhoria de desempenho da rede ALOHA foi desenvolvida a rede chamada ALOHA com intervalos ou *Slotted* ALOHA adicionando as regras: os pacotes devem ter um comprimento fixo e as transmissões só podem começar em pontos determinados no eixo do tempo. Essa subdivisão no tempo de transmissão do pacote é denominado de lote de tempo (ou *slot*) (Roberts, 1975) e permite que vários emissores compartilhem o mesmo canal de comunicação, desde que seja respeitada essa janela de tempo para envio, isso reduz as chanses que mais de um emissor acesse um canal de transmissão ao mesmo tempo (NAMISLO, 1984). A Figura 4 mostra

um cenário de transmissão no *slotted* Aloha com 6 emissores, exibindo duas colisões entre pacotes enviados no mesmo *slot* de tempo (da Silva; de Moraes, 2020).

Figura 4 – Rede com 6 emissores utilizando protocolo de comunicação *Slotted* Aloha subdividida em 9 slots de tempo, com colisões nos *slots* 4 e 6.



Fonte: O autor

Atualmente o *slotted* Aloha vem sendo utilizado principalmente em sistemas de grande escala que lidam com baixas taxas de transferências oriundas de conexões do tipo P2P entre máquinas denominadas *Machine-to-Machine* (M2M) a exemplo das redes WSN e IoT que requisitam protocolos energeticamente eficiente de baixa complexidade para dispositivos com bateria e memória limitadas (BALEVI; RABEE; GITLIN, 2018)

- **CSMA/CA (IEEE 802.11)**: O CSMA/CA do padrão IEEE 802.11 é um protocolo de camada MAC, com acesso múltiplo e verificação de portadora que conta com um sistema de prevenção de colisão. Esse protocolo aplica um sistema de verificação prévia do canal, que analisa a disponibilidade de transmissão por um período de tempo distribuído (do inglês, *Distributed InterFrame Space* (DIFS)). Uma vez identificada a colisão pelo emissor, expressa pela não confirmação do recebimento do pacote, este passará por um período de espera de tempo aleatório denominado de *backoff*, para que possa tentar transmitir novamente (Niu; Liu, 2020).

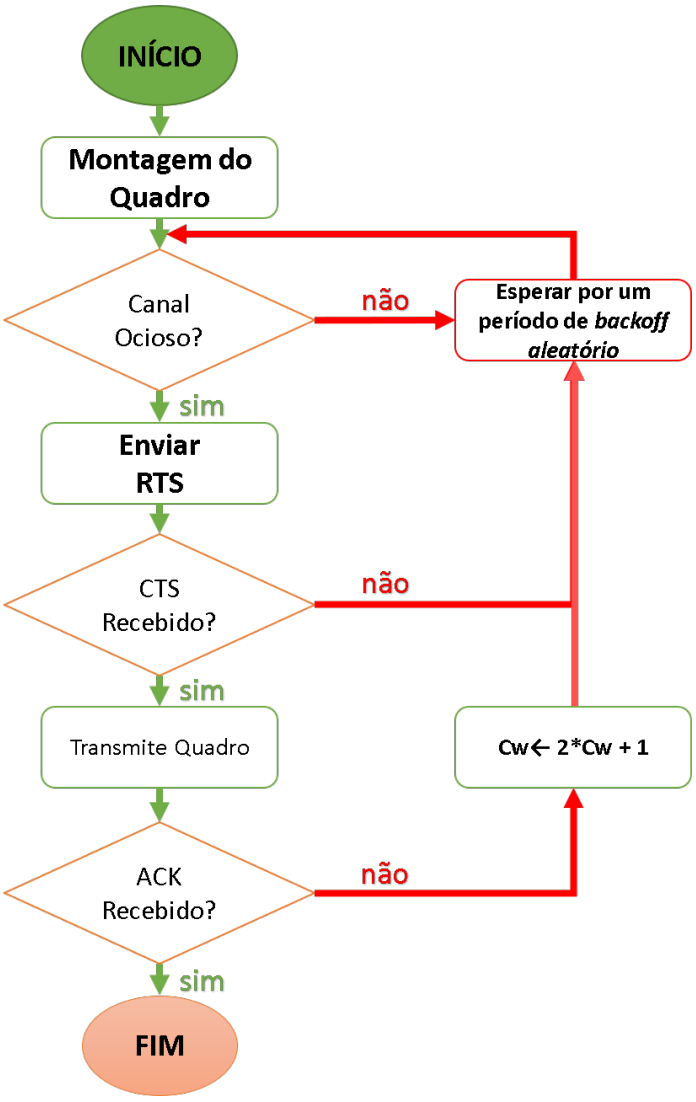
  Para que isso seja possível, o CSMA/CA atua desde o estabelecimento da comunicação utilizando a técnica de *handshake* bidirecional que verifica, através de bits de controle *Acknowledgment* (ACK), a disponibilidade do receptor em estabelecer uma comunicação. Uma vez verificada a disponibilidade da comunicação e iniciada

a transmissão, o dispositivo receptor enviará para a fonte uma confirmação imediata do recebimento de cada pacote de dados via ACK (da Silva, 2019).

O funcionamento detalhado do CSMA/CA é apresentado através do fluxograma na figura 5). Nele é possível analisar os passos para envio de pacote executado em X passos. No primeiro passo é montado o quadro de transmissão, o segundo verifica a disponibilidade do canal verificando as transmissões em cursos, no passo três o emissor envia um bit de requisição para envio denominado *Request to Send* (RTS).

Quando o receptor detecta o quadro RTS ela responde através do envio do quadro CTS (*Clear to Send*) indicando a disponibilidade de se iniciar uma transmissão de dados. Caso o emissor não receba esse CTS, ele entrará em *backoff* aguardando esse tempo para uma nova tentativa de estabelecer comunicação com a estação destino (da Silva, 2019). Após cada transmissão malsucedida, o valor da janela de contenção (do inglês, *Contention Window* (CW)) é dobrado até o valor máximo da janela de contenção $CWmax = 2^m CWmin$, onde $CWmin$ é um tempo predefinido e m o numero de tentativas de retransmissões (Niu; Liu, 2020).

Figura 5 – Fluxograma simplificado do protocolo CSMA/CA



Fonte: baseado em (da Silva, 2019)

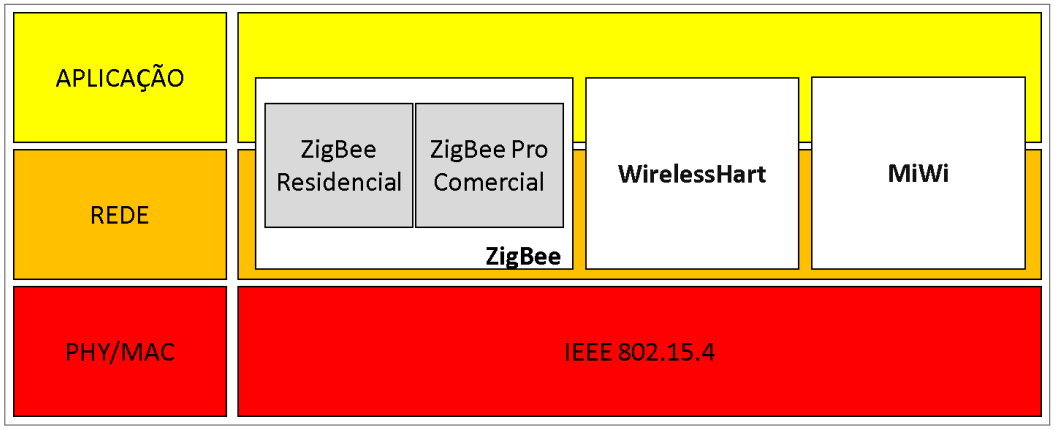
- **IEEE 802.15.4**: Este padrão e seus protocolos são amplamente utilizados na camada física em sistemas IoT e WSN, em redes de área pessoal sem fio de baixa potência (LP-WPAN), principalmente devido ao seu baixo custo e código aberto (Choudhury et al., 2020). A Subcamada MAC do IEEE 802.15.4 contém várias características do protocolo IEEE 802.11 utilizando o protocolo de acesso ao meio CSMA/CA adaptado para as limitações energéticas da LR-WPANs (*Low Rate Wireless Personal Area Networks*), eliminando assim mensagens de Request to Send (RTS) / Clear to Send (CTS), usadas no IEEE 802.11 para reduzir da probabilidade de colisões, que consomem muita energia (KO; LIU; FANG, 2006).

  O protocolo MAC do IEEE 802.15.4 conta ainda com um modo de operação que realiza multiplexação do tempo de transmissão com um método CSMA/CA organizando a rede através de *beacons* gerados periodicamente para identificar os dis-

positivos existentes na rede e sincroniza-los (SANTOS, 2013). Esse Compartimentos de Tempo Garantido (GTS) realiza reserva de horários de transmissão entre os dispositivos da rede, viabilizando o provimento de QoS para aplicações compatíveis, alta confiabilidade e segurança da rede a um baixo custo computacional (Choudhury et al., 2020), além de garantir uma transmissão mais rápida pois os dispositivos não precisam aguardar o início de um compartimento para iniciar suas transmissões e nem concluir suas transmissões antes do término de um compartimento garantindo um melhor aproveitamento do canal de comunicação (KO; LIU; FANG, 2006).

O IEEE 802.15.4 é usado basicamente para WSN de baixa complexidade e de curto alcance como ZigBee, Wireless HART e MiWi, com cada um deles possuindo suas próprias camadas superiores pré-definidas conforme demonstra a Figura 6. Portanto outros protocolos são utilizados em camadas superiores de acesso dessas tecnologias a Internet, tornando esses dispositivos habilitados para diferentes dispositivos IoT (Li et al., 2011).

Figura 6 – Estrutura em camadas do IEEE 802.15.4



Fonte: O autor

- **WIFI para WSN**: Tecnologias baseadas no IEEE 802.15.4 também tem limitações, como a baixa taxa de transmissão de dados, distância curta de transmissão e a inabilidade lidar com grandes atenuações de sinais (Samijayani et al., 2020).

  Com a evolução da tecnologia sem fio foram desenvolvidos muitos tipos de chips de sensores compatíveis com a tecnologia WiFi para aplicações de baixa potência originando um novo tipo de Rede de Sensor Sem Fio denominada *WiFi-based WSN*. Esta tecnologia consiste em nós de baixo consumo energético, autogerenciados, distribuídos em áreas de detecção, combinando os recursos de uma rede WiFi tradicional, centrada na rede, e WSN, centrada em captação de dados (Faid; Sadik; Sabir, 2020).

  Em comparação com a tecnologias baseadas em IEEE 802.15.4, a rede de sensores sem fio WiFi tem vantagens significativas como Alta largura de banda, alcançando até 300 Mbps, menor susceptibilidade a barreira nas transmissões (*Non-line-of-sight*

Transmission), maior área de cobertura alcançando um raio de transmissão com até 30 m, maior custo-beneficio, embora haja uma alta interoperabilidade entre os dispositivos com protocolos IEEE 802.15.4, a WSN baseados em WIFI se torna ainda mais económica pois utiliza toda estrutura de hardware pré-existente da rede WiFi tradicional (Samijayani et al., 2020). O *WiFi-based WSN* também apresenta alta escalabilidade, maior confiabilidade de transmissão, maior largura de banda em transmissões de múltiplos saltos e baixo consumo energético dos nós de sensores, garantindo que uma bateria AA seja suficiente para 5 a 10 anos (Li et al., 2011).

- ***Bluetooth Low Energy* (BLE)**: Conhecido também como Bluetooth inteligente, esse protocolo é utilizado para suporte a diversas operações entre dispositivos *mobile*, pois funciona em diferentes sistemas operacionais como Android, Blackberry e IOS e também em sistemas computacionais desktop como Macintosh, Windows e Linux (Tracey; Sreenan, 2019).

  O BLE possui recursos de segurança que toram muito robusto a troca de informações entre IoTs, consumindo uma energia dez vezes menos que o Bluetooth clássico. Isso é possível graças a sua arquitetura *master/slave*, onde o *master* informa ao *slave* sobre seu ciclo de despertar mantendo o dispositivo conectado a esse controlador acordado apenas durante o envio das informações, fazendo com que o *slave* volte ao modo sono assim que a conexão for terminada, economizando energia dos dois dispositivos envolvidos na transmissão. Esse controle rígido do acesso ao dispositivo e da energia gasta na comunicação faz o BLE muito adequado para sistemas IoTs (Baert et al., 2019).

- ***Long Term Evolution Advanced* (LTE-A)**: O protocolo LTE-A, é uma evolução do LTE, projetado para uma comunicação direta entre maquinas (M2M) realizada entre dispositivos IoTs. O LTE-A possui maior taxa de escalabilidade e duas vezes a velocidade do LTE existente. O LTE-A usa várias antenas do tipo 2xMIMO no lado do remetente e do receptor, fornecendo aos dispositivos conectados a essa rede uma velocidade mais rápida com um fluxo de dados transmitidos com mais eficiência. O LTE-A usa o sistema de Acesso Múltiplo por Divisão de Frequências Ortogonais (OFDMA) para seu MAC, dividindo a frequência utilizada em várias bandas que podem ser acessadas individualmente (Mukhopadhyay; Das, 2020).

- **Rede de baixa potência para área amplas (LoRaWAN, do inglês *Long Range WAN*)**: Este protocolo é principalmente destinado a dispositivos IoTs que operam com bateria a exemplo das Redes de Sensores sem fio WSN, dando suporte à mobilidade, comunicação bidirecional, segurança, localização e baixo consumo energético. Esse protocolo ainda dispões de suporte a escalabilidade a grande número de dispositivos conectados simultaneamente (Sisinni et al., 2020)

2.4.2 Protocolos para Camada de Rede

Os protocolos destinados à camada de rede tentam resolver problemas como deterioração do desempenho da rede durante sua execução, falhas na segurança e gerenciamento de endereçamento das informações. Para isso os protocolos se subdividem em duas subcamadas denominadas camada de roteamento e camada de encapsulamento.

Os protocolos de roteamento são responsáveis por gerenciar a transferência de pacotes, determinando o caminho que os dados percorrerão do transmissor ao receptor, enquanto a camada de encapsulamento é responsável pela formatação dos pacotes de dados enviados.

Esta seção exibe alguns protocolos destinados as subcamadas de roteamento e encapsulamento de dados em dispositivos IoTs.

2.4.2.1 Protocolos de Roteamento

- **Protocolo de roteamento para redes de baixa potência e com perdas (RPL)**: Os protocolos de transporte e infraestrutura de IoT mais comumente usados na camada de roteamento é o IPv6. O RPL é um protocolo baseado em IPv6, e suporta protocolos de ligação a diferentes camadas, definindo requisitos mínimos para uma padronização do roteamento em conexões ponto-a-ponto (P2P), ponto-a-multiponto (P2MP) ou multiponto-a-ponto (MP2P).

  O RPL utiliza um padrão de roteamento baseado no sistema de Grafo Direcionado Acíclico Orientado ao Destino (DODAG), onde um nó intermediário (folha) possui as informações sobre seu nó raiz e mantém uma rota salva dele até o nó raiz, fazendo com que o roteamento de um possível emissor tenha um desempenho mais rápido na conexão entre o nó origem e o nó destino (Chuang et al., 2017).

- **Protocolo de roteamento com reconhecimento de canal (CARP)**: O CARP é um protocolo de roteamento IoT destinado à comunicação de sensores subaquática. Para que isso seja possível esse protocolo é executado em dois tempos, o primeiro realiza a inicialização da rede e o segundo encaminha os dados obtidos pelos sensores. No processo de inicialização, o nó coletor transmite uma mensagens de *HELLO* para todos os nós dentro da rota, na segunda etapa os nós sensores enviam por pacotes seus dados para o nó coletor realizando vários saltos dentro dessa rota.

  O principal problema do protocolo CARP é que este não oferece suporte à reutilização de dados, sendo necessária uma nova avaliação da rota sempre que alterações forem observadas pelo nó coletor, gerando assim mensagens de controle desnecessárias. Protocolos como o E-CARP superaram esses problemas aumentando a eficiência energética dos sensores sem fio subaquático. E-CARP reduz o consumo de energia evitando que mensagens de controle durante o processo de seleção de nó da

rota de transmissão, enviando os dados apenas para o nó coletor quando a mudança é observada (Srinidhi; Kumar; Banu, 2017).

2.4.2.2 Protocolos de Encapsulamento

- **Redes de área pessoal sem fio IPv6 sobre baixa potência (6LoWPAN)**: Esse protocolo tenta garantir que as regras e especificações definidas no protocolo IPv6 se mantenham mesmo com as restrições encontradas nos dispositivos que utiliza redes do tipo IEEE 802.15.4. Para isso o 6LoWPAN encapsula os pacotes de 1280 bytes do IPv6 em um formato 127 bytes através de técnicas de fragmentação e compressão dos cabeçalhos usados, removendo até 40 bytes de informações não essenciais do cabeçalho IPV6 compactando-os em um cabeçalho de apenas 2 bytes. O principal objetivo do 6LoWPAN é corrigir as incompatibilidades entre as tecnologias mais antigas e as mais recentes (Beniwal; Nikolova; Iliev, 2019).
- **IPv6 para BLE**: Esta tecnologia é destinada a redes Bluetooth 4.0. Esse protocolo aplica um sistema de compactação do cabeçalho IPv6 dentro do protocolo BLE. Porém, diferente do 6LoWPAN, essa compactação é realizada através de um protocolo de adaptação de controle de link lógico (L2CAP), remontando a cargas úteis dos dados em pacotes de 27 bytes. Essa compactação é simples pois o BLE possui um nó central que funciona como roteador entre nós de borda sem suporte a uma transmissão multi-saltos como visto nas redes IEEE 802.15.4 (Baert et al., 2019).

2.4.3 Protocolos da Camada de Aplicação

A camada de aplicação é a responsável por prover interface e suporte a serviços como acesso à internet, transferência de arquivos, serviço de identificação de dispositivos, entre outros. Para isso essa camada gerencia as camadas inferiores dos dispositivos IoTs, construindo, via aplicações, um modelo de objeto que garanta aos dispositivos novos requisitos de QoS (Hejazi et al., 2018).

Esse acesso a camadas inferiores da rede faz com que a camada de aplicação seja recorrentemente usada para implementação de protocolos que garantam controle de banda, gerenciamento de múltiplas conexões, controle na participação de usuários a rede, segurança de dados e eficiência energéticas dos dispositivos (Hamad et al., 2020).

Entre os principais protocolos destinado a camada de aplicações podem ser destacados:

- **Protocolo de Aplicação Restrita (CoAP)**: Esse protocolo foi inspirado no modelo de comunicação cliente-servidor do protocolo HTTP (do inglês, *Hypertext Transfer Protocol*), sendo modelado para atender as limitações energéticas e de memória dos dispositivos IoTs. Por esse motivo o CoAP é simples e leve, gerando o

mínimo de tráfego possível a rede. Para tanto, o CoAP não suporta comunicação TCP / IP utilizando o modelo UDP (do inglês, *User Datagram Protoco*) sobre IP como base para garantir um conexões com alta confiabilidade enquanto economiza recursos energia.

O CoAP garante recursos REST (do inglês, *Representational State Transfer*), implementando funcionalidades como observar, executar e descobrir recursos, ler e escrever, serviços especificos para redes de sensores não disponíveis no protocolo HTTP padrão (Hejazi et al., 2018).

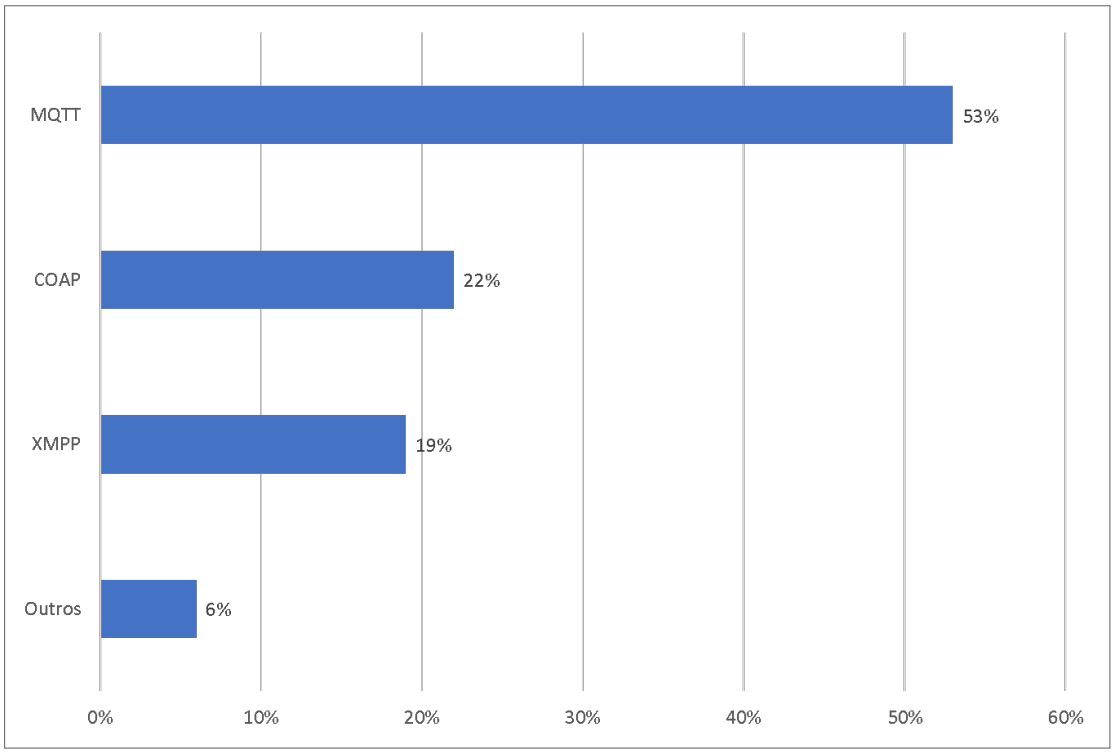
Além disso o CoAP proporciona às plataformas suporte a implementação de outros protocolos de comunicação que se adequem a outras necessidades do IoT como por exemplo protocolos baseados em pacotes (Yassein; Shatnawi; Al-zoubi, 2016).

- **Transporte de Filas de Mensagem de Telemetria (MQTT)**: é um protocolo de mensagens implementado a partir do TCP / IP, para publicação e assinatura de mensagens leves de maior latência, destinado a redes com altas falhas de transmissão composto por dispositivos com grande limitação de largura de banda. Para viabilizar suas operações esse protocolo é composto por três elementos, *Subscriber*(Subscrito), inscrito no tópico e irá ter o papel de receptor, *Publisher* (Publicador), responsável por ser o emissor e enviar os dados para um determinado tópico, e *"Broker"*, o intermediário que faz uma ponte de comunicação entre o *Publisher* e o *Subscriber*, se tornando responsável por fazer o recebimento, enfileiramento e envio das mensagens. Em um exemplo de uma rede de sensores com três clientes, caso dois deles se inscrevam em um mesmo tópico, temperatura, um terceiro cliente que publique seus dados sobre temperatura será ouvido por todos os demais dispositivos da rede cadastrados sob esse tópico. Em relação à segurança, o MQTT suporta os protocolos SSL (do inglês, *Sockets Layer*) e o TLS (do inglês, *Transport Layer Security*) (Hejazi et al., 2018).

- **Protocolo extensível de mensagens e presença (XMPP)**: Esse protocolo foi inicialmente usado em aplicativos de bate-papo e troca de mensagens, porém hoje é usado por aplicações IoTs e em rede definida por software (SDN), dado à sua eficiência na internet e reutilização de dados devido ao XML. Para isso esse protocolo usa os métodos solicitação-resposta e publicar-assinar para serviços como consumo de *web-services*. A grande desvantagem deste protocolo é que ele não fornece suporte a QoS em rede, isso se dá, devido aos muitos cabeçalhos e tags existentes, tornando esse protocolo, muitas vezes, inadequado para aplicativos de IoT *low-energy* (Cho et al., 2019).

A Figura 7 apresenta o percentual dos protocolos da camada de aplicação utilizada pelos dispositivos IoT segundo o estudo realizado por (Skerrett, 2017).

Figura 7 – Gráfico com percentagem de uso dos protocolos da camada de aplicação encontrados em dispositivos IoT segundo o estudo realizado por (Skerrett, 2017)



baseado em (Skerrett, 2017)

A Tabela 2 apresenta um resumo dos protocolos da camada de aplicação apresentados nesta seção.

Tabela 2 – Comparação entre os principais Protocolos da Camada de Aplicação mais utilizados segundo pesquisa (Skerrett, 2017)

Protocolo	**Transporte**	**Mensagem**	***Memória Requerida***	***Eficiencia Escalar***	**QoS**	**Segurança**
MQTT	TCP	*Pedido/Resposta* *Publicar/Assinar*	10Ks RAM/Flash	Boa	Sim	TLS/SSL
CoAP	UDP	*Pedido/Resposta*	10Ks RAM/Flash	Exelente	Sim	DTSL
XMPP	TCP	*Publicar/Assinar* *Pedido/Resposta*	10Ks RAM/Flash	Regular	Não	TLS/SSL

Fonte: adaptado de (Srinidhi; Kumar; Banu, 2017)

2.5 SISTEMAS OPERACIONAIS PARA IOTS/WSN

A extrema limitação na memória, energia e segurança dos dispositivos IoTs faz com que aumente a importância dos sistemas operacionais (SO) no controle rigoroso destes recursos. Um sistema operacional adequado com um kernel, rede, gerenciamento das ca-

pacidades em tempo real e muito mais pode tornar a usabilidade desses dispositivos mais flexíveis (Javed et al., 2018).

Esta seção fornece uma comparação detalhada dos sistemas operacionais projetados para dispositivos de IoT com base em seus modelos de programação, capacidade de reprogramação, métodos de gerenciamento de energia. Esta seção também avaliará os SO segundo fatores importantes como eficiência energética, escalabilidade, resiliência a falhas e interoperabilidade. Alguns SOs principais dos IoTs são discutidos abaixo:

- **Contiki**: Trata-se de um SO desenvolvidos com foco em sistemas WSNs, porém hoje sofreu melhorias possuindo um estilo de arquitetura modular, com suporte a agendamento e preempção de *multi-threading*. Esse Sistema Operacional implementa o Rime como sua pilha de rede e visa a alta eficiência no gerenciamento de energia e memória (Dunkels; Gronvall; Voigt, 2004).

  O Contiki foi escrito na linguagem C com programação simultânea baseada em eventos denominado *protothreads* que garante uma programação paralela eficiente através do compartilhamento de multiplas tarefas de forma cooperativa. O Contiki não permite manipuladores de interrupção de eventos necessitando sempre que um evento seja concluído para inicialização de outro (Dunkels; Gronvall; Voigt, 2004).

  Uma das grandes vantagens deste SO é que o código principal e o código do programa implementado são mantidos separados na ROM, sendo divididos em duas partes, o núcleo composto por um kernel, drivers do dispositivo, aplicação, bibliotecas de idiomas e tabela de símbolos. E os programas implementáveis carregados em tempo de execução a partir da ROM ou da RAM (Javed et al., 2018).

  O Contiki também possui biblioteca para gerenciamento dinâmico de memória e vinculação dinâmica dos programas, implementando serviços que podem ser alterados em tempo de execução, alocando memórias que podem ser gerenciadas e compartilhadas entre os processos (Srinidhi; Kumar; Banu, 2017).

  Quanto ao consumo energético o Contiki não contém abstrações explícitas de gerenciamento de energia, porém possui recursos de controle do estado dos dispositivos fazendo com que ligue, desligue ou permaneçam em sono profundo. Isso permite que os programadores implementem mecanismos para gerenciamento de energia (Javed et al., 2018).

  Para um maior controle de consumo de energia o Contiki precisa de protocolos IoTs energeticamente eficientes, sendo possível ver esse SO associado a protocolos como o CoAP ou XMPP (Srinidhi; Kumar; Banu, 2017).

- **TinyOS**: O TinyOS é chamado de sistema operacional padrão para WSNs, junto com o Contiki compõem a maior parte do mercado de SO para IoTs *low memory*

(Skerrett, 2017). Devido a recursos como suporte a vários dispositivos e facilidade de programação, também pode ser usado em outros dispositivos (Perrone; Nicol, 2002).

O TinyOS suporta arquitetura monolítica e um modelo de programação orientado a eventos com técnicas de agendamento com suporte a vários algoritmos. Além disso, possui sua própria linguagem de programação o NesC que deriva da linguagem C. Através do NesC, esse SO conseguiu melhorar suas técnicas de eficiência e gerenciamento de energia e memória, agregando a essas técnicas mecanismos próprio de comunicação e troca de mensagem em rede, fornecendo assistência, simplicidade, redução da complexidade do código e erradicação de bugs.

O modelo de programação TinyOS herda alguns dos conceitos de programação baseados em componentes, suportando simultaneidade através do uso de componentes categorizados em abstração, eventos e tarefas. Sempre que um comando é originado, ele é apresentado em uma mensagem de solicitação. Os componentes dos dispositivos também são fornecidos com a interface.

O NesC é sensível a interrupções deixando a cargo do hardware em IoTs os métodos para lidar com elas (Amjad et al., 2016). Essa linguagem também não suporta gerenciamento de memória dinâmica, porém não se mostra um problema pois esse SO é executado em hardwares de quantidade fixa e limitada de memória (Javed et al., 2018).

Embora originalmente o TinyOS não tenha integrado nenhum suporte para *multithread*, o novo TinyOS oferece essa tecnologia através de threads controláveis que fornece um modelo de programação intuitivo e eficiente do kernel orientado a eventos (Srinidhi; Kumar; Banu, 2017).

O modelo de programação do SO tem um grande impacto na memória, por este motivo o TinyOS apresenta a abordagem *UnStacked C* que ajuda a melhorar a eficiência da memória guardando poucas informações nela (Amjad et al., 2016).

Basicamente, o que torna o UnStacked C poderoso é sua preempção preguiçosa. Por meio dessa abordagem, o TinyOS obtém uma melhor simultaneidade dos dados. Seu tradutor C não empilhado converte o código de vários segmentos sem a necessidade de empilhamentos de comandos, no entanto, o código do aplicativo não será modificado resultando em uma eficiência de 35% maior no uso de RAM (Javed et al., 2018).

Para resolver o problema de eficiência energética o TinyOS possui APIs para controlar e gerenciar o uso de energia, minimizando o ciclo de trabalho através de técnicas como a *fine grain*.

- **RIOT**: O RIOT é um sistema operacional de código aberto sob a licença GNU *lesser*. Em contraste com TinyOS e Contiki, o RIOT foi lançado em 2013 exclusi-

vamente direcionado às redes IoTs dando suporte ao 6LoWPAN e agendamento de tarefas em tempo real através da sua arquitetura de micro-kernel e seu modelo de suporte a rede (Baccelli et al., 2013).

Foi desenvolvido nas linguagens de programação C e C ++ possuindo assim um ambiente de programação simples em sua arquitetura. Além da linguagem C pura, o C ++ também é usado, tendo total suporte as coleções de compiladores GNU, embora não dê suporte ao unix GNU (Baccelli et al., 2013).

O kit de desenvolvimento de software (SDKs), para o desenvolvimento de aplicativos no RIOT utiliza a famosa coleção de compiladores GNU (GCC), depurador GNU (GDB) e Valgrind, dando suporte a programação de aplicativos em C e C++ (Javed et al., 2018).

Ao contrário do TinyOS e Contiki, o RIOT possui recursos de manipulação para interrupção de tarefas, com latência zero e comutação de contexto. O encadeamento de maior prioridade nunca será interrompido mais do que um ciclo fixo. O kernel do RIOT é executado de forma modesta e inclui, além do planejador e do sistema de segmentação, uma comunicação mutex (*mutual exclusion object*), técnica utilizada para evitar que dois processos ou threads tenham acesso simultaneamente a um mesmo recurso como drivers e sistemas de arquivos (Challouf; Kriaa; Saidane, 2019).

O RIOT suporta alocação de memória estática e dinâmica possuindo uma eficiência com ordem de grandeza $O(n^1)$ para a execução da tarefa, garantindo o controle no tempo de execução em sistemas estáticos (Javed et al., 2018).

Para controle energético o RIOT possui um agendador sem marcadores que funciona sem a necessidade de ocorrência de eventos periódicos, garantindo uma suspensão profunda mais eficiente aos dispositivos IoTs. Quando não há tarefa na fila, o sistema entra no modo de suspensão podendo ser alterado para o estado de inativo, onde a execução do próprio SO tem uma menor prioridade o que aumenta ainda mais a eficiência de energia (Challouf; Kriaa; Saidane, 2019). Essa estratégia favorecerá o cenário em que a interação do usuário é mínima e o próprio dispositivo decide alternar ou interromper as tarefas em execução (Baccelli et al., 2013).

- **Nano-RK**: O Nano-RK é um sistema operacional em tempo real (RTOS) totalmente preventivo centrado em recursos, focado exclusivamente em redes de sensores com múltiplos saltos (ESWARAN; ROWE; RAJKUMAR, 2005).

  Muito parecido com o TinyOS, o Nano-RK também possui uma arquitetura monolítica e sistema orientado a eventos. No entanto, ele suporta aplicativos em tempo real escritos em C, o que vem facilitando seu uso pelos desenvolvedores (Javed et al., 2018).

Esse SO também é famoso por seu método exclusivo de reserva de recursos com execuções controladas das prioridades e frequências das operações, fazendo com que o sistema execute múltiplas tarefas diferentes de forma simultânea o que, indiretamente, da suporte a trocas de informações entre vários nós remetentes e nós receptores em redes reais (Athreya et al., 2014).

O Nano-RK usa a linguagem de programação C para escrever suas tarefas implementando uma abordagem estática em sua criação. Por outro lado, fornece API para configuração dinâmica dos processos, agregado a um Sistema de Controle de Tarefas (TCB) para gestão da execução colaborativa e sincronizada de tarefas. No Nano-RK, o TCB fica lotado durante a inicialização e a chamada do sistema (ESWARAN; ROWE; RAJKUMAR, 2005).

O TCB utiliza registros referenciados e pilha de tarefas com definições de prioridade, tamanho da reserva de memória e requisição de pontualidade na execução. Isso garante um rígido controle do uso da CPU em prol da eficiência energética (ESWARAN; ROWE; RAJKUMAR, 2005).

Além do controle rígido do uso da CPU o Nano-RK também da suporte ao controle da largura de banda da rede usada nas transmissões, tanto para remetente quanto para receptor. O SO fornece também controle ao estado dos sensores que permanecem, por padrão, desligado, até que alguma operação solicite mudança de seu estado para ativado, fazendo com que o dispositivo realize aferição e posteriormente desligue automaticamente através da API do sistema (Javed et al., 2018).

Para uma garantia total do controle energético o Nano-RK mapeia todo o recurso que está sendo usado, estima o uso de energia e a vida útil do dispositivo, podendo modifica-los caso seja identificado sobrecarga de uso na CPU, buffer de rede e demais hardwares do sensor.

- **FreeRTOS**: Criado em 2002 por Richard Barry, o FreeRTOS é um dos RTOS mais usados para IoTs (Al-Taleb; Min-Allah, 2019). Foi desenvolvido com o objetivo de ser um sistema operacional com simples abstração, fácil programação, reprogramação e uso, além de portátil para dispositivos IoTs *low memory* (Barry, 2020).

  Esse SO é baseado na arquitetura de microkernel e implementa um modelo de programação multithread utilizando a linguagem de programação C com arquitetura mais simples composto por apenas quatro arquivos. É um sistema operacional multi-encadeado que implementa os sistemas de mutex, semáforos, alocação dinâmica de memória e um agendador preventivo implementado através do algoritmo de agendamento round-robin (Challouf; Kriaa; Saidane, 2019).

  Esse agendador de tarefas é muito simples e por isso possui um rápido tempo de resposta, pois reparte o processamento executado em hardware em pedaços iguais

entre os processos. Sua desvantagem é que nenhum processo termina antes de todos processos agendados sejam executados ao menos uma vez. Além disso, o FreeRTOS fornece suporte a gerenciamento de sistema de arquivos (Al-Taleb; Min-Allah, 2019).

O FreeRTOS suporta tarefas em tempo real usando operações determinísticas fazendo com que todos processos sejam executados em sequência. Ele não fornece técnicas de rede precisando de outros sistemas ou protocolos definidos em hardware para isso (Javed et al., 2018).

Assim como no RIOT, o FreeRTOS também implementa a técnica do modo sono profundo para reservar energia. Para isso o SO usa um sistema de *Idle task hook* que coloca o microcontrolador em estados que variam entre ocioso e inativo. Esse método simples não se mostra muito eficiente devido à necessidade de sair periodicamente do estado de inativo para checagem de agenda ou processar interrupções (Challouf; Kriaa; Saidane, 2019).

Quando não há tarefas de aplicativo a serem executadas, o modo ocioso do FreeRTOS mata o processamento de interrupção periódica para permitir que o microcontrolador permaneça em um profundo estado de economia de energia até que ocorra uma interrupção ou uma tarefa pré-agendada, fazendo que o hardware assuma o estado pronto (Al-Taleb; Min-Allah, 2019).

O FreeRTOS também garante um controle energético do hardware fornecendo algumas macros que permite que aplicações usem códigos específicos de economia de energia (Al-Taleb; Min-Allah, 2019).

- **Mbed OS**: O Mbed OS é um sistema operacional de código aberto desenvolvido em 2009 pela empresa ARM em parceria com a Silicon Labs (Al-Taleb; Min-Allah, 2019). O Mbed OS é baseado em uma arquitetura de microkernel, implementando um modelo de programação multithread confiável com alocação de memória estática e dinâmica (Muhammad et al., 2019).

  A alocação de memória estática é usada durante o tempo de compilação, enquanto a alocação de memória dinâmica é usada durante o tempo de execução. Esse SO também fornece um sistema de arquivos incorporado no chip flash (Al-Taleb; Min-Allah, 2019).

  Sua estrutura de aplicativos é escrita em C++ e garante uma maior segurança além da reutilização de suas bibliotecas. O MbedOS implementa IPv4 e IPv6 em sua camada de rede, 6LoWPAN em sua camada de encapsulamento e suporta HTTP ou CoAP em sua camada de aplicação (Javed et al., 2018).

  Para economia de energia o MbedOS, oferece um modelo de agendamento preempitivo que fornece APIs eficientes no gerenciamento de energia com um modo de

suspensão temporária do SO e de seus suportes, tornando-o o sistema preferido para microcontroladores com alta restrição energética (Javed et al., 2018).

Além desses modos de suspensão, há também o modo DeepSleep, o modo Power-Down e o modo DeepPowerDown. Esses modos economizam energia adicional; no entanto, eles também levam mais tempo para serem ligados novamente e exigem uma configuração um pouco mais complexa do modelo de comunicação dos dispositivos em tempo real, sendo nescessário um controle rígido das interrupções externas (Sabri; Kriaa; Azzouz, 2017).

O Mbed também oferece a possibilidade de variação da taxa de clock entre 48MHz a 128MHz, reduzindo o desempenho do dispositivo em até 50% e consequentemente seu consumo energético (Sabri; Kriaa; Azzouz, 2017).

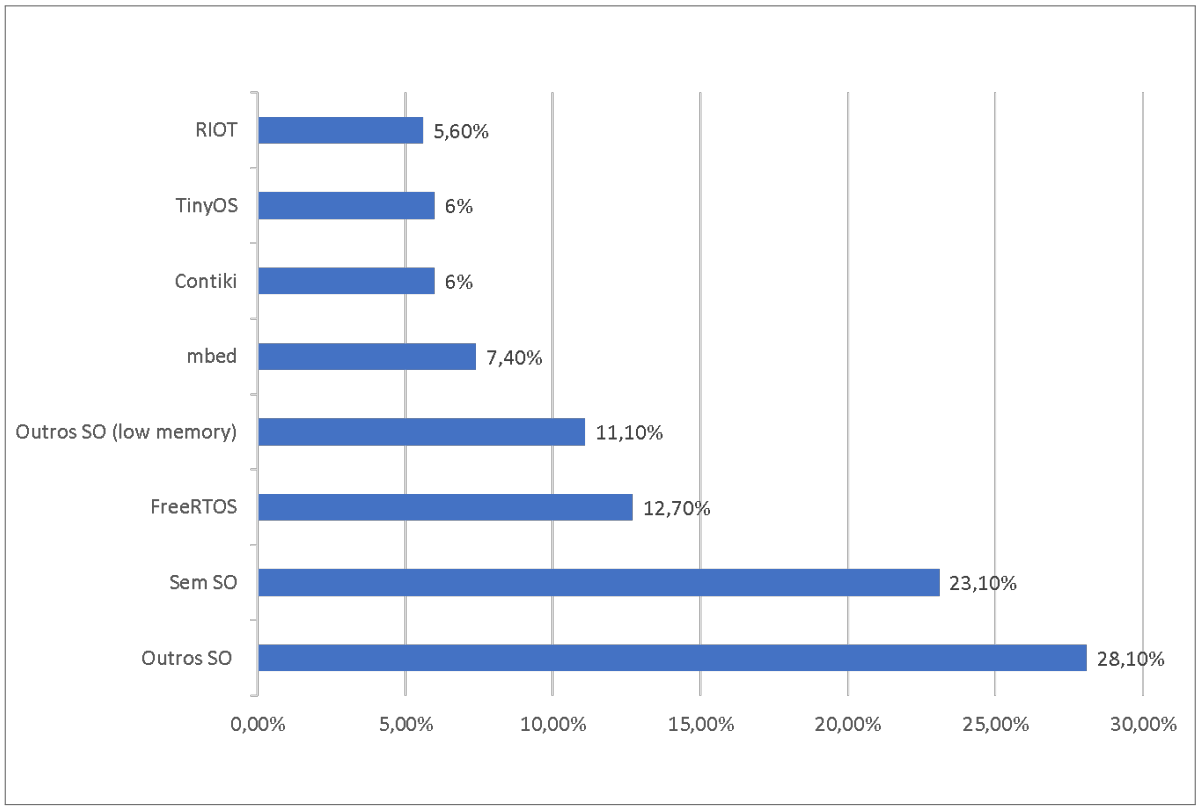
- **Outros SOs**: Existem diversos outros sistemas operacionais para dispositivos *low memory* que são amplamente conhecidos, porém não representam uma parcela expressiva dos SO dos dispositivos IoTs. (Skerrett, 2017) Dentre os que ainda se destacam em pesquisa cientifica recentes incluem-se o ERIKA OS muito estudado devido ao seus sistema eficiente de multitarefas (Choi, 2020).

  O OpenWSN, que entre muitos recursos, implementa o 6iTCH como sua pilha de rede, ainda utilizado na formação de sistemas de protocolos híbridos (Gardasevic; Plavsic; Vasilievic, 2019). Android, que, embora muito usado em sistemas *smartphone*, se torna cada vez mais incompatível com sistemas IoT de baixa memória e bateria (Ahmed et al., 2020). Outros sistemas operacionais de código aberto conhecidos são: nuttX(Pessacg et al., 2017) e o uClinux(Yangjian, 2017).

A Figura 8 apresenta a percentagem de uso dos sistemas operacionais mais utilizados pelos dispositivos IoT segundo o trabalho (Skerrett, 2017).

A Tabela 3 apresenta um resumo das informações de todos Sistemas operacionais IoTs de código aberto discutidos nesta seção.

Figura 8 – Grafico com a percentagem de uso dos sistemas operacionais mais utilizados pelos dispositivos IoT segundo o trabalho (Skerrett, 2017).



baseado em (Skerrett, 2017)

Tabela 3 – Comparação entre os principais Sistemas Operacionais

	Contiki	*TinyOS*	*RIOT*	*Nano-RK*	*FreeRTOS*	*Mbed OS*
Lançamento	2004	2000	2013	2005	2002	2009
Versão	3.0	2.1	1.0	1.0	10.3	6.0
Arquitetura	Modular	Monolitica	Microkernel RTOS	Monolitica	Microkernel RTOS	Microkernel RTOS
Agendamento	Cooperativo	Cooperativo	Preempitivo, tickless	Preempitivo	Preempitivo	Preempitivo
Suporte em tempo real	PARCIAL	NÃO	SIM	SIM	SIM	SIM
Modelo de Programação	Protothreads, Orientado a Eventos	Orientado a Eventos	Multi-threading	Multi-threading	Multi-threading	Multi-threading
Linguagem de Programação	C	NesC	C, C++	C	C	C++
Gerenciamento de Memoria	SIM	SIM	SIM	SIM	SIM	SIM
Gerenciamento do consumo energético	SIM	SIM	SIM	SIM	SIM	SIM
simuladores	Cooja	TOSSIM	Cooja, IoT-Lab	AVR Studio	MingW,GCC	Mbed Simulator
shell	SIM	NÃO	SIM	NÃO	NÃO	SIM
Protocolos Suportados	802.15.4 BLE 6Lowpan RPL Coap MQTT	802.15.4 ZigBee 6Lowpan RPL Coap MQTT	802.15.4 BLE ZigBee LPWAN 6Lowpan RPL Coap MQTT	802.15.4 BLE ZigBee LPWAN 6Lowpan RPL Coap MQTT	802.15.4 ZigBee 6Lowpan RPL Coap MQTT	802.15.4 BLE ZigBee LPWAN 6Lowpan RPL Coap MQTT
Hardwares Suportados	AVR, MSP430, ARM7, PIC32, x86, 6502, ARM Cortex-M	AVR, MSP430, px27ax	AVR, MSP430, ARM7, x86 , TI MSP430, ARM Cortex-M	AVR, MSP430	AVR,MSP430, ARM,x86,8052, Renesasc	ARM Cortex-M
Memória ROM Utilizada	30	4	5	18	9	16
Memória RAM Utilizada	2	1	1.5	2	4	4

Fonte: O autor

2.6 DISPOSITIVOS IOTS/WSN

Uma revisão detalhada apresentando informações relevantes sobre os dispositivos IoTs/-sensores programáveis, compatíveis com os padrões, protocolos e sistema operacional discutidos neste capítulo pode ser visto na tabela 4.

Tabela 4 – Dispositivos IoTs/sensores

Nome	Modelo	Clock	RTC	SO Compatíveis	Conectividade	Interfaces	Sensores	Consumo energético	Memória RAM	Memória ROM	Memória Flash	Serviços
eXtremeScale Mote(XSM)	Atmel ATmega128L	8MHz	32.768kHz	TyniOS	802.15.4 6LowPan	I2C/TWI SPI UART GPIO USART	Temperatura Luminosidade Acelerômetro Acústico GPS	Transmitindo:16.5mA Recebendo:9.6mA Sleep:N/A	4kB	4Mbit (SerialFlash)	128kB	IPV6
Fleck1	Atmel ATmega128	16MHz	32kHz	TyniOS	802.15.4 6LowPan	I2C/TWI SPI UART GPIO USART	Temperatura Luminosidade	Transmitindo:12.5-29.5mA Recebendo:18.5-22.5mA Sleep:N/A	4kB	N/A	128kB	IPV6
Fleck2	Atmel ATmega128	16MHz	32kHz	TyniOS	802.15.4 6LowPan	I2C/TWI SPI UART GPIO USART	Temperatura Acelerômetro GPS	Transmitindo:12.5-29.5mA Recebendo:18.5-22.5mA Sleep:N/A	4kB	N/A	128kB	IPV6
Iris	Atmel ATmega1281	16MHz	32kHz	TyniOS ContikiOS	Wi-Fi 802.15.4 ZeegBee 6LowPan WirelessHart	I2C/TWI SPI UART GPIO USART	Temperatura Humidade Luminosidade Pressão Acelerômetro Acústico	Transmitindo:10-17mA Recebendo:16mA Sleep:8µA	8kB	512kB (SerialFlash)	128kB	HTTP UDP TCP IPv4 IPv6
MicaZ	Atmel ATmega128L	8MHz	32kHz	TyniOS	Wi-Fi 802.15.4 ZeegBee 6LowPan	I2C/TWI SPI UART GPIO USART	Temperatura Humidade Luminosidade Pressão Acelerômetro Acústico	Transmitindo:11–17.4mA Recebendo:19.7mA Sleep:15µa	4kB	512kB	128kB	IPv6
Pluto	TI MSP430F149	8MHz	N/A	TyniOS	Wi-Fi 802.15.4 ZeegBee 6LowPan	SPI UART GPIO USART	Temperatura Acelerômetro	Transmitindo:17.4mA Recebendo:18.8-19.7mA Sleep:N/A	2kB	N/A	60kB	IPv6
SENTIO	Atmel ATmega128L	8MHz	32kHz	ContikiOS	802.15.4 ZeegBee 6LowPan	I2C/TWI SPI UART GPIO USART	Temperatura Humidade Luminosidade Giroscópio Acelerômetro Acústico	Transmitindo:16.8mA Recebendo:16.9mA Sleep:0.2µa	4kB +32kB (Externo)	N/A	128kB	HTTP UDP TCP IPv4 IPv6
Telos/ Tmote	TI MSP430F149	8MHz	32kHz	TyniOS	802.15.4 ZeegBee 6LowPan	I2C/TWI SPI UART GPIO USART USB	Temperatura Humidade Luminosidade RadiaçãoSolar	Transmitindo:17.4-21mA Recebendo:18.8-23mA Sleep:5.1-21µA	10kB	N/A	60kB	IPv6
TmoteMini (Plus)	TI MSP430F1611	8MHz	N/A	TyniOS Moteiv'sOS	802.15.4 ZeegBee 6LowPan	I2C/TWI SPI UART GPIO USART	Temperatura	Transmitindo:17.4mA Recebendo:18.8-19.7mA Sleep:1µA	10kB	N/A	48kB	IPv6
WiSMote (Dev)	TI MSP430F5437	8MHz	32kHz	ContikiOS	802.15.4 ZeegBee 6LowPan WirelessHart	I2C/TWI SPI UART GPIO USB	Temperatura Humidade Luminosidade Acelerômetro CO2 Acústico	Transmitindo:25.8-33.6mA Recebendo:18.5mA Sleep:2µA	16kB	1-8MB(ExternalFlash)	256kB	HTTP UDP TCP IPv4 IPv6
WiSMote (mini)	Atmel ATmega128RFA2	16MHz	N/A	ContikiOS	802.15.4 ZeegBee 6LowPan WirelessHart	I2C/TWI SPI UART GPIO USART	Temperatura Humidade Luminosidade Acelerômetro CO2 Acústico	Transmitindo:18.6mA Recebendo:12.5-16.6mA Sleep:20nA	16kB	N/A	128kB	HTTP UDP TCP IPv4 IPv6

Fonte: O autor

2.7 RESUMO DO CAPÍTULO

Este capítulo deixa claro que com o surgimento da IoT, novos protocolos regulatórios para gerir sua energia, escalabilidade, segurança, privacidade, interferência humana no circuito, big data, etc. tornaram-se necessários.

Esses protocolos devem ser capazes de se adequar a essa revolução na conectividade de diferentes dispositivos, em que rede de sensores, dispositivos *mobile* e aplicações em tempo real, estão interconectados e ligados à internet através de tecnologias baseadas em IP.

Além desses protocolos os sistemas operacionais tradicionais, os sistemas executados em hosts e o sistema operacional típico para redes de sensores também devem ser capazes de atender os diversos requisitos de uma ampla gama de dispositivos.

Portanto, nesta pesquisa bibliográfica foi apresentado também, diferentes sistemas operacionais selecionados de acordo com sua novidade e uso no domínio IoT nos últimos anos. Foi possível notar muitos desafios em abertos como a necessidade de arquitetura de kernel leve, que deve reduzir os serviços do SO e ocupar pouco espaço com uma execução eficiente de tarefas, a compatibilidade em tempo real, a minimização do consumo de energia e a interoperabilidade entre os dispositivos IoT.

Este trabalho apresentou a fragilidade de cada Sistema Operacional IoT com 9 preocupações principais que são: Arquitetura do sistema operacional, agendador e recursos em tempo real, modelo de programação, conectividade de rede, suporte de hardware, em especial aos problemas espaço de memória, suporte a protocolos de rede e eficiência energética. Comparando os sistemas operacionais escolhidos de acordo com cada preocupação.

Embora o sistema operacional ideal esperado para a IoT ainda não tenha nascido, deve-se levar em consideração que muitos outros desafios estudados neste capítulo devem ser analisados, considerando o gerenciamento da escalabilidade dos IoTs no espaço da rede.

A análise abrangente de todos os protocolos, sistema operacional e conceitos forneceu um conhecimento amplo nas questões e desafios existentes, deixando claro a necessidade de uma cadencia eficiente e otimizada entre hardware, protocolos, aplicações e sistemas, que garantam, de forma inteligente e adaptativa, uma comunicação mais eficiente entre esses agentes, os dispositivos que utilizam diferentes sistemas e todos os requisitos estabelecidos pela rede.

3 APRENDIZADO DE MÁQUINA

> *Aprender é como remar contra a corrente: sempre que se para, anda-se para trás.*
>
> Confúcio,

Nos últimos anos, problemas complexos de otimização como multidimensionalidade, multimodalidade, multiobjetivos, diferenciabilidade e diferentes características combinatórias são enfrentadas diariamente, por vezes, requisitando uma solução ideal em tempo real (Memeti et al., 2019).

Essa demanda por uma solução em tempo real resultou no desenvolvimento de técnicas precisas e computacionalmente poderosas, que podem ser compreendidas em três grandes áreas, Métodos de Extração, Métodos Heurísticos e Métodos de Aprendizado (Rajakumar; Dhavachelvan; Vengattaraman, 2016).

Métodos de extração, como programação lógica e matemática, estão diretamente envolvidos na resolução de problemas NP-completos, enquanto os Método Heurísticos se mostram superiores na resolução de problemas de otimização NP-difíceis ou NP-completo, obtendo rapidamente soluções denominadas sub-ótimas que atendem a demanda desses problemas, especialmente se comparado ao tempo de obtenção de soluções ótimas obtidas pelos métodos tradicionais (De León-Aldaco; Calleja; Aguayo Alquicira, 2015).

Métodos Heurísticos são definidos formalmente como processos de geração iterativa que orientam uma heurística combinando de forma inteligente diferentes conceitos para explorar de forma ampla ou minuciosa o espaço de busca focando na obtenção imediata de um resultado exato ou aproximado.

A forte tendência no uso de técnicas heurísticas, como método substituto às técnicas exatas se deve ao fato comprovado de que vários problemas precisam de soluções eficientes quase que em tempo real, ainda que não seja a solução ótima, algo que não é possível obter com algoritmos exatos que, independentemente do poder computacional, utiliza tempo de execução incompatíveis com aplicações em tempo reais (Memeti et al., 2019).

Os algoritmos meta-heurísticos também são frequentemente usados para o gerenciamento e uso eficiente de recursos para aumentar a produtividade, em casos que o espaço de busca é tão grande que tornam a busca exaustiva, e métodos iterativos ou heurísticas simples impraticáveis (Memeti et al., 2019).

O grande problema de boa parte dos algoritmos heurísticos é a alta complexidade computacional utilizada na obtenção de soluções, podendo gerar os mesmos problemas de desempenho computacional das técnicas tradicionais (Barbosa; Menezes, 2017).

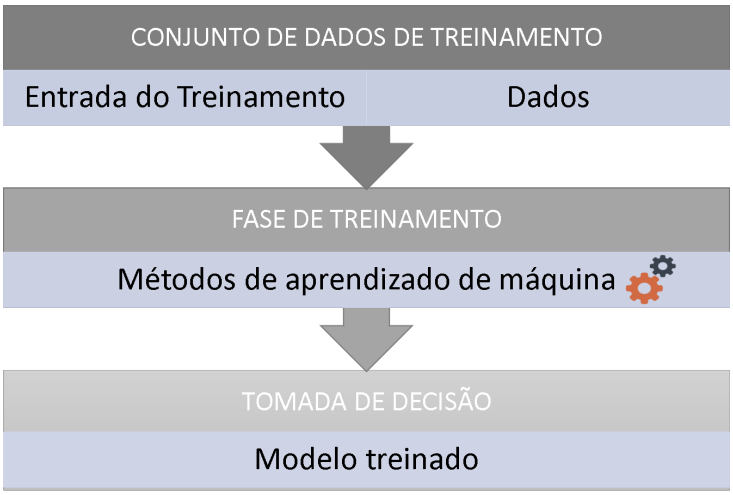
Esses problemas se agravam, pois parte das técnicas tradicionais e heurísticas são simuladas e observadas somente através de computações numéricas e análises teóricas, tornando sua implementação em sistemas reais ainda um obstáculo, principalmente em dispositivos com processamento limitado (Mao; Hu; Hao, 2018).

Outra dificuldade encontrada é a pouca autonomia imposta por essas técnicas que acabam dependente de uma interferência humana para a obtenção de seus resultados, tornando seu uso inviável principalmente em sistemas onde a tomada de decisão deve ser automática, inteligente e autônoma; a exemplo do controle de requisitos em redes de computadores reais (Fadlullah et al., 2017).

Nas últimas décadas, os métodos de aprendizado de máquina foram muito explorados pois implementam uma coleção de técnicas poderosas da áreas de IA que permite ao sistema aprender de forma mais inteligente os padrões e modelos estruturais de comportamento úteis na implantação de tomadas de decisões em sistemas autônomos (Xie et al., 2019).

Uma abordagem baseada em aprendizado de máquina geralmente consiste em duas fases principais: fase de treinamento e fase de inferência, conforme ilustrado na Figura 9. Na fase de treinamento, os métodos de aprendizado de máquina são aplicados para aprender a distribuição estatística dos dados do problema usando o conjunto de dados de treinamento (GONÇALVES et al., 2008). Na inferência, o sistema obtém a saída estimada para cada nova entrada usando o modelo treinado.

Figura 9 – O procedimento geral de processamento de uma abordagem de aprendizado de máquina.



Fonte: O autor

A seção a seguir apresenta, de forma resumida, algumas técnicas de aprendizado de máquina que precisaram ser estudadas e testadas.

3.1 ALGORITMOS DE APRENDIZADO DE MÁQUINA

O aprendizado de máquina é o estudo científico de algoritmos e modelos estatísticos que os sistemas computacionais usam para executar uma tarefa específica sem serem explicitamente programados (MAHESH, 2020).

O objetivo de vários algoritmos desses é criar métodos de aprendizado para que dispositivos aprendam automaticamente sem qualquer intervenção ou assistência humana (BONACCORSO, 2017).

A principal vantagem de usar o aprendizado de máquina é que, uma vez que um algoritmo aprende o que fazer com os dados, ele pode fazer seu trabalho automaticamente (MAHESH, 2020).

Técnicas de aprendizado de máquinas podem ser diferenciadas em quatro categorias: aprendizado supervisionado, não supervisionado, semissupervisionado e por reforço. Nesta seção, serão apresentados os algoritmos de aprendizado de máquina amplamente usados na literatura na resolução de problemas em sistemas autônomos. Cada algoritmo é explicado resumidamente com alguns exemplos sobre a teoria do aprendizado de máquina e seus conceitos clássicos. Todos Os algoritmos apresentados foram testados se sobrepujando-se o método de Redes Neurais Artificiais.

3.1.1 Aprendizado Não-Supervisionada

As técnicas de aprendizado não-supervisionado recebem um conjunto de entradas sem rótulos (ou seja, sem dados de saída), fazendo com que seus algoritmos busquem sozinho por padrões, estruturas ou conhecimento em dados, agrupando amostras em diferentes grupos de acordo com a similaridade das informações. Algoritmos não supervisionados são utilizados quase que unanimemente para agrupamento e agregação de dados (Xie et al., 2019). Entre as técnicas mais conhecidas estão o k-means (Krishna; Narasimha Murty, 1999) e o mapa auto-organizado (SOM) (Xie et al., 2019) .

- **O algoritmo k-means**: é um algoritmo popular de aprendizado não supervisionado, usado para agrupar um conjunto de dados não rotulados em diferentes clusters. Para implementar o algoritmo k-means, são necessários apenas parâmetros como o conjunto de dados inicial de entrada e o número desejado de clusters k. Para resolver o problema, o k-means (1) inicializa k centroides do cluster escolhendo aleatoriamente k nós; (2) usa uma função de distância para rotular cada nó com o centroide mais próximo; (3) atribui novos centroides de acordo com os membros do nó atual e (3) para o algoritmo se a condição de convergência for válida ou, caso contrário, volta para a etapa (2) (Krishna; Narasimha Murty, 1999).

- **Mapa auto-organizado (SOM)**: também conhecido como Mapa de recursos de auto-organização (SOFM), é um dos modelos de rede neural não supervisionados mais populares. O SOM é frequentemente aplicado para realizar redução de dimensionalidade e armazenamento de dados em cluster. Em geral, o SOM tem duas camadas, a primeira de entrada e segunda camada de mapeamento. Quando o SOM é usado para realizar o agrupamento de dados, o número de neurônios na camada

do mapa é igual ao número desejado de clusters. Cada neurônio possui um vetor de peso. As etapas para resolver o problema de agrupamento de dados usando o algoritmo SOM são: (1) inicializar o vetor de peso de cada neurônio na camada do mapa; (2) escolher uma amostra de dados pertencente ao conjunto de dados de treinamento; (3) usar uma função de distância para calcular a similaridade entre a amostra de dados de entrada e todos os vetores de peso. O neurônio cujo vetor de peso tem a maior similaridade é chamado de Melhor Unidade Correspondente (BMU). O algoritmo SOM é baseado no aprendizado competitivo, o que significa que há apenas um BMU de cada vez. Na fase (4) de funcionamento do SOM a vizinhança do BMU é calculada, em (5) os vetores de peso dos neurônios na vizinhança do BMU (incluindo o próprio BMU) são ajustados para a amostra de dados de entrada. Quando o algoritmo consegue o mapeamento e agrupamento ideal atendendo às condições definidas para a convergência ele finaliza (6), caso contrário ele retornará a etapa (2) (Xie et al., 2019).

3.1.2 Aprendizado Semi-supervisionado

No aprendizado semi-supervisionado são utilizados como entrada para sua operação dados rotulados e não rotulados. O aprendizado semi-supervisionado é útil para aplicações onde a aquisição de dados rotulados é custosa ou difícil de se obter, enquanto a aquisição de uma grande quantidade de dados não rotulados é relativamente fácil e barata. Essa técnica também é eficaz no uso de dados não rotulados durante o processo de treinamento, o que tende a melhorar o desempenho do modelo treinado (Kingma et al., 2014).

Para fazer o melhor uso dos dados não rotulados são implantados ao sistema suposições como suposição de suavidade, suposição de cluster, suposição de separação de baixa densidade e suposição de variedade.

A *Pseudo Labeling* é o melhor exemplo de técnica de aprendizado semisupervisionado pois é simples e eficiente. A ideia principal da técnica é usar os dados rotulados para treinar um modelo, em seguida, usar o modelo treinado para prever pseudos rótulos dos dados não rotulados. Por fim a técnica combina os dados rotulados e os dados pseudo-rotulados criados para treinar o modelo novamente.

Existem outros métodos de aprendizado semissupervisionados, como Maximização de Expectativas (EM), co-treinamento, SVM transdutivo e métodos baseados em grafos. A indicação de cada método depende das suposições sobre os dados de entrada. Por exemplo, o EM baseia-se na suposição de cluster, o SVM transdutivo baseia-se na hipótese de separação de baixa densidade, enquanto os métodos baseados em grafos baseiam-se na hipótese de variedade (Zhu et al., 2019).

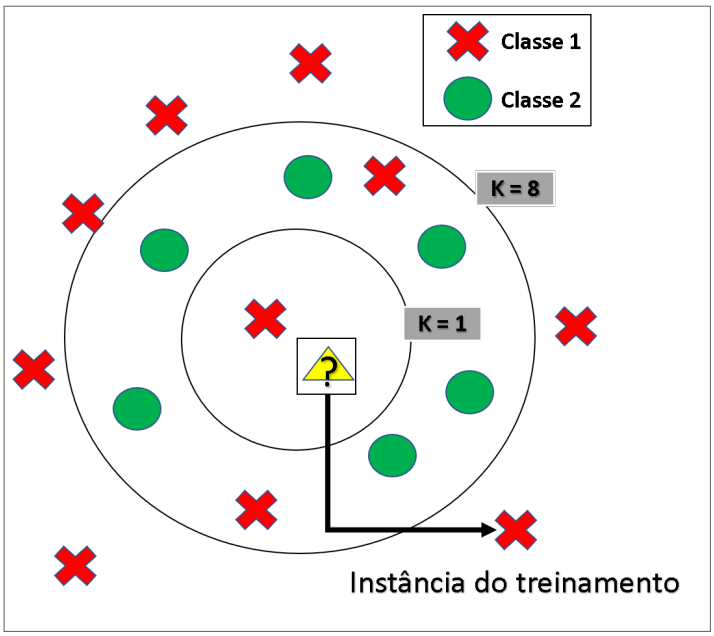
3.1.3 Aprendizado Supervisionado

Aprendizado supervisionada são técnicas utilizadas para classificação e regressão de dados. Os algoritmos de aprendizado supervisionados recebem um conjunto de dados de treinamento com entradas e saídas conhecidas, e constrói um modelo de sistema que representa a relação aprendida entre a entrada e a saída. Após o treinamento, quando uma nova entrada é alimentada no sistema, o modelo treinado tenta classificar essa entrada buscando associa-la a uma saída conhecida ou aproximada.

Entre os exemplos de aprendizado supervisionado amplamente utilizados (Xie et al., 2019) estão k-vizinho mais próximo (Hu et al., 2016), árvore de decisão(Chanama; Wongwirat, 2018), random forest (Breiman, 2001; Mengmeng; Yian, 2018), máquina de vetores de suporte (Yu et al., 2014; Xie et al., 2019), teoria de Bayes (Hartigan, 2012; Ahmadi; Bouallegue, 2015) e Modelo oculto de Markov (Salvo Rossi; Ciuonzo; Ekman, 2015).

- **k-vizinho mais próximo (k-NN)**: O k-NN é uma técnica de aprendizado supervisionado, em que a classificação de uma amostra é determinada com base em um número predefinido k de vizinhos mais próximos dessa amostra não classificada. O processo do algoritmo k-NN é muito simples: se a maioria desses k vizinhos mais próximos pertencerem a uma determinada classe, a amostra não classificada será classificada nessa classe. A Figura 10 mostra um exemplo simples de como o algoritmo k-NN funciona.

Figura 10 – Funcionamento da técnica de aprendizado K-NN



Fonte: O autor

No k-NN métricas importantes como k e a função utilizada para definir a distancia entre os vizinhos são decisórias para a qualidade da classificação. Pois, quando

maior for o valor de k, maior será a qualidade da classificação pois tende a minimizar os efeitos de ruídos durante a classificação. A distância é a principal métrica do algoritmo k-NN, várias funções podem ser aplicadas para definir o critério utilizado na avaliação do distanciamento entre a amostra não rotulada e seus vizinhos. Entre as técnicas mais conhecidas para essa definição estão as Chebyshev, *City-block*, Minkowsky, Euclidiana, Cosseno e Qui-quadrado (Hu et al., 2016).

- **Arvore de decisão (DT)**: O DT é uma técnica que realiza sua classificação por meio de uma árvore de aprendizagem. Na árvore, cada nó representa uma característica (atributo) de um dado, todos os ramos representam conjunções dessas características que levam às classificações e cada nó folha é um rótulo de classe. A amostra não rotulada pode ser classificada comparando seus valores de características com os nós da árvore de decisão (Chanama; Wongwirat, 2018).

  O DT tem muitas vantagens, como expressão de conhecimento intuitiva, implementação simples e alta precisão de classificação. O ID3, C4.5 e CART são algoritmos de árvore de decisão amplamente usados na realização de classificação automática do conjunto de dados de treinamento. A maior diferença entre eles são os critérios de divisão que são usados para construir árvores de decisão. Os critérios de divisão usados por ID3, C4.5 e CART são o ganho, razão para o ganho e impureza de Gini respectivamente (Xie et al., 2019).

- ***Random Forest***: Esse método também é conhecido como floresta de decisão aleatória, pode ser usado para tarefas de classificação e regressão. O *Random Forest* consiste em um conglomerado de muitas árvores de decisão. Isso faz com que essa técnica mitigue sobre ajustes melhorando sua precisão através de uma escolha aleatória do subconjunto do espaço de recursos na construção de suas árvores de decisão (Breiman, 2001).

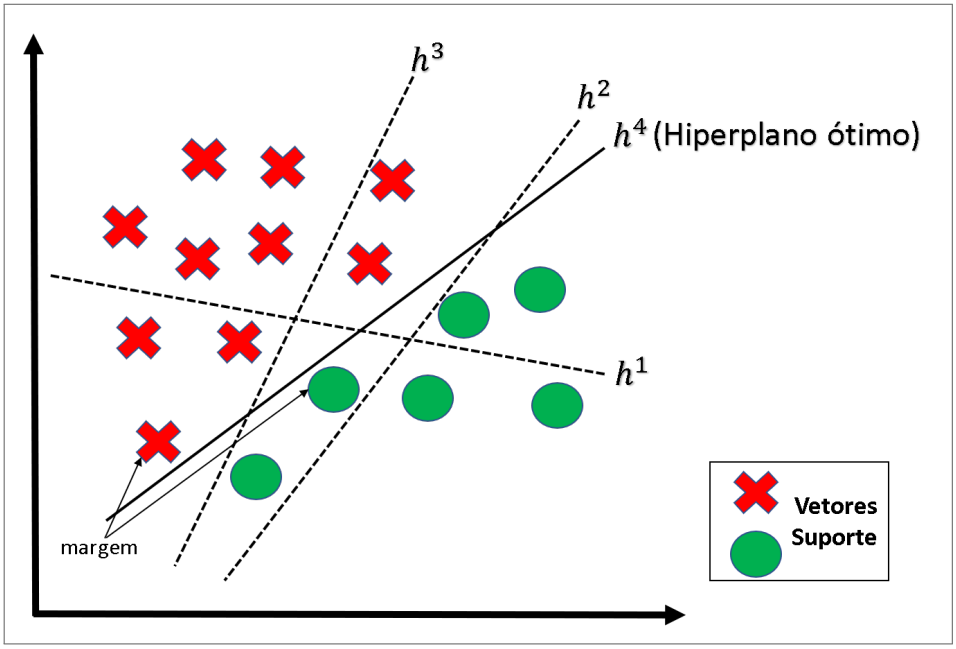
  Para isso a técnica *Random Forest* utiliza três etapas para classificar uma nova amostra de dados que são: (a) colocar a amostra de dados em cada árvore da floresta. (b) Cada árvore dá um resultado de classificação, que é o "voto"da árvore. (c) A amostra de dados será classificada na classe com mais votos (Mengmeng; Yian, 2018).

- **Máquina de vetores de suporte (SVM)**: SVM é outro método popular de aprendizagem supervisionada amplamente utilizado em classificação e reconhecimento de padrões. A ideia básica do SVM é mapear os vetores de entrada em um espaço de multidimensionais. Este mapeamento é obtido através da aplicação de diferentes funções *kernel*, como linear, polinomial e a *Radial Based Function (RBF)*. A seleção da função do kernel é uma tarefa importante no SVM, pois tem efeito direto na precisão da classificação (Yu et al., 2014).

A seleção da função do kernel depende do conjunto de dados de treinamento, exemplo: a função de kernel linear funciona bem se o conjunto de dados for linearmente separável, a função polinomial este conjunto de dados não for linearmente separável, e a RBF juntas as duas funções de kernel, fazendo com que esta última obtenha desempenho relativamente melhor do que as outras duas funções do kernel (Xie et al., 2019).

O objetivo do SVM é encontrar um hiperplano de separação no espaço de recursos para maximizar a margem entre as diferentes classes. Observe que a margem é a distância entre o hiperplano e os pontos de dados mais próximos de cada classe. Os pontos de dados mais próximos correspondentes são definidos como vetores de suporte. Um exemplo de classificador SVM é mostrado na Figura 11 onde é exibido os diferentes hiperplanos de separação possíveis entre duas classes, mas apenas um hiperplano de separação ótimo pode maximizar a margem.

Figura 11 – Classificador SVM com um hiperplano linear ótimo, duas classes, cada classe possuindo um vetor de suporte. Como pode ser visto, existem muitos hiperplanos de separação possíveis entre duas classes, como h^1, h^2, h^3 e h^4, mas apenas um hiperplano de separação ideal (h^4) que maximiza corretamente a margem



Fonte: O autor

- **Modelo oculto de Markov (HMM)**: HMM é um tipo de modelo de Markov. Os modelos de Markov são amplamente usados em ambientes dinâmicos aleatórios que obedecem à propriedade sem memória. A propriedade sem memória dos modelos de Markov significa que a distribuição de probabilidade condicional de estados futuros se relaciona apenas ao valor do estado atual e é independente de todos os estados anteriores. Existem outros modelos de Markov, como Cadeias de Markov (MC). A principal diferença entre o HMM e outros modelos é que o HMM é frequentemente

aplicado em ambientes onde os estados do sistema são parcialmente visíveis ou não são visíveis.(Salvo Rossi; Ciuonzo; Ekman, 2015)

- **Teorema de Bayes**: O Teorema de Bayes usa a probabilidade condicional para calcular a probabilidade de um evento ocorrer, dado o conhecimento prévio das condições que podem estar relacionadas ao evento. O Teorema de Bayes é definido por

$$P(H|E) = \frac{P(E|H) \times P(H)}{P(E)}, \tag{3.1}$$

onde E é uma evidência, H uma hipótese, $P(H|E)$ a probabilidade *a posteriori* da hipótese H se sustentar dada uma nova evidência E, $P(E|H)$ é a probabilidade posterior da evidência E condicionada ao hipótese H, $P(H)$ é a probabilidade anterior da hipótese H, independente da evidência E, e $P(E)$ a probabilidade de ocorrência da evidência E (Hartigan, 2012).

Esse teorema em aprendizado de máquina é utilizado em problemas de classificação, onde a teoria de Bayes aprende um modelo de probabilidade usando o conjunto de dados de treinamento. A evidência E é uma amostra de dados, e a hipótese H é a classe a ser atribuída à amostra de dados. A probabilidade*a posteriori* $P(H|E)$ representa a probabilidade de uma amostra de dados pertencer a uma determinada classe. A fim de calcular a probabilidade posterior $P(H|E)$, $P(H)$, $P(E)$ e $P(E|H)$ precisam ser obtidos com base no conjunto de dados de treinamento usando as teorias de probabilidade e estatística, que é o processo de aprendizagem do modelo de probabilidade (Ahmadi; Bouallegue, 2015).

Ao classificar uma nova amostra de dados de entrada, o modelo de probabilidade pode ser usado para calcular múltiplas probabilidades posteriores para diferentes classes. A amostra de dados será classificada na classe com a maior probabilidade posterior $P(H|E)$. A vantagem da teoria de Bayes é que requer um número relativamente pequeno de conjunto de dados de treinamento para aprender o modelo de probabilidade. No entanto, há uma importante suposição de independência ao usar a teoria de Bayes. Para facilitar o cálculo de $P(E|H)$, as características das amostras de dados no conjunto de treinamento são consideradas independentes umas das outras (Xie et al., 2019).

Outra técnica de aprendizado supervisionado muito importante é a Rede Neural Artificial (RNA) apresentada na seção a seguir.

3.2 REDES NEURAIS ARTIFICIAIS (RNA)

Assim como no sistema nervoso humano, as RNAs são compostas por um numeroso conjunto de neurônios, em sua maioria, fortemente conectadas que processam e trocam informações em uma estrutura de grafo orientado, onde os nós deste grafo são unidades de processamento simples denominados neurônios artificiais (McCulloch; Pitts, 1943).

Esses neurônios artificiais operam em paralelo, realizando cálculos com formulações complexas de expressões não lineares, obtendo um conhecimento experiencial a partir de dados históricos. Esse aprendizado é realizado através de funções de ativação que modificam o estado interno do neurônio que por sua vez propaga esse seu estado de ativação para as outros neurônios através de pesos sinápticos ou conexões (Perkov, 2020).

As mudanças realizadas nos pesos sinápticos e nas ponderações dos pesos de interconexão entre os neurônios são responsáveis pelas alterações no comportamento da rede fazendo com que esses neurônios assimilem o aprendizado de um novo comportamento (Osório, 1999).

Essas modificações recorrentes no estado de ativação na saída da rede em resposta a configuração na entrada fazem com que a RNA seja capaz de estabelecer uma associação entrada-saída entre as informações, como na relação estímulo e resposta de uma Rede Neural humana que se adapta a diferentes situações propostas. Essas modificações no comportamento da rede são denominadas como regra de aprendizado (Taulli, 2020).

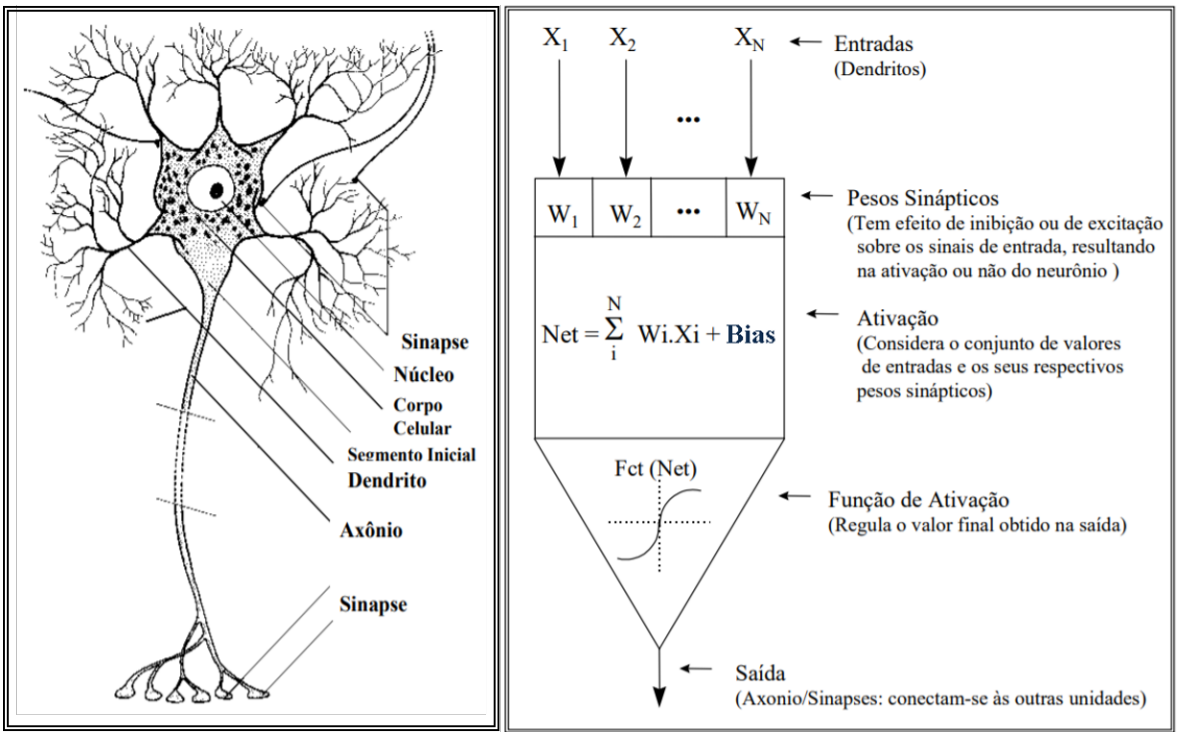
O conhecimento adquirido e as regras de aprendizado que descrevem o comportamento da rede ficam codificados nas conexões e dentro dos neurônios representados na forma de vetores de valores numéricos. Uma vez codificados esses valores ficam difícil à análise e interpretação humana, não fazendo sentido a relação numérica entre os valores (Osório, 1999). A Figura 12 mostra a relação entre os neurônios naturais e o modelo de neurônio artificial.

Especificamente no aprendizado supervisionado essa RNA deve adaptar seus pesos de forma a conhecer as associações existentes entre um conjunto de entrada ligadas as saídas de maneira a responder a novas entradas com um estado de ativação compatível com o esperado (Taulli, 2020).

A otimização do desempenho de uma rede neural está relacionada diretamente à configuração de seus parâmetros e hiperparâmetros. Dependendo do problema de domínio e dos conjuntos de dados, implementar um modelo que produza resultados melhores e mais rápidos depende do ajuste de hiperparâmetros como: O número de camadas de neurônios, o algoritmo gradiente descendente, a função de ativação dos neurônios, a taxa de aprendizado, o tamanho dos lotes de treinamento, o número de épocas de treinamento, entre outros que precisam ser ajustados durante o treinamento (Huan et al., Sep. 2016), (Mohammadi et al., 2018).

Uma rede neural do tipo MultiLayer Perceptron (MLP) foi utilizada para controle mais

Figura 12 – Exemplo de um neurônio natural e um neurônio artificial



Fonte:(Osório, 1999)

rigoroso da dimensionalidade e custos computacionais. As redes multicamadas se distinguem das redes de camada única pelo número de neurônios de rede nas camadas ocultas (Huan et al., Sep. 2016). Esses neurônios tornam um MLP capaz de realizar projeções ou classificações de inferência, reduzindo a ordem de complexidade assintótica de técnicas mais complexas e consequentemente economizando memória e energia, pré-requisitos básicos de dispositivos sensores com recursos limitados.

As subseções a seguir buscam explicar melhor a importância desses parâmetros e os ajustes de suas configurações para a otimização de ação de uma rede neural.

3.2.1 Algoritmos de Descida do Gradiente

A Descida do Gradiente é uma ferramenta comum utilizada na minimização de funções complexas, através da busca dos coeficientes de uma reta que aproximem a função de custo do zero, geralmente utilizando um algoritmo iterativo de otimização (Homem; Ufes, 2020).

Um dos processos de uma rede neural é a minimização da perda durante o processo de classificação ou predição tornando-a um modelo de melhor ajuste com o menor erro. Para tanto, ajustes e sobre ajuste dos pesos dos neurônios dependem de configuração realizadas por uma perda mais baixa e na direção da mudança da inclinação durante a descida do gradiente, evitando o problema de captura de mínimos locais, em oposição à obtenção do valor mínimo global absoluto desejável, bem como outros desafios associados à descida de

gradiente (Dogo et al., 2018).

Problemas não convexos são o caso mais comum para redes neurais, portanto, escolher uma estratégia de otimização que busque encontrar os ótimos globais nessas redes costuma ser um desafio devido ao processo de estimativa de um número muito grande de parâmetros em um espaço de busca de alta dimensão (Dogo et al., 2018).

Existem três tipos diferentes de método de gradiente descendente, a saber, "Gradiente descendente em lote", "Gradiente descendente estocástico"e "Descida gradiente em *minibatch*". Esses métodos diferem uns dos outros de acordo com a quantidade de dados usados no cálculo do gradiente.

O método *Batch Gradient Descent* usa todo o conjunto de treinamento para calcular o gradiente da função de custo. No método *Stochastic Gradient Descent* (SGD), um processo de atualização é executado para cada exemplo de treinamento. Portanto, ele funciona mais rápido, porém pode atingir o mínimo local mais cedo (Goodfellow et al., 2016).

O método *Mini-Batch Gradient Descent* foi desenvolvido considerando as vantagens de ambos os métodos. O processo de atualização é realizado para cada exemplo de n conjuntos de treinamento (Yazan; Talu, 2017).

Alguns algoritmos utilizados para otimização desse gradiente foram propostos nos últimos anos, a tendência baseada principalmente em uma rápida adaptação de estimativas que funcionam automaticamente e permitem uma minimização no ajuste em outros hiperparâmetros. Entre os mais disseminados na literatura destacam-se:

- ***Momentum***: *Momentum* foi proposto como um método que acelera o método SGD e tem efeito na redução das oscilações no SGD. Quando o momentum é aplicado ao método SGD, as oscilações ocasionadas pela aproximação ao valor mínimo diminuem (Figura 13). Ao atualizar os parâmetros históricos, o método *Momentum* leva em consideração não apenas o gradiente naquela iteração, mas também os gradientes anteriores, assumindo valores probabilísticos entre 0 e 1 para determinar quanto dos gradientes anteriores serão levados em consideração, em geral é considerado um valor em torno de 0,9 (Goodfellow et al., 2016).

- **Adagrad**: O método Adagrad é um método que elimina o problema decorrente do coeficiente constante de aprendizado no método gradiente descendente. De acordo com esse método, o coeficiente de aprendizado também é atualizado a cada etapa. Porém neste método é levado em consideração uma matriz do mesmo tamanho que o número de parâmetros que contém as informações do gradiente histórico. Cada elemento desta matriz representa a soma dos quadrados dos gradientes anteriores do parâmetro relevante. Esta matriz é usada para dimensionar o coeficiente de aprendizado. Assim, o coeficiente de aprendizado é atualizado dependendo dos parâmetros de cada etapa. No entanto, é necessário determinar inicialmente um coeficiente de

aprendizado. Outro ponto a ser observado é o uso de valores para evitar a divisão por zero, geralmente é considerado o valor 10^8 (Duchi; Hazan; Singer, 2011).

- **Adadelta**: Uma fraqueza do método Adagrad é que o coeficiente de aprendizado se aproxima de zero com o tempo, o crescimento da matriz de gradientes ao longo do tempo faz com que o coeficiente de aprendizado diminua. Este é um dos motivos que levou ao desenvolvimento do método Adadelta. Outra é a necessidade do parâmetro do coeficiente de aprendizado inicial. O método Adadelta eliminou essas duas desvantagens do método Adagrad. O método Adadelta pega a soma dos quadrados do gradiente em uma certa proporção, não todos como no método Adagrad, e define isso como uma constante, como no método *Momentum* (Zeiler, 2012).

- **RMSProp**: Proposto por Geoff Hinton incluído em suas notas de aula como uma variação do método Adadelta, este método é semelhante ao método Adadelta, mas usa o parâmetro do coeficiente de aprendizado considerando o valor do parâmetro como 0,9 e o valor do coeficiente de aprendizado como 0,001 para obtenção dos melhores resultados (Tieleman; Hinton, 2012).

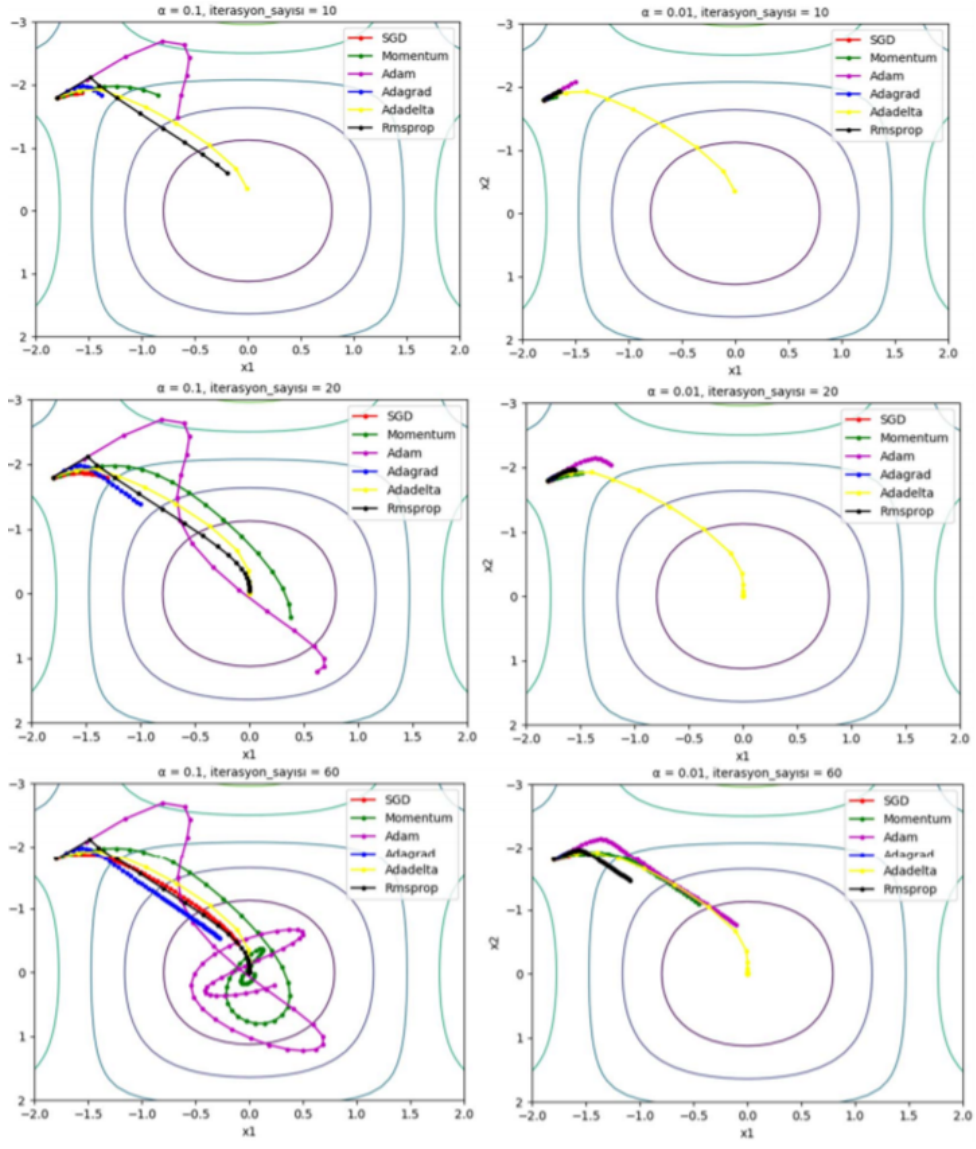
- **Adam**: O método *Adaptive Moment Estimation* (Adam) é um dos métodos que atualiza o coeficiente de aprendizado a cada iteração. Ele não só leva em consideração os quadrados dos gradientes anteriores, como nos métodos Adadelta e RMSProp, mas também leva em consideração os gradientes anteriores não quadráticos, como *Momentum*. Este método foi desenvolvido para combinar as vantagens dos métodos Adagrad e RMSProp. No estudo, foi afirmado que este método reage como uma atualização com maior precisão do método Adagrad. A diferença entre esse método e o RMSProp é que, enquanto RMSProp realiza a atualização do parâmetro usando um momento no gradiente escalado, neste método, as atualizações são expressas como a realização do gradiente usando as médias do primeiro e do segundo momento (Reddi; Kale; Kumar, 2019).

A Figura 13 expressa o comportamento dos gradientes apresentados nesta seção na obtenção do ótimo global para a função de *benchmark* Griewank (Locatelli, 2003).

3.2.2 Função de Ativação

Frequentemente referidos na literatura como função de transferência, as Funções de Ativação (AFs) são funções usadas em redes neurais são definidas sobre a soma ponderada de entradas e vieses. Essas funções são responsáveis também por analisar e manipulando os dados apresentados pelo gradiente descendente, classificar o grau de relevância dessa informação para a rede neural e definir se essa informação irá ativar ou não o neurônio que

Figura 13 – Gráfico comparativo da convergência entre os método, SGD, *Momentum*, Adam, Adagrad, Adadelta e Rmsprop sob a função Griewank, coeficiente de aprendizado α variando entre 0,1 (100%) e 0,01 (10%) e número de iterações (*iterasyon_sayisi*) variando entre 10, 20 e 60.



Fonte:(Yazan; Talu, 2017)

a possui, ignorando ou repassando essa informação ao longo da rede (Figura 12) (Nwankpa et al., 2018). Nesta seção destacam-se os tipos de AFs:

- **Função sigmoide (*sig*)**: A sigmoide, referida como a função logística ou função de esmagamento, é uma AF real, não linear, diferenciável limitada, usado principalmente em redes neurais *feedforward* (Turian; Bergstra; Bengio, 2009). Nela são definidas valores reais de entrada, com derivadas positivas em todos os lugares. A função sigmoide é representada pela equação

$$f(x) = \frac{1}{1 + e^{-x}}. \tag{3.2}$$

  A função sigmoide é usada para prever a saída baseada em probabilidade e foi aplicada com sucesso em problemas de classificação binária, modelagem de tarefas de regressão logística, bem como outros domínios de rede neural. Suas principais vantagens são fácil entendimento e rápida convergência em redes rasas (Glorot; Bengio, 2010). No entanto, suas principais desvantagens são a saturação de gradiente, a convergência lenta em redes neurais profundas e os valores de saída com centro diferente de zero, que faz as atualizações de gradiente se propagarem em diferentes direções. Outras formas de AFs foram propostas para remediar algumas dessas desvantagens sofridas pela FA sigmóide (Goodfellow et al., 2016).

- **Função tangente hiperbólica (*tanh*)**: A função tangente hiperbólica é outro tipo de AF usado em RN em aplicações. A função tangente hiperbólica conhecida como função *tanh*, é uma função centrada em zero, mais suave que a *sig*, cujo intervalo está entre -1 e 1, expressa pela equação

$$f(x) = \left(\frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \right). \tag{3.3}$$

  A função *tanh* tornou-se preferida em comparação com a função sigmóide, pois oferece melhor desempenho durante o treinamento mesmo em redes neurais multicamadas. A principal vantagem fornecida pela função é que ela produz saída centrada em zero, auxiliando assim o processo de retropropagação das redes neurais (Alom et al., 2018).

  Uma propriedade da função *tanh* é que ela só pode atingir um gradiente de 1 quando o valor da entrada for 0, ou seja, quando x for zero. Isso faz com que a função *tanh* produza pesos de ativação que resulta no valor zero da função gradiente, deixando neurônios mortos durante o processamento. Essa limitação na função *tanh* estimulou mais pesquisas em funções de ativação para resolver o problema, e deu origem à função de ativação da unidade linear retificada (ReLU). Porém a função *tanh* ainda tem sido muito usada principalmente em redes neurais recorrentes para

processamento de linguagem natural e tarefas de reconhecimento de voz (Nwankpa et al., 2018).

- **Softmax Function($softmax$)** : É uma função de ativação usada em computação neural para calcular a distribuição de probabilidade de um vetor de números reais. A função Softmax produz uma saída compreendida entre 0 e 1, com a soma das probabilidades sendo igual a 1. A função Softmax é calculada usando a relação

$$f(x_i) = \left(\frac{e^{x_i}}{\sum_j e^{x_j}} \right). \tag{3.4}$$

  A função Softmax é usada em modelos de classificação multiclasses, onde retorna as probabilidades de cada classe, com a classe alvo tendo a maior probabilidade. A função Softmax aparece principalmente em quase todas as camadas de saída nas arquiteturas de redes neurais profundas principalmente para tarefa de reconhecimento de imagens, onde são mais comumente utilizadas (Goodfellow et al., 2016).

  A principal diferença entre as funções de ativação Sigmoid e o Softmax é que a *sig* é usado na classificação binária, enquanto o $softmax$ é usado para tarefas de classificação multivariada (Nwankpa et al., 2018).

- **Função de Unidade Linear Retificada ($ReLU$)**: A função de ativação da unidade linear retificada (ReLU) foi proposta em (Nair; Hinton, 2010) e, desde então, tem sido a função de ativação mais amplamente usada para aplicações de aprendizado, obtendo os melhores resultados até hoje.

  A ReLU é uma AF para aprendizado rápido e eficiente provando ser a função mais bem-sucedida e amplamente utilizada atualmente. Ela oferece o melhor desempenho e generalização no aprendizado profundo em comparação com as funções de ativação *sig* e *tanh*. O ReLU representa uma função quase linear e, portanto, preserva as propriedades dos modelos lineares que os tornam fáceis de otimizar os métodos de gradiente descendente (Tang et al., 2017).

  A função de ativação ReLU realiza uma operação de limite para cada elemento de entrada onde os valores menores que zero são definidos como zero segundo a expressão

$$f(x) = max(0, x) = \begin{cases} x_i, & \text{se } x_i \geq 0 \\ 0, & \text{se } x_i < 0. \end{cases} \tag{3.5}$$

  Esta função corrige os valores das entradas menores que zero, forçando-os a zero e eliminando problema na função de gradiente, o que não é uma preocupação das AFs anteriores. A função ReLU tem sido usada dentro das camadas ocultas das redes neurais profundas as vezes acompanhado com outro AF usado nas camadas

de saída da rede principalmente em problemas de classificação e reconhecimento de voz (Agarap, 2018).

A principal vantagem de usar a *ReLU* em rede neurais está no fato de essa técnica não realizar cálculos exponenciais ou divisões, obtendo assim menor custo computacional em relação às demais. Outra propriedade do ReLU é que ele introduz esparsidade nas unidades ocultas à medida que espreme os valores entre zero e o máximo. No entanto, o ReLU tem uma limitação de superajuste (do inglês, *overfit*) em comparação com a função *sig*, embora a técnica de *dropout* venham sendo adotada para reduzir o efeito de *overfit* da ReLU (Tachibana; Otsuka, Sep. 2018).

O ReLU e suas variantes têm sido usados em diferentes arquiteturas de aprendizado profundo, que incluem as máquinas de Boltzmann restritas e as arquiteturas de rede neural convolucional, embora (Nair; Hinton, 2010) destaque que o ReLU tem sido usado em várias arquiteturas devido à sua simplicidade e confiabilidade (Tang et al., 2017).

O ReLU tem uma limitação significativa de que às vezes é frágil durante o treinamento, fazendo com que alguns dos gradientes morram. Isso leva à morte de alguns neurônios também, fazendo com que as atualizações de peso não sejam ativadas em pontos de dados futuros, dificultando o aprendizado, pois os neurônios mortos fornecem ativação zero (Cao; Gu, 2020). Para resolver esses problemas técnicas derivadas como *Leaky ReLU*, *Parametric ReLU*, *Randomized Leaky ReLU* e *S-shaped ReLU* foram propostas tentando reduzir a quantidade de neurônios mortos durante o treinamento do modelo (Nwankpa et al., 2018).

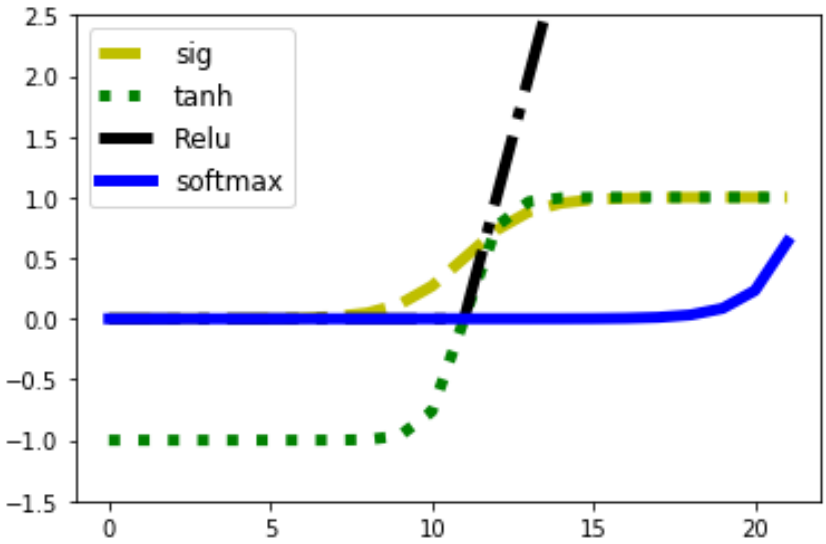
- **ELiSH**: O AF de *Squashing Exponencial linear* conhecido como função ELiSH é um dos AF mais recentes, proposto por Basirat e Roth, 2018. A função ELiSH é uma atualização da Sigmoid, fornecida pela expressão

$$f(x) = max(0, x) = \begin{cases} \frac{x}{1+e^{-x}}, & \text{se } x \geq 0 \\ \frac{e^{x}-1}{1+e^{-x}}, & \text{se } x < 0. \end{cases} \tag{3.6}$$

  As propriedades da função ELiSH variam nas partes negativas e positivas, conforme definido pelos limites. A parte sigmóide da função ELiSH melhora o fluxo de informações, enquanto as partes lineares eliminam os problemas nos gradientes. A função ELiSH foi aplicada com sucesso em redes neurais profundas no conjunto de dados ImageNet usando diferentes arquiteturas convolucionais (Basirat; Roth, 2018).

A Figura 14 apresenta o comportamento das principais funções apresentadas nesta subseção.

Figura 14 – Gráfico comparativo das funções de ativações Sigmoid(sig), Tangente hiperbólica ($tanh$), $Relu$ e $softmax$.



Fonte: O autor

3.2.3 Taxa de aprendizado

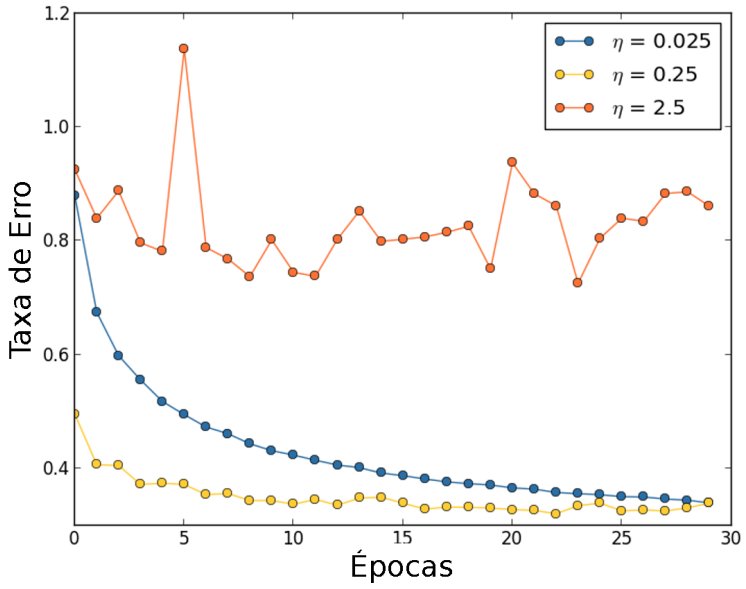
A taxa de aprendizado pode ser definida como o passo dado por uma rede neural em direção à busca de sua convergência ideal. Esse passo indica o ritmo com que os pesos dos neurônios serão atualizados durante a fase de treinamento da rede (Mikolov et al., 2011).

A cada interação o vetor do gradiente descendente é multiplicado por esse escalar para determinar a direção que será assumida ao próximo ponto, buscando minimizar o erro entre os dados reais e os estimados. Conforme se aplica essa taxa, o aprendizado da rede neural pode assumir posturas diferentes em direção a minimização do erro de estimativa (Jacobs, 1988).

A primeira é uma "descida" mais íngreme, ou seja, os valores dos pesos de cada neurônio são modificados mais rapidamente utilizando uma alta taxa de aprendizado. A segunda é uma descida suavizada onde os pesos dos neurônios serão modificados mais lentamente ao logo de cada interação (Jacobs, 1988).

A aleatoriedade na busca do mínimo erro global faz com que valores muito pequenos para a taxa de aprendizado torne o processo de aprendizado mais demorado, porém, taxas grandes demais, pode fazer com que a rede nunca encontre o melhor peso referente ao menor erro (Mikolov et al., 2011). A Figura 15 apresenta um comparativo entre a evolução do erro de estimativa para diferentes valores da taxa de aprendizado (η).

Figura 15 – Gráfico do comportamento da taxa de erro durante 30 épocas de treinamento para as taxas de aprendizado $\eta = 0.025$, $\eta = 0.25$ e $\eta = 2.5$, sob os mesmos hiperparametros.



Fonte: (Goodfellow; Bengio; Courville, 2016)

Na Figura 15 com o valor de $\eta = 0.025$, o erro no treinamento diminui lentamente até a época final. Em $\eta = 0.25$ o mínimo erro global é encontrado rapidamente com pouco mais de 20 épocas podendo gerar o *overfit* do modelo durante as épocas subsequente ao tempo de convergência ótimo. No entanto, a grande oscilação dos passos de busca da taxa $\eta = 2.5$ faz com que a descida estocástica do gradiente responsável por levar gradualmente a função de minimização a um vale de valores ótimos não convirja. (Goodfellow; Bengio; Courville, 2016). Deste modo, a definição da taxa de aprendizado ideal pode ser decisiva para a conversão rápida e salutar do modelo.

3.2.4 Tamanho do Lote de Treinamento

O aprendizado do gradiente descendente pode ser feito usando o método em lote único (do inglês, *batch*) ou o método on-line. No treinamento em lote único, as alterações nos pesos dos neurônios são acumuladas com base no conjunto total de aprendizado durante uma época de treinamento, enquanto o treinamento on-line atualiza os pesos após a inserção de cada exemplo de treinamento (instância) (Zhang et al., 2019).

Uma alternativa intermediária entre os métodos apresentados é a subdivisão do lote de treinamento denominado de *mini-batch*. Neste modelo as alterações de peso são acumuladas ao longo de um número predefinido de instâncias antes de realmente atualizar os pesos do neurônio, utilizando a taxa de aprendizado para ajustar o tamanho dessas alterações (Wilson; Martinez, 2003).

O treinamento em *mini-batch* é utilizado principalmente quando o uso de um lote único se torna impraticável como em casos onde o conjunto de treinamento é extenso, gerando tempo elevado para o treinamento, principalmente quando é necessário o uso de taxa de aprendizado pequenas para se aprender toda base (Zhang et al., 2019).

O uso de *mini-batch* também resolve o problema de sobajuste (do inglês, *underfit*) gerado pelos modelos de treinamento on-line. Esse problema é causado devido à inserção de uma instância por época, fazendo com que a rede neural seja norteada por exemplos, muitas vezes, pouco representativos do conjunto de treinamento total (Wilson; Martinez, 2003).

Por estes motivos, a definição do tamanho do lote se torna um parâmetro crítico para a rede neural que utiliza esses dados como guia para seus modelos de previsão. Embora esses lotes sirvam apenas para nortear a busca da rede em cada interação do processo de aprendizado, lotes muito pequenos causarão erros expressivos na estimativa gerada pelo gradiente, por outro lado, tamanhos excessivos podem sobrecarregar o hardware utilizado e requisitar muito tempo para uma atualização do modelo no treinamento (Goodfellow; Bengio; Courville, 2016).

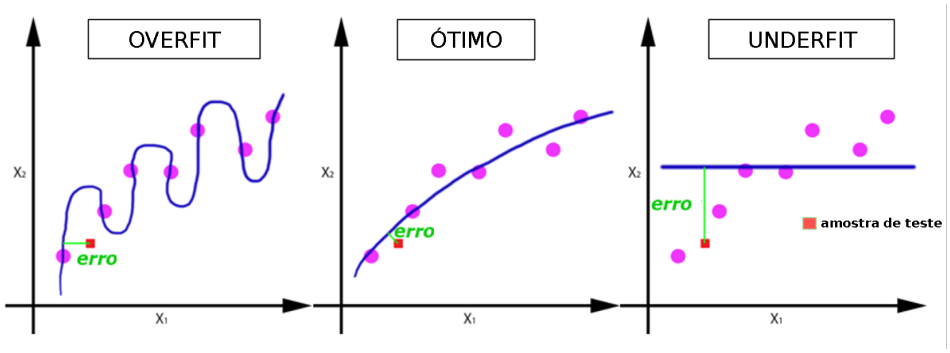
O tamanho do *batch* deve ser determinado de acordo com a complexidade do problema e o tamanho do conjunto de treinamento, através de uma análise do trade-off entre velocidade/qualidade de convergência e eficiência computacional. Pois, a divisão em mini-bath permite ajustar o tamanho dos dados de entrada na rede neural de acordo com a memória do computador, mantendo a vetorização durante as épocas (Goodfellow; Bengio; Courville, 2016).

3.2.5 Épocas de Treinamento

Entre todos os parâmetros a época de treinamento desempenha um papel significativo no desempenho final da rede neural. A época é responsável por definir o tempo que a rede neural analisará o conjunto de vetores de treinamento (entrada/alvo) para o cálculo de novos pesos e vieses (Panchal et al., 2011).

Esse tempo de treinamento deve ter um valor ótimo de forma a ajustar com eficiência a rede, pois, épocas pequenas não serão capazes de treinar a rede completamente ocasionando a não convergência (*underfit*) porém, se o tempo de treinamento for muito superior ao necessário a rede neural perderá sua capacidade de generalização (*underfit*), necessária para a previsão de novas amostras não apresentadas ao modelo. A Figura 16 apresenta um comparativo entre uma previsão realizada por uma rede neural sobajustada (*underfit*), uma sobreajustada (*overfit*) e uma com época de treinamento ótima (Sinha et al., 2010).

Figura 16 – Gráfico comparativo entre redes neurais sob diferente treinamento



Fonte: O autor

Em geral, a determinação da quantidade de épocas necessárias para a convergência é baseada em processos empíricos, de acerto e erro, durante o treinamento. Porém, em problemas que se utilizam de bases de dados grandes onde o treinamento em *mini-bach* não é possível, pode-se fazer uso de técnicas de definição do número de treinamento ótimo, a exemplo da técnica apresentada em (Sinha et al., 2010) que utiliza mapas auto organizados para encontrar a época de treinamento ótima e o trabalho de (Panchal et al., 2011) que propõe regras e etapas baseadas em processos empíricos para a definição adequada da duração do treinamento.

3.3 RESUMO DO CAPÍTULO

Este capítulo apresentou um breve levantamento das técnicas atuais de aprendizado de máquina fornecendo uma visão geral dos principais algoritmos utilizados em problemas previsão como classificação e regressões.

Entre os algoritmos discutidos, as Redes Neurais Artificiais se mostraram em evidência, ganhando muita popularidade e amplo uso em vários campos da ciência da computação. Contudo, a qualidade das soluções obtidas pela RNA está diretamente ligada a definição de parâmetros e hiperparâmetro como algoritmos de descida do gradiente, funções de ativação dos neurônios, taxa de aprendizado, tamanho do lote de treinamento e épocas de treinamento.

Neste estudo foi apresentado cinco diferentes algoritmos de descida do gradiente avaliando as vantagens e desvantagens dos métodos com função de custo exibido graficamente. Como resultado, observou-se que o método Adam apresenta vantagens quando comparado a outros métodos usando as informações de gradiente anteriores a uma certa taxa para reduzir as dificuldades em determinar o coeficiente de aprendizado e recurso para uma convergência rápida.

Quanto às funções de ativação, foram comparados os modelos sigmoide, tangente hiperbólica, Softmax, *ReLU*, e ELiSH. Entre as AFs abordadas destaca-se a *ReLU* com

aprendizado mais rápido e menor custo computacional em comparação as demais. Isso se deve ao fato da não realização de cálculos exponenciais ou divisões durante o processo de treinamento.

Este capítulo também apresentou um breve discussão sobre a importância da definição correta de parâmetros como taxa de aprendizado, tamanho do lote de treinamento e épocas de treinamento, e sua influência no aprendizado da Rede Neural Artificial. Essas variáveis se mostram decisivas para um treinamento eficiente do modelo, podendo evitar problemas como *underfit* e *overfit*.

4 ESTADO DA ARTE

> *Aqueles que se sentem satisfeitos sentam-se e nada fazem. Os insatisfeitos são os únicos benfeitores do mundo.*
>
> Walter S. Landor,

Em redes sem fio, o gerenciamento de variáveis como níveis de potência, taxa de transmissão ideal, tamanho do pacote de dados, frequência da portadora, coeficiente de atenuação, distância entre nós e densidade espectral de ruído, por exemplo, são variáveis críticas para o controle eficiente de energia, especialmente em dispositivos minúsculos (Luong et al., 2019).

Diante disto, este capítulo busca apresentar as técnicas de otimização de eficiência energética que opera em diferentes protocolos IoTs encontradas no estado-da-arte da literatura descrevendo as principais contribuições realizadas por elas.

4.1 TÉCNICAS DE EFICIÊNCIA ENERGÉTICA PARA SISTEMAS WSN/IOTS

Inúmeras soluções são sugeridas na literatura para redução do desperdício de energia. A seguir estão listadas maneiras de reduzir o consumo de energia em IoT.

4.1.1 Controle de Atividade de Sensores

Esta solução consiste na utilização de diferentes tipos de sensores para controle de uma mesma aplicação para torná-la mais eficiente. Se um sensor está controlando a aplicação, então outro sensor permanece em sono profundo podendo ser programado para despertar em algum tempo específico de forma que a aplicação sempre se mantenha ativa mesmo com a morte gradual dos sensores da rede. Para isso são utilizadas técnicas de escalonamento de dispositivos (Singh; Chana, 2019).

Em (Abedin et al., 2015) é exibido um algoritmo que usa três estágios diferentes para agendar sensores. Os sensores em estágio de serviço acordam programados para um funcionamento com capacidade total e quando estão ociosos, por um determinado tempo, entram no estágio de pré-desativação onde escutam o canal apenas para recebimento e transmissão de comandos essenciais. O último estágio é definido como *off* fazendo com que alguns sensores, dependendo do seu nível de atividade na rede, entrem em estados de hibernação, suspensão e desligamento, fazendo com que economize o máximo de energia.

O grande problema da técnica supracitada é a inabilidade de realizar controles de carga nos dispositivos fazendo com que nós sobrecarregados morram rapidamente enquanto os demais estejam em *off*, isso faz com que a rede possa perder comunicação até que um nó chave volte ao estado de escuta do canal.

Em (Wang et al., 2016) foi proposto um mecanismo de sono e despertar utilizando um equilíbrio de consumo baseado na restrição do tráfego em cada dispositivo. A melhoria no controle energético se torna mais eficiente pois o mecanismo de agendamento de sono é introduzido tornando os dispositivos inativos temporariamente sempre que houver uma sobrecarga de tráfego naquele nó. Isso faz com que a rede de sensores seja obrigada a redirecionar seu tráfego para nós mais ociosos da rede.

O grande problema das técnicas de controle de atividades são as interrupções abruptas realizadas no tráfego e a inabilidade de gerir recursos de forma eficiente, fazendo com que seja possível a interrupção na comunicação durante um processo de transmissão sem a necessidade de uma avaliação ou checagem da possibilidade de restruturação do tráfego da rede.

4.1.2 Transferência Periódica de Dados

Técnicas de controle estratégico de transferência de dados buscam promover de forma inteligente um controle no tráfego através de monitoramento remoto de sistemas IoT (Aakash et al., 2018).

O envio de dados continuamente na nuvem consome muita energia principalmente no *gateway*. Quando aplicações não requisitam monitoramento nem transferências contínua de informações, a técnica de transferência periódica de dados pode se tornar uma solução eficiente para a economia de energia. Nela os dados são enviados na nuvem periodicamente, ou seja, em um intervalo de tempo fixo, consequentemente, o consumo de energia dos dispositivos é reduzido durante o tempo em que nenhum dado é enviado na nuvem (Singh; Chana, 2019).

Em (Aakash et al., 2018) é proposta uma abordagem de *Data Mining* (DM) utilizando a técnica de aprendizado de máquina SVM para sensoriamento em tempo real. Nesta abordagem os dados são enviados por sensores sem fio de tecnologia ZigBee, pré-processados em nós individuais de *fog computing*, acumulados e enviados em forma de *big data*. Os dados coletado são posteriormente utilizados pela SVM para previsões de áreas sujeitas ou futuramente afetadas por incêndios florestais repentinos.

O grande problema de técnicas baseada em transferência periódica de dados é a incapacidade de monitoramento em tempo real em casos de eventos não programados. Isso se deve a falta de uma maior autonomia dos sensores na rede.

4.1.3 Filtragem de Dados

Dados detectados por dispositivos IoT criam muito tráfego na rede, enquanto registros duplicados que misturam os dados redundantes e os essenciais aumentam o tamanho do

armazenamento em nuvem (Elbanoby; Aborizka; Maghraby, 2019).

Técnicas de filtragem de dados podem ser úteis para uma eficiência energética entre esses dispositivos, pois, menos dados produzidos geram redução de transmissão, podendo economizar energia principalmente em *hardwares* envolvidos, como o transceptor (OLIVEIRA et al., 2020).

Essas técnicas podem ser implementadas em diferentes estágios da rede IoT, permitindo que os dados detectados possam ser pré-analisados após o estágio de detecção no próprio dispositivo (SANTOS; MINI, 2021). Além disso, dispositivos pré-programáveis podem garantir filtragem dos dados coletados, o que ajuda a decidir quais dados são redundantes ou ruidosos e, consequentemente, filtra esses dados antes do seu envio para a nuvem. Portanto, o tamanho dos dados filtrados enviados é menor em comparação aos envios originais requirindo assim menos energia de dispositivos transceptores. Além disso, ele economiza energia no armazenamento, processando e enviando a nuvem apenas dados essenciais (OLIVEIRA et al., 2020).

Em (Elbanoby; Aborizka; Maghraby, 2019), é proposta uma estrutura de tomada de decisão dos dados recebidos em tempo real para filtrar e fundir os dados utilizando um mecanismo de controle de *feedback* (FCM) na camada de rede. A estrutura proposta reduz o número de registros de IoT. Além disso, evita muito tráfego na rede, aumenta a vida útil de todos dispositivos envolvidos, e minimiza o consumo de energia. A abordagem proposta visa ajudar a aumentar o desempenho e reduz o custo em setores industriais. Os resultados apresentados neste trabalho mostraram uma redução de 82,34% na transferência de dados duplicados, melhorando o consumo de energia no tráfego na camada de rede, gerando dados com menos ruídos.

O grande problema de técnicas de filtragem de dados é o custo computacional, de memórias RAM e ROM, utilizado na realização de todas as pré-análises dos dados. Essa checagem força o dispositivo sensor a armazenar e verificar os dados recolhidos e enviados previamente. Esse problema pode ocasionar sobrecarga em dispositivos IoTs que não possuem arquiteturas focadas em processamento e armazenamento de grande número de dados (JUNIOR et al., 2022).

4.1.4 Programação Eficiente

Realizar uma programação eficiente em dispositivos IoT desempenha um papel importante na economia de energia pois conseguem otimizar a programação do dispositivo desde o hardware do microcontrolador do IoT. Com base na otimização da codificação do software, eles podem economizar energia predefinindo modelos de eficiência energética, protocolos de interrupção, sono-despertar e assim por diante (Abedin et al., 2015).

Modelos matemáticos e algoritmos baseados em funções lineares podem ser utilizados como estrategia de para programação eficiente por, em sua maioria, não requisitarem de

um alto consumo computacional de memória e fluxo continuo de processamento, podendo ser executados rapidamente em dispositivos de baixa complexidades como WSN/IoT (Singh; Chana, 2019).

O trabalho proposto em (da Silva; de Moraes, 2020), por exemplo, é um modelo matemático de otimização energética de abordagem *crosslayer* que garante uma vida operacional mais longa a dispositivos de rede ad hoc que empregam *slotted* Aloha e CSMA/CA como protocolos MAC de comunicação. Para isto, o trabalho obtém a potência de transmissão ideal em função de múltiplos parâmetros das camadas físicas e de enlace dos dispositivos, como taxa de transmissão e tamanho do pacote, resultando em um consumo de energia otimizado por bit transmitido com sucesso.

Para isso, o trabalho atualiza estudos da abordagem da comunicação proposto por (Zhang; Gorce, Mar. 2008), e revisa as formulações dos protocolos acrescentando ao modelo formulações de otimizações contidas nos trabalhos (Ghobad; de Moraes, Sep. 2012; Zhang; Gorce, Mar. 2008; Gao, 2002; Gorce; Zhang; Parvery, 2007; Panichpapiboon; Ferrari; Tonguz, May. 2004), tornando o modelo proposto em (da Silva; de Moraes, 2020) robusto para otimização energética.

Apesar desse modelo apresentar excelentes resultados, foram aplicados, simulados e observados somente através de computações numéricas e análises teóricas. A implementação em sistemas reais ainda enfrenta muitos obstáculos em particular pela inabilidade desse e de outros modelos em lidar com a dinamicidade da rede, que requisita de uma otimização dos parâmetros em tempo real de execução (Barbosa et al., 2019).

4.1.5 Análise e Processamento de Dados em Tempo Real

Embora o uso de sistemas IoT e sensores estejam diretamente ligados ao acesso a dados em tempo real, o processamento e a análise desses dados localmente pode consumir muita energia e tempo (Ferreira; Nogueira; Macedo, 2019).

Esse problema torna técnicas de tomada de decisão necessárias para que os dispositivos sejam capazes de avaliar quando se torna necessária o processamento dos dados. Com base nessas técnicas, os dados recolhidos em tempo real podem ser processados e/ou enviados a um *gateway* ou para a Internet após semanas ou meses, a depender do dispositivo e aplicação de monitoramento. Com isso, os dispositivos IoT do tipo sensores podem obter dados e tomar as ações adequadas, apenas após a observância dos padrões predefinidos pelas aplicações. Essa análise em tempo não real economiza energia dos dispositivos em tempo real (Bibri, 2018).

Em (Islam et al., 2019) é proposto um sistema de processamento de dados em tempo real específico para redes com dispositivos *low process* e grande fluxo de dados, a exemplo de redes IoT de *streaming*. O sistema proposto neste trabalho, avalia os dados recebidos, e quando necessário, utiliza processamento paralelo de dados, subdividindo a carga de

trabalho entre os dispositivos da rede através do Intel Software Guard Extensions (SGX), uma plataforma segura que processa os dados em um curto espaço de tempo. Esse sistema conta com dois módulos de construção para a análise de dados em tempo real.

No primeiro módulo, é proposto uma subdivisão de tarefas com criptografia de dados ponta-a-ponta e sistema de *Trusted Execution Environment* (TEE), garantindo que o código e os dados processados sejam protegidos, mantendo a confidencialidade e integridade dos dados, mesmo com a coleta de dados em tempo real sendo obtida de vários dispositivos de IoT em paralelo.

O segundo modulo propõe uma estrutura genérica baseada em aprendizado de máquina para simplificar o processo de coleta, processamento e armazenamento de imagens provenientes de diversos dispositivos IoT drones do tipo VANETS (*Vehicular Ad Hoc Networks*). Para isso, utiliza o modelo de aprendizado profundo em miniatura para detecção de objetos denominado *nano yolo* que realiza processamento de imagem de baixa resolução com reduzido custo computacional.

4.2 RESUMO DO CAPÍTULO

Este capítulo apresentou diferentes abordagens e técnicas que atuam diretamente ou indiretamente na otimização da eficiência energética em dispositivos IoT. Mostrando as principais soluções para sistemas de economia de energia como técnicas de resolução por controle de atividade de sensoriamento, controle de envios de dados, filtragem de informações, otimização de codificação de programas e APIs e análise e processamento de dados em tempo real.

Neste capítulo foi possível apresentar técnicas como a proposta por (da Silva; de Moraes, 2020), modelo matemático que utiliza funções matemáticas para prolongar a vida operacional de dispositivos que operam sob protocolos como *slotted* Aloha e CSMA/CA. Porém, apesar do modelo apresentar excelentes resultados teóricos, este não consegue lidar com a dinamicidade da rede, que exige uma otimização dos parâmetros envolvidos em tempo real.

Com este estudo foi possível perceber a necessidade de protocolos inteligentes que avaliem os cenários de rede reais e garantam maior autonomia do dispositivo, redução de transmissões e processamento envolvendo atividade de controle.

Diante de todos os desafios ainda enfrentados pelo estado da arte, esta tese de doutorado apresentará um modelo inteligente baseado em aprendizado de máquina que agrega às boas práticas apresentada neste capítulo, em especial as de programação eficiente e processamento em tempo real, fazendo com que dispositivos sejam capazes de avaliar o cenário de rede e tomar as ações adequadas em tempo real de forma inteligente com o intuito de reduzir o consumo energético envolvido nas transmissões. Para isso será levado em consideração especificações e limitações de hardware e de software de dispositivos IoTs

simples como os utilizados em WSN.

5 REGRESSÃO DE MCE VIA RNA

“ O começo de todas as ciências é o espanto de as coisas serem o que são. ”

Aristóteles,

Em redes sem fio, o gerenciamento de variáveis como níveis de potência, taxa de transmissão, tamanho do pacote de dados, frequência da portadora, coeficiente de atenuação, distância entre nós e densidade espectral de ruído, são variáveis críticas para o controle eficiente de energia, especialmente se em dispositivos limitados (Luong et al., 2019).

A literatura apresenta diferentes modelos e algoritmos para otimizar o uso de recursos em redes sem fio (Sun et al., Jul. 2017). Um número significativo dessas técnicas é composto por algoritmos iterativos alimentados por parâmetros de rede obtidos em tempo real. Esses dados são processados e avaliados, através de funções potencialmente custosas, para a formação de estratégias de otimização dos recursos da rede (Mao; Hu; Hao, 2018).

Embora o uso algoritmos que operam dentro de diferentes protocolos de rede apresentem excelentes resultados, sua execução em sistemas reais ainda enfrentam muitos obstáculos, associados ao seu alto custo de processamento. Isso é mais evidente em redes extensas compostas por dispositivos com limitações de hardware, como WSN (Sinche et al., 2020).

Encontrar as configurações que reduzam o consumo de energia em diferentes cenários de transmissão, como proposto em (da Silva; de Moraes, 2020), por exemplo, requer processos computacionalmente caros por lidar com de múltiplas variáveis simultaneamente. Embora essa abordagem busque otimizar vários parâmetros usando um modelo matemático linear, ela ainda precisa de algoritmos de pesquisa exaustivos para cruzar todos os parâmetros envolvidos na transmissão e identificar as configurações ideais de transmissão para cada cenário específico. Porém, operações como essa em redes reais devem ser realizadas em milissegundos devido aos parâmetros do sistema, como condições do canal, número de usuários, que mudam a cada transmissão, etc.

Neste contexto, propomos como primeira parte dessa tese a criação de uma nova técnica, baseado em redes neurais artificiais, que visa aprender as relações de entrada / saída de diferentes modelos matemáticos utilizados para gerenciamento de controle energético, como uma “caixa preta” para aprender os critérios de otimização e as relações que resultam na redução do consumo de energia (Sun et al., Jul. 2017).

A partir de uma compreensão do funcionamento desses modelos matemáticos, considerando diferentes dispositivos e cenários, a abordagem proposta pode adaptar sua resposta ao dispositivo em uso, identificando o protocolo da camada MAC, ALOHA ou CSMA/CA, e fazer o dispositivo operar com a melhor configuração possível. Nosso objetivo é substituir modelos de otimização tradicionais de gerenciamento de recursos de energia por um modelo inteligente que atue em ambos protocolos. Diante disto a parte 1 deste trabalho apresenta as seguintes contribuições:

- Uma nova técnica que combina informações de modelos matemáticos robustos utilizados para controle de consumo de energia em um modelo inteligente que funciona em redes CSMA/CA ou ALOHA com *slots*.
- Uma nova abordagem de economia de energia baseada em aprendizado de máquina considerando múltiplas variáveis de camadas físicas (PHY) e camada de enlace (MAC).
- Apresentar ganhos de desempenho em relação ao tempo de processamento, custo computacional e autoadaptação de parâmetros (PHY/MAC) considerando métodos clássicos.
- Um modelo robusto avaliado através de extensos experimentos numéricos simulados a partir de parâmetros reais de rede.

O restante deste Capítulo se destina a descrever o método e a técnica propostos e suas etapas de operação.

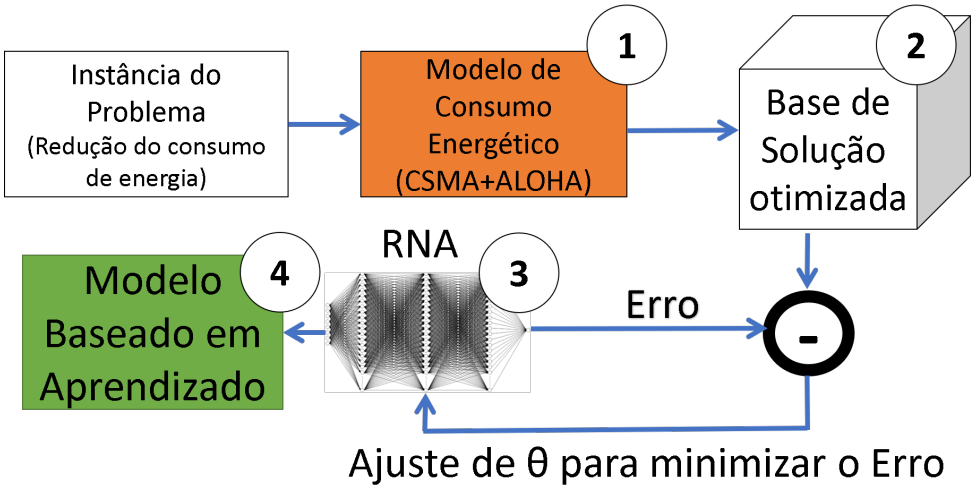
5.1 UMA ABORDAGEM DE APRENDIZADO DE MÁQUINA PARA REDUZIR O CONSUMO DE ENERGIA

Para redução do consumo energético foram otimizados parâmetros importantes da rede (da Silva; de Moraes, 2020; SOUZA et al., 2016) como número de bits por quadro (N_b), potência de transmissão ótima (P_0), número de saltos (i) do remetente ao destino, e taxa de transmissão (R) de dados. Foi implementando um modelo que utiliza aprendizado de máquina para se adaptar instantaneamente ao cenário de transmissão atual e gerar uma configuração energética para essas variáveis que minimize o consumo energético durante cada transmissão.

A abordagem proposta compreende uma sequência de quatro etapas, conforme é mostrado na Figura 5.1. Primeiramente, dois modelos de consumo de energia, um para CSMA/CA e outro para *slotted* ALOHA, são escolhidos para servir de instância (base) do problema a ser aprendido pela Rede neural artificial. Na segunda etapa são usados esses modelos matemáticos para gerar soluções otimizadas como exemplos de treinamento para a RNA. A etapa três consiste em encontrar e ajustar os parâmetros de treinamento ideais que criam o modelo proposto (*etapa* 4) para aproximar as soluções de treinamento.

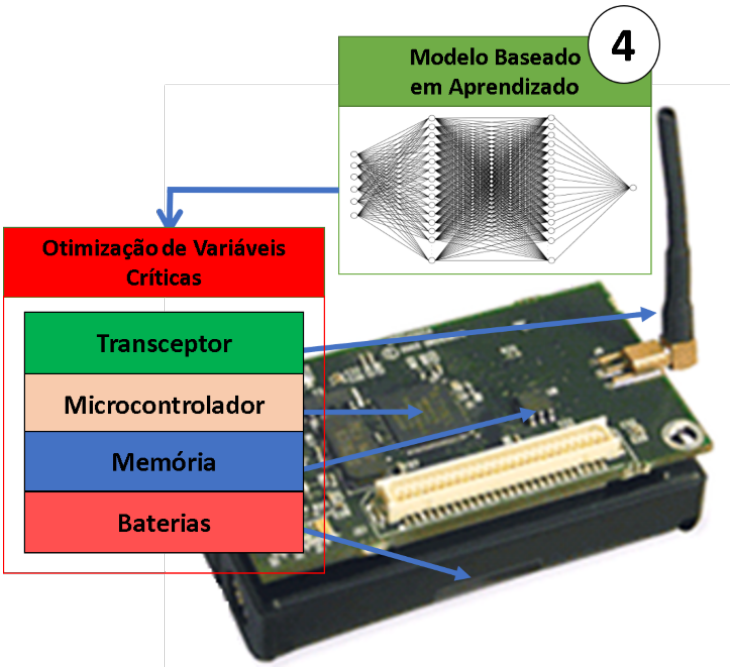
O modelo proposto, uma vez inserido em um dispositivo IoT, identificará os recursos de hardware e protocolos disponíveis e otimizará inúmeros parâmetros envolvidos no processo de transmissão e consumo de energia deste dispositivo (Figura 18). As Subseções 5.1.1, 5.1.2 e 5.1.3 detalham essas etapas.

Figura 17 – Etapas de operação do método baseado em aprendizado de máquina: 1) Definição dos modelos de consumo de energia; 2) Geração de um conjunto de soluções otimizadas para treinamento; 3) Treinar o modelo para aproximar soluções ótimas de algoritmos clássicos; 4) Implementação do modelo.



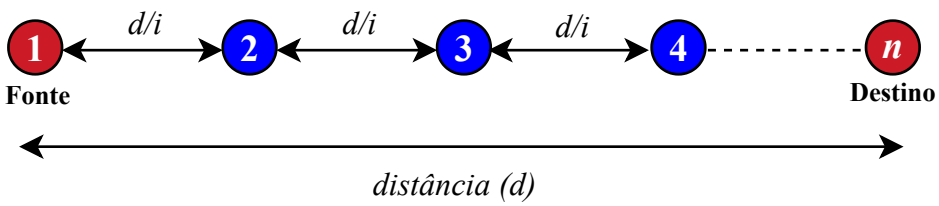
Fonte: O autor

Figura 18 – O modelo baseado no aprendizado (4) é capaz de identificar faixas de variáveis como a taxa de transmissão, número de bits por frame e potência de transmissão, disponibilizada pelo dispositivo, e criar uma configuração em tempo real com foco na economia de energia.



Fonte: O autor

Figura 19 – Exemplo de rede linear de computadores com n nós e i saltos equidistantes



Fonte: O autor

5.1.1 Modelo de Consumo de Energia (MCE)

Nesta etapa foram definidos modelos matemáticos de otimização energética capazes de otimizar simultaneamente parâmetros multiobjetivos conflitantes como potência e taxa de transmissão, tamanho do pacote de dados e a distância entre os nós para servir como base de aprendizado pela rede neural.

Usamos dois protocolos de controle de acesso ao meio para projetar nossa abordagem para reduzir o consumo de energia em redes *ad hoc slotted* Aloha e CSMA/CA. O modelo de consumo de energia proposta por (da Silva; de Moraes, 2020) foi utilizado como ponto de partida.

Para obtenção da variável distância foi definido que existe um caminho dispostos geograficamente em uma configuração linear com saltos equidistantes que torna todos os nós acessíveis. Com isso foi possível garantir um conhecimento generalista da RNA que uma vez treinada com valores de d e $d + 1$, obtém previsões de valores não treinados como $d/2$, por exemplo.

A Figura 19 ilustra um processo de transmissão que poderia representar um ramo de uma topologia de árvore ou uma linha/coluna em uma rede em grade.

O MCE proposto (da Silva; de Moraes, 2020) utiliza a probabilidade de uma transmissão bem-sucedida para um pacote de dados de tamanho N_b bits, dado por $pr_s(\gamma) = (1 - BER(\gamma))^{N_b}$ onde a taxa de erro de bit $BER(\gamma)$ está em função da relação sinal/-ruído γ proposta por (Zhang; Gorce, Mar. 2008) para aqui incluir os protocolos CSMA/CA (Tinnirello; Bianchi; Xiao, 2010) e *slotted* ALOHA (Roberts, 1975). A relação sinal-ruído é representada por $\gamma = C_2 P_t \left(\frac{d}{i}\right)^{-\alpha} R^{-1}$, com $C_2 = \frac{G_T G_R \lambda^2}{(4\pi)^2 N_0}$, onde α é o coeficiente de atenuação do sinal, P_t é a potência de transmissão e G_T e G_R os ganhos da antena durante transmissão e recepção, respectivamente. N_0 é a densidade espectral de potência de ruído do canal de comunicação, $\lambda = \frac{c}{f_c}$ é o comprimento de onda do sinal transmitido (portadora), c é a velocidade da luz e f_c é a frequência portadora.

Aqui, usamos o canal de comunicação de desvanecimento Rayleigh, pois é amplamente utilizado em redes *ad hoc* e de sensores para modelar ambientes sem fio. Consequentemente para uma relação sinal-ruído maior que 5, que é o caso dos parâmetros empregados aqui, a taxa de erro de bit pode ser aproximada por $BER(\gamma) \approx \frac{\alpha_m}{2\beta_m \gamma}$ (Goldsmith, 2005), onde

α_m e β_m são constantes que dependem do tipo de modulação empregado. Escolhemos a modulação *Binary Phase Shift Keying* (BPSK) comumente usada em dispositivos de comunicação sem fio para envio de cabeçalhos da PHY, por exemplo. Assim, $\alpha_m = 1$ e $\beta_m = 2$.

A probabilidade de uma transmissão bem-sucedida para n nós para o protocolo CSMA/CA é dada por $\zeta = \frac{n\tau(1-\tau)^{n-1}}{1-(1-\tau)^n}$, em que τ é a probabilidade de um dispositivo transmitir durante um intervalo de tempo independentemente de outras transmissões é representado em (Tinnirello; Bianchi; Xiao, 2010) pela equação

$$\tau = \frac{1}{1 + \frac{1-p}{2(1-p^{Ret+1})}\left[\sum_{j=0}^{Ret} p^j(2^jW - 1) - (1 - p^{Ret+1})\right]}, \tag{5.1}$$

em que *Ret* é o número de tentativas de retransmissão, W é o tamanho mínimo da janela de contenção, p é a probabilidade de colisão de pacotes e j representa o estágio de *backoff* j^{th}.

A probabilidade de acesso ao meio com sucesso no *slotted* ALOHA é $\delta = \frac{\left(\frac{i}{i+1}\right)^{i-1}}{i+1}$ (da Silva; de Moraes, 2020), onde i representa o número de saltos dados em função do número de nós n, com $i = n - 1$.

Assim, a probabilidade de transmissão bem-sucedida de um pacote nos protocolos *slotted* ALOHA e CSMA/CA, são dados, respectivamente, por

$$Pr_s = pr_s(\gamma)\delta = \frac{(1-BER(\gamma))^{N_b}\left(\frac{i}{i+1}\right)^{i-1}}{i+1} \tag{5.2}$$

e

$$Pr_s = pr_s(\gamma)\zeta = \frac{(1-BER(\gamma))^{N_b}\tau(i+1)(1-\tau)^i}{1-(1-\tau)^{i+1}}. \tag{5.3}$$

Este modelo não considera a interferência causada por outro nó da rede na relação sinal ruido porque o protocolo de controle de acesso ao meio trata estes casos por meio da resolução de colisão. Assim, se dois ou mais nodos transmitem simultaneamente, seus pacotes de dados colidem, e a informação é perdida, exigindo retransmissão. Esse efeito é incorporado aqui em ambos os protocolos estudados.

O consumo de energia em uma transmissão de salto único é a soma da energia consumida no transmissor (E_{Tx}) e no receptor (E_{Rx}). Assim, o consumo de energia por bit (E_b) em Joules por bit (J/bit) para um único salto é definido e fornecido por (Zhang; Gorce, Mar. 2008)

$$E_b = \frac{E_{Tx} + E_{Rx}}{N_b} = \frac{2T_{start}P_{start}}{N_b} + \frac{P_{txElec} + P_{rxElec} + \alpha_{amp} + \beta_{amp}P_t}{R}, \tag{5.4}$$

onde P_{txElec} é a potência elétrica do circuito do transmissor, P_{rxElec} é a potência elétrica do circuito do receptor, α_{amp} é um nível de potência constante e β_{amp} é uma constante de proporcionalidade da potência de transmissão P_t e T_{start} e P_{start} são, respectivamente, o tempo e a potência de inicialização do circuito.

O consumo médio de energia por bit de dados (em unidades de Joules/bits) para um único salto é dado por

$$\overline{E}_{1hop} = E_b \sum_{k=1}^{7} kPr_s(1-Pr_s)^{k-1} \approx \frac{E_b}{Pr_s}, \tag{5.5}$$

onde k é o número de transmissões necessárias para enviar com sucesso um pacote em um salto, limitado ao número máximo de 7 tentativas de retransmissão, aceito pelo protocolo CSMA/CA (Karl; Willig, 2007).

Portanto, a energia média gasta para transmitir com sucesso um bit por meio de vários saltos, desconsiderando o tempo de *back-off* do dispositivo, é $\overline{E}_{ihop} = \overline{E}_{1hop} \cdot i$. Para o CSMA/CA, a partir de Eqs. (5.3), (5.4) e (5.5), resulta que a energia média total por bit para entregar o pacote de dados por meio de i saltos entre o par de nós de comunicação (fonte e destino), é

$$\overline{E}_{ihop} = \frac{E_b}{\left(1 - \frac{\alpha_m}{2\beta_m C_2 P_t \left(\frac{d}{i}\right)^{-\alpha} R^{-1}}\right)^{N_b}} \cdot i \cdot \frac{1-(1-\tau)^{i+1}}{\tau(i+1)(1-\tau)^i}. \tag{5.6}$$

Para *slotted* ALOHA, com base nas Eqs. (5.2, 5.4 e 5.5), resulta que

$$\overline{E}_{ihop} = \frac{E_c + K_1 P_t}{\left(1 - \frac{\alpha_m}{2\beta_m C_2 P_t \left(\frac{d}{i}\right)^{-\alpha} R^{-1}}\right)^{N_b}} \cdot \frac{(i+1)^i}{i^{i-2}}. \tag{5.7}$$

Neste trabalho estas formulações são denominadas de modelos de consumo de energia (MCE).

A potência de transmissão ótima (P_0) é obtida a partir de

$$\left.\frac{\partial \overline{E}_{ihop}}{\partial P_t}\right|_{P_t = P_0} = 0, \tag{5.8}$$

com P_0 em função da taxa de transmissão de dados (R) e do tamanho do pacote de dados (N_b). Da Eq. (5.8), resulta que

$$P_0(R, N_b) = \frac{\left(\frac{d}{i}\right)^{-\alpha} \alpha_m (1+N_b)}{4C_2\beta_m R^{-1}} + \frac{\sqrt{\left(\frac{d}{i}\right)^{-\alpha} \beta_{amp}\alpha_m \left[\left(\frac{d}{i}\right)^{-\alpha} \beta_{amp}\alpha_m (1+N_b)^2 + 8\beta_m C_2 R^{-1}(C_3 R + C_4 N_b)\right]}}{4C_2\beta_{amp}\beta_m R^{-1}}. \tag{5.9}$$

C_3 e C_4 são derivados da Eq. (5.4), onde $C_3 = 2T_{start}P_{start}$, e $C_4 = P_{txelec} + P_{rxelec} + \alpha_{amp}$.

Portanto, a energia média ($\overline{E}_{ihop}$) consumida por bit de dados transmitido com sucesso da fonte ao destino pode ser dada em função de R e N_b, para i saltos, considerando a potência ótima de transmissão $P_0(R, N_b)$. Consequentemente, a partir das Eqs. (5.6) e (5.9) para CSMA/CA e das Eqs.(5.7) e (5.9) para *slotted* ALOHA, $\overline{E}_{ihop}$ é calculado, respectivamente, por

$$\overline{E}_{ihop}(P_t = P_0(R, N_b)) = \frac{C_3R+(C_4+\beta_{amp}P_0(R,N_b))N_b}{RN_b\left(1-\frac{\alpha_m}{2\beta_m C_2 P_0(R,N_b)\left(\frac{d}{i}\right)^{-\alpha}R^{-1}}\right)^{N_b}} \cdot \frac{i\left[1-(1-\tau)^{i+1}\right]}{\tau(i+1)(1-\tau)^i} \tag{5.10}$$

e

$$\overline{E}_{ihop}(P_t = P_0(R, N_b)) = \frac{C_3R+(C_4+\beta_{amp}P_0(R,N_b))N_b}{RN_b\left(1-\frac{\alpha_m}{2\beta_m C_2\left(\frac{d}{i}\right)^{-\alpha}R^{-1}P_0(R,N_b)}\right)^{N_b}} \cdot \frac{(i+1)^i}{i^{i-2}}. \tag{5.11}$$

Os parâmetros das camadas físicas e MAC, para dispositivos IoT sensores, padrão μ-AMPS-1 (Karl; Willig, 2007), são apresentados na Tabela 5.

Tabela 5 – Parâmetros PHY-MAC empregados na formação da base de dados ALOHA + CSMA (Zhang; Gorce, Mar. 2008)

Parâmetro	Descrição	Valor
P_{rxElec}	Potência do circuito de recepção	$279mW$
P_{start}	Potência para inicialização	$58,7mW$
T_{start}	Tempo de inicialização	$446\mu s$
P_{txElec}	Potência do circuito de transmissão	$151mW$
α_{amp}	Nível de potência	$174mW$
β_{amp}	Constante de proporcionalidade	5
N_0	Densidade espetral de ruído	$-154dBm/Hz$
f_c	Frequência da portadora	$2,4GHz$
c	Velocidade da luz	$3\times10^8 m/s$
G_T	Ganho da antena de transmissão	1
G_R	Ganho da antena de recepção	1
ΔR	Taxa de transmissão	50 a $10^7 bps$
ΔN_b	Número de bits por quadro	1 a $8\times10^6 bits$
$\Delta\alpha$	Coeficiente de atenuação	2 e 4
Δd	Distância entre fonte e destino	10 a $250m$
Δi	Número de saltos até o destino	$1, 2, 4, 8$

Fonte: O autor

É importante ressaltar que os valores altos do intervalo Δ foram escolhidos com o propósito de aumentar o espaço de busca da RNA pela melhor configuração de eficiência energética. Por exemplo, N_b, mesmo utilizando valores que chegaram a $8Mbit$, N_b estabilizou automaticamente com valores entre $10bit$ e $10kbit$, como serão apresentados nos resultados do capitulo 7.

5.1.2 Geração da Base de Treinamento para a RNA

O objetivo desta etapa é criar um banco de dados de treinamento combinando todas as variáveis envolvidas nas Eqs. (5.10) e (5.11), usando os parâmetros da Tabela 5 para ambos os protocolos.

Essa estratégia tenta criar soluções para diferentes cenários de transmissões capacitando a RNA encontrar soluções eficientes mesmo em cenários onde não é possível controlar essas variáveis, em especial valores de α e d. Sendo capaz de realizar todas as combinações dos intervalos (Δ) definidos na Tabela 5 para as variáveis de entrada R, N_b, α, d e i, para isso, foi necessário criar um algoritmo (Algoritmo 1) para enumerar todos os possíveis candidatos à solução do problema e assim encontrar a solução global ótima para a saída $\overline{E}_{ihop}$, esse algoritmo é denominado de busca exaustiva (LIASHCHYNSKYI; LIASHCHYNSKYI, 2019).

Com o Algoritmo 1 foi criado cenários de simulação que contemplaram todas as combinações geradas pelo cruzamento dos valores Δs descrito na Tabela 5, gerando um total de 200.000 amostras representativas para cada protocolo.

As amostras geradas serão utilizadas para treinamento da RNA na tentativa de provar que um algoritmo determinístico, com iterações de mapeamentos contínuos, também pode ter seu comportamento aprendido através de uma RNA bem treinada (Sun et al., Jul. 2017). Com isso pretende-se gerar um modelo capaz de aprender o comportamento de funções de otimização energética de diferentes protocolos simultaneamente.

Para simplificar a identificação de cada protocolo dos dispositivos e otimizar os parâmetros com maior eficiência, foi definido uma variável de controle denominada *protocolkey* (Pk). O Pk foi adicionado para permitir a RNA identificar, durante o aprendizado, sob qual protocolo a rede WSN/IoT deve operar, para isso utilizamos o identificador 0 para *slotted* Aloha e 1 para CSMA/CA. A base de dados completa está registrada em (Barbosa et al., 2020).

Algoritmo 1: Busca exaustiva das melhores configurações

Data: Δ_R, Δ_{Nb}, Δ_α, Δ_d, Δ_i ranges
Result: $NNTrainingSet$

```
k = 0;
for α ∈ Δ_α do
    for d ∈ Δ_d do
        for i ∈ Δ_i do
            j = 0;
            Ē_ihop = MAX_Value;
            for R ∈ Δ_R do
                Ē_ihop = MAX_Value;
                for Nb ∈ Δ_Nb do
                    TempĒ_ihop = Calculate_Eihop(MCE);
                    if Ē_ihop > TempĒ_ihop then
                        Ē_ihop = TempĒ_ihop
                        NNTrainingSet[k, j] ← Ē_ihop, R, Nb, α, d, i;
            j = j + 1;
        k = k + 1;
```

5.1.3 Regressão de MCE Otimizado via Aprendizado de Máquina

Esta etapa da técnica proposta tem como principal objetivo criar um modelo de regressão baseado em aprendizado de máquina.

Após estudos e testes empíricos, utilizando bibliotecas como scikit-learn, Tensorflow e Keras (GÉRON, 2022) para avaliar se as técnicas apresentadas no Capítulo 3 atendiam ao requisito do modelo proposto. Este trabalho optou por utilizar a técnica redes neurais artificiais supervisionadas por seus resultados promissores e sua escalabilidade, ou seja, a capacidade de adaptar a quantidade final de neurônios utilizados de acordo com as limitações de hardware do dispositivo utilizado nos testes, sem perder qualidade de convergência nos resultados (de Almeida Florencio et al., 2018). Essa escalabilidade permite que uma RNA seja dimensionada manualmente ou automaticamente garantindo uma redução de custos no seu processamento e armazenamento, tornando-o um método promissor para inserção em dispositivos IoT com limitados recursos.

Para um controle mais rigoroso da dimensionalidade e dos custos computacionais foi utilizada uma RNA de múltiplas camadas do tipo MLP (do inglês, *MultiLayer Perceptron*). As redes de múltiplas camadas distinguem-se das redes de camada simples pelo número de camadas intermediárias compostas por neurônios ocultos. Estes neurônios tornam a RNA capaz de realizar previsões ou classificações de modelos podendo diminuir a ordem

de complexidade assintótica de técnicas mais complexas (Affonso et al., 2010).

O algoritmo de treinamento utilizado na MLP se baseia em aprendizado por correção de erros. Esta técnica realiza um aprendizado supervisionado que busca, a cada interação, corrigir os pesos dos neurônios (Θ) através de uma retro-propagação do erro calculado entre a saída predita e a saída esperada. Assim, a cada interação os pesos de cada neurônio são atualizados e os erros são novamente calculados (Hecht-Nielsen, 1992).

O desempenho de uma MLP está relacionado à configuração adequada dos parâmetros e hiper-parâmetros, como o número de épocas para treinamento, número de camadas ocultas, função de ativação, tamanho dos lotes de treinamento, taxa de aprendizado, entre outros, e precisam ser ajustados durante o treinamento (Huan et al., Sep. 2016).

Os valores do número de neurônios nas camadas ocultas e a quantidade de camadas ocultas da rede neural foram obtidos empiricamente, após exaustivos testes avaliando os *trade-offs* entre o tempo de convergência e a qualidade de predição do modelo durante a fase de treinamento. Assim, buscamos reduzir o tempo de convergência do modelo sem perder qualidade do seu desempenho. Escolhemos a abordagem de rede neural rasa em vez de modelos de aprendizado profundo por causa da baixa complexidade, memória e consumo de energia, pré-requisitos de dispositivos com pouca memória e poder de processamento limitado.

Para implementação, foram selecionados os *frameworks* Keras e TensorFlow Lite (Shuangfeng, 2020) que obtiveram o melhores resultados em menor tempo de convergência, qualidade de soluções e tamanho final da estrutura da RNA. Outras bibliotecas, a exemplo do scikit-learn e TensorFlow foram testadas (Pedregosa et al., 2011).

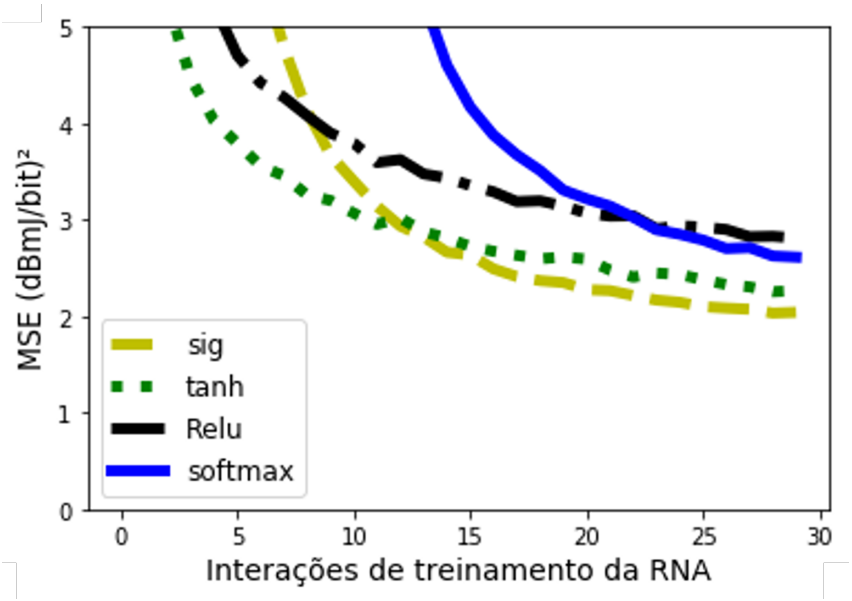
Foi utilizado o erro quadrático médio (MSE) como função de avaliação de convergência. Também usamos um fator de erro mínimo para interromper o treinamento, evitando *overfitting*. Após vários testes empíricos, a convergência mínima para MSE foi o valor de 0,5 $(dBmJ/bit)^2$ usando a média de 25 épocas para modelar a convergência.

As funções de otimização e ativação também são parâmetros decisivos para uma boa convergência do modelo. O algoritmo de aprendizado usado foi *Adam*. Também investigamos as técnicas *Adagrad* e *RMSProp* (Zou et al., 2019), mas *Adam* apresentou o melhor desempenho como sugerido em (Dogo et al., 2018).

Após testar diferentes funções de ativação, como as Funções Tangentes Hiperbólicas, Sigmóide Logístico e $Softmax$ apresentando uma média dos resultados muito semelhante (Figura 20), o estudo optou por utilizar a função de ativação $ReLU$, que garantiu uma convergência rápida sem comprometer a qualidade final das soluções (Nwankpa et al., 2018).

Embora funções como Sigmóide Logístico e Tangentes Hiperbólicas apresentem resultados melhores durante a convergência, o uso da $ReLU$ para nossa aplicação, permitiu um maior controle de ativação dos neurônios tornando 0 neurônios pouco relevantes para a convergência do modelo, facilitando a poda de conexões, neurônios ou até mesmo camadas esparsificando e/ou reduzindo assim o tamanho final da rede neural (Serra et al., 2021).

Figura 20 – Gráfico comparativo entre os MSEs obtidos pelas funções de ativações Sigmoid (*sig*), Tangente hiperbólica (*tanh*), *ReLU* e *softmax* durante o treinamento da base de dados (Barbosa et al., 2020).



Fonte: O autor

Além disso, a característica retificadora da *Relu* possibilitou a retirada da chamada da função de ativação do cálculo final dos peso de cada neurônio (ECKLE; SCHMIDT-HIEBER, 2019), o que reduz o custo de processamento envolvido na busca das soluções.

Neste trabalho foram usados minilotes com atualização de 200 amostras por lote para reduzir o tempo de treinamento e o custo computacional no processo de aprendizado (Zheng; Ma; Zhou, Oct. 2016).

Todas as definições e afinação (*fine tuning*) dos hiper-parâmetros do modelo proposto foram definidas por meio de múltiplas análises empíricas. A melhor configuração de parâmetros para cada modelo está resumida nas Tabelas 6 e 7, esses valores são usados para configurar os modelos na Figura 21 (para *slotted* ALOHA) e Figura 22 (para CSMA/CA), respectivamente.

Utilizamos os frameworks Keras e TensorFlow Lite na linguagem Python para a implementação da MLP utilizada (Shuangfeng, 2020).

Tabela 6 – Parâmetros para RNA-*slotted* ALOHA

Descrição	Valor
Quantidade de camadas escondidas	1
Quantidade de neurônios em cada camada escondida	100
Número máximo de épocas para treinamento	25
Taxa de aprendizado	$0,001$
Fator de erro mínimo	$0,18$
Algoritmo de otimização do gradiente estocástico	*Adam*
Função de ativação de neurônios	*ReLU*
Quantidade de batchs	200
Quantidade de exemplos de treinamento	180,000

Fonte: O autor

Tabela 7 – Parâmetros para RNA-CSMA

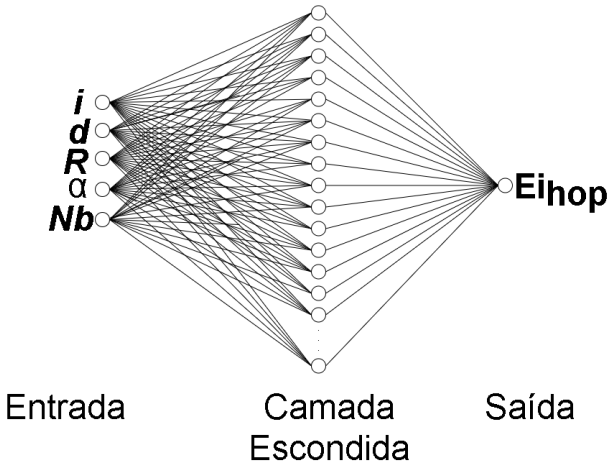
Descrição	Valor
Quantidade de camadas escondidas	2
Quantidade de neurônios em cada camada escondida	100
Número máximo de épocas para treinamento	20
Taxa de aprendizado	$0,001$
Fator de erro mínimo	$0,47$
Algoritmo de otimização do gradiente estocástico	*Adam*
Função de ativação de neurônios	*ReLU*
Quantidade de batchs	200
Quantidade de exemplos de treinamento	180,000

Fonte: O autor

O Pk também foi um parâmetro determinante para a eficiência do modelo proposto, identificando os grupos de dados por protocolo para manter o número de neurônios e capacidade de generalização simultaneamente para ambos os protocolos (Tabela 8) de acordo com a arquitetura do modelo apresentada na Figura 23.

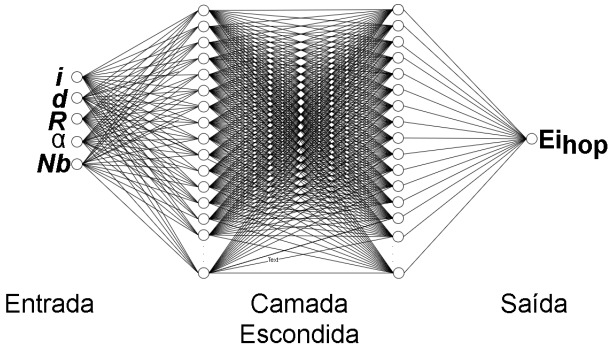
Essa estratégia facilita a implantação do modelo em diferentes dispositivos se adaptando ao protocolo de camada MAC utilizada por ele.

Figura 21 – Arquitetura RNA utilizada para treinamento do modelo *slotted* ALOHA.



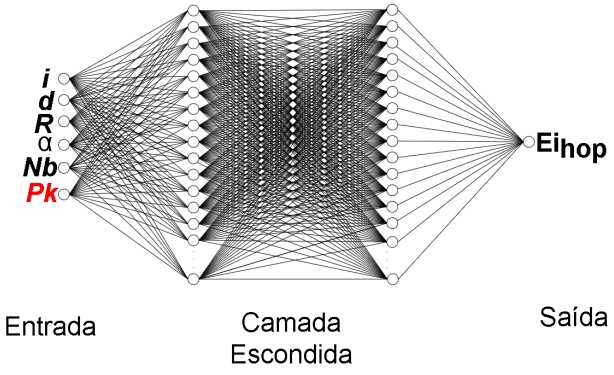
Fonte: O autor

Figura 22 – Arquitetura RNA utilizada para treinamento do modelo CSMA/CA.



Fonte: O autor

Figura 23 – Arquitetura RNA utilizada para treinamento do modelo *slotted* ALOHA+CSMA



Fonte: O autor

Tabela 8 – Parâmetros para RNA *slotted* ALOHA+CSMA

Descrição	Valor
Quantidade de camadas escondidas	2
Quantidade de neurônios em cada camada escondida	100
Número máximo de épocas para treinamento	30
Taxa de aprendizado	$0,001$
Fator de erro mínimo	$0,5$
Algoritmo de otimização do gradiente estocástico	*Adam*
Função de ativação de neurônios	*ReLU*
Quantidade de batchs	200
Quantidade de exemplos de treinamento	360,000

Fonte: O autor

5.2 DADOS DO EXPERIMENTO

Como o treinamento da RNA não precisa ser realizado dentro do dispositivo sensor, ao contrário da técnica MCE, que precisa realizar um novo processamento para obter soluções sempre que os parâmetros de cenários de rede mudam, foi utilizado um computador Intel Core^{TM} i7 8^a geração com 16 GB de memória RAM e placa de vídeo Nvidea GTX 1060 com 6 GB GDDR5 para treinar o modelo (Mavrogiorgou; Kiourtis; Kyriazis, 2017).

Para testes e as simulações foi utilizado um notebook Intel Core ^{TM} Dual Core de 2,4 GHz com 4 GB de RAM para maior aproximação aos hardware IoT mobile padrão, conforme analise do estudo publicado em (SINGH; KAPOOR, 2017).

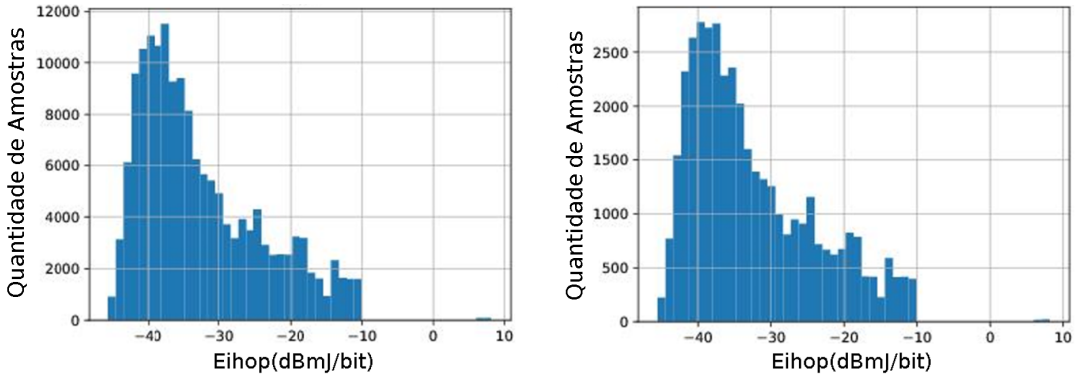
Para a criação do banco de dados teste, foram utilizados 20% do tamanho total do banco de dados, não aprendido pela rede neural no processo de treinamento, preservando a distribuição individual dos parâmetros mostrados na tabela 5. Para essa divisão aleatória dos conjuntos de treino e teste foi utilizada a função *train_test_split* importada da biblioteca *sklearn* que, além de garantir uma divisão justa dos dados permite, caso necessário, definir "*backups*"desses conjuntos através da sua funcionalidade *random_state* (SALAZAR et al., 2022).

As Figuras 24, 25 e 26 mostram as distribuições das amostra de teste e treino para cada uma das bases de dados utilizadas.

Para uma melhor avaliação da robustez e escalabilidade do modelo metaheurístico proposto, rodamos cenários obtidos por 30 variações, com novos valores para todos os pesos de inicialização da rede neural (Barbosa et al., 2019). Essa metodologia é essencial porque a maioria das técnicas heurísticas, como a abordagem utilizada, emprega inicialização aleatória dos pesos, o que apresenta uma variação nos resultados finais.

A tabela 9 apresenta os requisitos de sistema utilizado na geração da base de dados *slotted* ALOHA+CSMA nos dois computadores utilizados na pesquisa.

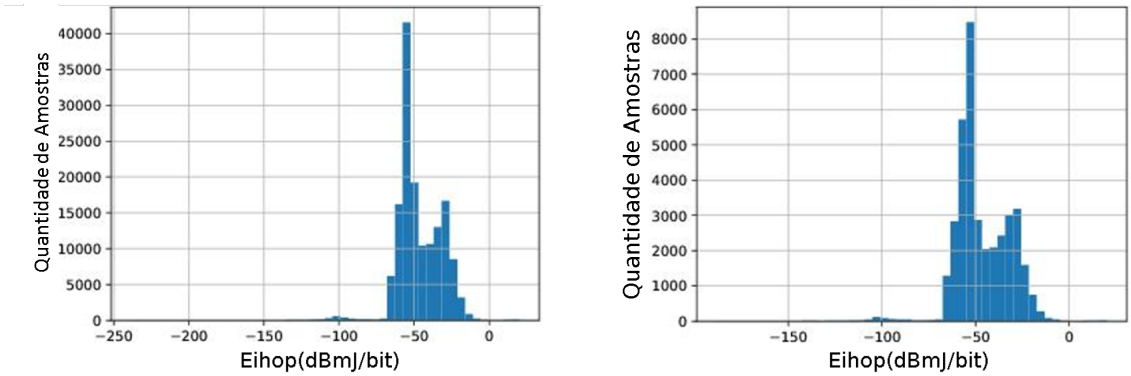
Figura 24 – Comparação entre a saída $\overline{E}_{ihop}$ dos conjuntos de treinamento e teste da base de dado *slotted* ALOHA.



(a) Treino (b) Teste

Fonte: O autor

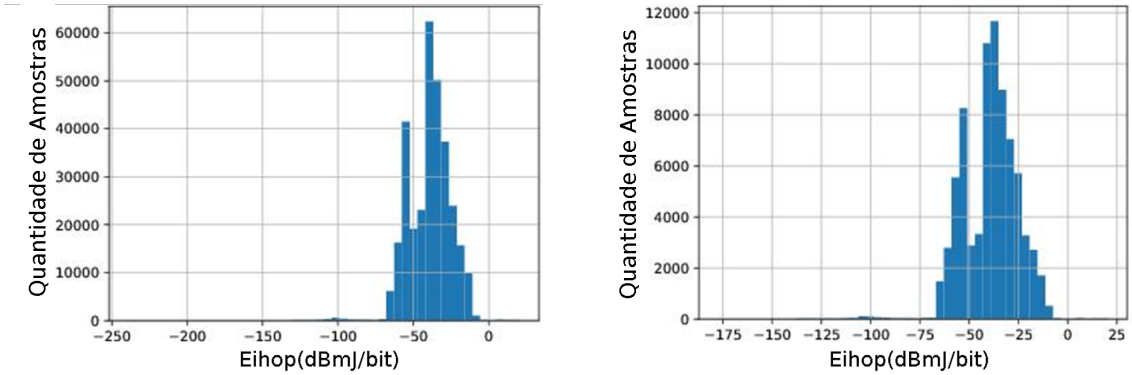
Figura 25 – Comparação entre a saída $\overline{E}_{ihop}$ dos conjuntos de treinamento e teste da base de dados CSMA/CA



(a) Treinamento (b) Teste

Fonte: O autor

Figura 26 – Comparação entre a saída $\overline{E}_{ihop}$ do conjunto de treinamento e teste da base de dado de dados *slotted* ALOHA + CSMA/CA.



(a) Treinamento (b) Teste

Fonte: O autor

Tabela 9 – Tabela de Requisitos dos sistemas utilizados na geração da base de dados *slotted* ALOHA+CSMA.

Requisitos						
SO	**Processador**	**Memória RAM**	**Espaço Disco**	**Software**	**Biblioteca**	**Tempo**
Linux	Intel Dual Core	4 GB	32Mb	Spider 3.3.1	-	15,4 dias
Windows	Intel i7	16 GB	32Mb	Spider 4.1.3	Numba. njit	8,3 dias

Fonte: O autor

5.3 RESUMO DO CAPÍTULO

Este capitulo apresentou uma proposta de abordagem multiprotocolo baseada em aprendizado de máquina que combina informações de algoritmos de controle de consumo de energia para gerenciar recursos de dispositivo que operem sob os protocolos CSMA/CA ou *slotted* ALOHA.

O modelo proposto considera múltiplas variáveis de camadas físicas PHY e MAC para a formação de uma configuração de hardware que otimiza, de forma multiobjetiva, o consumo energético.

Para possibilitar essas características a execução do modelo se deu em 3 etapas: A etapa 1 Modelo de Consumo de Energia (MCE), tem como objetivo definir e implementar modelos de otimização energético clássicos, capazes de otimizar simultaneamente parâmetros conflitantes como potência e taxa de transmissão, tamanho do pacote de dados e a distância entre os nós, em diferentes protocolos. O modelo de consumo de energia proposta por (da Silva; de Moraes, 2020) foi utilizado como ponto de partida.

A etapa 2, Geração da Base de Treinamento, tem como objetivo criar um banco de dados de treinamento robusto a partir da combinação de todas variáveis no MCE escolhido. Para isso, foi necessário criar um algoritmo de busca exaustiva, e assim encontrar as configurações de otimização global que encontrasse de forma exata a melhor configuração que resulta no menor consumo de energia durante diferentes cenários de transmissão.

A etapa 3, denominada Regressão de MCE via aprendizado de máquina, tem como principal objetivo criar um modelo de regressão multiprotocolo, baseado na técnica Redes Neurais Artificiais utilizando os parâmetros ótimos da etapa dois. Com isto pretende-se mostrar que é possível aproximar e otimizar funções exatas em casos onde os parâmetros ótimos não podem ser mapeados matematicamente (Sun et al., Jul. 2017).

Embora o modelo apresentado conte com várias etapas de implementação, sugerindo um alto custo computacional, uma vez que a RNA é treinada, o consumo computacional utilizado para predições ou inferências torna-se compatível com as configurações de dispo-

sitivos sem fio de baixa memória como será apresentado nos resultados. Portanto, pode-se inferir que o uso do aprendizado de máquina pode criar novos sistemas inteligentes que promovem uma comunicação eficiente independentemente do protocolo e da capacidade de processamento dos dispositivos utilizados.

O capítulo 6 apresentará a robustez do método, avaliando através de análises teóricas obtidas através de extensos experimentos numéricos simulados a partir de parâmetros de redes reais.

6 RESULTADOS PRELIMINARES E DISCUSSÕES

> *O homem não teria alcançado o possível se, repetidas vezes, não tivesse tentado o impossível.*
>
> Max Weber,

Para uma rede neural obter as soluções esperadas, mantendo seu poder de generalização, é preciso que se defina bem seus hiper-parâmetros. Variáveis como a quantidade de épocas de treinamento, critério de parada do aprendizado e o método de comparação entre os estimadores durante a avaliação da evolução da rede definem o comportamento durante o treinamento do modelo, podendo gerar situações de *underfitting* ou *overfitting* (Luxhøj; Williams; Shyur, 1997). Por este motivo, foi utilizado como comparação de estimadores o MSE (*Mean Squared Error*), que serviu como valor de referência para a quantidade de épocas e de avaliação entre as técnicas durante as iterações.

Para validação da proposta foram utilizados como referência os protocolos *benchmark* o *slotted* ALOHA e o CSMA/CA, realizando experimentos numéricos extensivos empregando dados simulados com parâmetros reais de redes com o intuito de:

1. Acompanhar a evolução de treinamento da RNA proposta e avaliar a capacidade de aprendizado da aproximação a soluções obtidas a partir de técnicas exatas.

2. Analisar a qualidade de predição da RNA a conjuntos de testes ainda não estudados pela técnica.

3. Comparar resultados obtidos pela técnica com os adquiridos a partir de algoritmos exatos, com o intuito de comprovar a capacidade de uma técnica heurística, baseada em aprendizado de maquina, de obter soluções tão eficientes quanto técnicas de busca de ótimos globais.

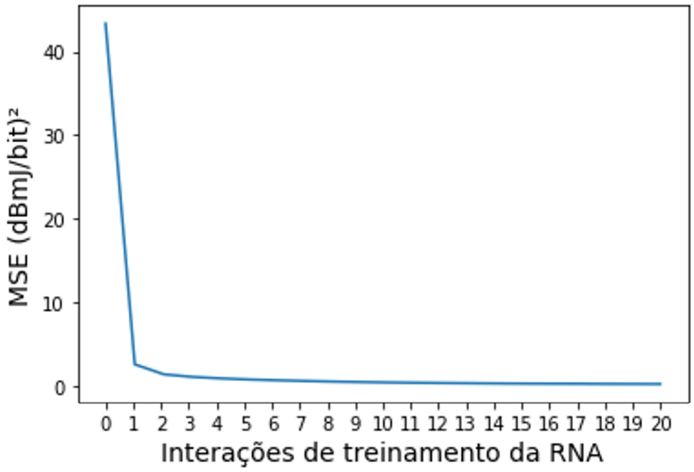
4. Medir o custo-benefício do modelo proposto em relação à técnica convencional de busca de melhores soluções, avaliando-os sob métricas críticas para sistemas IoT/WSN como custo de processamento e tempo de execução dos modelos na obtenção de soluções.

As seções a seguir apresentam os resultados obtidos em cada uma das redes.

6.1 *SLOTTED* ALOHA

A Figura 27 apresenta a evolução do MSE durante o processo de treinamento da RNA. Inicialmente, a rede apresenta uma queda acentuada na convergência do MSE já na

Figura 27 – Evolução do Erro Quadrático Médio (MSE) da saída E_{ihop} durante o treinamento *slotted* ALOHA.



Fonte: O autor

primeira iteração, obtendo o MSE mínimo definido com, em média, 4 iterações mantendo poucas variações ao decorrer das épocas de treinamento.

Em todos os testes realizados a RNA convergiu ao MSE mínimo definido, precisando de aproximadamente 20 interações para isso. Em termos gerais, os resultados obtidos foram bastante satisfatórios, se comparados ao valor de início do MSE.

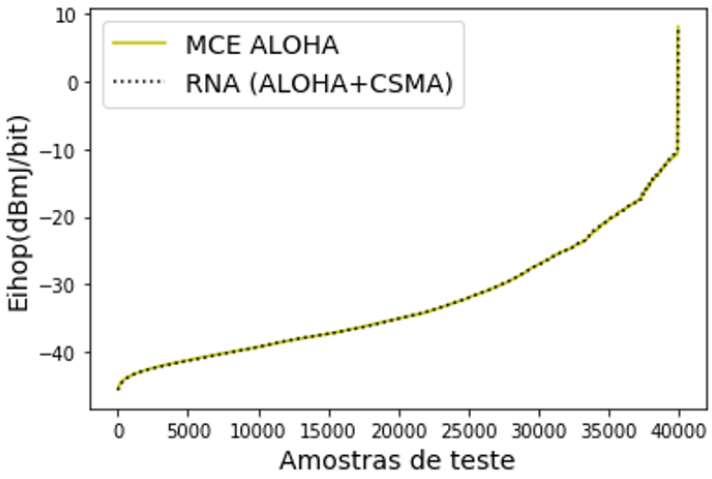
RNAs conseguem uma queda acentuada na convergência obtendo o MSE mínimo definido com, em média, 5 iterações mantendo poucas variações ao decorrer das épocas de treinamento, mostrando uma rápida convergência do modelo mesmo com as redes possuindo apenas um neurônio cada.

A Figura 28 apresenta um comparativo ordenado entre os valores de E_{ihop} obtidos pelo MCE, definidos como conjunto de testes *slotted* ALOHA, e as predições dos valores encontrados pela RNA. As linhas sobrepostas do gráfico demonstra que o MSE médio se manteve muito próximo do máximo definido no treinamento, indicando convergência do método. Isso evidencia a capacidade da rede em aprender as relações não lineares do algoritmo comparado.

As Figuras 29a e 29b ilustram o desempenho do modelo *slotted* ALOHA gerado a partir da Eq. (5.11) e a RNA (*slotted* ALOHA + CSMA/CA), para $\overline{E}_{ihop}$ em função de R e expoente de perda de caminho $\alpha = 2$, para cenários com ambientes pouco atenuados e $\alpha = 4$ para ambientes muito atenuados. A distância total entre o transmissor e o receptor é $d = 80$ metros (m), o número de saltos i varia em 1, 2, 4 e 8 com distâncias entre saltos de 80, 40, 20 e 10 metros, respectivamente.

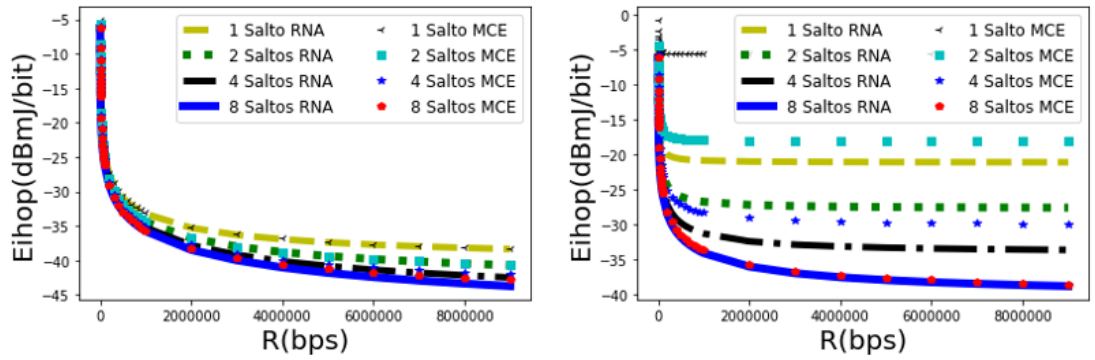
Nessas figuras, embora o modelo proposto por (da Silva; de Moraes, 2020) otimize matematicamente o parâmetro P_0, ele não é capaz de obter resultados superiores ao esquema aqui proposto. Isso porque a rede neural otimiza todos os parâmetros multiobjetivos envolvidos simultaneamente, trazendo uma melhor solução para o problema. Esta análise é

Figura 28 – Comparativo entre o E_{ihop} *slotted* ALOHA encontrado pelo MCE e o E_{ihop} predito pela RNA.



Fonte: O autor

Figura 29 – $\overline{E}_{ihop}$ em função da taxa de transmissão (R). Uma comparação entre o modelo apresentado por (da Silva; de Moraes, 2020) para o protocolo *slotted* ALOHA e o modelo *slotted* ALOHA + CSMA/CA proposto, para (a) $\alpha = 2$ e (b) α=4.



(a) $\alpha = 2$ (b) $\alpha = 4$

Fonte: O autor

evidenciada mais fortemente na Figura 29(b), na qual parâmetros como N_b, não otimizado por (da Silva; de Moraes, 2020), tornaram-se cruciais na redução do consumo de energia principalmente em transmissões com 1, 2 e 4 saltos.

A Tabela 10 analisa a ordem de complexidade assintótica do modelo para o pior caso utilizando a notação Big O, a média do custo computacional e o tempo de processamento aferidos para a obtenção das 40.000 melhores soluções em ambos os métodos. Neste aspecto, os custos totais para treinamento da RNA são separados nesta análise por não terem sido executadas no hardware de teste.

Podemos inferir com a Tabela 10 que uma rede neural além de produzir saídas passíveis de serem utilizadas em sistemas reais, realizando uma economia de aproximadamente 97,7% no uso da CPU e até 24.047,3 segundos (s) no tempo total de busca de soluções.

Tal desempenho foi alcançado tendo em vista que uma RNA treinada se limita a utilizar operações simples como multiplicação de matrizes, reduzindo sensivelmente o consumo na busca das soluções.

Para obtenção dos valores de Uso Médio de CPU e MEM, definidos nas Tabelas 10 e 11 foi usado o comando *htop* executado no *terminal* Linux do dispositivo utilizado para testes.

Tabela 10 – Comparativo da porcentagem de custo computacional (CPU e memória RAM (MEM)) e Tempo Total de Convergência entre os modelos RNA (*slotted* ALOHA+CSMA/CA) e MCE *slotted* ALOHA proposto em (da Silva; de Moraes, 2020)

Técnica	Big O	Uso Médio	Tempo Total
DNN Treinada	$O(n^2)$	$2,1\%$ CPU; $0,1\%$ MEM	$0,678s$
(da Silva; de Moraes, 2020) *slotted* ALOHA	$O(n^5)$	$99,7\%$ CPU; $2,2\%$ MEM	$6,68h$

Fonte: O autor

6.2 CSMA/CA

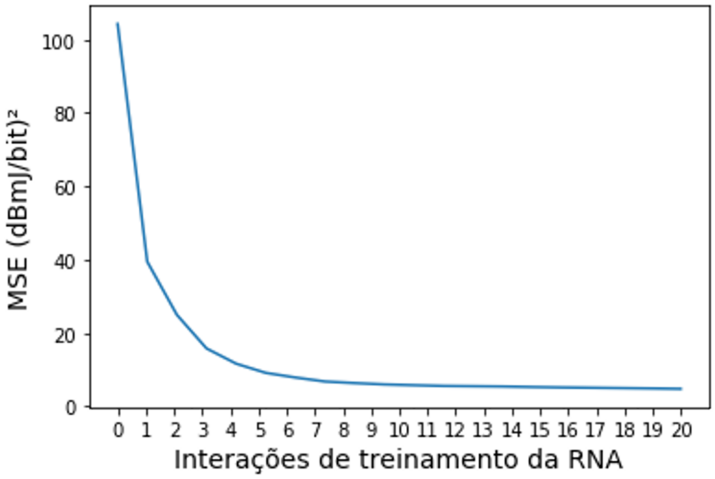
A Figura 30, apresenta a evolução do MSE durante o processo de treinamento da RNA CSMA/CA.

Diferente da RNA *slotted* ALOHA que obteve uma queda brusca no erro de previsão já nas duas primeiras iterações (Figura 27), a Figura 30 mostra que o modelo CSMA/CA precisou de em média 10 épocas para convergência, a partir daí entrando em fase de ajuste fino da previsão denominada *exploitation*.

As diferentes curvas de convergência podem ser entendidas pela diferença na complexidade das Eqs. (5.11) e (5.10), exigindo um esforço maior para aprender as relações de otimização entre todas as variáveis na Tabela 5 principalmente no modelo CSMA/CA que, embora tenha usado menos épocas para atingir MSE definido, exigiu uma arquitetura mais complexa para aprendizado desse MCE.

Após diversos testes empíricos, avaliou-se que a fase de *exploitation* foi necessária em ambos os modelos para garantir um valor de MSE sem perder a capacidade de generalização. Nos testes foi possível concluir também que modelos treinados com mais de 20 épocas apresentaram *underfit*.

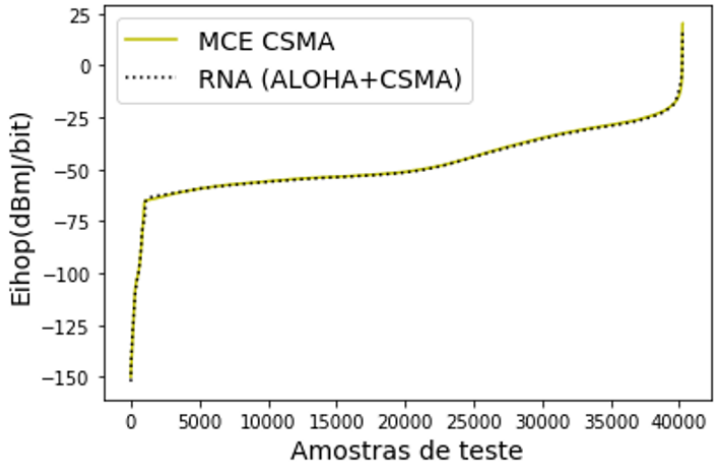
Figura 30 – Evolução do Erro Quadrático Médio (MSE) da saída E_{ihop} durante o treinamento CSMA/CA.



Fonte: O autor

A Figura 31 apresenta um comparativo ordenado entre os valores de E_{ihop} obtidos pelo MCE, definidos como conjunto de testes CSMA/CA, e as predições dos valores encontrados pela RNA. A análise do gráfico mostra que o modelo foi eficiente no mapeamento da entrada, encontrando os valores de $\overline{E}_{ihop}$ ótimos obtidos pelo método de treinamento proposto. Porém, mesmo com a adição de outras camadas ocultas, a RNA proposta não foi eficiente na predição de pontos fora da curva (*outlier*), a exemplo do valor $\overline{E}_{ihop} = 25$ dBmJ/bit, para o qual elevou o MSE médio da predição.

Figura 31 – Comparativo entre o E_{ihop} CSMA/CA encontrado pelo MCE e o E_{ihop} predito pela RNA.

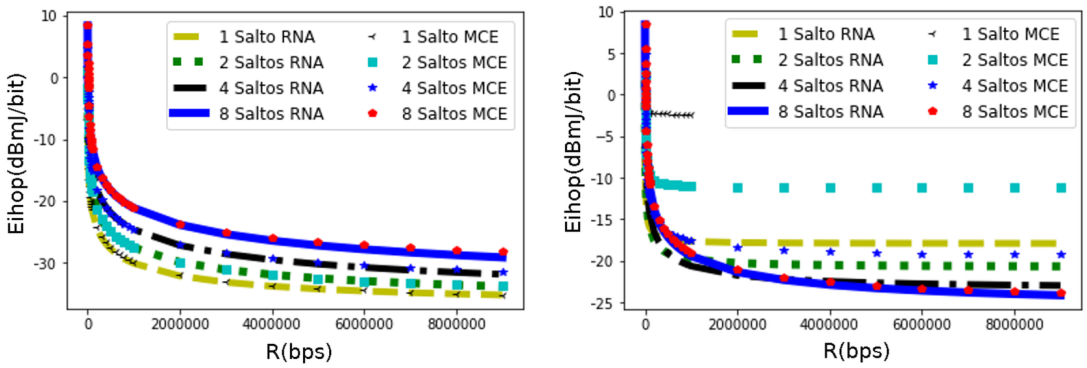


Fonte: O autor

As Figuras 32(a) e 32(b) apresentam os resultados obtidos do modelo CSMA/CA encontrado por (da Silva; de Moraes, 2020) gerado a partir da Eq. (5.10) e a RNA *slotted* ALOHA + CSMA/CA, para $\overline{E}_{ihop}$ em função de R para expoente de perda $\alpha = 2$, para ambientes pouco atenuados e $\alpha = 4$ para ambientes muito atenuados. A distância entre a

origem e o destino $d = 80$ metros (m) e o número de saltos $i = 1$, 2, 4 e 8 com distâncias entre os saltos de 80, 40, 20 e 10 metros, respectivamente.

Figura 32 – $\overline{E}_{ihop}$ em função da taxa de transmissão (R). Uma comparação entre o modelo apresentado por (da Silva; de Moraes, 2020) para o protocolo CSMA/CA e o modelo *slotted* ALOHA+CSMA/CA proposto, para (a) $\alpha = 2$ e (b) α=4.



(a) $\alpha = 2$ (b) $\alpha = 4$

Fonte: O autor

A Figura 32(a) mostra que a Eq. (5.10) obtém desempenho semelhante à nossa abordagem, porém a Figura 32(b), que simula ambientes com grande perda de dados durante as transmissões, a RNA proposta superou na otimização do tamanho do pacote N_b e na potência de transmissão P_0. A abordagem proposta por (da Silva; de Moraes, 2020) não conseguiu otimizar satisfatoriamente essas variáveis devido ao aumento substancial na complexidade do modelo matemático.

A análise de desempenho exposto na Tabela 11 mostra que rede neural realizou uma economia de aproximadamente 97,1% no uso da CPU e 113.322.733% no tempo total de busca de soluções. Essa grande economia, principalmente no tempo total de execução, foi dada principalmente pela ineficácia do trabalho proposto por (da Silva; de Moraes, 2020) em encontrar rapidamente as soluções ótimas, precisando pesquisar e cruzar, de forma exaustiva, todas as combinações das 5 variáveis de entrada R, N_b, α, d e i em diferentes cenários de transmissão para encontrar as 40.000 melhores soluções, enquanto a RNA proposta, após seu treinamento, é capaz de se adaptar rapidamente a esses cenários.

Tabela 11 – Comparativo da porcentagem de custo computacional (CPU e memória RAM (MEM)) e Tempo Total de Convergência entre os modelos RNA (*slotted* ALOHA+CSMA/CA) e o modelo CSMA/CA proposto em (da Silva; de Moraes, 2020)

Técnica	Big O	Uso Médio	Tempo Total
DNN Treinada	$O(n^2)$	$2,1\%$ CPU; $0,1\%$ MEM	$0,678s$
(da Silva; de Moraes, 2020) CSMA/CA	$O(n^5)$	$99,2\%$CPU; $2,2\%$MEM	$8,3$ dias

Fonte: O autor

6.3 RNA MULTIPROTOCOLO

Esta seção apresentará os resultados obtidos pela RNA (*slotted* ALOHA + CSMA/CA) treinada com uma base de dados composta pelos dois protocolos simultaneamente, requisitando apenas receber como instrução, do dispositivo, a variável de controle Pk, definida como mais um parâmetro de entrada para a Rede Neural, conforme Figura 23. O Pk identifica sob qual protocolo a rede WSN/IoT deveria operar com identificador 0 para *slotted* ALOHA e 1 para CSMA/CA. Assim, a RNA foi capaz de obter os resultados apresentados nas Figuras 29 e 32.

Em um sensores sem fio real o Pk serviria como um sistema de controle para Camada MAC auto-adaptável (SAML) que seleciona e alterna dinamicamente os protocolos MAC para obter as características desejadas em resposta a mudanças nas condições ambientais e nos requisitos da aplicação (SHA et al., 2013).

A rápida convergência da RNA (*slotted* ALOHA + CSMA/CA) mostrado na Figura 33 só foi possível por causa da variável Pk. Este parâmetro permitiu que, durante o treinamento, a RNA identificasse o protocolo de operação e aprendesse a relação de otimização existente entre a entrada e saída de cada protocolo individualmente.

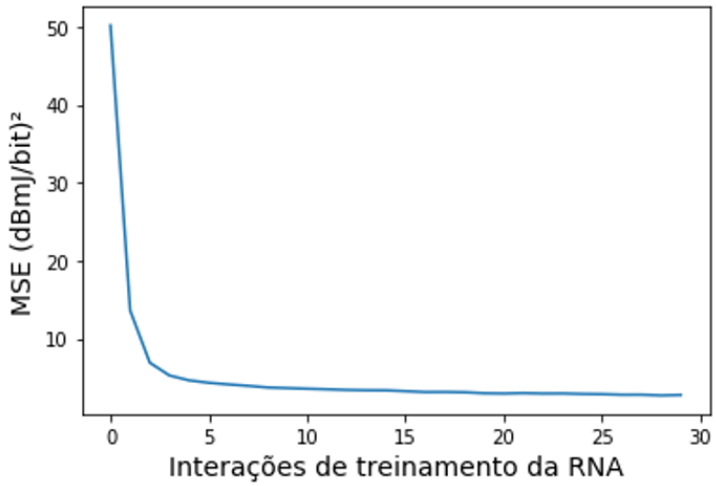
A análise do gráfico mostra que o MSE converge para o valor médio de 0,5 $(dBmJ/bit)^2$ predefinido no treinamento, sugerindo a convergência do método. Isso demonstra que a RNA proposta tem uma capacidade excepcional de aprender as relações não lineares entre as variáveis que interferem na otimização energética de vários protocolos diferentes simultaneamente. Esse fato indica que um modelo baseado em Aprendizado de Máquina, se bem treinado, pode agregar conhecimento de diferentes protocolos em sua estrutura, podendo ser utilizado para formar um sistema inteligente adaptável único que garanta eficiência energética em diferentes dispositivos independentemente do protocolo de camada MAC utilizado.

A Figura 34 apresenta um comparativo ordenado entre os valores de E_{ihop} obtidos pelo MCE, definidos como conjunto de teste (*slotted* ALOHA + CSMA/CA), e as predições dos valores encontrados pela RNA.

A RNA (*slotted* ALOHA + CSMA/CA) também foi responsável pela otimização de recursos energéticos apresentados nas Tabelas 10 e 11. Esse desempenho é garantido pelo treinamento eficiente do modelo, que, embora computacionalmente custoso, usando 346,85% de processamento da CPU, durou em média 8 minutos. Porém, o treinamento do modelo não precisa ser realizado dentro do dispositivo de baixa energia, ao contrário da técnica de MCE que precisa de um novo cálculo para obtenção de novas soluções, sempre que o parâmetro da rede mudarem.

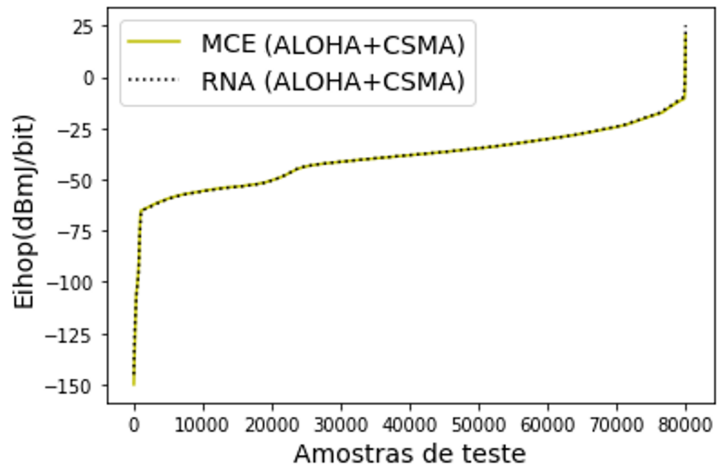
A principal diferença entre o modelo proposto e as técnicas exatas de busca de soluções é o fato de que, uma vez treinada previamente em um computador robusto, como o utilizado para estes experimentos, a rede neural pode ser salva em arquivos de modelo

Figura 33 – Evolução do Erro Quadrático Médio (MSE) da saída E_{ihop} durante o treinamento da base *slotted* ALOHA + CSMA/CA.



Fonte: O autor

Figura 34 – Avaliação de predição de E_{ihop} do conjunto de Teste das bases *slotted* ALOHA + CSMA/CA.



Fonte: O autor

de 25 kB de memória ROM em média, sem a necessidade de executar processos computacionalmente caros e demorados de busca de soluções, podendo obter resultados por predições mesmo em meio a novos cenários que se apresentam em redes reais. Por sua vez, os dispositivos de baixo consumo de energia executariam este modelo como uma função simples. Isso faz com que nosso modelo supere ainda mais as técnicas de computação tradicionais.

6.4 RESUMO DO CAPÍTULO

Uma das contribuições deste trabalho é um modelo de baixo processamento baseado RNA que otimiza múltiplas variáveis envolvidas no consumo energético de dispositivos que operam sob protocolos de camada MAC *slotted* ALOHA e/ou CSMA/CA. Desta forma, o aprendizado de máquina e a otimização multi-objetivo de recursos para eficiência energética podem ser alcançados em redes reais.

Extensos experimentos numéricos empregando dados simulados e parâmetros reais de rede, obtidos a partir de um modelo teórico rigoroso foram considerados para demonstrar a eficiência do modelo proposto em distintos cenários das redes, mostrando que uma rede neural pode ser usada para aproximar e otimizar funções exatas, onde os parâmetros necessários para a convergência do modelo não podem ser mapeados matematicamente.

O modelo proposto considera a energia disponível e a capacidade de processamento dos dispositivos para operar, apresentando configurações compatíveis com a grande maioria dos dispositivos apresentados na Seção 2.6. Nós o mostramos otimizando os resultados obtidos a partir das técnicas exatas executáveis em arquiteturas MIPS.

Este trabalho também mostrou que, embora o treinamento de um RNA seja caro e muitas vezes demorado, uma vez a rede treinada, seu uso e consumo computacional diminui consideravelmente. Também é importante notar que o uso de aprendizado de maquina permite obter resultados ideais com mais rapidez e eficiência. Dessa forma, obtivemos resultados tão bons quanto os das técnicas exatas com custos computacionais mais baixos do que as técnicas de busca gulosa que garante uma eficiência de 100% dos na busca dos melhores resultados.

7 RNA PARA CONTROLE ENERGÉTICO DENTRO DE SENSORES

> *A tarefa não é tanto ver aquilo que ninguém viu, mas pensar o que ninguém ainda pensou sobre aquilo que todo mundo vê.*
>
> Arthur Schopenhauer,

Na primeira parte desta pesquisa foi provado, através de testes em cenários controlados, a hipótese que uma RNA pode aproximar ou até mesmo melhorar de forma rápida e eficiente os resultados de modelos matemáticos com formulações complexas utilizados no gerenciamento de eficiência energética em protocolos como *slotted* ALOHA e CSMA/CA. Assim, foram apresentados resultados que mostram a redução do consumo energético envolvido na busca de soluções otimizando as variáveis de controle (no caso parâmetros das camadas PHY e Enlace-MAC).

A grande limitação encontrada na primeira etapa da pesquisa foi o uso apenas de modelos matemáticos para definir o comportamento e cenários hipotéticos de transmissão, pois, embora o modelo matemático proposto em (da Silva; de Moraes, 2020) consiga modelar o comportamento de transmissão utilizando inúmeras variáveis, esse modelo desconsideram outras importantes que aparecem nos cenários reais como as diferentes modulações utilizada por cada dispositivo (Gerasimov; Bender, 2000), as tentativas de transmissão de múltiplos dispositivos simultâneos que interfere diretamente no quantidade de retransmissão necessária para um envio de pacote, o tempo de *back-off* gasto, isto é, o tempo que um dispositivo aguarda para realizar uma nova transmissão após a ocorrência de erros como colisão ou perda de pacote, influência real no sinal provocada pelo ambiente que deve considerar a interferência causada por outros dispositivos e seu impacto no alcance do sinal entre outros (Rajandekar; SIKDAR, 2015).

Para resolver tais limitações e tornar o modelo proposto mais compatível com sistemas WSN/IoT reais, aqui se propõe como segunda parte desta pesquisa a criação de uma nova RNA implementada em hardware capaz de atuar diretamente em variáveis das camadas PHY e MAC inferiores como potência de transmissão, número de bits por quadro e taxa de transmissão, além de funções da camada MAC superior e de aplicação como distância entre os nós e números de saltos ideais do remetente ao destino.

Para isso, ao invés de se utilizar Modelos de Consumo de Energia (MCE) matemáticos, será utilizado um modelo de aferição medindo o consumo energético através de experimentos em simuladores (TAMIOSSO et al., 2020). Inicialmente esses dados serão utilizados como "caixa preta" para aprender os critérios de otimização e as relações que resultam no consumo de energia considerando a dinamicidade dos cenários reais de transmissão. Posteriormente é aberta a "caixa preta" identificando os pesos dos neurônios e *bias* utilizados pela RNA na otimização dos resultados, com isso é remodelada a RNA em uma função do primeiro grau com operações de soma e multiplicação reproduzindo apenas a funcionalidade *feedforward* da MLP.

O objetivo é inserir, dentro de um dispositivo sensor, uma RNA focada em otimização do consumo de energia, treinada em diferentes cenários e condições reais de transmissão.

O restante deste capítulo se destina a descrever o método e a técnica propostos e suas etapas de operação.

7.1 UMA ABORDAGEM DE APRENDIZADO DE MÁQUINA PARA REDUZIR O CONSUMO DE ENERGIA EM DISPOSITIVOS SENSORES

Para redução do consumo de energia empregando $\overline{E}_{ihop}$ foram otimizados os parâmetros de camada MAC: número de bits por quadro (N_b), potência de transmissão (P_0), taxa de transmissão (R) de dados e o parâmetro de camada de rede: número de saltos (i) do remetente ao destino. Todos os parâmetros foram otimizados para cenários com diferentes distâncias (d) levando em consideração outros parâmetros sensíveis não controláveis da rede como ocorrência de erros como colisão ou perda de pacote, atenuação real do ambiente, interferência de dispositivos e variações no alcance do sinal. Assim foi pretendido que a RNA, depois de treinada nesses diferentes cenários, se adaptasse gerando uma configuração para essas varáveis de forma a minimizar o consumo de energia.

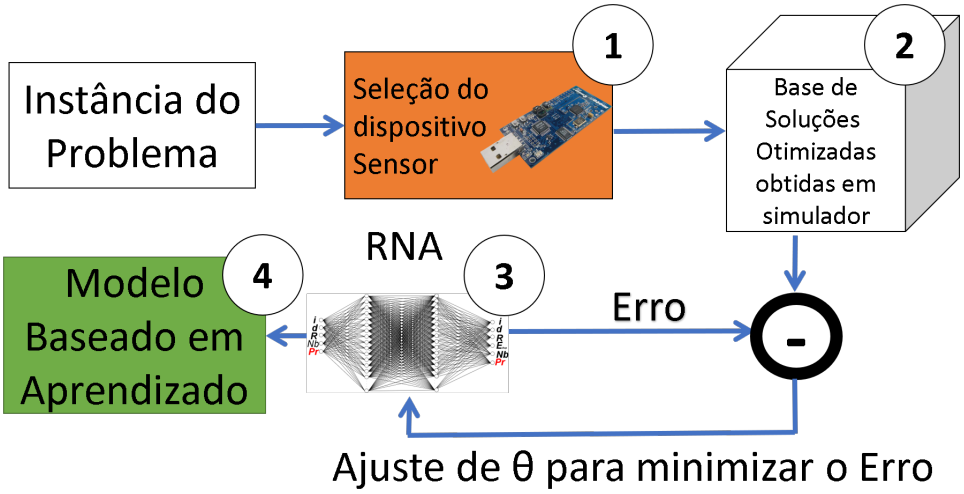
A abordagem proposta compreende uma sequência de quatro etapas, conforme é mostrado na Figura 35.

Primeiramente, foi escolhido um dispositivo sensor do tipo de baixa energia e baixo processamento utilizado como instância (base) do problema. A segunda etapa é submeter exaustivamente por simulação este dispositivo a diferentes cenários modificando, dentro das limitações definidas pelo fabricante, os parâmetros utilizados em cada transmissão e registrar as configurações de rede que resultaram em um menor consumo energético.

A etapa três consiste em encontrar e ajustar os parâmetros de treinamento ideais que criam o modelo proposto (*passo* 4) para aproximar as soluções de treinamento, capacitando assim a Rede neural artificial. Ao final do processo a RNA treinada é inserida dentro do dispositivo sensor capacitando este a operar de forma inteligente auto ajustando seus parâmetros para uma maior eficiência energética.

O modelo baseado em aprendizado dentro do dispositivo receberá como entrada os parâmetros de rede do seu fabricante, identificará e se adaptará às suas limitações de hardware e definirá as configurações necessárias para um menor consumo energético durante as transmissões. As Subseções 7.1.1, 7.1.2, 7.1.3 e 7.1.4 detalham melhor estas etapas.

Figura 35 – Etapas de operação do método baseado em aprendizado de máquina: 1) Definição do dispositivo; 2) Geração de um conjunto de soluções otimizadas para treinamento; 3) Treinamento do modelo para aproximar soluções ótimas obtidas em diferentes cenários de transmissão; 4) Implementação do modelo.



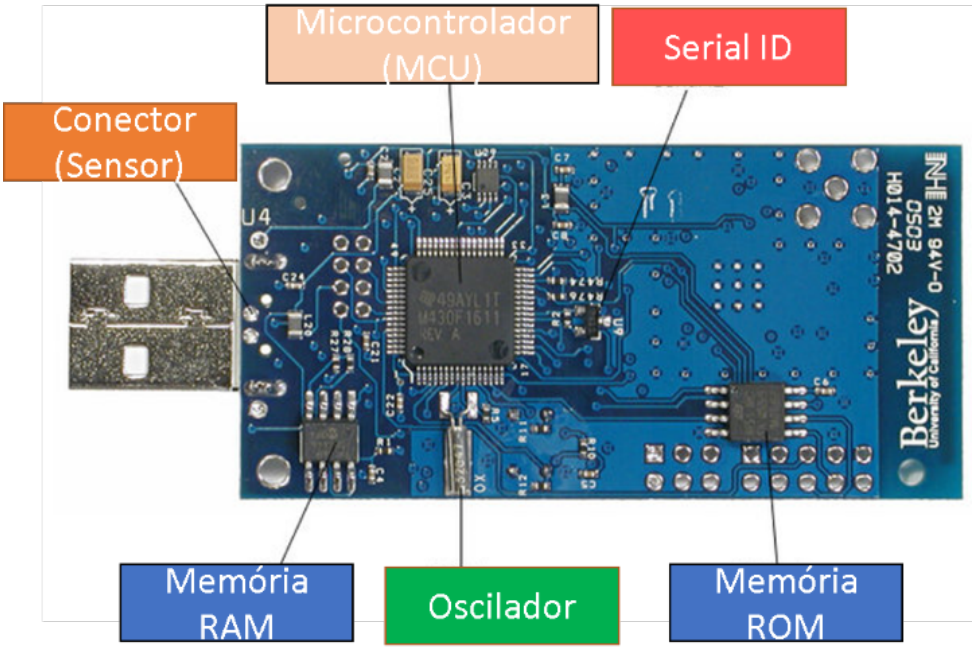
Fonte: O autor

7.1.1 Etapa 1: Seleção do Dispositivo Sensor

Nesta etapa foi definido um dispositivo sensor com memória e processador limitados porém que possibilitasse a manipulação e modificação das variáveis de camada MAC: potência, taxa de transmissão, número de bits por quadro, e tamanho do pacote de dados; e com um protocolo de rede que proporcionasse a obtenção do número de saltos percorrido pelo pacote do nó origem ao nó destino.

Entre todos os dispositivos que atendiam os pré-requisitos da pesquisa, apresentados na tabela 4, empregou-se o sensor Tmote Sky (Figura 36) com parâmetros PHY-MAC (Tabela 12), escolhido por sua limitação de hardware, compatibilidade com protocolos baseado em IPv6, sistema operacional programável TinyOS compatível com um simulador de rede Contiki Simulator 3.0 e interoperabilidade com outros dispositivos WSN/IoT (Johnson et al., 2009).

Figura 36 – Visão do verso do módulo Tmote Sky Ultra low power IEEE 802.15.4



Fonte: (DATASHEET, 2006)

Tabela 12 – Informações do dispositivo Tmote Sky (DATASHEET, 2006)

Descrição	Valor
Sistema Operacional	$TinyOS$
Processador	$8MHz$
Memoria RAM	$10kB$
Memória Flash (ROM)	$48kB$
Protocolo de Camada MAC	IEEE 802.15.4
Frequência da portadora	$2,4GHz$
Tensão de alimentação máxima durante a execução do programa	$3,6V$
Consumo máximo em *idle* (MCU ocioso/rádio desligado)	$1,2mA$
Consumo de corrente máximo em TX	$21mA$
Consumo de corrente máximo em RX	$23mA$
Taxa máxima de transmissão	$250kbps$
Número máximo de bits por quadro	$128bits$
Alcance máximo	$125m$

Fonte: O autor

7.1.2 Etapa 2: Geração da Base de Treinamento para a RNA

O objetivo desta etapa é criar uma base de dados de treinamento computando todos os dados obtidos pelas combinações entre as variáveis de entrada R, ΔN_b, Δd e Δi, definidos na Tabela 13, que resultam nas configurações de saída: $\overline{E}_{ihop}$, P_0, N_b e i, respeitando as limitações do dispositivo Tmote Sky (DATASHEET, 2006).

Tabela 13 – Parâmetros PHY-MAC empregados na formação da base de dados Tmote Sky (DATASHEET, 2006)

Parâmetro	Descrição	Valor
$B_{voltage}$	Tensão de alimentação da bateria	$3,6V$
Tx_{amp}	Consumo de corrente para receber	$19,5mA$
Rx_{amp}	Consumo de corrente para transmitir	$21,8mA$
cpu_{amp}	MCU idle e Radio desligado	$54,5\mu A$
$cpuS_{amp}$	MCU standby	$5,1\mu A$
R	Taxa de transmissão	$250kbps$
ΔN_b	Número de bits por quadro	8 a $128bits$
Δi	Número de saltos até o destino	1 a 8 saltos
Δd	Distância entre fonte e destino	25 a $360m$

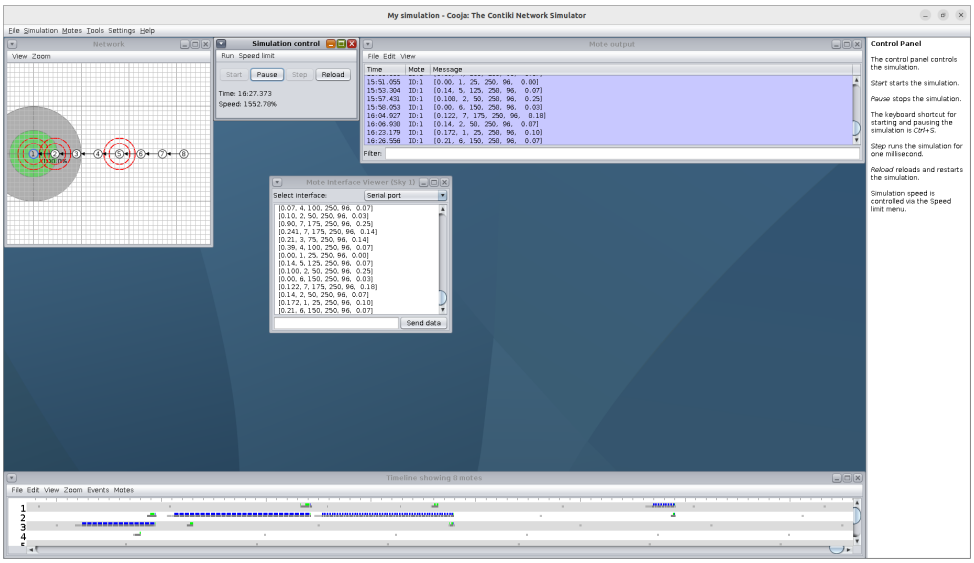
Fonte: O autor

Para obtenção dos dados seria necessário utilizar diversos dispositivos Tmote Sky sob testes em diferentes cenários de transmissão. Assim, utilizou-se um simulador/emulador de rede capaz de simular o dispositivo, criar topologias e fazer a investigação do comportamento da rede (MARTINS et al., 2017). Com este propósito utilizou-se a plataforma de simulação/emulação Cooja versão 3.0 (CONTIKI, 2012) projetada especificamente para redes de sensores sem fio e redes IoT com capacidades limitadas (VELINOV; MILEVA, 2016b).

Convenientemente, o simulador Cooja também conta com um Ambiente de Desenvolvimento Integrado (IDE) que proporciona uma apresentação visual detalhada do comportamento dos dispositivos durante cada interação de transmissão da rede como mostra a Figura 37. Além disso, todos os códigos executados no Cooja podem ser transferidos e executados diretamente em dispositivos reais que utilizam ContikiOS ou TinyOS (Guimaraes et al., 2019).

No simulador Cooja as medições de consumo de energia foram efetuadas utilizando duas ferramentas principais o PowerTrace, que coleta o consumo de energia dos componentes em uma determinada faixa de tempo, e o NETSTACK (Mugdhe, 2012), uma biblioteca que importa uma pilha de protocolos de rede com rádio modelo cc2420, protocolo MAC IEEE 802.15.4 e RPL para roteamento, todos compatíveis com o dispositivo Tmote Sky.

Figura 37 – Tela ilustrativa do IDE Cooja versão 3.0.



Fonte: o autor

O PowerTrace é dividido em campos que obtém o consumo acumulado gasto pelo MCU e rádio em atividades como transmissão, recepção ou *idle*. Todos essas informações podem ser obtidas dentro de um determinado espaço de tempo, tornando-a uma ferramenta útil para calcular o consumo instantâneo de energia durante a realização de uma transmissão, por exemplo (Guimaraes et al., 2019).

Com o NETSTACK foi possível definir e aferir as variáveis de transmissão Número de bits por quadro da camada MAC e número de saltos da camada de rede. A variável distância entre os nós foi definida como *Random positioning* diretamente na IDE contiki definindo apenas o alcance máximo de $360m$. O valor da taxa de transmissão se manteve constante em $250kbps$ dada a limitação do dispositivo que não aceita a modificação dessa variável (DATASHEET, 2006).

Então, utilizou-se o PowerTrace para obter o consumo de energia total $\overline{E}_{ihop}$ necessário para transmitir um pacote em i saltos dado pela equação de consumo energético da seguinte forma (Guimaraes et al., 2019; VELINOV; MILEVA, 2016a; Amirinasab Nasab et al., 2020):

$$\overline{E}_{ihop} = \frac{I_{ihop} * T_{State} * B_{voltage}}{Clock_{Res}}, \tag{7.1}$$

onde, I_{ihop} é o consumo de corrente, em amperes, gastos pelos componentes do dispositivo para realizar uma transmissão, T_{State} o tempo acumulado de um estado, no sistema de temporização do dispositivo de uma transmissão e ClockRes a frequência do relógio (32.768 Hz). O campo *timer* utilizado por T_{State} é dado no PowerTrace pelo cronômetro do SO do dispositivo, que possui fator de escala 256; assim, o *timer* do SO tem uma frequência de 128 Hz então se tem que (Guimaraes et al., 2019):

$$\frac{Time}{128Hz} = \frac{CPUon + CPU_{idle}}{32,768Hz}, \tag{7.2}$$

com $CPUon$ sendo o número de tiques de relógio em que a CPU esteve no modo ativo e CPU_{idle} o número de tiques em que a CPU esteve no modo inativo.

A base de dados final foi obtida em cenários que consideraram o primeiro nó como o emissor e os intermediários retransmitindo essa mensagem para um nó destino enquanto concorrem pelo canal tentando realizar suas próprias transmissões. Isso resultou em uma base de dados sólida com aproximadamente 30.000 amostras representativas do problema, que levou em consideração problemas reais como a ocorrência de erros de colisão e perda de pacote, necessidade de retransmissões, diferentes tempos de *back-off*, atenuação causada por interferência entre antenas e alterações no alcance do sinal. A base de dados completa está registrada em (Barbosa; Zanchettin; Mariz, 2023).

7.1.3 Etapa 3: Regressão via Aprendizado de Máquina

Esta etapa da técnica proposta tem como principal objetivo criar a RNA do tipo MLP que será inserida parcialmente no dispositivo Tmote Sky, respeitando as limitações de processamento e armazenamento do dispositivo e compatibilidade com o sistema operacional TinyOS e linguagem computacional NesC ou C definidas pelo fabricante.

Diferente da primeira etapa da pesquisa, o objetivo da RNA que atuará no sensor é receber os parâmetros: Consumo de energia esperado $E_{ihopMAX}$, definido como padrão em 0 para consumo mínimo, alcance máximo d_{max}, taxa máxima de transmissão R_{max} e Número máximo de bits por quadro N_{bmax} fornecidos pelo fabricante. Utiliza-se esses dados como balizador para que a RNA encontre, como saída, as melhores configurações de operação para as varáveis: E_{ihop}, i, N_b e P_0. Com isso a RNA adaptará sozinha suas saídas aos diferentes cenários reais de transmissão que o dispositivo for submetido.

Para suprir as limitações de hardware do dispositivo foi realizado um rigoroso controle na dimensionalidade da RNA através de exaustivos testes empíricos que garantissem resultados para convergência mínima com MSE=0,5 para todas as saídas da RNA. Um ajuste fino no tamanho final da rede neural também foi realizado, podando cautelosamente camadas e neurônios nulos definidos pela função de ativação dos neurônios $ReLU$ que torna 0 os pesos dos neurônios pouco relevantes para a qualidade final do modelo (Serra et al., 2021).

Assim como na primeira parte da pesquisa, foi utilizado o algoritmo de aprendizado *Adam* que novamente obteve melhores resultados na convergência do modelo.

Contudo foi possível concluir que a melhor solução para o problema foi a criação de 4 RNAs de 1 neurônio com 1 camada escondida, cada uma otimizando uma saída específica do modelo. As melhores configurações dos parâmetros e hiper-parâmetros estão resumidas na Tabela 14. Esses valores foram usados para configurar a RNA da Figura 38.

7.1.4 Etapa 4: Inserção do Modelo baseado em aprendizado no Tmote Sky

Depois da RNA treinada com o MSE em um nível satisfatório, ela já pode ser utilizada para gerar predições das melhores configurações para as saídas desejadas. Porém, para viabilizar a inserção da RNA-MLP em um dispositivo de baixa memória ela precisa ser modificada para formato de uma função, com utilização apenas no modo progressivo ($feed - forward$) (Svozil; Kvasnicka; Pospichal, 1997), ou seja, as entradas apresentadas serão processadas nas camadas intermediárias gerando resultados na camada de saída sem a retro-propagação e aprendizado.

Desta forma é inserida no sensor apenas os valores das saídas determinado pelas seguintes equações (Svozil; Kvasnicka; Pospichal, 1997; NORIEGA, 2005):

$$x_i = f(\xi_i), \tag{7.3}$$

$$\xi_i = \sum_{j \in \Gamma_i} \omega_{ij} x_j + \vartheta_i, \tag{7.4}$$

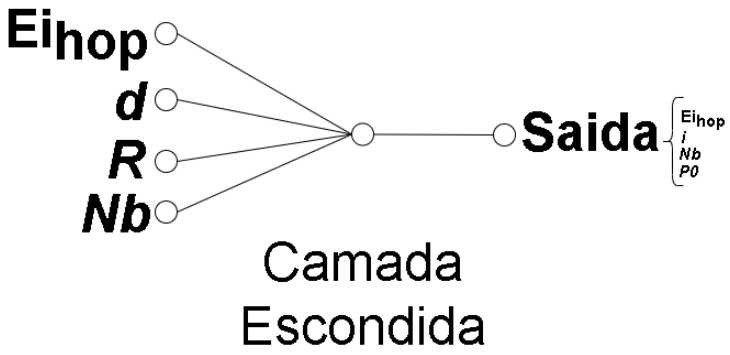
onde ξ_i é o potencial do i-ésimo neurônio, $f(\xi_i)$ é a função de transferência, com Γ_i representando um conjunto de neurônios predecessores conectados ao neurônio i onde cada neurônio em uma determinada camada está conectado com todos os neurônios na próxima camada, ω_{ij} é o peso da conexão entre o i-ésimo e o j-ésimo neurônio, e ϑ_i o coeficiente de limiar do i-ésimo neurônio. Especificamente para o modelo com apenas uma camada intermediária de um neurônio pode-se reduzir Γ_i a um conjunto unitário.

Tabela 14 – Parâmetros da RNA treinada para o Sensor Tmote Sky

Descrição	Valor
Quantidade de camadas escondidas	1
Quantidade de neurônios em cada camada escondida	1
Número máximo de épocas para treinamento	10
Taxa de aprendizado	$0,001$
Fator de erro mínimo	$0,05$
Algoritmo de otimização do gradiente estocástico	*Adam*
Função de ativação de neurônios	*ReLU*
Quantidade de batchs durante treinamento	200
Quantidade de exemplos de treinamento	$26,840$

Fonte: O autor

Figura 38 – Arquitetura da RNA



Fonte: O autor

7.2 DADOS DO EXPERIMENTO

As etapas descritas nas Seções 7.1.1, 7.1.2, 7.1.3 requisitam um alto custo computacional porém não precisam ser executadas em um dispositivo sensor.

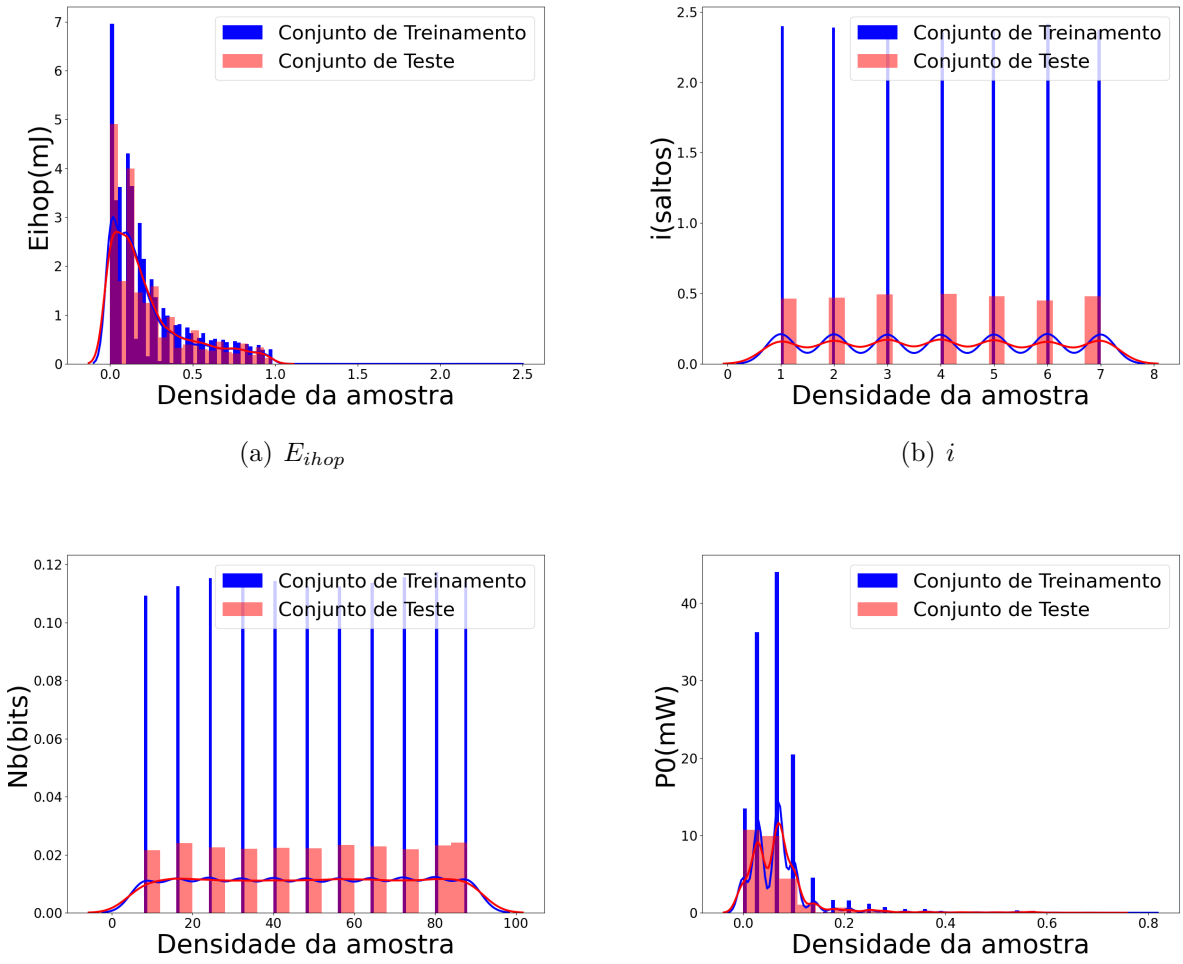
Para obtenção de dados de simulações, etapa 2, e simulações da etapa 4, foi utilizado um computador Intel Core^{TM} i7 8^{a} geração com 16 GB de memória RAM e placa de vídeo Nvidea GTX 1060 com 6 GB GDDR5.

Para os exaustivos testes empiricamente de ajustes de parâmetros e hiperparâmetros de treinamento da RNA, etapa 3, foi utilizado um super computador, disponível em servidor, com 4 TB de RAM e placa de vídeo NVIDIA Tesla V100 32GB cedido pela Nvidea através do programa *NVIDIA Academic Hardware Grant Application*.

Para a criação da base de dados teste, foram utilizados 10% do tamanho total base gerado na etapa 7.1.2, preservando a distribuição individual dos parâmetros mostrados na Tabela 13, novamente divididos através da função *train_test_split* do *sklearn* (SALAZAR et al., 2022). A Figura 39 mostra as distribuições das amostra para cada uma das saída das RNAs.

Com a inserção da RNA no dispositivo sensor Tmote Sky outros testes foram realizados analisando a qualidade das soluções e comparando com o consumo pretendido durante as simulações.

Figura 39 – Comparação entre as saídas E_{ihop}, i, N_b e P_0 do conjunto de treinamento e teste da base de dados Tmote Sky.



(a) E_{ihop} (b) i

(c) N_b (d) P_0

Fonte: O autor

7.3 RESUMO DO CAPÍTULO

Este capítulo apresentou a segunda parte da pesquisa que teve como objetivo apresentar os processos necessários para a viabilização de uma RNA-MLP em sistemas WSN/IoT reais.

Essa RNA implementada em hardware é capaz de atuar diretamente em funções estáveis das camadas PHY e MAC melhorando a eficiência energética através da otimização de variáveis como o números de saltos do remetente ao destino, o número de bits por quadro e a potência de transmissão.

Para possibilitar essas características a execução do modelo se deu em 4 etapa: A etapa 1 - Definição do dispositivo, selecionando um dispositivo de baixa memória que será utilizado como instância do problema para assim identificar, através de simulações

assistidas, o comportamento das variações dos parâmetros que influenciam no consumo de energia durante a transmissão.

A etapa 2, Geração da Base de Treinamento, criando um banco de dados de treinamento robusto a partir da combinação de diversas variáveis no dispositivo escolhido. Para isso, foi necessário submeter o dispositivo sensor a diferentes cenários de transmissão e aferir em simulador o consumo gasto em cada transmissão encontrando as configurações de otimização energética que resultaram no menor consumo de energia em cada cenário.

A etapa 3, denominada Regressão via Aprendizado de Máquina, tem como principal objetivo criar um modelo de regressão baseado na técnica Redes Neurais Artificiais do tipo *MultiLayer Perceptron* capaz de encontrar rapidamente as configurações PHY-MAC que resultem em um menor consumo energético.

A etapa 4, Inserção do Modelo, exibiu os detalhes para viabilizar a implantação de RNA em um dispositivo sensor Real modelando a RNA em formato de uma função linear para inseri-la em dispositivos IoT com limitados recursos de memória e processamento. Com isto é possível inserir uma RNA em um dispositivo limitado, e criar um modelo de regressão mesmo em casos onde há parâmetros que ainda não podem ser mapeados por funções matemáticas.

Embora o modelo apresentado conte com várias etapas que demandam um alto custo computacional, uma vez que a RNA é treinada, o consumo computacional utilizado para predições ou inferências torna-se equivalente a de uma função linear com apenas operações de soma e multiplicação, facilmente inserida em dispositivos de baixa memória como será apresentado nos resultados.

O capítulo 8 apresentará a robustez do método, avaliando o modelo através de extensos experimentos em simuladores.

8 RESULTADOS E DISCUSSÕES

“ A Skynet decidiu o nosso destino em um microssegundo: o extermínio. ”

Kyle Reese,

Um dos objetivos desta tese é propor um modelo inteligente de otimização da eficiência energética em dispositivos WNS/IoT capaz de se adaptar automaticamente em diferentes cenários de transmissão. Para isso, foram realizados experimentos extensivos em cenários de transmissões simulados em um emulador sensor com o intuito de:

1. Acompanhar a evolução de treinamento de cada uma das RNAs avaliando a capacidade e velocidade de aprendizado.

2. Analisar a qualidade de predição das RNAs durante as transmissões em cenários reais.

3. Comparar as soluções propostas pelas RNAs com os resultados adquiridos por um dispositivo sensor durante a transmissão de dados avaliando a capacidade de otimização energética do modelo.

4. Medir o custo-benefício do modelo proposto comparando o custo de processamento e tempo de execução de um sensor com e sem a RNA.

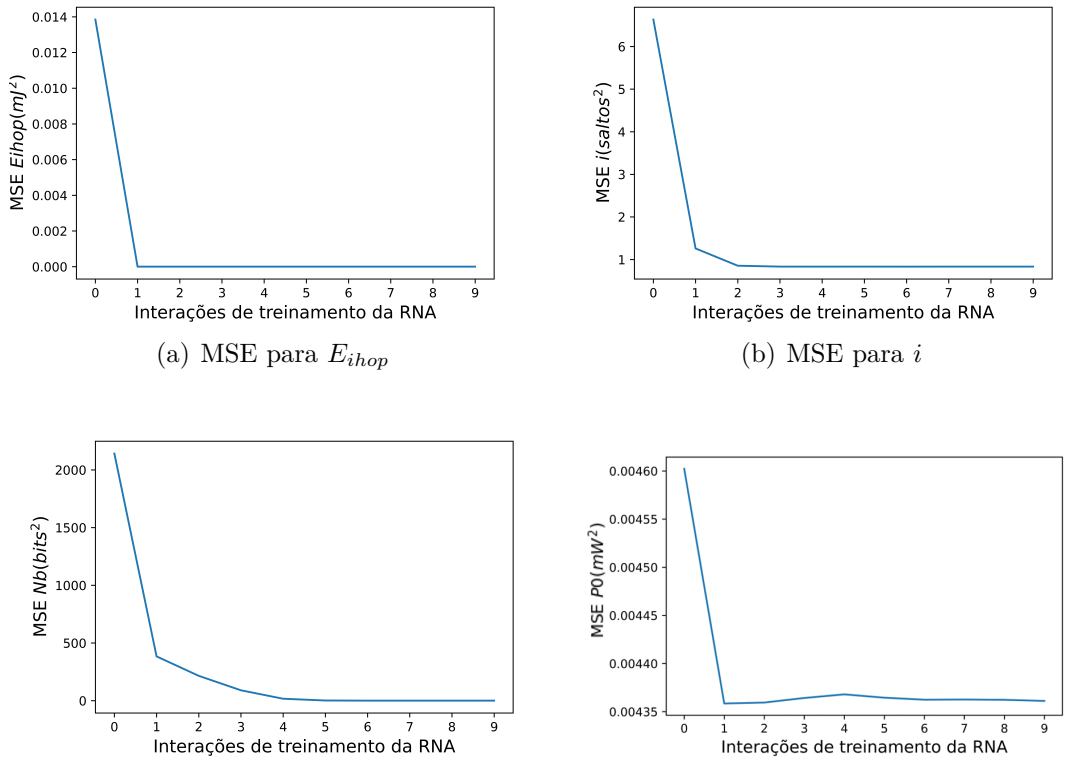
As Seções as seguir apresentam respectivamente: os resultados de predição obtidos por cada uma das redes neurais treinadas em um sistema computacional (8.1) e a avaliação de desempenho das RNAs inseridas no Tmote Sky (8.2).

8.1 AVALIAÇÃO DAS SOLUÇÕES DA RNA

A Figura 40 apresenta a evolução do MSE durante o processo de treinamento das RNAs RNA_{Eihop}, RNA_i, RNA_{N_b} e RNA_{P_0}.

Em todas evoluções de MSE apresentados na Figura 40 as RNAs conseguem uma queda acentuada na convergência obtendo o MSE mínimo definido com, em média, 5 iterações mantendo poucas variações ao decorrer das épocas de treinamento, mostrando uma rápida convergência do modelo mesmo com as redes possuindo apenas um neurônio cada. Essa eficiência da RNA pode ser melhor evidenciada na Figura 40(c) onde, embora o MSE inicial fosse de aproximadamente 2000, há uma convergência rápida para o MSE de $0,5\ bits^2$. Outro destaque é a Figura 40(d) onde a RNA_{P_0} consegue evoluir seu MSE para $0,005\ mW^2$ já na primeira iteração de treinamento. Todas as redes obtiveram resultados satisfatórios, se comparados ao valor inicial do MSE.

Figura 40 – Evolução do Erro Quadrático Médio (MSE) para cada umas das saídas: E_{ihop}, i, N_b e P_0, durante o treinamento do modelo.



(a) MSE para E_{ihop}

(b) MSE para i

(c) MSE para N_b

(d) MSE para P_0

Fonte: O autor

A Figura 41 apresenta um comparativo ordenado entre os valores de E_{ihop}, i, N_b e P_0 obtidos pelo Tmote Sky e as predições dos valores encontrados pelas RNAs. A análise das Figuras 41(a) e 41(c) constata que houve um rápido aprendizado do modelo, observado pela obtenção de bons resultados obtidos em tempo real durante as transmissões de dados no Tmote Sky. Como foi utilizado apenas um neurônio na camada escondida da rede isso pode indicar que talvez exista uma relação linear entre as entradas do dispositivo sensor Tmote Sky: distância entre os nós d, taxa de transmissão R e tamanho do pacote N_b. Trabalhos futuros podem explorar melhor essa relação.

A Figura 41(b) mostra que a rede neural RNA_i não conseguiu identificar uma relação discreta na quantidade de saltos necessária para uma transmissão, embora tenha conseguido em RNA_{Nb}, levando a crer que essa substituição de valores discretos de i para um modelo contínuo de saltos esteja intrinsecamente ligada aos valores contínuos da distâncias entre os nós emissor e receptor.

Como nos cenários reais realizados nos testes de transmissão a distância não é algo controlável essa definição de valores contínuos para i pode se apresentar como uma tentativa de compensação pelo d inconstante, podendo ser um ponto positivo na busca das melhores soluções expandindo o espectro de i dentro do MSE proposto. É importante

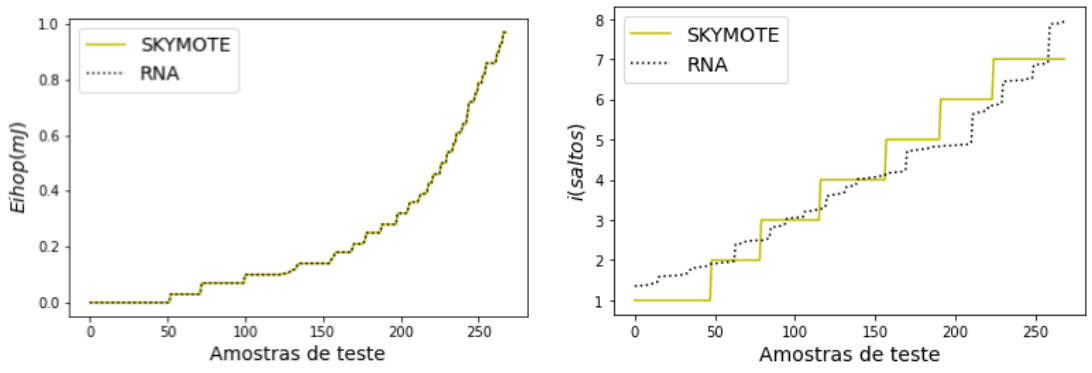
esclarecer que esses valores contínuos não influenciaram negativamente na qualidade de predições do modelo, que continuou apresentando MSE de 0,5 para a RNA_i.

A Figura 41(d) mostra que a RNA_{P_0}, embora tenha obtido MSE de 0,5 como previsto, não conseguiu acompanhar o aumento rápido da potência de transmissão em alguns cenários reais atípicos de transmissão, como por exemplo colisão e/ou perda de pacote recorrentes provocada por anomalias do ambiente ou mau funcionamento de outros dispositivos, isso pode ser notado pela incapacidade do modelo em gerar os *outliers* de P_0. Esse comportamento da RNA foi propositalmente deixado durante o treinamento de forma a desestimular subidas rápidas de P_0 durante as simulações de transmissões, fazendo com que as RNAs buscassem soluções alternativas que minimizasse esse valor ainda que requisitasse novas retransmissões.

Resumidamente a Figura 41 evidencia a capacidade das redes neurais em aprender as relações lineares e não lineares envolvidas nas transmissão reais entre dispositivos sensores.

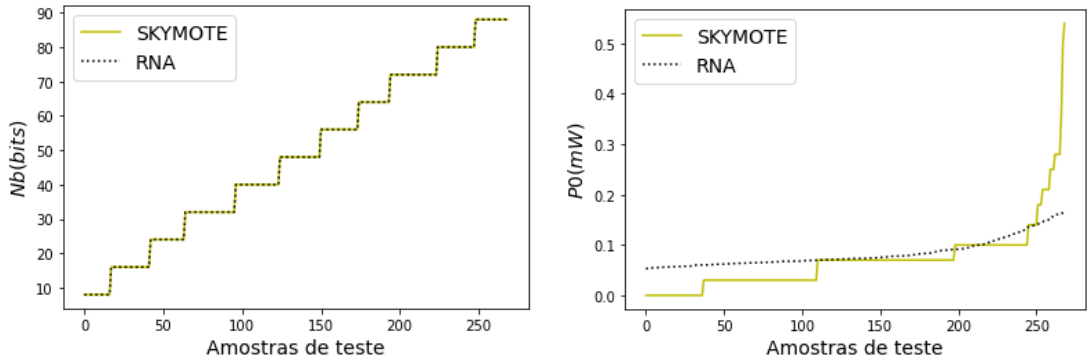
Porém, estudos avançados sobre o comportamento da RNA em cenários de transmissões reais ainda precisam ser realizados para compreender a causa dessas e de outras tomadas de decisões da RNA operando dentro do Mote.

Figura 41 – Comparativo entre os valores de E_{ihop}, i, N_b, P_0 encontrados pelo Tmote Sky e o E_{ihop}, i, N_b, P_0 preditos pelas RNAs.



(a) E_{ihop}

(b) i

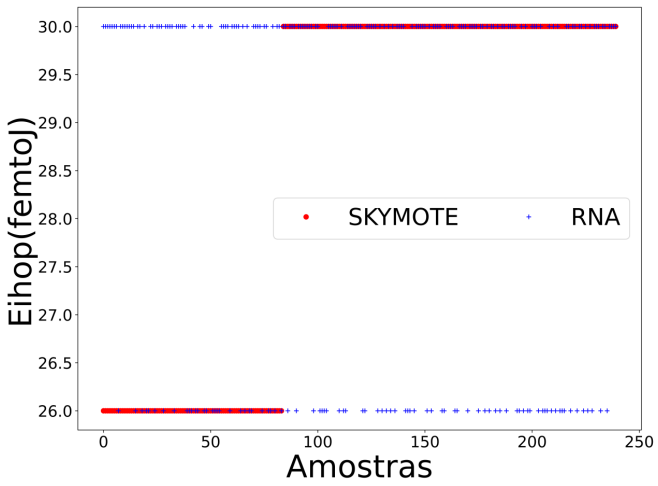


(c) N_b

(d) P_0

Fonte: O autor

Figura 42 – Consumo energético da função RNA no Tmote Sky



Fonte: O autor

8.2 EFICIÊNCIA ENERGÉTICA DA RNA DENTRO DO SENSOR

Para medir a eficiência energética do modelo proposto também se avaliou na Figura 42 o custo de processamento utilizado pela função RNA executada dentro do Tmote Sky. Para isso foi utilizada a equação 7.1 limitando I_{ihop} apenas à aferição de custos ligados aos componentes $CPUon$ e CPU_{idle} antes e depois da chamada de função de predição da RNA.

Na Figura 42 pode-se constatar que o uso da RNA não apresenta qualquer influência no aumento de processamento no Tmote Sky que mostrou um consumo energético estável entre 30 e 26 fJ com e sem a execução do modelo durante 250 transmissões. Esse comportamento é facilmente explicável tendo em vista que as RNAs inseridas no dispositivo se limitam apenas às funções lineares da $feed - forward$ com somas e multiplicação como expressa na equação 7.4. Isso potencializado pela convergência do modelo com apenas 1 neurônio na sua única camada escondida, deixando a função RNA com notação assintótica de grandeza $O(n^1)$.

A principal diferença entre o modelo proposto e das técnicas que se utilizam de funções matemáticas ou heurísticas é a capacidade da rede neural em seu modo $feed - forward$, após treinada previamente em um computador robusto, poder ser salva em um arquivo de 3,59 kB de memória ROM sem a necessidade de importação adicionais de bibliotecas de sistema a um custo computacional extremamente baixo como comprovado por este estudo. Mantendo suas predições fiéis aos resultados obtidos em treinamento mesmo em meio a novos cenários que se apresentaram na rede de sensores. Por sua vez, o dispositivo de baixo consumo de energia executou este modelo como uma função linear. Isso faz com que o modelo proposto supere ainda mais as técnicas de computação tradicionais.

8.3 RESUMO DO CAPÍTULO

Uma das contribuições desta pesquisa é a comprovação da possibilidade de inserção de um sistema complexo como Redes Neurais Artificiais em dispositivos de baixo processamento garantindo os bons resultados dessa técnica na predição de valores que ajudam o dispositivo sensor de baixo processamento realizar um controle inteligente de eficiência energética.

A qualidade das predições da RNA se mantivera o mesmo durante os experimentos simulados com diferentes parâmetros de cenários reais realizadas no emulador Tmote Sky, garantindo que as soluções encontradas não sofreram com os processos de adaptação do modelo para as limitações de hardware do sensor de 10 kB de RAM e 48 kB de ROM.

Resultados como a notação assintótica de grandeza $O(n^1)$ da RNA, seu baixo consumo de operação, entre 30 e 26 fJ e seu tamanho de 3,59 kB de memória ROM sem a necessidade de importação adicionais de bibliotecas de sistema garantem que é possível inserir um sistema complexo como uma RNA em outros dispositivos de especificações tão limitadas como a grande maioria dos dispositivos apresentados na Seção 2.6.

Podemos concluir também, com os resultados apresentados neste capitulo, que a RNA aprendeu o comportamento de transmissão da rede de sensores sem a necessidade de funções complexas de controle energético mostrando-se uma ótima solução identificando a influências existentes de varáveis controláveis como alcance de transmissão do dispositivo, taxa máxima de transmissão, número máximo de bits por quadro, número de saltos e potência de transmissão, além de variáveis não controláveis como tentativas de transmissão e retransmissão de múltiplos dispositivos simultâneos, tempo de *back-off* gasto, erros como colisão ou perda de pacote, atenuação do sinal por interferência causada por antenas de outros dispositivos.

9 CONCLUSÃO

> *Talvez não tenha conseguido fazer o melhor, mas lutei para que o melhor fosse feito. Não sou o que deveria ser, mas Graças a Deus, não sou o que era antes*
>
> Marthin Luther King,

As conclusões desse trabalho estão subdivididos em duas seções: 9.1 Regressão de MCE via RNA, fase 1 da pesquisa; 9.2 RNA para controle energético dentro de sensores, fase 2.

9.1 REGRESSÃO DE MCE VIA RNA

Primeiramente se propôs uma técnica para regressão simultânea de dois algoritmos iterativos complexos, a exemplo de modelos de consumo energético (MCE) para protocolos *slotted* ALOHA e CSMA/CA, utilizando de aprendizado de máquina com foco na redução do tempo e do custo de processamento utilizado para obtenção das solução (Barbosa et al., 2019; Barbosa; Menezes, 2017).

Neste contexto foi criada uma nova técnica, baseado em redes neurais artificiais, que visa aprender as relações de entrada/saída de diferentes modelos matemáticos, utilizados para gerenciamento de controle energético, como uma "caixa preta" para aprender os critérios de otimização e as relações que resultam no consumo de energia reduzido.

Para isso o modelo proposto utilizou 3 etapas: A etapa 1 definiu e implementou dois modelos clássicos de eficiência energética, capazes de otimizar simultaneamente parâmetros conflitantes como potência e taxa de transmissão, tamanho do pacote de dados e a distância entre os nós, nos protocolos *slotted* ALOHA e CSMA/CA. O modelo de consumo de energia proposta por (da Silva; de Moraes, 2020) foi utilizado como ponto de partida.

A fase 2 criou um banco de dados de treinamento a partir da combinação de todas variáveis nos MCEs escolhidos. Para isso, foi necessário criar um algoritmo de busca exaustiva, e assim encontrar as configurações de otimização global que encontrasse de forma exata a melhor representação que resulta no menor consumo de energia durante diferentes cenários de transmissão. A base de dados completa está registrada em (Barbosa et al., 2020).

Na fase 3, foi criado um modelo de regressão multiprotocolo, baseado em Redes Neurais Artificiais utilizando os parâmetros ótimos da fase dois, para simplificar a identificação de cada protocolo e otimizar os parâmetros com maior eficiência. Foi definido uma variável de controle denominada *protocolkey* (Pk) que permite a RNA identificar, durante o aprendizado, sob qual protocolo a rede WSN/IoT deve operar, para isso utiliza-se o identificador 0 para *slotted* Aloha e 1 para CSMA/CA.

Após extensos experimentos numéricos empregando dados simulados e parâmetros reais de rede, obtidos a partir de um modelo teórico rigoroso foi possível aproximar e otimizar funções exatas em casos onde os parâmetros ótimos não podem ser mapeados matematicamente. Também mostrou que, embora o treinamento de um RNA seja caro e complexo, uma vez a rede treinada, seu uso e consumo computacional diminui consideravelmente.

Também é importante notar que o uso de aprendizado de maquina permite obter resultados ideais com mais rapidez e eficiência. Dessa forma, obtivemos resultados tão bons quanto os das técnicas exatas com uma economia máxima de aproximadamente 97,1% no uso da CPU e 113.322.733% no tempo total de busca de soluções, podendo ser salva em um arquivos de 25 kB de memória ROM em média, sem a necessidade de executar processos complexos e demorados de busca de soluções, podendo obter resultados por predições mesmo em meio a novos cenários que se apresentam em redes reais.

Os resultados da primeira fase da pesquisa encorajou a uma evolução no estudo que estabeleceu como nova meta: a inserção da RNA em hardwares inferiores (empregados em sensores) aos utilizados nos testes da primeira fase da pesquisa.

9.2 RNA PARA CONTROLE ENERGÉTICO DENTRO DE SENSORES

A segunda fase propôs um modelo de consumo energético para dispositivo de baixo processamento que opere dentro do sensor, permitindo que ele gerencie de forma autônoma múltiplas variáveis relacionadas ao consumo de energia durante as transmissões.

Para possibilitar essas características a execução do modelo se deu em 4 fases: A fase 1 definiu o dispositivo utilizado como instância do problema selecionando o sensor Tmote Sky por sua limitação de hardware, compatibilidade com protocolos baseado em IPv6, sistema operacional programável, compatibilidade com um simulador de rede e interoperabilidade com outros dispositivos WSN/IoT.

A fase 2 criou uma banco de dados de treinamento a partir da combinação de diversas variáveis no dispositivo escolhido. Para isso, foi necessário submeter o dispositivo sensor a diferentes cenários de transmissão e aferir em simulador o consumo gasto em cada transmissão encontrando as configurações de otimização energética que resultaram no menor consumo de energia.

A fase 3, teve como principal objetivo criar a RNA do tipo MLP capaz de encontrar rapidamente as configurações PHY-MAC que resultem em um menor consumo energético, respeitando as limitações de processamento ($8MHz$), armazenamento ($48kB$ ROM e $10kB$ RAM), compatibilidade com o sistema operacional TinyOS e linguagem computacional NesC ou C definidas pelo fabricante.

A fase 4 exibiu os detalhes para viabilizar a implantação de RNA em um dispositivo sensor modelando a RNA em formato de uma função linear para inseri-la em dispositivos. Com isso foi possível inserir uma RNA em um dispositivo limitado, e criar um modelo de

regressão mesmo em casos onde há parâmetros que ainda não podem ser mapeados por funções matemáticas.

Embora o modelo apresentado conte com várias etapas que demandam um alto custo computacional, uma vez que a RNA é treinada, o consumo computacional utilizado para predições ou inferências torna-se equivalente ao de uma função linear com apenas operações de soma e multiplicação, facilmente inserida em dispositivos de limitadas capacidade e memória.

Os resultados comprovaram a proposta, a qualidade das predição da RNA se mantiveram o mesmo durante os experimentos simulados com diferentes parâmetros de cenários reais. A RNA inserida obteve uma notação assintótica de grandeza $O(n^1)$, consumo de operação entre 30 e 26 fJ e tamanho final de 3,59 kB de memória ROM sem a necessidade de importação adicionais de bibliotecas de sistema, mostrando que é possível inserir essa RNA em outros dispositivos de especificações tão limitadas quanto o sensor utilizado na pesquisa.

Pode-se concluir também, com os resultados apresentados, que a RNA compreendeu o comportamento de transmissão da rede de sensores sem a necessidade de funções complexas de controle energético mostrando-se uma ótima solução identificando influências existentes de variáveis controláveis como alcance de transmissão do dispositivo, taxa máxima de transmissão, número máximo de bits por quadro, número de saltos e potência de transmissão, além de variáveis não controláveis como tentativas de transmissão e retransmissão de múltiplos dispositivos simultâneos, tempo de *back-off* gasto, erros como colisão ou perda de pacote, atenuação do sinal por interferência causada por outros dispositivos.

A partir dos resultados apresentados na fase dois da pesquisa surgiram outras metas a serem desenvolvidas futuramente como expressa na seção 9.3.

9.3 TRABALHOS FUTUROS

Como uma evolução da pesquisa é possível vislumbrar os seguintes aspectos a serem tratados futuramente:

- **Análise da inteligência emergente da RNA no Mote**: Após testes realizados com a RNA no Tmote Sky constatou-se que, embora a RNA tenha sido projetada para otimização dos parâmetros individuais de cada dispositivo, a autonomia dos nós fez emergir um comportamento que buscou reduzir o consumo em todas transmissões da rede. Esse comportamento foi notado em especial na definição do tamanho do pacote que em todas simulações tendiam a diminuir à um tamanho estável entre 70 e 54 bits. Estudos mais complexos precisam ser realizados para avaliar as causas e o efeito desse comportamento em diferentes cenários e protocolos.

- **Adição de novos protocolos à RNA multi-protocolo**: Na primeira fase da

pesquisa foram utilizados os protocolos *slotted* Aloha e CSMA/CA, e uma vertente de evolução seria a inclusão de MCEs de outros protocolos de camada MAC à RNA a exemplo de tecnologias como 4G, 5G, IEEE 802.15.4, RFID, Bluetooth, WiFi para WSN, etc (seção 2.4.1) com outros modelos de desvanecimento substitutivo como Rice *fading*, por exemplo, substituindo o modelo Rayleigh utilizado (DELIBASIC, 2023).

- **Implantação de sistema de controle para MAC auto-adaptável**: Na primeira fase da pesquisa definiu-se a variável de controle Pk, como um parâmetro que identifica sob qual protocolo a rede WSN/IoT deveria operar com identificador 0 para *slotted* Aloha e 1 para CSMA/CA. Compreende-se como evolução a implantação em dispositivo um sistema de controle para o MAC auto-adaptável (SAML) que alterna dinamicamente os protocolos MAC sempre que a RNA identificar que a melhor escolha seja a alteração da variável Pk (SHA et al., 2013).

- **Criação de uma RNA multi-dispositivos**: Assim como foi possível a implantação de uma RNA multi-protocolo (fase 1) e a inserção de uma RNA em um dispositivo sensor (fase 2) pode-se definir como trabalhos futuros a criação de um sistema que alterne entre protocolo e dispositivo, para isso sugere-se a criação de uma variável de controle que, assim como Pk, seja utilizada para definir a qual dispositivo (WSN/IoT) a RNA deve operar, adaptando seus dados a diferentes protocolos e dispositivos a exemplo dos apresentados nas seções 2.4.1 e 2.6, respectivamente.

- **Possível Transformação da RNA em função matemática**: Resultados apresentados na seção 8.1 trazem evidencias que talvez exista uma relação linear entre as entradas distância entre os nós, taxa de transmissão e tamanho do pacote em dispositivos sensores Tmote Sky ainda não explorado na literatura. Uma vertente de pesquisa seria a tentativa de transformação da RNA de um neurônio em uma função matemática removendo assim múltiplas variáveis utilizada na matriz de peso e *bias* da RNA, simplificando mais ainda o modelo proposto.

9.4 OUTRAS CONTRIBUIÇÕES

- Barbosa, P. F., da Silva, B. A., Macedo, D., Zanchettin, C., & de Moraes, R. M. Otimização do Consumo de Energia em Redes Ad Hoc Aloha Empregando Deep Learning. In: Anais do XVIII Workshop em Desempenho de Sistemas Computacionais e de Comunicação (WPerformance). SBC, 2019.

- Barbosa, P. F., da Silva, B. A., Zanchettin, C., & de Moraes, R. M. *Energy Consumption Optimization for CSMA/CA Protocol Employing Machine Learning*. In: IEEE Vehicular Technology Conference: VTC2020-Spring, Antwerp, Belgium, 2020.

- Barbosa, P. F. C.; Silva, B. A. d.; Zanchettin, C.; de Moraes, R. M. *Energy Consumption CSMA/CA and slotted ALOHA (-AMPS)*. IEEE Dataport, 2020. Disponível em: http://dx.doi.org/10.21227/99vd-vh94

- P. F. C. Barbosa, B. A. d. Silva, C. Zanchettin eR. M. de Moraes. *Software to Energy Consumption Optimization for CSMA/CA Protocol Employing Machine Learning*. 2020. CIN - Centro de Informática da UFPE. BR512020001888-2, 29 jan. 2020, 25 maio 2020.

- Premiação de um hardware virtual no *NVIDIA Academic Hardware Grant Program*, 2021. Disponível em: https://portal.cin.ufpe.br/2021/09/20/doutorando-do-cin-tem-proposta-premiada-pela-nvidia-e-ganha-500-horas-de-computacao-no-sistema-cloud-da-saturncloud/

- O artigo intitulado *A Multi-protocol Energy Optimization Method for an Adaptable Wireless MAC System through Machine Learning*, encontra-se submetido a revista *Springer Nature Annals of Telecommunications*, aguardando pela resposta dos revisores.

- O artigo intitulado *Neural Network inside Sensors for Intelligent Power Control*, encontra-se em revisão para submissão.

- Barbosa, P. F. C.; Zanchettin, C.; de Moraes, R. M. *Energy Consumption Tmote Sky Ultra low power IEEE 802.15.4*. IEEE Dataport, 2023. Disponível em: https://dx.doi.org/10.21227/xsph-hf71

9.5 CONSIDERAÇÕES

Como um motivacional para pesquisas em áreas correlatas de aprendizado de máquina e rede de computadores apresenta-se algumas dificuldades encontradas durante a execução da pesquisa entre elas destacam-se:

1. Falhas técnicas em rede elétrica e rede de computadores e a necessidade de upgrade de hardware em especial para as fases de formação da base de dados e treinamento da RNA que ocasionaram a perca de dados gerados durante 21 dias de execução do modelo.

2. Pandemia COVID-19, que resultou na impossibilidade de reuniões presenciais para acompanhamento detalhado da pesquisa.

3. Delonga na geração da base de dados, requisitando o uso de bibliotecas de paralelização de hardware a exemplo da *njit*.

4. Delonga nos testes de definição das configurações de hiper-parâmetros e parâmetros utilizados no treinamento das RNAs.

5. Falta de material bibliográfico na área de aprendizado de máquina para dispositivos sensores.

6. Delonga na modelagem matemático-computacional do sistema final da RNA_{Mote} implementado em hardware, requisitando testes de inúmeras bibliotecas baseadas em .py e .c .

A existência dos problemas relatados não atrapalhou diretamente no cronograma final utilizado para a conclusão doutorado.

9.6 RESUMO DO CAPÍTULO

Define-se como principal contribuição desta pesquisa de doutorado a possibilidade de inserção de um sistema complexo como Redes Neurais Artificiais em dispositivos de baixo processamento garantindo um controle inteligente da eficiência energética multi-protocolo.

Após todos os testes realizados e apresentados nesta tese, sugerimos o uso da fase 1 da pesquisa quando o Modelo de Consumo de Energia (MCE) apresentar dados confiáveis quanto ao consumo de energia durante as transmissões, tendo em vista que a extração de dados de uma função iterativa é mais rápida que a obtenção de dados aferidos em simuladores. No entanto, como nem sempre é possível a obtenção de funções complexas de mapeamento, indicamos a fase 2 da pesquisa como estratégia generalista possível de ser reproduzida em qualquer dispositivo independentemente de protocolo pois, esse método consegue englobar os comportamentos controláveis e não controláveis que aparecem durante transmissões.

Entre as limitações da proposta que ainda precisam ser estudadas estão o aprofundamento da analise comportamental da inteligência emergente no mote, estudos em cenários onde apenas alguns dos nós possuem inteligência, cenários com diferentes modelos de dispositivos atuando sob uma RNA, entre outros estudos que, embora fujam do escopo inicial da pesquisa, requisitavam de maior detalhamento e aprendizado.

REFERÊNCIAS

Aakash, R. S.; Nishanth, M.; Rajageethan, R.; Rao, R.; Ezhilarasie, R. Data mining approach to predict forest fire using fog computing. In: *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*. [S.l.: s.n.], 2018. p. 1582–1587.

Abdelouahid, R. A.; Oqaidi, M.; Marzak, A. Towards to a new iot interoperability architecture. In: *2018 IEEE International Conference on Technology Management, Operations and Decisions (ICTMOD)*. [S.l.: s.n.], 2018. p. 148–154.

Abedin, S. F.; Alam, M. G. R.; Haw, R.; Hong, C. S. A system model for energy efficient green-iot network. In: *2015 International Conference on Information Networking (ICOIN)*. [S.l.: s.n.], 2015. p. 177–182.

Affonso, E. T.; Silva, A. M.; Silva, M. P.; Rodrigues, T. M.; Moita, G. F. Uso redes neurais multilayer perceptron (mlp) em sistema de bloqueio de websites baseado em conteúdo. *Mecánica Computacional*, v. 29, n. 93, p. 9075–9090, 2010.

Agarap, A. F. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

Ahmadi, H.; Bouallegue, R. Comparative study of learning-based localization algorithms for wireless sensor networks: Support vector regression, neural network and naïve bayes. In: *2015 International Wireless Communications and Mobile Computing Conference (IWCMC)*. [S.l.: s.n.], 2015. p. 1554–1558.

Ahmed, T.; Bin Nuruddin, A. T.; Latif, A. B.; Arnob, S. S.; Rahman, R. A real-time controlled closed loop iot based home surveillance system for android using firebase. In: *2020 6th International Conference on Control, Automation and Robotics (ICCAR)*. [S.l.: s.n.], 2020. p. 601–606.

Al-Taleb, N.; Min-Allah, N. A study on internet of things operating systems. In: *2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*. [S.l.: s.n.], 2019. p. 1–7.

Aljarrah, E.; Yassein, M. B.; Aljawarneh, S. Routing protocol of low-power and lossy network: Survey and open issues. In: *2016 International Conference on Engineering MIS (ICEMIS)*. [S.l.: s.n.], 2016. p. 1–6.

Alom, M. Z.; Taha, T. M.; Yakopcic, C.; Westberg, S.; Sidike, P.; Nasrin, M. S.; Van Esesn, B. C.; Awwal, A. A. S.; Asari, V. K. The history began from alexnet: A comprehensive survey on deep learning approaches. *arXiv preprint arXiv:1803.01164*, 2018.

Alves, C. P.; da Silva Filho, J. B. Um estudo sobre o modelo zigbee de rede sem fio ieee 802.15. 4. *Revista Eletrônica de Iniciação Científica em Computação*, v. 18, n. 2, 2020.

Amirinasab Nasab, M.; Shamshirband, S.; Chronopoulos, A. T.; Mosavi, A.; Nabipour, N. Energy-efficient method for wireless sensor networks low-power radio operation in internet of things. *Electronics*, MDPI, v. 9, n. 2, p. 320, 2020.

Amjad, M.; Sharif, M.; Afzal, M. K.; Kim, S. W. Tinyos-new trends, comparative views, and supported sensing applications: A review. *IEEE Sensors Journal*, v. 16, n. 9, p. 2865–2889, 2016.

Arellanes, D.; Lau, K. Decentralized data flows in algebraic service compositions for the scalability of iot systems. In: *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*. [S.l.: s.n.], 2019. p. 668–673.

Athreya, A. P.; Chan-Maestas, H.; Katz, E.; Tague, P.; Iannucci, B. Energy-governed resilient networked systems. In: *2014 IEEE 11th Consumer Communications and Networking Conference (CCNC)*. [S.l.: s.n.], 2014. p. 56–61.

Baccelli, E.; Hahm, O.; Günes, M.; Wählisch, M.; Schmidt, T. C. Riot os: Towards an os for the internet of things. In: *2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. [S.l.: s.n.], 2013. p. 79–80.

Baert, M.; Camerlynck, P.; Crombez, P.; Hoebeke, J. A ble-based multi-gateway network infrastructure with handover support for mobile ble peripherals. In: *2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*. [S.l.: s.n.], 2019. p. 91–99.

BALEVI, E.; RABEE, F. T. A.; GITLIN, R. D. Aloha-noma for massive machine-to-machine iot communication. In: *2018 IEEE International Conference on Communications (ICC)*. [S.l.: s.n.], 2018. p. 1–5.

Barbosa, P. F.; da Silva, B. A.; Macedo, D.; Zanchettin, C.; de Moraes, R. M. Otimização do consumo de energia em redes ad hoc aloha empregando deep learning. In: SBC. *Anais do XVIII Workshop em Desempenho de Sistemas Computacionais e de Comunicação*. [S.l.], 2019.

Barbosa, P. F.; Zanchettin, C.; Mariz, R. *Energy Consumption Tmote Sky Ultra low power IEEE 802.15.4*. IEEE Dataport, 2023. Disponível em: <https://dx.doi.org/10.21227/xsph-hf71>.

Barbosa, P. F. C.; Menezes, L. Spea — ma: A social evolutionary algorithm for managing multiple transmissions. In: *2017 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*. [S.l.: s.n.], 2017. p. 1–7.

Barbosa, P. F. C.; Silva, B. A. d.; Zanchettin, C.; de Moraes, R. M. *Energy Consumption CSMA/CA and slotted ALOHA (-AMPS)*. IEEE Dataport, 2020. Disponível em: <http://dx.doi.org/10.21227/99vd-vh94>.

Barry, R. Freertos, http. *www.freertos.org*, 2020.

Basirat, M.; Roth, P. M. The quest for the golden activation function. *arXiv preprint arXiv:1808.00783*, 2018.

Beniwal, R.; Nikolova, K.; Iliev, G. Performance analysis of mm-speed routing protocol implemented in 6lowpan environment. In: *2019 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*. [S.l.: s.n.], 2019. p. 1–5.

Bibri, S. E. The iot for smart sustainable cities of the future: An analytical framework for sensor-based big data applications for environmental sustainability. *Sustainable cities and society*, Elsevier, v. 38, p. 230–253, 2018.

BONACCORSO, G. *Machine learning algorithms*. [S.l.]: Packt Publishing Ltd, 2017.

Breiman, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001.

Cao, Y.; Gu, Q. Generalization error bounds of gradient descent for learning over-parameterized deep relu networks. In: *AAAI*. [S.l.: s.n.], 2020. p. 3349–3356.

CARRION, P.; QUARESMA, M. Internet da coisas (iot): Definições e aplicabilidade aos usuários finais. *Human Factors in Design*, v. 8, n. 15, p. 049–066, 2019.

Challouf, S.; Kriaa, L.; Saidane, L. A. Power consumption comparison of synchronized iot devices running freertos and riot. In: *2019 8th International Conference on Performance Evaluation and Modeling in Wired and Wireless Networks (PEMWN)*. [S.l.: s.n.], 2019. p. 1–5.

Chanama, L.; Wongwirat, O. A comparison of decision tree based techniques for indoor positioning system. In: *2018 International Conference on Information Networking (ICOIN)*. [S.l.: s.n.], 2018. p. 732–737.

Cho, C.; Chen, W.; Liao, C. C.; Liao, J.; Chen, F.; Shen, T. Building on the distributed energy resources iot based iec 61850 xmpp for tpc. In: *2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS)*. [S.l.: s.n.], 2019. p. 61–66.

Choi, Y. Os-aware interaction model for the verification of multitasking embedded software. *IEEE Access*, v. 8, p. 134987–134999, 2020.

CHOUDHURY, G.; RAPPAPORT, S. Diversity aloha-a random access scheme for satellite communications. *IEEE Transactions on Communications*, IEEE, v. 31, n. 3, p. 450–457, 1983.

Choudhury, N.; Matam, R.; Mukherjee, M.; Lloret, J. A performance-to-cost analysis of ieee 802.15.4 mac with 802.15.4e mac modes. *IEEE Access*, v. 8, p. 41936–41950, 2020.

Chuang, Y.; Wang, T.; Chuang, C.; Chen, H. A rpl-based self-organization mechanism for wireless sensor networks. In: *2017 IEEE International Symposium on Consumer Electronics (ISCE)*. [S.l.: s.n.], 2017. p. 74–75.

Chung, C.; Jeong, J. P. A design of iot device configuration translator for intent-based iot-cloud services. In: *2020 22nd International Conference on Advanced Communication Technology (ICACT)*. [S.l.: s.n.], 2020. p. 52–56.

CONTIKI, O. The open source os for the internet of things. 2012. Acessado em 18 de abril de 2022. Disponível em: <http://www.contiki-os.org>.

da Silva, B. A. *Consumo de energia ótimo em redes Ad Hoc ALOHA e Csma/Ca considerando as camadas física e de enlace*. Dissertação (Mestrado) — Universidade Federal de Pernambuco, 2019.

Da Silva, B. A.; De Moraes, R. M. Um estudo sobre o consumo de energia em redes ad hoc lineares aloha com saltos equidistantes. In: *Anais do XXXVI Simpósio Brasileiro de Telecomunicações e Processamento de Sinais - Diponível em: https://www.dropbox.com/s/89k2t7swo064afk/Anais_SBrT_2018_v2.pdf?dl=0*. Campina Grande, PB: [s.n.], 2018.

da Silva, B. A.; de Moraes, R. M. Optimized energy consumption for slotted aloha and csma/ca multihop ad hoc networks. *Journal of Communication and Information Systems*, v. 35, n. 1, p. 189–197, 2020.

DATASHEET, T. S. Moteiv corporation. *Saatavissa (viitattu 1.10. 2017)*, 2006. Acessado em 18 de abril de 2022. Disponível em: <http://www.crew-project.eu/sites/default/files/tmote-sky-datasheet.pdf>.

Datta, P.; Sharma, B. A survey on iot architectures, protocols, security and smart city based applications. In: *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. [S.l.: s.n.], 2017. p. 1–5.

de Almeida Florencio, F.; Moreno, E. D.; Teixeira Macedo, H.; de Britto Salgueiro, R. J. P.; Barreto do Nascimento, F.; Oliveira Santos, F. A. Intrusion detection via mlp neural network using an arduino embedded system. In: *2018 VIII Brazilian Symposium on Computing Systems Engineering (SBESC)*. [S.l.: s.n.], 2018. p. 190–195.

De León-Aldaco, S. E.; Calleja, H.; Aguayo Alquicira, J. Metaheuristic optimization methods applied to power converters: A review. *IEEE Transactions on Power Electronics*, v. 30, n. 12, p. 6791–6803, 2015.

DELIBASIC, M. Performance improvements for small-cells with cooperative diversity in rayleigh/rice fading environment. Research Square, 2023. Disponível em: <https://doi.org/10.21203/rs.3.rs-1386095/v1>.

Derhamy, H.; Eliasson, J.; Delsing, J. Iot interoperability—on-demand and low latency transparent multiprotocol translator. *IEEE Internet of Things Journal*, v. 4, n. 5, p. 1754–1763, 2017.

Dey, A.; Stuart, K.; Tolentino, M. E. Characterizing the impact of topology on iot stream processing. In: *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*. [S.l.: s.n.], 2018. p. 505–510.

Dogo, E. M.; Afolabi, O. J.; Nwulu, N. I.; Twala, B.; Aigbavboa, C. O. A comparative analysis of gradient descent-based optimization algorithms on convolutional neural networks. In: *2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)*. [S.l.: s.n.], 2018. p. 92–99.

dos Santos, W. G. V.; Costa, W. S.; Faber, M. J.; Silva, J. A. L.; Rocha, H. R. O.; Segatto, M. E. V. Sensor allocation in a hybrid star-mesh iot network using genetic algorithm and k-medoids. In: *2019 IEEE Latin-American Conference on Communications (LATINCOM)*. [S.l.: s.n.], 2019. p. 1–6.

Duchi, J.; Hazan, E.; Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, v. 12, n. 7, 2011.

Dunkels, A.; Gronvall, B.; Voigt, T. Contiki - a lightweight and flexible operating system for tiny networked sensors. In: *29th Annual IEEE International Conference on Local Computer Networks*. [S.l.: s.n.], 2004. p. 455–462.

ECKLE, K.; SCHMIDT-HIEBER, J. A comparison of deep networks with relu activation function and linear spline-type methods. *Neural Networks*, Elsevier, v. 110, p. 232–242, 2019.

Ejaz, M.; Kumar, T.; Ylianttila, M.; Harjula, E. Performance and efficiency optimization of multi-layer iot edge architecture. In: *2020 2nd 6G Wireless Summit (6G SUMMIT)*. [S.l.: s.n.], 2020. p. 1–5.

Elbanoby, Y.; Aborizka, M.; Maghraby, F. Real-time data management for iot in cloud environment. In: *2019 IEEE Global Conference on Internet of Things (GCIoT)*. [S.l.: s.n.], 2019. p. 1–7.

ERICSSON, M. R. 5g uptake even faster than expected. *acessado em Janeiro*, 2023. Disponível em: <www.ericsson.com/49d1d9/assets/local/mobilityâĂŘreport/documents/2019/ericssonâĂŘmobilityâĂŘreportâĂŘjuneâĂŘ2019.pdf>.

ESWARAN, A.; ROWE, A.; RAJKUMAR, R. Nano-rk: an energy-aware resource-centric rtos for sensor networks. In: IEEE. *26th IEEE International Real-Time Systems Symposium (RTSS'05)*. [S.l.], 2005. p. 10–pp.

Fadlullah, Z.; Tang, F.; Mao, B.; Kato, N.; Akashi, O.; Inoue, T.; Mizutani, K. State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems. *IEEE Communications Surveys & Tutorials*, IEEE, v. 19, n. 4, p. 2432–2455, 2017.

Faid, A.; Sadik, M.; Sabir, E. Iot-based low cost architecture for smart farming. In: IEEE. *2020 International Wireless Communications and Mobile Computing (IWCMC)*. [S.l.], 2020. p. 1296–1302.

Farhan, L.; Shukur, S. T.; Alissa, A. E.; Alrweg, M.; Raza, U.; Kharel, R. A survey on the challenges and opportunities of the internet of things (iot). In: *2017 Eleventh International Conference on Sensing Technology (ICST)*. [S.l.: s.n.], 2017. p. 1–5.

Farshchi, S.; Nuyujukian, P. H.; Pesterev, A.; Mody, I.; Judy, J. W. A tinyos-enabled mica2-basedwireless neural interface. *IEEE Transactions on Biomedical Engineering*, v. 53, n. 7, p. 1416–1424, 2006.

Ferreira, L. N.; Nogueira, J.-M.; Macedo, D. F. Distribuição de carga de processamento para dispositivos iot e smartphones com arquitetura multi-nível. In: SBC. *Anais Principais do XXXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*. [S.l.], 2019. p. 566–579.

Forouzan, B. A. *Comunicação de dados e redes de computadores*. [S.l.]: AMGH Editora, 2009.

Gao, J. Analysis of energy consumption for ad hoc wireless sensor networks using a bit-meter-per-joule metric. *IPN Progress Report*, Jet Propulsion Laboratory, v. 42, n. 150, p. 1–19, 2002.

Gardasevic, G.; Plavsic, P.; Vasilievic, D. Experimental iot testbed for testing the 6tisch and rpl coexistence. In: *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. [S.l.: s.n.], 2019. p. 917–918.

Gerasimov, V.; Bender, W. Things that talk: using sound for device-to-device and device-to-human communication. *IBM Systems Journal*, IBM, v. 39, n. 3.4, p. 530–546, 2000.

GÉRON, A. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. [S.l.]: "O'Reilly Media, Inc.", 2022.

Ghazanfar, S.; Hussain, F.; Rehman, A. U.; Fayyaz, U. U.; Shahzad, F.; Shah, G. A. Iot-flock: An open-source framework for iot traffic generation. In: *2020 International Conference on Emerging Trends in Smart Technologies (ICETST)*. [S.l.: s.n.], 2020. p. 1–6.

Ghobad, P. C.; de Moraes, R. M. Ber analysis with an appropriate friis formula for multi-hop aloha dense ad hoc networks. In: *Vehicular Technology Conference (VTC Fall)*. Quebec City, Canada: [s.n.], Sep. 2012.

Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. [S.l.: s.n.], 2010. p. 249–256.

Goldsmith, A. *Wireless communications*. [S.l.]: Cambridge university press, 2005.

GONÇALVES, M. L.; NETTO, M. d. A.; JR, J. Z.; COSTA, J. A. F. Classificação não-supervisionada de imagens de sensores remotos utilizando redes neurais auto-organizáveis e métodos de agrupamentos hierárquicos. *Revista Brasileira de Cartografia*, v. 60, n. 1, p. 17–29, 2008.

Goodfellow, I.; Bengio, Y.; Courville, A. Deep learning. book in preparation for mit press. *URL¡ http://www. deeplearningbook. org*, v. 1, 2016.

Goodfellow, I.; Bengio, Y.; Courville, A.; Bengio, Y. *Deep learning*. [S.l.]: MIT press Cambridge, 2016. v. 1.

Gorce, J.-M.; Zhang, R.; Parvery, H. Impact of radio link unreliability on the connectivity of wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking*, Springer, n. 1, p. 1–16, 2007. Doi: 10.1155/2007/19196.

Guimaraes, V. G.; Moraes, R. M. de; Obraczka, K.; Bauchspiess, A. A novel iot protocol architecture: Efficiency through data and functionality sharing across layers. In: IEEE. *2019 28th International Conference on Computer Communication and Networks (ICCCN)*. [S.l.], 2019. p. 1–9.

Hamad, S. A.; Sheng, Q. Z.; Zhang, W. E.; Nepal, S. Realizing an internet of secure things: A survey on issues and enabling technologies. *IEEE Communications Surveys Tutorials*, v. 22, n. 2, p. 1372–1391, 2020.

Hartigan, J. A. *Bayes theory*. [S.l.]: Springer Science & Business Media, 2012.

Hecht-Nielsen, R. Theory of the backpropagation neural network. In: *Neural networks for perception*. [S.l.]: Elsevier, 1992. p. 65–93.

Hejazi, H.; Rajab, H.; Cinkler, T.; Lengyel, L. Survey of platforms for massive iot. In: *2018 IEEE International Conference on Future IoT Technologies (Future IoT)*. [S.l.: s.n.], 2018. p. 1–8.

Homem, W. L.; Ufes, P. E. M. Apostila de machine learning. 2020.

Hu, L.-Y.; Huang, M.-W.; Ke, S.-W.; TsaI, C.-F. The distance function effect on k-nearest neighbor classification for medical datasets. *SpringerPlus*, SpringerOpen, v. 5, n. 1, p. 1–9, 2016.

Huan, Y.; Qin, Y.; YOU, Y.; ZHENG, L.; ZOU, Z. A multiplication reduction technique with near-zero approximation for embedded learning in iot devices. In: *Proc. of IEEE SOCC*. Seattle, USA: [s.n.], Sep. 2016.

Islam, M. S.; Verma, H.; Khan, L.; Kantarcioglu, M. Secure real-time heterogeneous iot data management system. In: *2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*. [S.l.: s.n.], 2019. p. 228–235.

Isolani, P. H.; Claeys, M.; Donato, C.; Granville, L. Z.; Latré, S. A survey on the programmability of wireless mac protocols. *IEEE Communications Surveys Tutorials*, v. 21, n. 2, p. 1064–1092, 2019.

Jacobs, R. A. Increased rates of convergence through learning rate adaptation. *Neural networks*, Elsevier, v. 1, n. 4, p. 295–307, 1988.

Javed, F.; Afzal, M. K.; Sharif, M.; Kim, B. Internet of things (iot) operating systems support, networking technologies, applications, and challenges: A comparative review. *IEEE Communications Surveys Tutorials*, v. 20, n. 3, p. 2062–2100, 2018.

Jo 3. BÍblia, n. t. *Tradução Almeida Corrigida Fiel*, Sagrada Bíblia: Antigo e Novo Testamentos, 16.

Johnson, M.; Healy, M.; Van de Ven, P.; Hayes, M. J.; Nelson, J.; Newe, T.; Lewis, E. A comparative review of wireless sensor network mote technologies. In: IEEE. *SENSORS, 2009 IEEE*. [S.l.], 2009. p. 1439–1442.

JUNG, C. F. *Metodologia para pesquisa e desenvolvimento: aplicada a novas tecnologias, produtos e processos*. [S.l.]: Axcel Books, 2004.

JUNIOR, C. M. L.; CASELLA, I. R.; CAMBERO, E. V.; CAPOVILLA, C. E. Comparison between po, pso, and fpa techniques applied to mppt of a low-power photovoltaic system for lpwa devices. In: SPRINGER. *Proceedings of the 7th Brazilian Technology Symposium (BTSym'21) Emerging Trends in Systems Engineering Mathematics and Physical Sciences, Volume 2*. [S.l.], 2022. p. 216–228.

Karl, H.; Willig, A. *Protocols and architectures for wireless sensor networks*. [S.l.]: John Wiley & Sons, 2007.

Khan, N.; Sakib, N.; Jerin, I.; Quader, S.; Chakrabarty, A. Performance analysis of security algorithms for iot devices. In: *2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*. [S.l.: s.n.], 2017. p. 130–133.

Kingma, D. P.; Mohamed, S.; Rezende, D. J.; Welling, M. Semi-supervised learning with deep generative models. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2014. p. 3581–3589.

KO, L.; LIU, Y.; FANG, H. Design and implementation of ieee 802.15.4 beacon-enabled network devices. In: *Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06)*. [S.l.: s.n.], 2006. p. 5 pp.–419.

Kotstein, S.; Decker, C. Reinforcement learning for iot interoperability. In: *2019 IEEE International Conference on Software Architecture Companion (ICSA-C)*. [S.l.: s.n.], 2019. p. 11–18.

Krishna, K.; Narasimha Murty, M. Genetic k-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, v. 29, n. 3, p. 433–439, 1999.

Kumar, N. V. R.; Bhuvana, C.; Anushya, S. Comparison of zigbee and bluetooth wireless technologies-survey. In: *2017 International Conference on Information Communication and Embedded Systems (ICICES)*. [S.l.: s.n.], 2017. p. 1–4.

Li, L.; Xiaoguang, H.; Ke, C.; Ketai, H. The applications of wifi-based wireless sensor network in internet of things and smart grid. In: *2011 6th IEEE Conference on Industrial Electronics and Applications*. [S.l.: s.n.], 2011. p. 789–793.

LIASHCHYNSKYI, P.; LIASHCHYNSKYI, P. Grid search, random search, genetic algorithm: a big comparison for nas. *arXiv preprint arXiv:1912.06059*, 2019.

Liu, X.; Sheng, Z.; Yin, C.; Ali, F.; Roggen, D. Performance analysis of routing protocol for low power and lossy networks (rpl) in large scale networks. *IEEE Internet of Things Journal*, v. 4, n. 6, p. 2172–2185, 2017.

Liu, Y.; Zhang, D.; Gooi, H. B. Optimization strategy based on deep reinforcement learning for home energy management. *CSEE Journal of Power and Energy Systems*, v. 6, n. 3, p. 572–582, 2020.

Locatelli, M. A note on the griewank test function. *Journal of global optimization*, Springer, v. 25, n. 2, p. 169–174, 2003.

Luong, N. C.; Hoang, D. T.; Gong, S.; Niyato, D.; Wang, P.; Liang, Y.; Kim, D. I. Applications of deep reinforcement learning in communications and networking: A survey. *IEEE Communications Surveys Tutorials*, v. 21, n. 4, p. 3133–3174, 2019.

Luxhøj, J. T.; Williams, T. P.; Shyur, H.-J. Comparison of regression and neural network models for prediction of inspection profiles for aging aircraft. *IIE transactions*, Springer, v. 29, n. 2, p. 91–101, 1997.

MAHESH, B. Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, v. 9, p. 381–386, 2020.

Mao, Q.; Hu, F.; Hao, Q. Deep learning for intelligent wireless networks: A comprehensive survey. *IEEE Commun. Surveys Tuts.*, IEEE, v. 20, n. 4, p. 2595–2621, 2018.

MARTINS, G. N.; COSTA, A. F. da; GOMES, R. C.; ALENCAR, M. S. de. Análise comparativa entre testbed e simulador em redes de sensores sem fio ieee 802.15. 4 com unslotted csma/ca. *Revista de Tecnologia da Informação e Comunicação*, v. 7, n. 2, p. 41–48, 2017.

Mavrogiorgou, A.; Kiourtis, A.; Kyriazis, D. A comparative study of classification techniques for managing iot devices of common specifications. In: Pham, C.; Altmann, J.; Bañares, J. Á. (Ed.). *Economics of Grids, Clouds, Systems, and Services*. Cham: Springer International Publishing, 2017. p. 67–77. ISBN 978-3-319-68066-8.

Mbacke, A. A.; Mitton, N.; Rivano, H. A survey of rfid readers anticollision protocols. *IEEE Journal of Radio Frequency Identification*, v. 2, n. 1, p. 38–48, 2018.

McCulloch, W. S.; Pitts, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, Springer, v. 5, n. 4, p. 115–133, 1943.

MEDINA, B. E. Internet das coisas em edifícios inteligentes: desenvolvimento de uma rede de sensores e atuadores sem fio para o controle de sistemas de climatização. *Campinas: Tese (Mestrado)-Programa de Pós-Graduação em Engenharia Elétrica da Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas, São Paulo*, 2017.

Memeti, S.; Pllana, S.; Binotto, A.; Kołodziej, J.; Brandic, I. Using meta-heuristics and machine learning for software optimization of parallel computing systems: a systematic literature review. *Computing*, Springer, v. 101, n. 8, p. 893–936, 2019.

Mengmeng, Z.; Yian, L. Signal sorting using teaching-learning-based optimization and random forest. In: *2018 17th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES)*. [S.l.: s.n.], 2018. p. 258–261.

Mikolov, T.; Deoras, A.; Povey, D.; Burget, L.; Černocký, J. Strategies for training large scale neural network language models. In: *2011 IEEE Workshop on Automatic Speech Recognition Understanding*. [S.l.: s.n.], 2011. p. 196–201.

Mohammadi, M.; Al-Fuqaha, A.; Sorour, S.; Guizani, M. Deep learning for iot big data and streaming analytics: A survey. *IEEE Commun. Surveys Tuts.*, IEEE, 2018.

Mugdhe, S. P. Contiki tutorials network protocol stack (netstack). *acessado em https://anrg.usc.edu/contiki/index.php/Network$_S$tack*, 2012.

Muhammad, A.; Afzal, B.; Imran, B.; Tanwir, A.; Akbar, A. H.; Shah, G. onem2m architecture based secure mqtt binding in mbed os. In: *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS PW)*. [S.l.: s.n.], 2019. p. 48–56.

Mukhopadhyay, A.; Das, G. Low complexity fair scheduling in lte/lte-a uplink involving multiple traffic classes. *IEEE Systems Journal*, p. 1–12, 2020.

Munari, A.; Clazzer, F.; Liva, G. Multi-receiver aloha systems - a survey and new results. In: *2015 IEEE International Conference on Communication Workshop (ICCW)*. [S.l.: s.n.], 2015. p. 2108–2114.

Nair, V.; Hinton, G. E. Rectified linear units improve restricted boltzmann machines. In: *ICML*. [S.l.: s.n.], 2010.

NAMISLO, C. Analysis of mobile radio slotted aloha networks. *IEEE Journal on Selected Areas in Communications*, v. 2, n. 4, p. 583–588, 1984.

Neshenko, N.; Bou-Harb, E.; Crichigno, J.; Kaddoum, G.; Ghani, N. Demystifying iot security: An exhaustive survey on iot vulnerabilities and a first empirical look on internet-scale iot exploitations. *IEEE Communications Surveys Tutorials*, v. 21, n. 3, p. 2702–2733, 2019.

Niu, L.; Liu, D. Performance evaluation of unslotted csma/ca algorithm in wireless sensor networks. In: *2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*. [S.l.: s.n.], 2020. p. 971–974.

NORIEGA, L. Multilayer perceptron tutorial. *School of Computing. Staffordshire University*, Citeseer, v. 4, p. 5, 2005.

Nwankpa, C.; Ijomah, W.; Gachagan, A.; Marshall, S. Activation functions: Comparison of trends in practice and research for deep learning. *arXiv preprint arXiv:1811.03378*, 2018.

OLIVEIRA, S. S. T. de; VAGNER, J.; MARTINS, W. S.; PALMEIRAS, A.; QUADRA, D.; SAMAMBAIA, C. Smart: Uso de aprendizado de máquina para filtragem e recuperação eficiente de dados espaciais e temporais em big data. In: *SBBD*. [S.l.: s.n.], 2020. p. 85–96.

Osório, F. S. Redes neurais artificiais: do aprendizado ao aprendizado artificial. *I Fórum de Inteligência Artificial, Canoas*, 1999.

Owada, Y.; Sato, G.; Temma, K.; Kuri, T.; Inoue, M.; Nagano, T. An implementation of layer 2 overlay mesh network and edge computing platform for iot. In: *2019 Twelfth International Conference on Mobile Computing and Ubiquitous Network (ICMU)*. [S.l.: s.n.], 2019. p. 1–6.

PANAHI, U.; BAYILMIŞ, C. Enabling secure data transmission for wireless sensor networks based iot applications. *Ain Shams Engineering Journal*, Elsevier, v. 14, n. 2, p. 101866, 2023.

Panchal, G.; Ganatra, A.; Shah, P.; Panchal, D. Determination of over-learning and over-fitting problem in back propagation neural network. *International Journal on Soft Computing*, Academy & Industry Research Collaboration Center(AIRCC), v. 2, n. 2, p. 40–51, 2011.

Panichpapiboon, S.; Ferrari, G.; Tonguz, O. K. Sensor networks with random versus uniform topology: Mac and interference considerations. In: *IEEE Vehicular Technology Conference (VTC Spring)*. Milan, Italy: [s.n.], May. 2004.

Patel, P. P.; Jhaveri, R. H. Soft computing techniques to address various issues in wireless sensor networks: A survey. In: *2016 International Conference on Computing, Communication and Automation (ICCCA)*. [S.l.: s.n.], 2016. p. 399–404.

Pechetti, S. V.; Jindal, A.; Bose, R. Channel-based mapping diversity for enhancing the physical layer security in the internet of things. In: *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. [S.l.: s.n.], 2017. p. 1–6.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V. et al. Scikit-learn: Machine learning in python. *J. Mach. Learn. Research*, v. 12, n. Oct, p. 2825–2830, 2011.

Perkov, T. The mcculloch–pitts paper from the perspective of mathematical logic. In: _____. *Guide to Deep Learning Basics: Logical, Historical and Philosophical Perspectives*. Cham: Springer International Publishing, 2020. p. 7–12. ISBN 978-3-030-37591-1. Disponível em: <https://doi.org/10.1007/978-3-030-37591-1_2>.

Perrone, L. F.; Nicol, D. M. A scalable simulator for tinyos applications. In: *Proceedings of the Winter Simulation Conference*. [S.l.: s.n.], 2002. v. 1, p. 679–687 vol.1.

Pessacg, F.; Nitsche, M.; Teijeiro, A.; Martın, D.; Cristoforis, P. D. Open-source embedded framework for unmanned ground vehicle control using ciaa. In: *2017 Eight Argentine Symposium and Conference on Embedded Systems (CASE)*. [S.l.: s.n.], 2017. p. 1–6.

Rajakumar, R.; Dhavachelvan, P.; Vengattaraman, T. A survey on nature inspired meta-heuristic algorithms with its domain specifications. In: *2016 International Conference on Communication and Electronics Systems (ICCES)*. [S.l.: s.n.], 2016. p. 1–6.

Rajandekar, A.; SIKDAR, B. A survey of mac layer issues and protocols for machine-to-machine communications. *IEEE Internet of Things Journal*, IEEE, v. 2, n. 2, p. 175–186, 2015.

Reddi, S. J.; Kale, S.; Kumar, S. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

Roberts, L. G. Aloha packet system with and without slots and capture. *ACM SIGCOMM Computer Communication Review*, ACM New York, NY, USA, v. 5, n. 2, p. 28–42, 1975.

ROTTA, G.; CHARÃO, A.; DANTAS, M. Um estudo sobre protocolos de comunicação para ambientes de internet das coisas. In: SBC. *Anais da XVII Escola Regional de Alto Desempenho do Estado do Rio Grande do Sul*. [S.l.], 2017.

Sabri, C.; Kriaa, L.; Azzouz, S. L. Comparison of iot constrained devices operating systems: A survey. In: *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*. [S.l.: s.n.], 2017. p. 369–375.

SALAZAR, J. J.; GARLAND, L.; OCHOA, J.; PYRCZ, M. J. Fair train-test split in machine learning: Mitigating spatial autocorrelation for improved prediction accuracy. *Journal of Petroleum Science and Engineering*, Elsevier, v. 209, p. 109885, 2022.

Salvo Rossi, P.; Ciuonzo, D.; Ekman, T. Hmm-based decision fusion in wireless sensor networks with noncoherent multiple access. *IEEE Communications Letters*, v. 19, n. 5, p. 871–874, 2015.

Samijayani, O. N.; Darwis, R.; Rahmatia, S.; Mujadin, A.; Astharini, D. Hybrid zigbee and wifi wireless sensor networks for hydroponic monitoring. In: *2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*. [S.l.: s.n.], 2020. p. 1–4.

Santos, D. R. G. dos; Volante, C. R. A importância da tecnologia sem fio na indústria 4.0. *Revista Interface Tecnológica*, v. 15, n. 2, p. 245–254, 2018.

SANTOS, E. L. F. dos. A ieee 802.15. 4 como plataforma de comunicação de dados. *Revista Ilha Digital*, v. 4, p. 97–105, 2013.

SANTOS, F. C. dos; MINI, R. A. de F. Iotsafe-uma arquitetura baseada em fog computing para prover segurança em iot. In: SBC. *Anais do XIX Workshop de Computação em Clouds e Aplicações*. [S.l.], 2021. p. 15–28.

Serra, T.; Yu, X.; Kumar, A.; Ramalingam, S. Scaling up exact neural network compression by relu stability. *Advances in Neural Information Processing Systems*, v. 34, p. 27081–27093, 2021.

SHA, M.; DOR, R.; HACKMANN, G.; LU, C.; KIM, T.-S.; PARK, T. Self-adapting mac layer for wireless sensor networks. In: *2013 IEEE 34th Real-Time Systems Symposium*. [S.l.: s.n.], 2013. p. 192–201.

SHANTHAMALLU, U. S.; SPANIAS, A.; TEPEDELENLIOGLU, C.; STANLEY, M. A brief survey of machine learning methods and their sensor and iot applications. In: IEEE. *2017 8th International Conference on Information, Intelligence, Systems & Applications (IISA)*. [S.l.], 2017. p. 1–8.

Shuangfeng, L. Tensorflow lite: On-device machine learning framework. *Journal of Computer Research and Development*, v. 57, n. 9, p. 1839, 2020.

Sinche, S.; Raposo, D.; Armando, N.; Rodrigues, A.; Boavida, F.; Pereira, V.; Silva, J. S. A survey of iot management protocols and frameworks. *IEEE Communications Surveys Tutorials*, v. 22, n. 2, p. 1168–1190, 2020.

SINGH, K. J.; KAPOOR, D. S. A survey of iot platforms: Create your own internet of things. *IEEE Consumer Electronics Magazine*, v. 6, n. 2, p. 57–68, 2017.

Singh, U.; Chana, I. Energy efficient solutions for iot based applications. In: *2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*. [S.l.: s.n.], 2019. v. 1, p. 1287–1292.

Sinha, S.; Singh, T.; Singh, V.; Verma, A. Epoch determination for neural network by self-organized map (som). *Computational Geosciences*, Springer, v. 14, n. 1, p. 199–206, 2010.

Sisinni, E.; Ferrari, P.; Fernandes Carvalho, D.; Rinaldi, S.; Marco, P.; Flammini, A.; Depari, A. Lorawan range extender for industrial iot. *IEEE Transactions on Industrial Informatics*, v. 16, n. 8, p. 5607–5616, 2020.

Skerrett, I. Iot developer survey 2017. *LinkedIn SlideShare*, 2017.

Sousa, G. L. d. Análise e desenvolvimento de subsistemas de comunicação e processamento de dados para integração e supervisão de processos de soldagem. Dissertação (Mestrado em Engenharia Mecânica) - Programa de Pós-Graduação em Engenharia Mecânica, Universidade Federal de Santa Catarina, Florianópolis, 2018.

SOUZA, F. A. de; CHANG, B. S.; BRANTE, G.; SOUZA, R. D.; PELLENZ, M. E.; ROSAS, F. Optimizing the number of hops and retransmissions for energy efficient multi-hop underwater acoustic communications. *IEEE Sensors Journal*, IEEE, v. 16, n. 10, p. 3927–3938, 2016.

Srinidhi, N. N.; Kumar, S. M. D.; Banu, R. Internet of things for neophytes: A survey. In: *2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*. [S.l.: s.n.], 2017. p. 234–242.

Sun, H.; Chen, X.; Shi, Q.; Hong, M.; Fu, X.; Sidiropoulos, N. D. Learning to optimize: Training deep neural networks for wireless resource management. In: *18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. Sapporo, Japan: [s.n.], Jul. 2017.

Svozil, D.; Kvasnicka, V.; Pospichal, J. Introduction to multi-layer feed-forward neural networks. *Chemometrics and intelligent laboratory systems*, Elsevier, v. 39, n. 1, p. 43–62, 1997.

Tachibana, K.; Otsuka, K. Wind prediction performance of complex neural network with relu activation function. In: *Proc. of SICE*. Nara, Japan: [s.n.], Sep. 2018.

TAMIOSSO, G.; BERNED, G.; LUIZELLI, M. C.; ROSSI, F.; NONAKA, J.; LORENZON, A. Avaliaçao do desempenho e consumo de energia de aplicaçoes implementadas com openmp simd. In: SBC. *Anais da XX Escola Regional de Alto Desempenho da Região Sul*. [S.l.], 2020. p. 69–72.

Tang, J.; Sun, D.; Liu, S.; Gaudiot, J.-L. Enabling deep learning on iot devices. *Computer*, IEEE, v. 50, n. 10, p. 92–96, 2017.

Taulli, T. *Introdução à Inteligência Artificial: uma abordagem não técnica, Editora Novatec*. [S.l.]: novatec, 2020.

Tieleman, T.; Hinton, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, v. 4, n. 2, p. 26–31, 2012.

Tinnirello, I.; Bianchi, G.; Xiao, Y. Refinements on ieee 802.11 distributed coordination function modeling approaches. *IEEE Trans. Veh. Technol.*, IEEE, v. 59, n. 3, p. 1055–1067, 2010.

Tracey, D.; Sreenan, C. How to see through the fog? using peer to peer (p2p) for the internet of things. In: *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*. [S.l.: s.n.], 2019. p. 47–52.

Tukur, Y. M.; Ali, Y. S. Demonstrating the effect of insider attacks on perception layer of internet of things (iot) systems. In: *2019 15th International Conference on Electronics, Computer and Computation (ICECCO)*. [S.l.: s.n.], 2019. p. 1–6.

Turian, J.; Bergstra, J.; Bengio, Y. Quadratic features and deep architectures for chunking. In: *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*. [S.l.: s.n.], 2009. p. 245–248.

VELINOV, A.; MILEVA, A. Power consumption analysis of application layer protocols for the internet of things. In: SPRINGER. *International Conference on ICT Innovations*. [S.l.], 2016. p. 193–202.

VELINOV, A.; MILEVA, A. Running and testing applications for contiki os using cooja simulator. *International Conference on Information Technology and Development of Education (ITRO 2016)*, 2016.

Wang, K.; Wang, Y.; Sun, Y.; Guo, S.; Wu, J. Green industrial internet of things architecture: An energy-efficient perspective. *IEEE Communications Magazine*, IEEE, v. 54, n. 12, p. 48–54, 2016.

Wilson, D. R.; Martinez, T. R. The general inefficiency of batch training for gradient descent learning. *Neural networks*, Elsevier, v. 16, n. 10, p. 1429–1451, 2003.

Xie, J.; Yu, F. R.; Huang, T.; Xie, R.; Liu, J.; Wang, C.; Liu, Y. A survey of machine learning techniques applied to software defined networking (sdn): Research issues and challenges. *IEEE Communications Surveys Tutorials*, v. 21, n. 1, p. 393–430, 2019.

Yangjian. Development of ralink's wireless network interface card based on uclinux. In: *2017 Chinese Automation Congress (CAC)*. [S.l.: s.n.], 2017. p. 3278–3281.

Yassein, M. B.; Shatnawi, M. Q.; Al-zoubi, D. Application layer protocols for the internet of things: A survey. In: *2016 International Conference on Engineering MIS (ICEMIS)*. [S.l.: s.n.], 2016. p. 1–4.

Yazan, E.; Talu, M. F. Comparison of the stochastic gradient descent based optimization techniques. In: *2017 International Artificial Intelligence and Data Processing Symposium (IDAP)*. [S.l.: s.n.], 2017. p. 1–5.

Yu, C.; Hu, J.; Li, R.; Deng, S.; Yang, R. Node fault diagnosis in wsn based on rs and svm. In: *2014 International Conference on Wireless Communication and Sensor Network*. [S.l.: s.n.], 2014. p. 153–156.

Zeiler, M. D. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Zhang, G.; Li, L.; Nado, Z.; Martens, J.; Sachdeva, S.; Dahl, G.; Shallue, C.; Grosse, R. B. Which algorithmic choices matter at which batch sizes? insights from a noisy quadratic model. In: Wallach, H.; Larochelle, H.; Beygelzimer, A.; ALCHé-BUC, F. d'; Fox, E.; Garnett, R. (Ed.). *Advances in Neural Information Processing Systems 32*. [S.l.]: Curran Associates, Inc., 2019. p. 8196–8207.

Zhang, R.; Gorce, J.-M. Optimal transmission range for minimum energy consumption in wireless sensor networks. In: *Proc. of IEEE WCNC*. Las Vegas, USA: [s.n.], Mar. 2008.

Zheng, J.; Ma, Q.; Zhou, W. Performance comparison of full-batch bp and mini-batch bp algorithm on spark framework. In: *Proc. of WCSP*. Yangzhou, China: [s.n.], Oct. 2016.

Zhu, J.; Gong, C.; Zhang, S.; Zhao, M.; Zhou, W. Foundation study on wireless big data: Concept, mining, learning and practices. *China Communications*, v. 15, n. 12, p. 1–15, 2018.

Zhu, P.; Zhang, S.; Luo, H.; Wu, Z. A semi-supervised method for classifying unknown protocols. In: *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*. [S.l.: s.n.], 2019. p. 1246–1250.

Zou, F.; Shen, L.; Jie, Z.; Zhang, W.; Liu, W. A sufficient condition for convergences of adam and rmsprop. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2019. p. 11119–11127.