

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E  
TECNOLOGIA FLUMINENSE**

**PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS APLICADOS À  
ENGENHARIA E GESTÃO**

**Carlos Márcio da Silva Freitas**

**REDES NEURAIIS ARTIFICIAIS APLICADAS AO GERENCIAMENTO DE  
ENERGIA EM MICROPROCESSADORES DE BAIXA POTÊNCIA.**

**Campos dos Goytacazes / Rio de Janeiro**

2021

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E  
TECNOLOGIA FLUMINENSE**

PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS APLICADOS À  
ENGENHARIA E GESTÃO

Carlos Márcio da Silva Freitas

REDES NEURAIIS ARTIFICIAIS APLICADAS AO GERENCIAMENTO DE ENERGIA EM  
MICROPROCESSADORES DE BAIXA POTÊNCIA.

Rogério Atem de Carvalho

Orientador

Rodrigo Martins Fernandes

Coorientador

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação do Instituto Federal de Educação, Ciência e Tecnologia Fluminense, no Curso de Mestrado Profissional em Sistemas Aplicados à Engenharia e Gestão (MPSAEG), como parte dos requisitos necessários à obtenção do título de Mestre em Sistemas Aplicados à Engenharia e Gestão.

Campos dos Goytacazes / Rio de Janeiro  
2021

Biblioteca Anton Dakitsch  
CIP - Catalogação na Publicação

F866r Freitas, Carlos Márcio da Silva  
REDES NEURAIAS ARTIFICIAIS APLICADAS AO  
GERENCIAMENTO DE ENERGIA EM MICROPROCESSADORES DE  
BAIXA POTÊNCIA. / Carlos Márcio da Silva Freitas - 2021.  
74 f.: il. color.

Orientador: Rogério Atem de Carvalho  
Coorientador: Rodrigo Martins Fernandes

Dissertação (mestrado) -- Instituto Federal de Educação, Ciência e  
Tecnologia Fluminense, Campus Campos Centro, Curso de Mestrado  
Profissional em Sistemas Aplicados à Engenharia e Gestão, Campos dos  
Goytacazes, RJ, 2021.  
Referências: f. 71 a 74.

1. Artificial Intelligence. 2. Low Power. 3. Power Management. 4.  
LSTM. I. Carvalho, Rogério Atem de, orient. II. Fernandes, Rodrigo  
Martins, coorient. III. Título.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA  
FLUMINENSE

PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS APLICADOS À  
ENGENHARIA E GESTÃO


Carlos Márcio da Silva Freitas

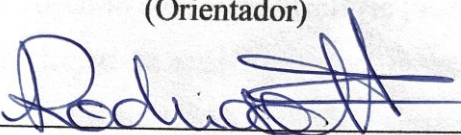
REDES NEURAS ARTIFICIAIS APLICADAS AO GERENCIAMENTO DE ENERGIA EM  
MICROPROCESSADORES DE BAIXA POTÊNCIA.

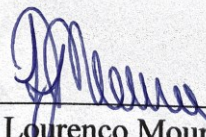
Dissertação de Mestrado apresentada ao Programa de Pós-Graduação do Instituto Federal de Educação, Ciência e Tecnologia Fluminense, no Curso de Mestrado Profissional em Sistemas Aplicados à Engenharia e Gestão (MPSAEG), como parte dos requisitos necessários à obtenção do título de Mestre em Sistemas Aplicados à Engenharia e Gestão.

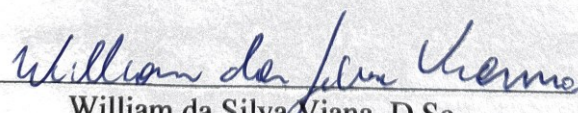
Aprovado(a) em 08 de outubro de 2021.

Banca Examinadora:

  
\_\_\_\_\_  
Rogério Atém de Carvalho, D.Sc.  
Instituto Federal Fluminense  
(Orientador)

  
\_\_\_\_\_  
Rodrigo Martins Fernandes, D.Sc.  
Instituto Federal Fluminense  
(Coorientador)

  
\_\_\_\_\_  
Luiz Gustavo Lourenço Moura, D.Sc.  
Instituto Federal Fluminense

  
\_\_\_\_\_  
William da Silva Viana, D.Sc.  
Instituto Federal Fluminense

## AGRADECIMENTOS

Primeiramente agradeço a *Deus*, por me conceder saúde e paz a cada jornada enfrentada, sem Sua presença, cuidado e conforto jamais seria completo na forma de viver. Não tenho como agradecer por tantas orações respondidas, pelas portas abertas e também fechadas para minha segurança, pelos amigos e irmãos que me enviou quando precisei, pelas chuvas temporã e serôdia que trouxeram tantas bênçãos sobre mim e geraram tantos frutos de alegria. “*Quando o SENHOR restaurou a sorte de Sião, ficamos como quem sonha. Então, a nossa boca se encheu de riso, e a nossa língua, de júbilo; então, entre as nações se dizia: Grandes coisas o SENHOR tem feito por eles*”. (Salmos 126:1-2).

À minha amiga e esposa *Josenir Nunes Freitas*, pelo incentivo incondicional a todos meus projetos, carreiras e planos de vida, por cuidar de mim com tanto carinho e dedicação, por ser uma serva de Deus exemplar na qual posso me espelhar a cada dia e por se a maior bênção que recebi das mãos de Deus.

Ao meu amigo e Pastor *Jesué Berilo* por sempre ver o melhor em mim, por me abraçar como um filho e me amar como um irmão, por cobrir a mim e minha esposa com suas orações e seu carinho, por sempre contribuir com o melhor para que eu possa ser quem sou hoje.

Ao meu orientador *Rogério Atem de Carvalho*, pela honra de ter sido aceito como orientado e por ter tido a oportunidade de conhece-lo como um excelente professor e pesquisador pelo qual tentarei me espelhar em toda minha trajetória acadêmica e profissional, por me incentivar a continuar minha trajetória acadêmica me mostrando a direção certa e os caminhos que preciso seguir. Por poder compartilhar meus objetivos e receber total apoio e reconhecimento.

Ao meu coorientador *Rodrigo Martins Fernandes* por compartilhar comigo a paixão pela engenharia elétrica, pelos ensinamentos, pelas dicas e pelo tempo a mim dedicado.

Aos integrantes da banca de Qualificação e de Defesa, pela atenção e tempo a mim concedidos, pelo valioso aprendizado, por contribuir na ampliação da minha capacidade técnica e científica e pelo incentivo de prosseguir como pesquisador.

A todos os professores que tive a honra de conhecer no SAEG e que contribuíram para meu crescimento científico, profissional e até mesmo pessoal.

Aos meu pais *José Carlos e Márcia* por me ensinarem o valor do trabalho e me mostrarem desde a infância o caminho de Deus.

## RESUMO

Sistemas computacionais que operam em locais remotos como satélites, estações meteorológicas remotas e robôs autônomos são altamente limitados quanto a disponibilidade de energia para sua operação. Esse trabalho tem o objetivo de empregar algoritmos de inteligência artificial no gerenciamento de energia de forma a se obter o máximo rendimento energético realizando a previsão da disponibilidade de energia. O trabalho apresenta os principais tipos de algoritmos utilizados em inteligência artificial e apresenta a criação de um protótipo que irá operar como um sistema low power alimentado por baterias e uma pequena placa solar, o protótipo realiza inferências no algoritmo de redes neurais LSTM de modo a prever a futura disponibilidade de energia, consequentemente o sistema de gerenciamento realiza a distribuição de energia a fim de se obter a máxima operação do protótipo sem que haja a descarga total das baterias. Para que o sistema de inteligência artificial pudesse ser embarcado no protótipo foi utilizado o framework TensorFlow Lite que permite realizar a inferência em dispositivos de baixo consumo e limitado poder de processamento.

**Palavras-chave:** Inteligência Artificial, Low Power , Gerenciamento de energia, LSTM.

## ABSTRACT

Computer systems that operate in remote locations such as satellites, remote weather stations and autonomous robots are highly limited in the availability of energy for their operation. This work aims to employ artificial intelligence algorithms in energy management in order to obtain the maximum energy yield and the prediction of energy availability to the system. The work presents the main types of algorithms used in artificial intelligence and presents the creation of a prototype that will operate as a low power system powered by batteries and a small solar plate, the prototype performs inferences in the LSTM neural network algorithm in order to predict the future availability of energy, consequently the management system performs the energy distribution in order to obtain the maximum operation of the prototype without total discharge of the batteries. So that the artificial intelligence system could be embedded in the prototype, the TensorFlow Lite framework was used, which allows the inference to be carried out in devices with low consumption and limited processing power.

**Keywords:** Artificial Intelligence, Low Power, Power Management, LSTM.

## LISTAS

### Lista de tabelas

Tabela 1: Objetivos específicos do projeto. Fonte: Elaborado pelo autor (2021).	16
Tabela 2: Etapas do projeto. Fonte: Elaborado pelo autor (2021). Fonte: Elaborado pelo autor (2021).	33
Tabela 3: Características do módulo INA-219. Fonte: Elaborado pelo autor (2021).	37
Tabela 4: Características do ESP32WROOM. Fonte: Elaborado pelo autor (2021).	38
Tabela 5: Características do circuito BMS. Fonte: Elaborado pelo autor (2021).	39
Tabela 6: Características da placa fotovoltaica. Fonte: Elaborado pelo autor (2021).	40
Tabela 7: Características da fresadora de placas de circuito impresso. Fonte: Elaborado pelo autor (2021).	45
Tabela 8: Pinos de entrada e saída utilizados no ESP-32. Fonte: Elaborado pelo autor (2021).	54
Tabela 9: Trecho dos dados armazenados no cartão de memória.	59
Tabela 10: Testes efetuados no modelo com 2 camadas LSTM. Fonte: Elaborado pelo autor (2021).	64
Tabela 11: Testes efetuados no modelo 3 camadas LSTM.	64
Tabela 12: Resultados do teste de campo. Fonte: Elaborado pelo autor (2021).	68

### Lista de figuras

Figura 1: Classificações dos algoritmos de inteligência artificial abordados. Fonte: Elaborado pelo autor (2021).	23
Figura 2: Divisões da área de machine learning. Fonte: Elaborado pelo autor (2021).	24
Figura 3: Logica Fuzzy aplicada a análise de capacidade de carga de baterias. Fonte: Elaborado pelo autor (2021).	26
Figura 4: Atividades para execução do projeto. Fonte: Elaborado pelo autor (2021).	34
Figura 5: Diagrama de blocos dos componentes do projeto. Fonte: Elaborado pelo autor (2021).	36
Figura 6: Fotos dos componentes. Fonte: Elaborado pelo autor (2021).	37
Figura 7: Diagrama de blocos do ESP32. Fonte: Espressif (2021)	39
Figura 8: Placa fotovoltaica instalada. Fonte: Elaborado pelo autor (2021).	42
<i>Figura 9: Vista frontal da placa de circuito em 3D. Figura 10: Vista inferior da placa de circuito.</i>	43
Figura 11: Placa de circuito impresso em desenvolvimento. Fonte: Elaborado pelo autor (2021).	44
Figura 12: Fresadora CNC de placas de circuito impresso. Fonte: <a href="http://www.ttp.ind.br">www.ttp.ind.br</a> , acessado em 10/08/2021.	45
Figura 13: Placa de circuito impresso finalizada. Fonte: Elaborado pelo autor (2021).	46
Figura 14: Módulo Raspberry Pi Zero W. Fonte: Elaborado pelo autor (2021).	47
Figura 15: ambiente de desenvolvimento utilizado para a construção do firmware. Fonte: Elaborado pelo autor (2021).	47
Figura 16: Visão geral dos componentes básicos do projeto. Fonte: Elaborado pelo autor (2021).	48
Figura 17: Diagrama do circuito elétrico do projeto. Fonte: Elaborado pelo autor (2021).	49
Figura 18: Limite de operação fixo de um sistema de gerenciamento de energia. Fonte: Elaborado pelo autor (2021).	50
Figura 19: Fluxograma básico do software de operação das previsões. Fonte: Elaborado pelo autor (2021).	57
Figura 20: Gráfico com a variação da tensão da placa fotovoltaica ( $V_A$ ) juntamente com a temperatura e umidade. Fonte: Elaborado pelo autor (2021).	60



Figura 21: Gráfico com a variação da tensão e corrente da bateria juntamente com a temperatura e humidade. Fonte: Elaborado pelo autor (2021).	61
Figura 22: Gráfico com a variação da tensão e corrente da placa fotovoltaica juntamente com a temperatura e humidade. Fonte: Elaborado pelo autor (2021).	61
Figura 23: Foto da montagem do conjunto e sensor DHT11 localizado externamente a caixa de proteção. Fonte: Elaborado pelo autor (2021).	62
Figura 24: Resumo do modelo LSTM. Fonte: Elaborado pelo autor (2021).	63
Figura 25: Gráfico de variação da perda loss. Fonte: Elaborado pelo autor (2021).	65
Figura 26: Gráfico de comparação entre dados previstos e dados de teste. Fonte: Elaborado pelo autor (2021).	65
Figura 27: Diagrama de blocos dos atributos previsores. Fonte: Elaborado pelo autor (2021).	67
Figura 28: Etapas para criação do modelo TensorFlow Lite. Fonte: Elaborado pelo autor (2021).	68
Figura 29: Medição da corrente da lâmpada utilizada como carga. Fonte: Elaborado pelo autor (2021).	69

## Lista de Mapas

Mapa 1: Modos de operação para economia de energia de um sistema embarcado. Fonte: LIN et al.,(2020).	22
Mapa 2: Mapa de localização do ponto de testes do sistema. Fonte: Google Maps (acessado em 02/082021).	41

## Lista de Quadros

Quadro 1: Primeira parte do código com a declaração das bibliotecas e checagem das versões.	51
Quadro 2: Criação do modelo LSTM e declaração das funções de callback que permitem otimizar o processo de treinamento.	52
Quadro 3: Preparação dos dados para treino da rede neural.	52
Quadro 4: Análise da perda e do erro absoluto médio e teste do modelo treinado através da base de dados de teste.	53
Quadro 5: Conversão do modelo keras para TensorFlow Lite.	54

## Lista de siglas

IoT - Internet das coisas

MEC - Ministério da Educação

SoC - State of charge

PCB - Placa de circuito impresso

eIoT - Energia da Internet das Coisas

MCU - Unidades de Microcontroladores

## SUMÁRIO

1.0 INTRODUÇÃO .....	13
1.1 OBJETIVOS.....	16
1.1.1 Objetivo Geral.....	16
1.1.2 Objetivos específicos.....	16
1.2 JUSTIFICATIVA .....	17
2.0 FUNDAMENTAÇÃO TEÓRICA.....	19
2.1 Sistemas Low Power.....	19
2.2.1. Aprendizagem Não Supervisionada.....	25
2.2.1.1 Redução de Dimensionalidade.....	25
2.2.1.2 Clustering.....	25
2.2.2 Aprendizagem Supervisionada.....	25
2.2.2.1 Classificação.....	25
2.2.2.2 Regressão.....	25
2.3 Aprendizagem por reforço.....	26
2.4 Lógica Fuzzy.....	26
2.5 Redes Neurais Artificiais.....	27
2.5.1 Camadas de uma rede Neural.....	27
2.5.2 O Perceptron.....	27
2.5.3 Rede Neural FeedForward.....	28
2.5.4 Rede Neural Recorrente.....	28
2.5.5 Redes neurais LSTM (Long Short Term Memory ).....	28
2.5.6 Autoencoders.....	29
2.5.7 Redes neurais convolucionais.....	29
3.0 Vantagens das Redes neurais LSTM em comparação a outros métodos de previsão. ....	30
3.0 METODOLOGIA .....	33
3.1 Classificação da Pesquisa .....	33
3.2 Etapas da Pesquisa.....	33
3.3 Fluxo de trabalho para criação de um modelo baseado em redes neurais. ....	34
4.0 MATERIAIS .....	36
5.0 SISTEMA PROPOSTO.....	48
6.0 MÉTODOS.....	51
6.1 Código utilizado para criar o modelo LSTM.....	51

6.3 Software de gerenciamento de energia .....	56
7.0 RESULTADOS.....	59
8.0 CONCLUSÃO .....	70
9.0 REFERÊNCIAS.....	71

## 1.0 INTRODUÇÃO

Os sistemas elétricos modernos, como veículos elétricos, robôs móveis, nano satélites e drones, exigem várias operações de demanda de energia para aplicações do usuário e manutenção do sistema. Isso, por sua vez, exige um gerenciamento avançado de energia, que considera em conjunto a demanda de energia das operações e o fornecimento de energia de várias fontes, como baterias, painéis solares e supercapacitores. No entanto, uma solução completa para o problema de programação de energia também precisa caracterizar armazenamentos e fontes de energia porque afetam a capacidade de energia para as demandas de operações e funcionamento do sistema.(KIM et al., 2020)

O correto gerenciamento de energia vem se tornando um dos principais fatores em projetos de dispositivos alimentados por baterias ou com recursos limitados de energia que utilizam energias renováveis como fonte de energia, como energia solar e eólica.

Dispositivos com alimentação intermitente, recentemente ganharam muito interesse em uma ampla variedade de campos de aplicação, de redes de sensores sem fio à Internet das Coisas (IoT), devido ao seu tamanho pequeno, baixo custo e requisitos de baixa manutenção. Esses dispositivos, são alimentados por recursos de energia intermitente, como luz do sol, calor, vibração ou sinais de radiofrequência, e possuem diversas aplicações, incluindo casa inteligente, agricultura inteligente e monitoramento de saúde, para citar apenas alguns.(KARIMI et al., 2021)

Recentemente, com o rápido desenvolvimento da microeletrônica, micromáquinas, materiais leves e outras tecnologias básicas, o uso de tecnologias de micro e nano satélites para exploração espacial, comunicação e navegação se tornaram um ponto de pesquisa. O uso de micro ou nano satélites possuem as características de baixo custo, alta confiabilidade, alto desempenho e alta flexibilidade. Vários satélites podem ser virtualizados como um grande satélite por meio da cooperação entre satélites para completar missões espaciais. No entanto, o ambiente espacial da formação de micro e nano satélites é complexo e é fácil de ser perturbado pelo ambiente externo ou por seus próprios problemas. O longo tempo de recuperação de falhas aumenta a perda e o congestionamento de pacotes de rede e afeta a capacidade de execução da missão espacial na formação de micro e nano satélites. Além disso, a energia dos satélites é geralmente limitada devido ao uso de bateria. Portanto, a recuperação rápida com consciência de conservação de energia por falha de rede de micro e nano satélite é um problema prático a ser resolvido.(ZHANG; LIU; DING, 2019)

Muitas aplicações tipicamente remotas e de difícil acesso como os nano satélites utilizam meios renováveis como forma de obtenção de energia elétrica.

Comparada com a energia baseada em combustíveis fósseis, a energia renovável tem sido cada vez mais atraente porque não se esgota e não é prejudicial ao meio ambiente. Operando como amortecedor de energia em sistemas de energia renovável, o armazenamento de energia pode aumentar a eficiência da energia elétrica dos clientes e o deslocamento de carga.(LI; XIAO; FAN, 2019)

A tecnologia de energia renovável está associada a esquemas de estimativa de energia para uma gestão astuta de energia. Assim, há a necessidade de empreender mecanismos ineptos de economia de energia, além de tecnologia de energia renovável a fim de atingir um status de alta confiabilidade. Os sensores podem incorporar tendências de comportamento dinâmico em face da energia estimada não ser capaz de sustentá-los no próximo ciclo de recarga. Portanto, eles podem otimizar parâmetros decisivos, como taxa de amostragem, potência de transmissão e ciclo de trabalho para adaptar seu consumo de energia de acordo com a periodicidade e magnitude da fonte de energia. (ALSHARIF; KIM; KURUOĞLU, 2019)

Com o avanço das tecnologias de internet das coisas (IoT), a demanda por sistemas com alimentação autônoma se torna cada vez maior. A autonomia das baterias influencia diretamente o tempo de operação de redes de sensores, estações meteorológicas, e robôs autônomos por exemplo.

A Internet das Coisas(IoT) é um paradigma que visa o avanço das telecomunicações em todas as esferas da vida humana. Em um futuro próximo, as tecnologias de simetria para a Internet das Coisas, junto com os aplicativos de simetria e assimetria para a segurança e privacidade da IoT irão redesenhar a morfologia do terreno humano socioecológico. Esse avanço leva a uma melhoria substancial na qualidade de vida humana e ao crescimento econômico mundial em geral. A IoT é considerada a espinha dorsal das aplicações emergentes, pois a inovação desempenha um papel fundamental na evolução massiva das comunicações entre máquinas. Estima-se que o tráfego máquina-a-máquina (M2M) corresponda a aproximadamente 45% do tráfego total da Internet em 2022. A IoT cria uma plataforma na qual objetos físicos podem imitar certas capacidades sensoriais humanas, como percepção, visão, audição, olfato e pensamento. (ALSHARIF; KIM; KURUOĞLU, 2019)

Os sensores sem fio são um componente integral dos aplicativos inteligentes baseados na tecnologia IoT. São dispositivos miniaturizados e baratos, equipados com a capacidade de detectar parâmetros de interesse e transmitir periodicamente os resultados ao ponto de coleta, e são alimentados por baterias. Os principais alvos da fonte de energia dos sensores sem fio devem ser sustentabilidade e confiabilidade, bem como a redução das emissões de gases de efeito estufa, esses fatores podem ser atendidos por meio de avanços na tecnologia de energia renovável. Além disso, a tecnologia de energia renovável é uma das maneiras promissoras de abordar a questão de eficiência energética das redes de sensores sem fio localizadas em áreas rurais e remotas. Em muitos perfis de terreno, o acesso as baterias é dificultado devido às limitações geográficas como altitude e distância. As células solares têm baixa necessidade de manutenção e alta confiabilidade, com vida útil esperada de 20-30 anos. (ALSHARIF; KIM; KURUOĞLU, 2019)

A correta análise das condições da bateria durante a operação do sistema remoto é fundamental para se estimar a duração da carga de baterias e até mesmo suas condições de operação e perda de capacidade.

O estado de carga ou SoC é uma quantidade que fornece as informações sobre a quantidade restante de carga em uma bateria como uma proporção de sua capacidade nominal, e portanto, fornece indiretamente informações sobre a quantidade restante de energia na bateria. Infelizmente, ele não pode ser medido diretamente e deve ser estimado indiretamente a partir dos parâmetros e variáveis observáveis da bateria. A estimativa de SoC é de extrema importância.(BHATTACHARJEE et al., 2021)

Devido aos fatores meteorológicos, como condições climáticas, irradiância, temperatura ambiente e velocidade do vento, bem como aos fatores não meteorológicos, como temperatura e local de instalação de componentes e peças, a potência de saída do sistema de geração de energia fotovoltaica sofre forte intermitência, volatilidade e incerteza, além de baixa precisão de previsão da geração de energia fotovoltaica. Porém com base nos dados históricos de geração de energia e nos dados meteorológicos reais do sistema fotovoltaico, a previsão de curto prazo da geração de energia fotovoltaica pode ser realizada usando o modelo de previsão de rede neural.( KAIJU et al., 2018)

## 1.1 OBJETIVOS

### 1.1.1 Objetivo Geral

Prototipar um sistema embarcado em placa de circuito impresso (PCB) capaz de realizar o gerenciamento de energia utilizando inteligência artificial do tipo redes neurais LSTM com capacidade de prever a disponibilidade de energia e realizar operações de medição, comunicação e acionamento de cargas mediante essa previsão. O sistema tem a finalidade de comprovar a eficiência energética obtida por um sistema de previsão baseado em inteligência artificial que gerencia o consumo de energia com base em um painel solar fotovoltaico, dessa forma comprovando meios de se obter o máximo aproveitamento da energia, sem que ocorra falhas de fornecimento para a correta operação do sistema.

### 1.1.2 Objetivos específicos

Visando alcançar o objetivo geral os seguintes objetivos específicos foram estabelecidos:

Nº do objetivo	Objetivos específicos
1	Fabricar uma placa de circuito impresso para um sistema embarcado de baixo consumo de energia capaz de processar algoritmos de inteligência artificial.
2	Desenvolver o firmware para o processador do sistema afim de coletar os dados dos sensores e enviar as informações através de rede sem fio.
3	Treinar e otimizar redes neurais recorrentes do tipo LSTM para utilização em sistemas embarcados utilizando a biblioteca TensorFlow Lite.
4	Desenvolver e testar o scheduler de gerenciamento de energia baseado em previsões de fornecimento de potência elétrica.

*Tabela 1: Objetivos específicos do projeto. Fonte: Elaborado pelo autor (2021).*



## 1.2 JUSTIFICATIVA

O aumento do número de dispositivos de internet das coisas com alimentação elétrica própria desenvolve um cenário onde as baterias e os sistemas de energia renováveis se tornam fundamentais para sua operação.

As previsões futuras de sensores sem fio para permitir a Internet das coisas (IoT) mostram um aumento de duas vezes entre 2018 e 2023, o que resultará em uma demanda significativamente maior por fontes de energia para cerca de 50 bilhões de sensores sem fio. As fontes de energia como placas solares e turbinas eólicas podem ser utilizadas como fontes de energia autônomas para nós de sensores sem fio para uma ampla gama de aplicações em diferentes ambientes, com os benefícios da captação de energia sendo bem reconhecidos para configurações comerciais e residenciais (NEWELL; DUFFY, 2019)

As múltiplas operações de demanda de energia impõem diferentes demandas de energia no sistema e podem ser acionadas em momentos diferentes, exigindo que o sistema disponibilize efetivamente um fornecimento de energia variável no tempo. (KIM et al., 2020)

Na aplicação de nano satélites as exigências e desafios de gerenciamento de energia são ainda maiores. Além de um esquema de distribuição de energia altamente eficiente, os esquemas de gerenciamento de energia a bordo desempenham um papel significativo para garantir que o satélite não falhe devido a uma falta de energia. A principal fonte de energia para a maioria dos pequenos satélites é o sol. Células solares não implantáveis são usadas para aproveitar essa energia. As limitações de tamanho, área de superfície e massa de um nano satélite limitam sua capacidade de geração de energia (THAKURTA et al., 2019)

Embora haja diversas pesquisas no ramo de gerenciamento de energia que utilizam algoritmos baseados em inteligência artificial, é raro encontrar trabalhos que demonstrem o gerenciamento de energia com inteligência artificial embarcados em um único dispositivo de baixo consumo. Um trabalho publicado em 2019 por LI; XIAO; FAN demonstrando a eficiência da previsão de carga de baterias utilizando inteligência artificial é um dos poucos trabalhos encontrados nessa linha de pesquisa, porém o mesmo não trata do processamento da inteligência artificial embarcada em dispositivos de baixo consumo.

Outra pesquisa considerável na área foi publicada por PARK em 2020 apresentando a previsão de capacidade útil de baterias utilizando redes neurais, porém o trabalho se refere a sistemas embarcados que não realizam o processamento de inteligência artificial de forma embarcada.

Dessa forma o gerenciamento de energia com inteligência artificial embarcada em sistemas de baixo consumo representa um vasto campo de pesquisa pelo qual esse trabalho tem o objetivo de explorar.

## 2.0 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são avaliados os princípios de sistemas low power e as características dos principais algoritmos de inteligência artificial que podem processar os dados afim de prever dados de gerenciamento de energia.

### 2.1 Sistemas Low Power.

Com os crescentes avanços da tecnologia em diversas áreas como medicina, telecomunicações, segurança patrimonial e até mesmo dispositivos embarcados em roupas e acessórios chamados de wearables, a demanda por sistemas alimentados por baterias também tende a ser maior a cada ano. Para ampliar a vida útil das cargas das baterias é necessário que o sistema embarcado possua dispositivos de baixo consumo de energia e que adote medidas para suspender o consumo de energia quando o sistema estiver inoperante ou aguardando por sinais externos.

A recarga de energia de dispositivos alimentados por bateria, como, dispositivos IoT e sensores, ganharam mais atenção nos últimos anos, pois é uma solução robusta e confiável para paradigma de aprovisionamento de energia. Conhecimento da carga do sistema, sua operação e requisitos de energia durante o tempo, são vitais para serem tratados por sistemas inteligentes responsáveis pelo gerenciamento de energia de sensores de IoT e módulos de comunicação.(FILIOS et al., 2020)

Uma das grandes aplicações de sistemas low power é no monitoramento contínuo de saúde e estilo de vida por meio de dispositivos médicos vestíveis que restringe o orçamento de energia devido ao tamanho restrito da bateria. Isso exige a necessidade de aquisição e processamento de biosinais de ultrabaixa potência. Além disso, o monitoramento de longo prazo leva a grandes quantidades de dados que precisam ser processados e transmitidos, o que resulta em um consumo significativo de energia.(PAMULA; VAN HOOFF; VERHELST, 2019)

Dispositivos médicos de ultrabaixa energia são essenciais na era da IoT. Sensores de cuidados de saúde capturam dados fisiológicos vitais para monitorar e diagnosticar pacientes. Os monitores de saúde são exemplos de dispositivos que fazem registro e monitoramento de dados vitais de forma contínua por 24 h. Eles são limitados pelo consumo de energia, uma vez que precisam operar por um período prolongado continuamente. Por outro lado, a plataforma de saúde IoT permite processamento

local mínimo e transfere dados para servidores conectados à nuvem que ajudam a resolver as desvantagens dos monitores e dispositivos semelhantes. (TEKESTE HABTE et al., 2019)

Para sensores de monitoramento de saúde vestíveis com telemetria sem fio, fonte de energia e consumo de energia tornaram-se critérios de design significativos. Para gerenciar a restrição de energia os projetistas de circuitos implementaram técnicas para reduzir o consumo geral de energia do sistema, ou para encontrar qualquer substituição da bateria convencional ou fonte de alimentação alternativa. Em geral, para um sistema de circuito integrado, o consumo total de energia compreende de dois tipos - estático e dinâmico. A energia estática é normalmente associada à corrente de fuga e corrente contínua das fontes, enquanto o consumo de energia dinâmica é dependente da frequência de consumo da potência total. Na abordagem de design de baixo consumo de energia, o consumo de corrente é reduzido minimizando a tensão de alimentação, complexidade do circuito, frequências de clock, valores de fonte de corrente contínua e até mesmo a capacitância dos nós de comutação. (SIU; INIEWSKI, 2018)

O rápido crescimento da energia da Internet das Coisas (eIoT) também impulsionou o desenvolvimento vigoroso de unidades de microcontroladores (MCUs). Para fornecer uma solução de custo mais baixo, mais e mais funções são integradas nos microcontroladores, como: interfaces de rede, interfaces de sensor de alta eficiência e outras, isso torna o design de baixo consumo de energia muito desafiante. Além disso, os cenários de aplicação de eIoT são mais complexos devido ao poder de múltiplos domínios, tornando o design de baixo consumo uma tarefa complexa. O eIoT é um sistema inteligente integrado distribuído, que utiliza fontes de energia renováveis escalonáveis e tecnologia da internet. Controlando o fluxo de energia através do fluxo de informações, onde a plataforma conecta usuários finais e estação de fornecimento de energia, percebendo o compartilhamento de informações e portanto, formando um ecossistema de energia. A tendência de desenvolvimento de eIoT é inseparável do mercado de MCU. Seja um pequeno nó para conexão ou um hub sensor para coleta e registro de dados, todos são baseados principalmente na plataforma de microcontroladores. Como uma das partes principais, o gerenciamento de energia em um microcontrolador para o eIoT é bastante desafiador. Em primeiro lugar, o microcontrolador para o eIoT geralmente tem vários domínios de fornecimento. Por exemplo, além da fonte de energia normal, microcontroladores para medidores inteligentes também possuem uma fonte de energia com bateria. Além disso, o bloco do relógio em tempo real (RTC) é necessário no microcontrolador para

medição de energia e precisa estar funcionando sem interrupções quando o medidor está armazenado ou em falta de energia. Devido a múltiplos domínios de alimentação, o microcontrolador para eIoT deve ser compatível com uma grande faixa de tensão de alimentação de 2,2 V a 5,5 V. Alternar entre diferentes domínios de fornecimento com alta confiabilidade também é um desafio para o gerenciamento de energia no microcontrolador. (LIN et al., 2020)

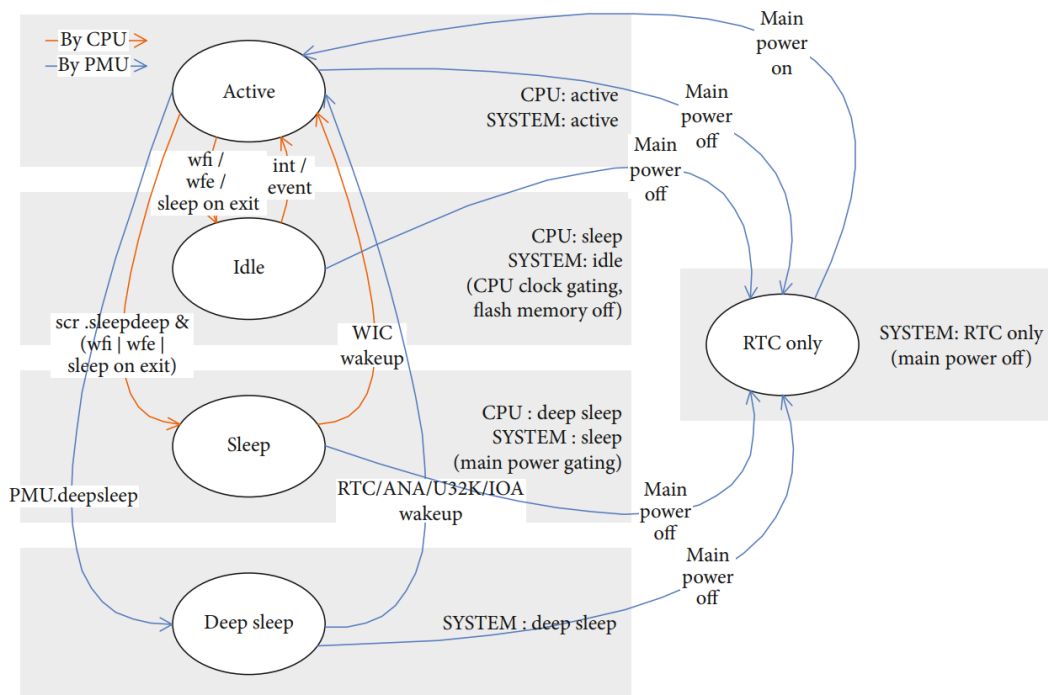
Em sistemas low power, dois esquemas de economia de energia são propostos. Um é para impor a programação do modo sleep em um nó sensor, alterando entre modo ativo e modo de hibernação. O segundo é a duração do modo sleep que pode ser escolhido com base nas condições ambientais. Considerando um espaço inteligente onde vários nós sensores colaboram em trabalhos de detecção, se um dos nós sensores detecta possíveis eventos anômalos, todos os sensores circundantes dos nós encurtarão seu período em modo sleep para a consciência do evento atual. A compressão de dados também contribui para o esquema de economia de energia encurtando os tamanhos dos dados do sensor. Este processo de compressão deve ser simples e eficiente para economizar tempo e poder de computação. (IIKUBO; LO, 2019)

Fisicamente dispositivos embarcados podem tirar proveito de diferentes modos de operação, como modos ativo e inativo, modo de hibernação e até mesmo desligando alguns componentes de hardware quando são desnecessários. Protocolos de comunicação com recursos de economia de energia geralmente reforçam a programação em modo sleep nos dados de transmissão. Agregação de dados no roteamento pode reduzir a carga de tráfego e então economizar energia de transmissão. A compressão de dados também é uma técnica potencial para a economia de energia durante a transmissão de dados. (IIKUBO; LO, 2019)

A programação do modo sleep é realizada periodicamente alternando o microcontrolador entre modo hibernação e modo ativo. Um programa watchdog é usado para despertar o dispositivo do modo de hibernação. Dois modos de hibernação costumam ser fornecidos: deep sleep e shallow sleep, a duração do sono do primeiro é maior do que o último. Um nó sensor começa com deep sleep, se o nó sensor detecta quaisquer valores anômalos que estão além de uma faixa normal pré-definida durante o modo ativo, este sensor enviará uma mensagem de alerta ao gateway do sensor que por sua vez, transmite a mensagem para todos os nós sensores conectados. Um nó sensor muda para shallow sleep sempre que um alerta de mensagem é recebido. Um sensor em shallow sleep ao descobrir que

os valores detectados caem dentro da faixa normal enviará uma mensagem clara para o gateway do sensor. Se a maioria dos sensores estiverem conectados ao mesmo gateway, então o gateway do sensor transmite as mensagens para todos os sensores conectados. Um sensor muda para deep sleep sempre que uma mensagem é recebida. (IIKUBO; LO, 2019)

No mapa 1 é possível visualizar a interação entre os modos de economia de energia comumente encontrados em microcontroladores e microprocessadores, o relógio RTC(Real Time Clock) permite a contagem do tempo, a manutenção da data e da hora em qualquer modo de operação e pode ativar e desativar os modos de economia de energia. O modo active permite a utilização de todos os recursos do dispositivo bem como utilização total do poder de processamento da CPU. O modo Idle é ativado quando a CPU não é exigida, o modo sleep mantém a operação de alguns periféricos e desativa as operações da CPU, dessa forma os periféricos são gerenciados pelo relógio RTC e podem ter acesso direto a memória. Já o modo deep sleep desliga a CPU e todos os periféricos afim de se obter o mínimo de consumo de energia.



Mapa 1: Modos de operação para economia de energia de um sistema embarcado. Fonte: LIN et al.,(2020).

## 2.2 Algoritmos de inteligência artificial.

Nos tempos atuais podem ser encontrados diversos algoritmos de inteligência artificial, como cada um deles possui características específicas, serão tratados os que mais se adequam ao projeto proposto.

A figura 1 resume alguns dos grupos de algoritmos utilizados em inteligência artificial.

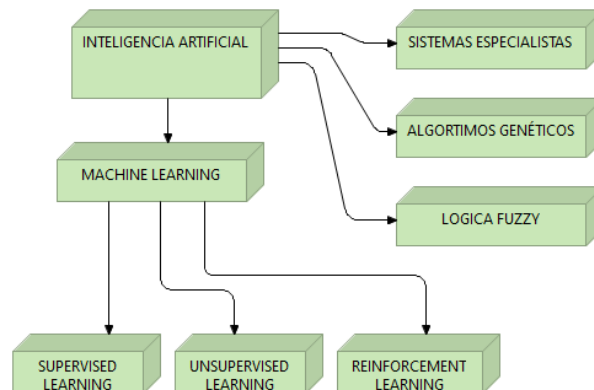


Figura 1: Classificações dos algoritmos de inteligência artificial abordados. Fonte: Elaborado pelo autor (2021).

Uma das áreas de inteligência artificial que mais tem se desenvolvido nos últimos anos é a área de machine learning ou aprendizado de máquina, que na verdade é um método automatizado de construção de modelos analíticos que são capazes de reconhecer padrões e realizar previsões. Para realizar essas previsões os modelos passam por processos de aprendizagem onde recebem informações úteis para criação de sua base de conhecimento que será utilizada durante as classificações e previsões desenvolvidas pelo algoritmo, funcionando de forma semelhante ao cérebro humano através da cognição (AKERKAR, RAJENDRA, 2018).

O ramo de machine learning pode ser classificado em três grandes áreas no que se refere ao tipo de supervisão que o sistema recebe durante o período de aprendizagem, sendo:

Aprendizado Supervisionado;

Aprendizado Não Supervisionado;

Aprendizagem por Reforço.

No aprendizado supervisionado o Sistema de inteligência artificial faz o aprendizado com dados previamente classificados e rotulados, como por exemplo, um sistema de reconhecimento de imagens irá receber figuras que já estão rotuladas como carro, celular e chave, ou seja, o sistema irá ser informado sobre qual imagem ele está sendo exposto. Dessa forma o sistema irá prever os próximos dados com base em informações já apresentadas (JOSHI, 2018).

Já no aprendizado não supervisionado o sistema não possui dados classificados para utilizar na aprendizagem, e busca características dos dados de forma autônoma, ou seja, ele busca por

informações em comum nos dados para efetuar a classificação dos mesmos. (AKERKAR, RAJENDRA, 2018).

Por exemplo, se um sistema não supervisionado fosse orientado para separar em 4 grupos os visitantes de um site que possuem características semelhantes, o sistema iria realizar automaticamente a separação baseando-se em características em comum dos dados como, horário de acesso, gênero, idade e local de origem e irá descobrir os principais grupos de perfis sem dados prévios de treinamento. Outro exemplo é descobrir a associação entre produtos comprados em um supermercado, pois ao ir as compras um cliente raramente compra apenas um produto

A área de reinforcement learning ou aprendizado por reforço, possui uma abordagem um pouco diferente, onde o sistema aprende qual é a melhor decisão a ser tomada dependendo obviamente das circunstâncias encontradas. O processo se baseia no princípio da recompensa e da punição, onde toda ação tomada resulta em uma punição ou uma recompensa, esse processo ocorre milhares de vezes até que o sistema aprende a melhor ação a ser tomada nas mais diversas situações (LI, 2019). Na figura 2 é possível observar as divisões dos algoritmos de machine learning.

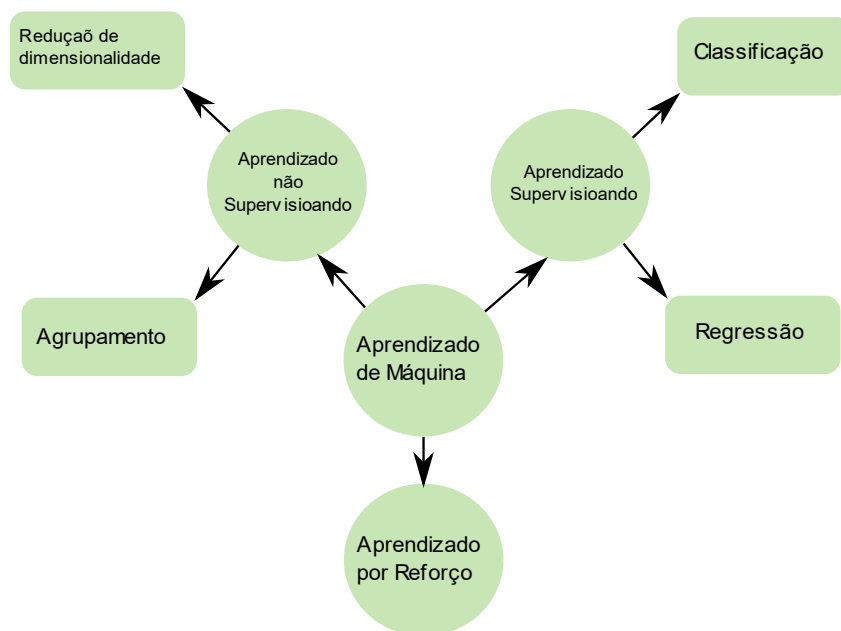


Figura 2: Divisões da área de machine learning. Fonte: Elaborado pelo autor (2021).



## 2.2.1. Aprendizagem Não Supervisionada;

### 2.2.1.1 Redução de Dimensionalidade

Os algoritmos de redução de dimensionalidade são utilizados no processo de redução do número de variáveis aleatórias em consideração, obtendo-se um conjunto de variáveis principais, em muitos casos, a maioria das variáveis é correlacionada e considerada redundante. Quanto maior o número de recursos, mais difícil é visualizar o conjunto de treinamento, dificultando os trabalhos no mesmo (JOSHI, 2018).

A redução de dimensionalidade é utilizada em visualização de big data, extração de recurso e reduz o tempo de computação e o espaço de armazenamento de dados.

### 2.2.1.2 Clustering

O clustering conhecido também por análise de agrupamento de dados, é o conjunto de técnicas de mineração de dados com o objetivo de agrupar dados automaticamente segundo algum padrão de associação. Essa técnica pode ser utilizada para customização de segmentos, sistemas de recomendação e para seleção de mercado alvo na área de marketing (CHOWDHARY, 2020).

## 2.2.2 Aprendizagem Supervisionada

### 2.2.2.1 Classificação

Os algoritmos de classificação são utilizados para classificar um conjunto de dados conforme suas características e padrões previamente determinados, funciona com base no reconhecimento de padrões. Um exemplo muito comum de classificação é a seleção de e-mails, onde se pode classificar uma mensagem como spam ou como mensagem regular.

### 2.2.2.2 Regressão

A regressão é um recurso de modelagem estatística e tem a função de estimar a relação entre duas ou mais variáveis utilizando como base dados de amostra. Um exemplo muito conhecido é o de regressão linear, onde o algoritmo traça uma reta próxima a concentração dos dados possibilitando obter informações dos intervalos de dados. (AKERKAR, RAJENDRA, 2018).

### 2.3 Aprendizagem por reforço.

Aprendizado por reforço é um modelo de aprendizado de máquina onde o algoritmo realiza uma sequência de decisões. Pois o algoritmo utiliza um processo de tentativa e erro para encontrar uma solução para o problema e recebe recompensas ou penalidades pelas ações que executa, onde o objetivo é maximizar as recompensas. É utilizado em sistemas de decisão em tempo real, navegação de robôs, games, aquisição de habilidades e tarefas de aprendizado (JOSHI, 2018).

### 2.4 Lógica Fuzzy

A lógica fuzzy ou também conhecida como lógica difusa tem o objetivo de fazer com que as decisões tomadas pela máquina se aproximem das decisões tomadas por seres humanos no cotidiano. É uma lógica desenvolvida para lidar com conceitos de verdade parcial e capaz de capturar informações vagas representadas por linguagem natural e converter para o formato numérico. É baseada em graus de verdade ou pertinência como por exemplo a temperatura de um objeto que pode ser considerado frio, morno, quente ou muito quente, onde os graus podem variar entre 0(falso) e 1(verdadeiro) (PAREDES, 2020).

Portanto a Logica fuzzy pode ser utilizada para avaliar níveis de pertinência de uma mesma variável, não oferecendo definição exata. Sendo muito úteis pra lidar com dados incompletos e imprecisos, realizando aproximação de funções, classificação, controle e previsão de dados.

Como no exemplo da figura 3 a logica Fuzzy pode ser utilizada para determinar o estado da carga da bateria, descrevendo a porcentagem da capacidade que ainda resta na bateria.

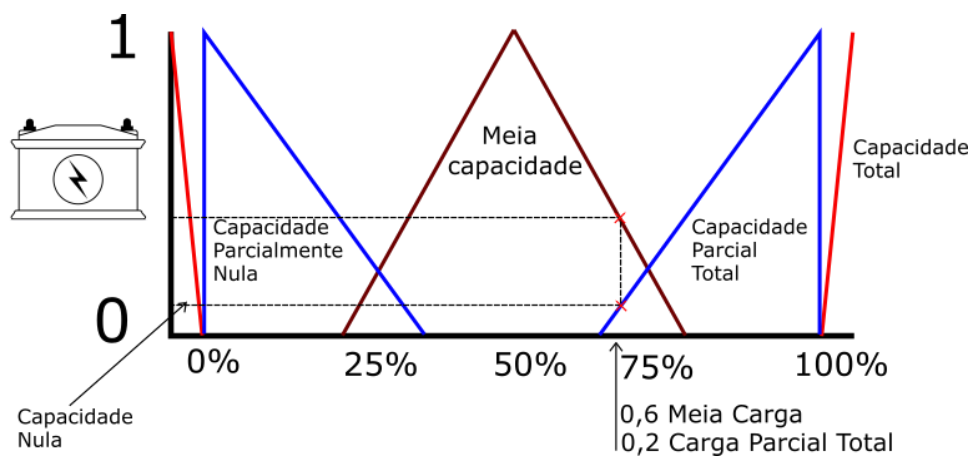


Figura 3: Logica Fuzzy aplicada a análise de capacidade de carga de baterias. Fonte: Elaborado pelo autor (2021).

No processo chamado de fuzzificação as variáveis linguísticas são definidas de forma subjetiva, e podem ser gerados diversos tipos de espaços como trapezoidal ou triangular utilizado no exemplo acima. Uma particularidade da Lógica fuzzy é que não possuem memória ou capacidade de aprendizado, os dados são expostos a lógica gerada inicialmente (PAREDES, 2020).

Para reduzir essa limitação é possível o desenvolvimento de sistemas híbridos onde a técnica de redes neurais pode ser combinada com a lógica fuzzy.

## 2.5 Redes Neurais Artificiais

O conceito de redes neurais artificiais foi introduzido pela primeira vez em 1943 pelo neurofisiologista Warren McCulloch e pelo matemático Walter Pitts. Em um artigo denominado, “Um cálculo lógico de ideias imanentes à atividade nervosa”, McCulloch e Pitts apresentaram um modelo computacional simplificado de como os neurônios biológicos podem trabalhar juntos em cérebros de animais para realizar cálculos complexos (CHOWDHARY, 2020).

Desde então os algoritmos baseados em redes neurais tem sido largamente utilizados para desenvolvimento de sistemas de inteligência artificial.

### 2.5.1 Camadas de uma rede Neural

A camada de entrada é a primeira camada de uma rede neural artificial que recebe as informações de entrada nas formas mais variadas como textos, números, áudio e imagem. Após a camada de entrada, os dados são direcionados para uma ou várias camadas ocultas, onde são executados vários tipos de cálculos e reconhecimento de padrões desses dados. Na camada de saída obtêm-se os cálculos realizados nas camadas ocultas.

### 2.5.2 O Perceptron

O perceptron é um dos modelos mais básicos de redes neurais artificiais. É um classificador simples de uma camada, podendo receber  $n$  atributos em diversas camadas de entrada, porém possui apenas uma única camada de saída que produz apenas dados binário, ou seja 0 e 1 ou verdadeiro/Falso (CHOWDHARY, 2020).

Cada atributo possui um peso na relevância do processamento do perceptron, ou seja, a variável  $X_1$  pode ter uma consideração maior do que a segunda variável para a tomada de decisão do conjunto.

O perceptron pode ser essencialmente útil para tarefas de classificação, mais adiante será tratado outros algoritmos mais completos.

### 2.5.3 Rede Neural FeedForward

Nas redes neurais tipo feedforward, o fluxo de dados ocorre apenas em uma direção, sendo da camada de entrada para a camada oculta e na sequência para a camada de saída. Não há loops de feedback presentes nessa rede neural, por isso é usada em casos em que os dados não são sequenciais por natureza. Caso os dados analisados sejam sequenciais ou dependentes do tempo como o som e dados de séries temporais é mais adequado a utilização de redes neurais do tipo feedback que possuem recursos de retenção de memória (BELOVA,2019).

Esse tipo de rede neural é usado principalmente no aprendizado supervisionado para classificação e reconhecimento de imagem.

### 2.5.4 Rede Neural Recorrente

Uma rede neural recorrente é um tipo de rede neural utilizada para a análise de dados sequenciais e temporais como por exemplo análise de linguagem natural, séries temporais como a variação da bolsa de valores, a variação da temperatura ambiente ao longo de um dia e na tradução automática de áudio e legendas de vídeos. É largamente utilizada para a previsão de dados baseada no histórico de variação dos mesmos, pois é capaz de armazenar informações ao longo do tempo.

### 2.5.5 Redes neurais LSTM (Long Short Term Memory )

As redes neurais recorrentes possuem uma limitação chamada de vanishing gradiente que dificulta o reconhecimento de alguns tipos de padrões e o treinamento da rede em alguns casos. Para solucionar essa limitação foram desenvolvidas as redes LSTM que é uma arquitetura de rede neural recorrente (RNN) capaz de lembrar valores em intervalos temporais arbitrários, por isso é capaz de processar e prever séries temporais com intervalos de tempo de duração desconhecida extraindo

informações do contexto dos dados, algo que a rede neural recorrente básica não é capaz de executar. (KHUMPROM e YODO, 2019).

Sua aplicação ocorre em reconhecimento de fala, tradução automática, análise de sentimentos e séries temporais encontradas na engenharia, economia, saúde e outros.

#### 2.5.6 Autoencoders

Autoencoders são um tipo de redes neurais artificiais, onde a entrada é igual à saída. Eles compactam a entrada em um código de menor dimensão e em seguida reconstruem a saída dessa representação. O código é um resumo ou compactação da entrada, ou seja, os autoencodificadores são um algoritmo de redução de dimensionalidade e compactação de dados do mesmo tipo, como imagens de números manuscritos ou fotografias. É um processo não supervisionado, ou seja, basta inserir os dados na camada de entrada da rede e a mesma se encarrega de criar os rótulos dos dados fornecidos. Vale lembrar que os dados compactados na saída não são exatamente como os mesmos fornecidos inicialmente, existe uma degradação no processo de codificação (VEERARAGHAVAN, 2017).

As principais aplicações práticas dos Autoencoders é a remoção de ruídos em dados e a redução de dimensionalidade para visualização de dados.

#### 2.5.7 Redes neurais convolucionais

Redes Neurais Convolucionais são um tipo de algoritmo de aprendizagem profunda que realiza o aprendizado com informações extraídas de imagens através de filtros, isso lhe permite aprender os objetos presentes em uma imagem e discernir uma imagem entre outras diferentes (KHUMPROM, 2019).

Durante o treinamento inicial, os filtros podem exigir ajustes manuais mas com o progresso do treinamento, a rede é capaz de se adaptar aos recursos aprendidos e desenvolver filtros próprios.

Sua principal aplicação é no reconhecimento de imagens para classificação, segmentação e localização de objetos em imagens. Embora também possam ser utilizadas para análise de imagem de gráficos como espectrograma de áudio, análise de dados gráficos e até mesmo associado a outras redes como LSTM para reconhecimento de fala (KHUMPROM, 2019).

### 3.0 Vantagens das Redes neurais LSTM em comparação a outros métodos de previsão.

Para realizar as previsões o projeto utilizou o modelo de redes neurais recorrentes tipo LSTM, esse escolha se deu pelos seguintes motivos:

- O modelo LSTM opera com eficiência mediante várias variáveis de entrada ou saída.
- Em padrões de dados não lineares o modelo LSTM oferece mais precisão.
- Elevada precisão mesmo sendo treinada por um número reduzido de amostras.
- Modelos LSTM possuem a capacidade de prever variações em curto e longo prazo.

A previsão de séries temporais é uma área de pesquisa muito exigente devido ao seu enorme potencial em diferentes aplicações como previsão de preços de ações, planejamento de negócios, previsão do tempo, alocação de recursos e vários outros. Apesar de a previsão poder ser considerada como um subconjunto de problemas de regressão supervisionados, algumas ferramentas são vitais por causa da ideia de percepção do mundo real. Alguns modelos de previsão de série temporal comuns são Autoregressão (AR), Média móvel autoregressiva (ARMA) e Média móvel integrada autoregressiva (ARIMA). Diferentes análises nos mercados de ações são frequentemente discutidas e considerado como um tipo único de série temporal. Alguns significativos padrões não podem ser capturados com precisão por meio de métodos tradicionais que dependem dos métodos de regressão linear por causa da natureza complexa dos modelos de séries temporais. Uma grande parte de séries temporais mostra não linearidade se esses modelos forem usados. Além disso, é difícil prever ou estimar as finanças sem o uso de técnicas mais robusta e não lineares. (ISTIAKE SUNNY; MASWOOD; ALHARBI, 2020)

A predição de série temporal multivariável tem sido amplamente estudada em energia, aerologia, meteorologia, finanças, transporte, etc. Os métodos de modelagem tradicionais têm padrões complexos e são ineficientes para capturar dependências multivariadas de dados de longo prazo para previsão com a precisão desejada. Para lidar com essas preocupações, existem vários modelos de aprendizado profundo baseados em Rede Neural Recorrente Métodos (RNN) e Rede Neural Convolutiva (CNN). (WAN et al., 2019a)

Utilizando métodos neurais profundos, é possível prever o valor futuro do preço das ações com demonstração excepcionalmente não linear. Uma rede neural se esforça para mapear e destacar as informações que são necessárias para estar familiarizado com uma função e assim, alcançar uma melhor previsão. O desempenho do Deep Learning é superior a qualquer outro modelo para fazer previsões não lineares dados em diferentes problemas de previsão de séries temporais. Um método de aprendizado profundo conhecido como Gated Recurrent Unit (GRU) consiste no mesmo tipo de estrutura como o modelo LSTM, exceto que o design da célula de memória é simplificado no modelo GRU. Para reduzir a estrutura celular da memória, o modelo GRU contém apenas duas portas, ou seja, a porta de reinicialização e a porta de atualização. O modelo GRU é notável porque leva menos tempo para treinar e funciona bem o suficiente em uma pequena quantidade de dados. No entanto, LSTM é mais eficiente em comparação com GRU, especialmente para lidar com a não linearidade em conjuntos de dados maiores. (ISTIAKE SUNNY; MASWOOD; ALHARBI, 2020)

Istiake publicou em 2020 um artigo que compara a precisão do método ARIMA e o modelo LSTM, como técnicas representativas na previsão de dados em séries temporais. Essas duas técnicas foram implementadas e aplicadas em um conjunto de dados financeiros e os resultados indicados pelo modelo LSTM era superior ao ARIMA. Mais especificamente, o algoritmo baseado em LSTM melhorou a previsão em 85% em média em comparação com ARIMA.

O sistema de previsão de série temporal envolve prever o comportamento do sistema no futuro, que é com base na informação do estado atual e passado do sistema. No passado, o problema de previsão de série temporal foi influenciado por métodos estatísticos lineares para atingir as atividades de previsão. (SAGHEER; KOTB, 2019)

Uma única variável dependente do tempo significa uma série temporal uni-variada, enquanto uma série temporal multivariada como dados ambientais tem mais de uma variável dependente do tempo, cada variável depende de seus valores anteriores e também de outras variáveis. As melhores soluções de problemas de modelagem para múltiplas variáveis de entrada são com redes neurais recorrentes e são a grande solução para vários problemas de previsão de série temporal, onde os métodos lineares clássicos não podem oferecer precisão elevada. (ALHIRMIZY; QADER, 2019)

Um dos maiores desafios da previsão de séries temporais multivariadas é a não linearidade e a periodicidade de dados originados por comportamento dinâmico de curto e longo prazo. Vários modelos foram estabelecido com base em métodos estatísticos clássicos ou algoritmos de aprendizagem de máquina. O modelo de série temporal univariada clássico proeminente é Autoregressivo (AR) com algoritmos de estatística clássica, o método AR é bem utilizado para séries temporais estacionárias. O modelos melhorados tais como média móvel autorregressiva integrada (ARIMA) , movimento autorregressivo média (ARMA) e auto-regressão vetorial (VAR), foram desenvolvidos incluindo técnicas de suavização exponencial. No entanto, para padrões temporais de longo prazo, esses modelos são inevitavelmente sujeitos a sobreajuste e alto custo computacional, especialmente para entradas de alta dimensão. Métodos alternativos, tratando os problemas de previsão de séries temporais como regressão geral com parâmetros variáveis no tempo foram aplicados por modelos de aprendizado de máquina, por exemplo, vetor de suporte linear (SVR) , floresta aleatória , regressão de crista e LASSO. Esses modelos são praticamente mais eficientes devido às soluções prontas para uso de alta qualidade na comunidade de aprendizado de máquina. Ainda assim, os modelos baseados em aprendizado de máquina podem ser incapazes de incluir dependências não lineares complexas de grandes conjuntos de dados multivariados. Enquanto isso, as redes neurais profundas bem construídas de Redes Neurais Convolucionais (CNNs) e Redes Neurais Recorrentes (RNNs) têm sido amplamente aplicadas na previsão de séries temporais, que são atribuídos às estruturas de aprendizado profundo de código aberto, como Keras. Alguns modelos representativos compreendem modelos LSTM e seus herdeiros, LSTM convolucional (ConvLSTM), e Redes LSTM Totalmente Convolucionais para Classificação de Séries Temporais LSTM-FCN, que superam os desafios envolvidos no treinamento de uma rede neural recorrente para uma mistura de horizontes de longo e curto prazo.(WAN et al., 2019b)



### 3.0 METODOLOGIA

Nesse capítulo são apresentadas as metodologias utilizadas para desenvolvimento do presente trabalho, dessa forma será apresentada a classificação da pesquisa e as etapas constituintes de sua elaboração.

#### 3.1 Classificação da Pesquisa

Do ponto de vista da natureza o trabalho se classifica como uma pesquisa aplicada. Com foco qualitativo pois é de caráter exploratório. Com o objetivo de descrever as características dos sistemas de inteligência artificial aplicáveis ao gerenciamento de energia, o presente trabalho se classifica como descritivo. (SILVA E MENEZES, 2005).

Levando em consideração que parte da pesquisa é baseada na análise de livros, artigos e conhecimentos já obtidos e divulgados pela comunidade científica a mesma se classifica como bibliográfica.

#### 3.2 Etapas da Pesquisa

O trabalho é dividido em quatro etapas compreendidas da seguinte forma:

Tabela 2: Etapas do projeto. Fonte: Elaborado pelo autor (2021). Fonte: Elaborado pelo autor (2021).

<b>Etapa</b>	<b>Descrição</b>
1	Apresentação dos dispositivos utilizados no levantamento da base de dados.
2	Projetar e fabricar uma placa de circuito impresso.
3	Desenvolver o firmware para o processador do sistema realizar o levantamento da base de dados.
4	Treinar e otimizar redes neurais recorrentes.
5	Desenvolver e testar o scheduler de gerenciamento de energia com um microprocessador BCM2835.

Fonte: Elaborado pelo autor (2021).

Na figura 4 é possível visualizar o diagrama de blocos da sequência das atividades realizadas para concluir a pesquisa.

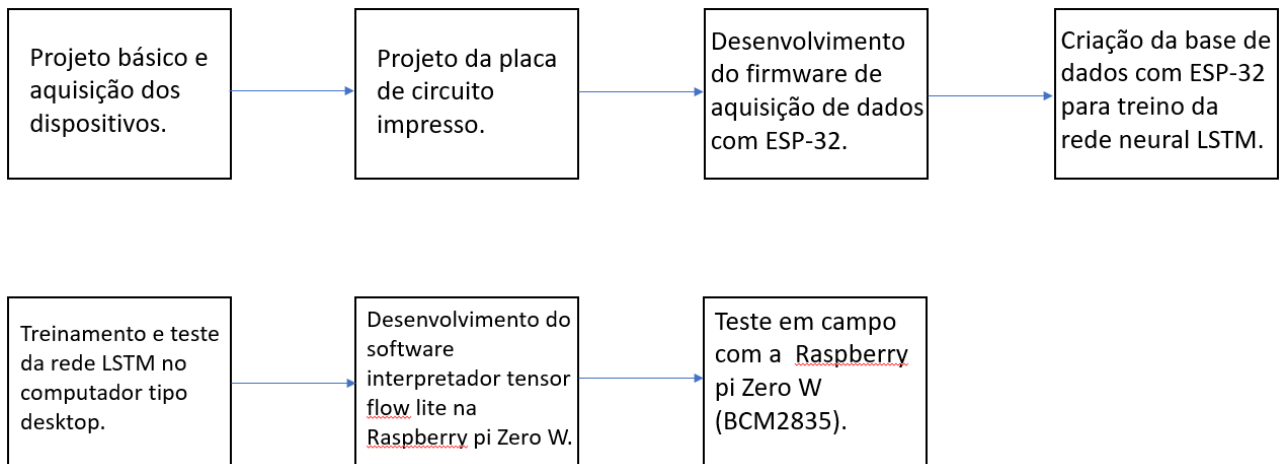


Figura 4: Atividades para execução do projeto. Fonte: Elaborado pelo autor (2021).

### 3.3 Fluxo de trabalho para criação de um modelo baseado em redes neurais.

Segundo Warden e Situnayake o fluxo de trabalho para criar e utilizar um modelo de inteligência artificial segue as seguintes etapas:

- 1 Estabelecer o objetivo;
- 2 Coletar os dados de base para o treinamento;
- 3 Escolher a arquitetura do modelo;
- 4 Treinar o modelo em um computador pessoal;
- 5 Converter o modelo para uso em microcontroladores;
- 6 Executar inferência;
- 7 Avaliação e otimização dos resultados;

Para utilização eficiente das redes neurais em dispositivos móveis foi utilizado o framework Tensor Flow Lite, que possibilita a criação de um modelo otimizado para ser executado em dispositivos móveis como smartphones e microcontroladores que possuem características limitadas de memória e processamento em comparação a outros sistemas computacionais.

Para atender a esses requisitos de tamanho menor para plataformas móveis, em 2017 o Google iniciou um projeto complementar para a linha principal do TensorFlow, chamado TensorFlow Lite. Esta biblioteca é destinada a executar modelos de rede neural de forma eficiente e fácil em

dispositivos móveis. Para reduzir o tamanho e a complexidade do framework, ele elimina recursos que são menos complexos nessas plataformas. Por exemplo, ele não oferece suporte a treinamento, apenas executar inferências em modelos que foram previamente treinados em uma plataforma de nuvem. Também não suporta a gama completa de tipos de dados (como double) disponíveis no Tensor da linha principal. Além disso, algumas operações menos usadas não estão presentes, como `tf.depth_to_space`. Em troca dessas compensações, o TensorFlow Lite pode caber em apenas algumas centenas de quilos bytes, tornando muito mais fácil caber em um aplicativo de tamanho limitado. Também tem bibliotecas altamente otimizadas para CPUs Arm Cortex-A-series, junto com suporte para API de rede neural do Android para aceleradores e GPUs por meio de OpenGL. (WARDEN; SITUNAYAKE, 2020)

Os algoritmos de inteligência artificial podem ser utilizados para determinar diversas variáveis em sistemas eletroeletrônicos, como por exemplo a análise de carga em baterias. Os modelos de inteligência artificial usam funções não lineares aprendíveis para identificar de forma adaptativa o modelo da bateria a partir das variáveis observáveis da bateria, como tensão terminal, corrente de entrada, temperatura da célula, etc. e, portanto, são capazes de estimar os estados internos da bateria diretamente dos dados. O aprendizado de máquina e os algoritmos de aprendizado profundo se enquadram nessa categoria. Eles são robustos ao ruído, fornecem baixos valores de erro de estimativa e são altamente escaláveis e facilmente implantáveis. Diferentes tipos de redes neurais têm sido usados anteriormente na literatura para fins de estimativa de estado de carga da bateria.(BHATTACHARJEE et al., 2021)

#### 4.0 MATERIAIS

Nessa etapa foram selecionados os dispositivos elétricos e eletrônicos que foram utilizados para a construção do protótipo. No diagrama de blocos da figura 5 é possível identificar os principais elementos do sistema e sua interação básica. Esse conjunto de dispositivos foi utilizado para o levantamento da base de dados usada posteriormente para treino da rede neural recorrente.

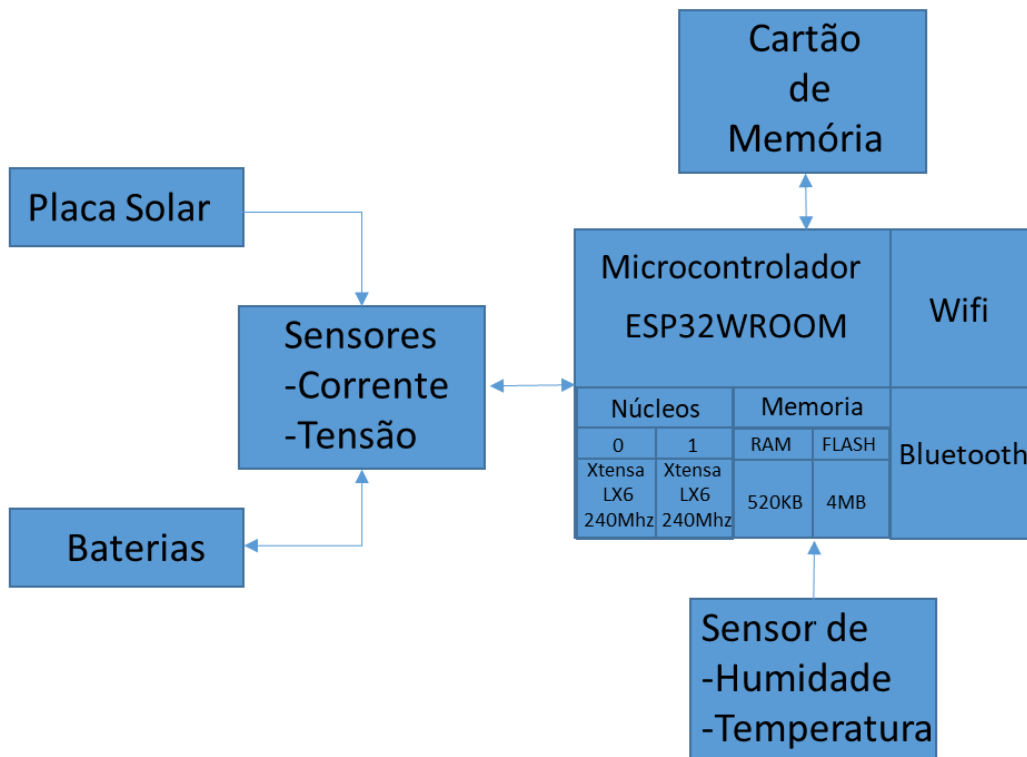


Figura 5: Diagrama de blocos dos componentes do projeto. Fonte: Elaborado pelo autor (2021).

Na figura 6 é possível visualizar as fotos dos componentes utilizados para realizar o levantamento da base de dados:

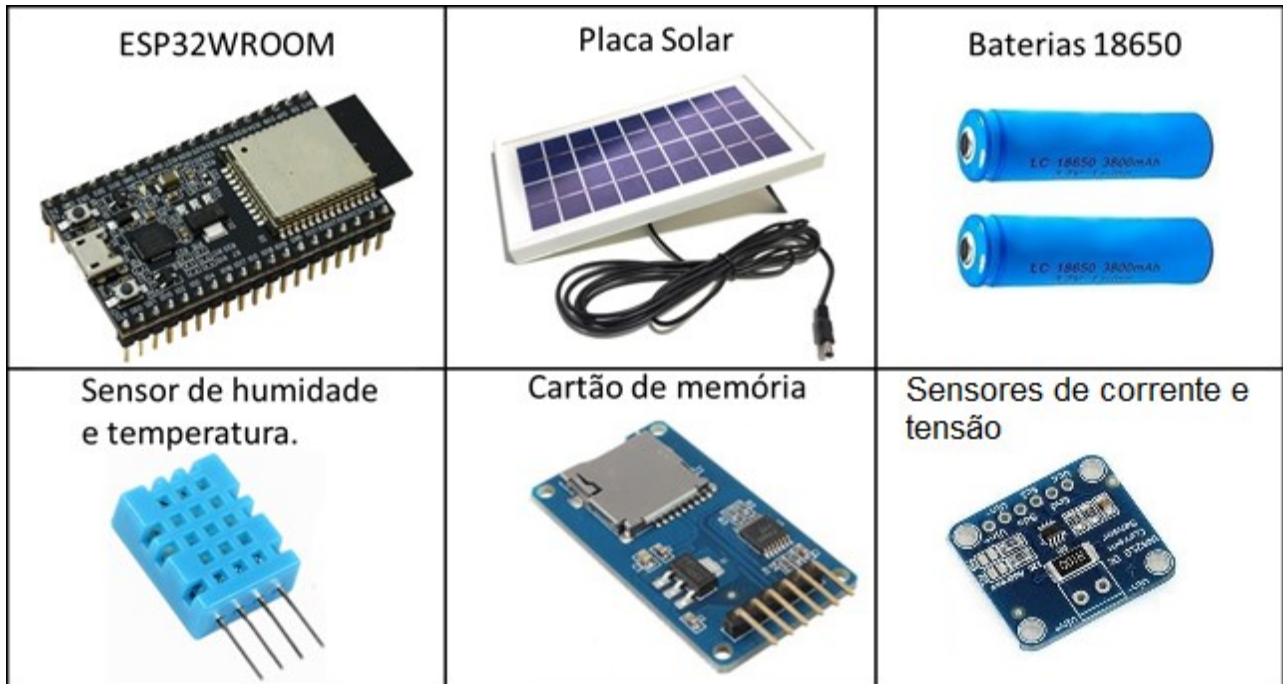


Figura 6:Fotos dos componentes. Fonte: Elaborado pelo autor (2021).

Os módulos sensor de corrente utilizam o circuito integrado INA219 que juntamente com um resistor de  $0.1\Omega$  forma um excelente sensor de corrente e tensão que opera mediante o protocolo de comunicação I2C, o mesmo ainda possui as seguintes características:

Tabela 3: Características do módulo INA-219. Fonte: Elaborado pelo autor (2021).

Módulo INA-219	Características
Tensão de alimentação.	3 à 5.5VDC.
Faixa de tensão no barramento de teste.	0 à 26VDC.
Corrente máxima de operação.	$\pm 3.2A$ com resolução de $0.8mA$ .
<b>Variáveis medidas.</b>	Tensão, Corrente e Potência.
<b>Comunicação.</b>	Protocolo I2C com endereço seleccionável.
Temperatura de operação.	$-40^{\circ}C$ a $125^{\circ}C$ .
Resistor Shunt	$0.1\Omega$

Fonte: Elaborado pelo autor (2021).

O microcontrolador utilizado para a criação da base de dados foi o ESP32WROOM que possui as seguintes características:

Tabela 4: Características do ESP32WROOM. Fonte: Elaborado pelo autor (2021).

<b>Bloco</b>	<b>Características</b>
MCU	<ul style="list-style-type: none"> <li>-ESP32-D0WD-V3 embedded, Xtensa® dual-core</li> <li>-32-bit LX6 microprocessor, up to 240 MHz</li> <li>-448 KB ROM for booting and core functions</li> <li>-520 KB SRAM for data and instructions</li> <li>-16 KB SRAM in RTC</li> </ul>
Wi-Fi	<ul style="list-style-type: none"> <li>-802.11b/g/n</li> <li>-Bit rate: 802.11n up to 150 Mbps</li> <li>-A-MPDU and A-MSDU aggregation</li> <li>-0.4 <math>\mu</math>s guard interval support</li> <li>-Center frequency range of operating channel: 2412 ~ 2484 MHz</li> </ul>
Bluetooth	<ul style="list-style-type: none"> <li>-Bluetooth V4.2 BR/EDR and Bluetooth LE specification</li> <li>-Class-1, class-2 and class-3 transmitter</li> <li>-AFH</li> <li>-CVSD and SBC</li> </ul>
Hardware	<ul style="list-style-type: none"> <li>-Interfaces: SD card, UART, SPI, SDIO, I2C, LED PWM, Motor PWM, I2S, IR, pulse counter, GPIO, capacitive touch sensor, ADC, DAC, Two-Wire Automotive Interface (TWAI®, compatible with ISO11898-1)</li> <li>-40 MHz crystal oscillator</li> <li>-4 MB SPI flash</li> <li>-Operating voltage/Power supply: 3.0 ~ 3.6 V</li> <li>-Operating temperature range: -40 ~ 85 °C</li> </ul>

Fonte: Elaborado pelo autor (2021).

O diagrama de blocos do ESP32WROOM pode ser visualizado na figura 7.

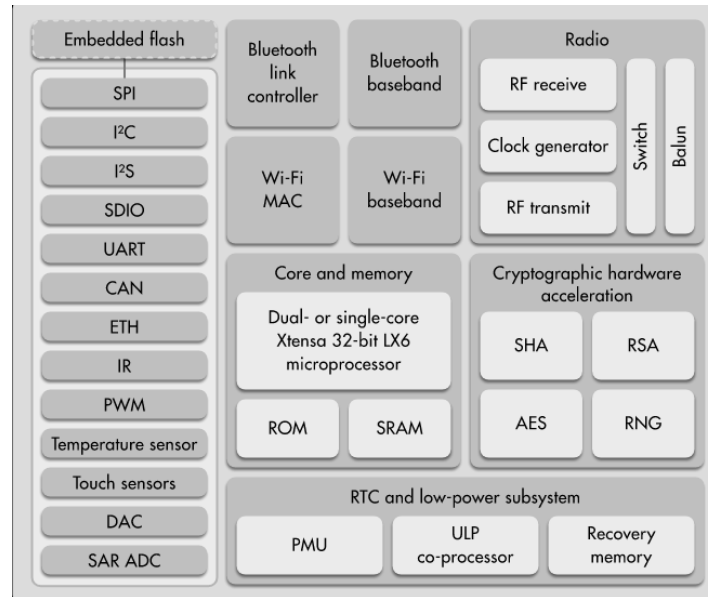


Figura 7: Diagrama de blocos do ESP32. Fonte: Espressif (2021)

Optou-se por utilizar o ESP-32 para aquisição de dados por 3 motivos principais:

- Baixo custo.
- Alta disponibilidade no mercado nacional.
- Facilidade de programação com 2 frameworks (ESP-IDF e Arduino).

Para armazenar a energia da placa fotovoltaica e permitir a operação do sistema durante períodos noturnos ou nublados, foram utilizadas duas baterias padrão 18650 de 3,7v e 2600mah fabricante Samsung ligadas em série, ambas as baterias estavam totalmente carregadas quando foram postas em operação. Para a proteção das baterias foi utilizado uma placa BMS (Battery Management System) modelo: FDC-2S-2, esse circuito oferece proteção contra sobrecargas no conjunto de baterias, prolongando sua vida útil no sistema, na tabela 5 é possível verificar as características da placa BMS.

Tabela 5: Características do circuito BMS. Fonte: Elaborado pelo autor (2021).

Características	Valores
Modelo	FDC-2S-2
Corrente máxima de carga	3A
Tensão de carga	8,4V a 9V
Faixa de identificação de sobretensão:	4,25V a 4,35V $\pm$ 0,05V

Corrente máxima de descarga continua	5A
Corrente máxima de pico de descarga	7A
Corrente de repouso	10uA
Resistência interna	0,1 $\Omega$
Dimensões	38x8x2,5mm

Fonte: Elaborado pelo autor (2021).

Como declarado por KIM em 2020 as baterias possuem vital importância em sistemas que utilizam energias renováveis como fonte principal de energia. Os estudos existentes sobre o problema de escalonamento de energia focaram na redução da potência de pico por meio do escalonamento de múltiplas operações de demanda de energia e sua análise. No entanto, uma solução completa para o problema de programação de energia também precisa caracterizar os armazenamentos / fontes de energia, pois afeta a capacidade de energia para as operações de demanda de energia. Além disso, as fontes de energia renováveis (por exemplo, energia solar, eólica e geotérmica) aumentam a capacidade de energia do sistema e reduzem a intensidade de carga nas baterias e supercapacitores, prolongando assim sua vida útil.(KIM et al., 2020)

A placa fotovoltaica utilizada para fornecer energia para o sistema possui dimensões de 25cm x 14cm x 1,7cm e pode fornecer até 3.5 watts de pico, na tabela 6 é possível visualizar todas as especificações fornecidas pelo fabricante Mighty Mule.

*Tabela 6: Características da placa fotovoltaica. Fonte: Elaborado pelo autor (2021).*

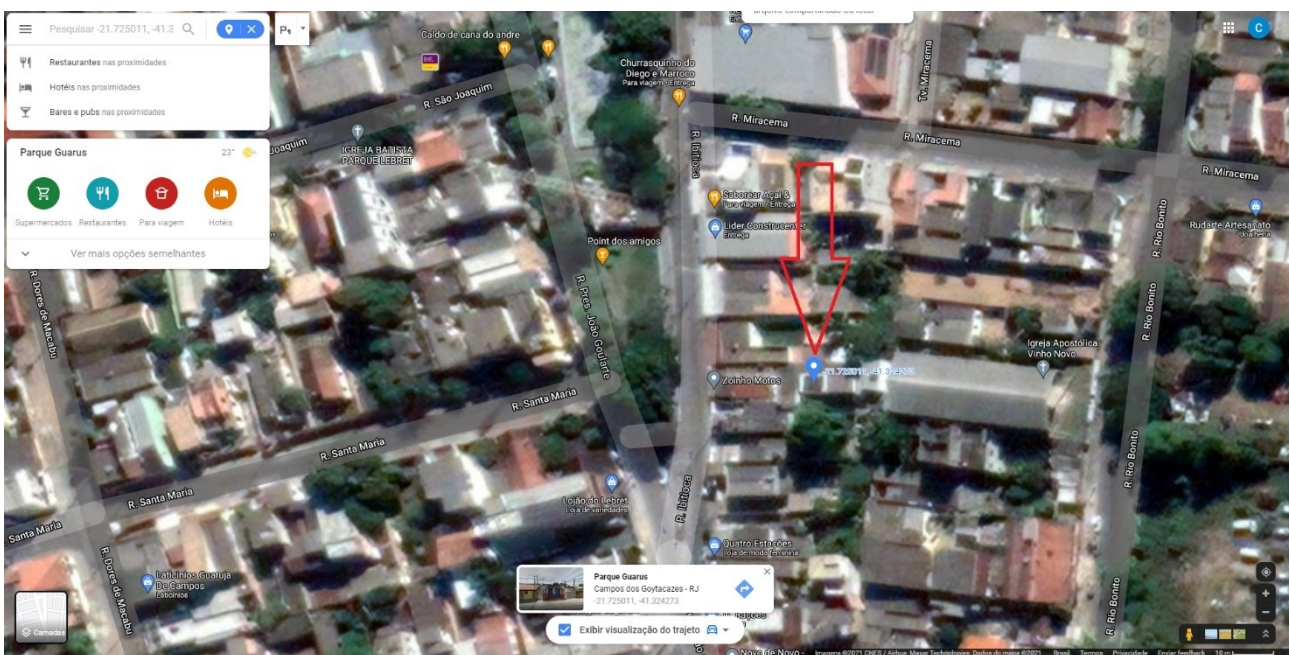
<b>Característica típica da placa fotovoltaica.</b>	<b>Valor</b>
Pico de Potência	3.5W
Tensão máxima (Vmp)	9V
Corrente máxima(Imp)	0.38A
Tensão em circuito aberto(Voc)	11.11V
Corrente de curto circuito(Isc)	0.418A
At STC(Standard Test Conditions): AM1.51000W/M <sup>2</sup>	25°C

Fonte: Elaborado pelo autor (2021).



Para estabilizar as tensões no protótipo foram utilizados reguladores de tensão da série 78XX, sendo o 7805 e 7833L, esses reguladores possuem uma considerável queda de tensão de 2 volts (Dropout Voltage = 2V), como a corrente de consumo total do projeto é de cerca de 100 miliampères não houve aquecimento excessivo dispensando o uso de dissipadores de calor. (SPARKFUN, 2020)

É possível a utilização de diversos tipos de outros circuitos integrados reguladores de tensão que podem oferecer uma queda de tensão menor convergindo para um rendimento superior a série 78XX.



### Parque Guarus

Campos dos Goytacazes - RJ

-21.725011, -41.324273

Mapa 2: Mapa de localização do ponto de testes do sistema. Fonte: Google Maps (acessado em 02/082021).



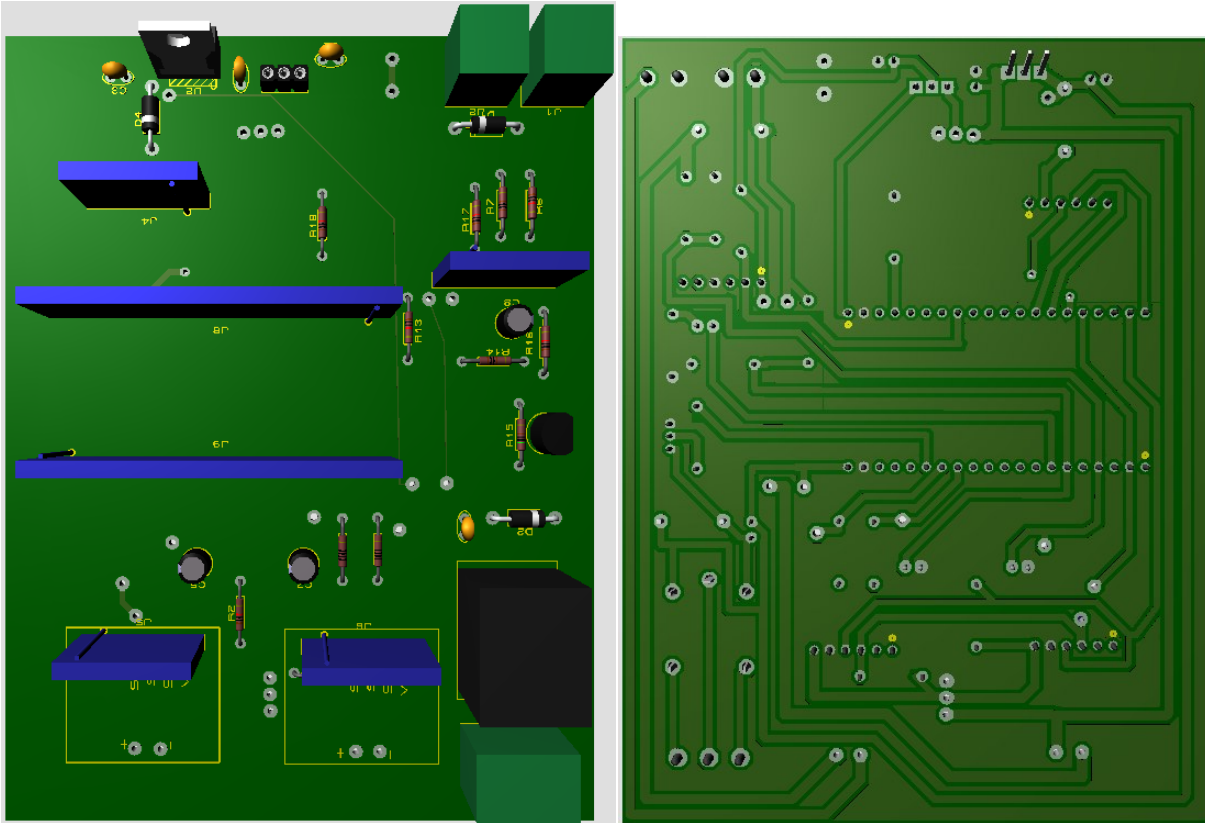
Figura 8: Placa fotovoltaica instalada. Fonte: Elaborado pelo autor (2021).

Na figura 8 é possível visualizar a posição da mini placa fotovoltaica instalada ao lado de um conjunto de placas fotovoltaicas que atendem a uma residência.

A placa fotovoltaica foi instalada no telhado de uma residência situada no bairro Parque Guarus, na cidade de Campos dos Goytacazes-RJ, Brasil, que possui Latitude  $-21.725011$  e Longitude  $-41.324273$  conforme a figura 11. A mesma está montada a 3 metros de altura ao lado de 10 placas fotovoltaicas que alimentam uma residência e com cerca de 30 graus de inclinação em relação ao solo.

Para operação do protótipo foi projetada a placa de circuito impresso utilizando software CAD dedicado para placas de circuito impresso, o software possibilita o roteamento automático das trilhas, restando poucas trilhas para posicionamento manual ou criação de jumper sobre a placa. O CAD oferece diversos recursos que facilitam a alteração do projeto como aumento de largura de trilhas,

alteração de dimensão de ilhas e checagem de falhas de conexão entre os pontos do circuito, as figuras 9 e 10, respectivamente, representam as projeções em 3D da vista superior e inferior da placa.



*Figura 9: Vista frontal da placa de circuito em 3D. Figura 10: Vista inferior da placa de circuito.*

Na figura 11 é possível visualizar as trilhas e os componentes durante o processo de edição da placa de circuito.

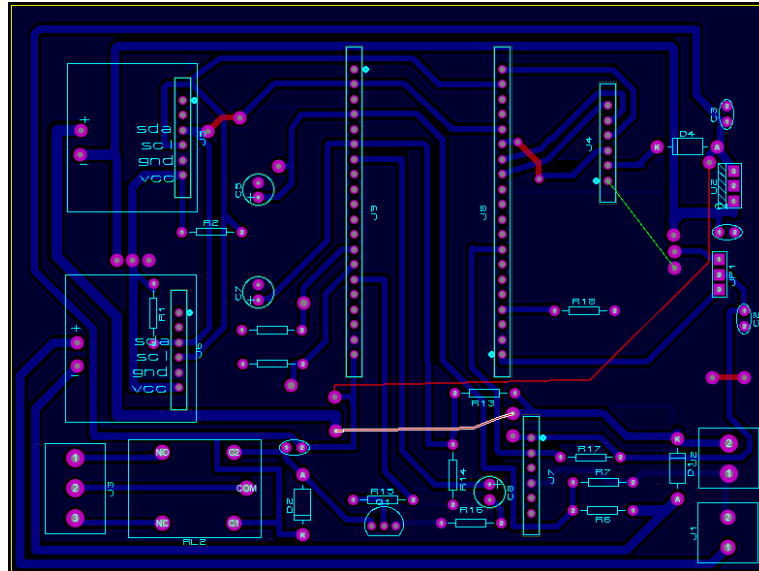


Figura 11: Placa de circuito impresso em desenvolvimento. Fonte: Elaborado pelo autor (2021).

Após finalizado o projeto da placa de circuito impresso, a mesma foi fabricada em uma fresadora de placa de circuito impresso modelo PCB-Proto 2 do fabricante TTW, pertencente ao Polo de Inovação do IFF em Campos dos Goytacazes/RJ, a figura 12 demonstra o aspecto da fresadora.

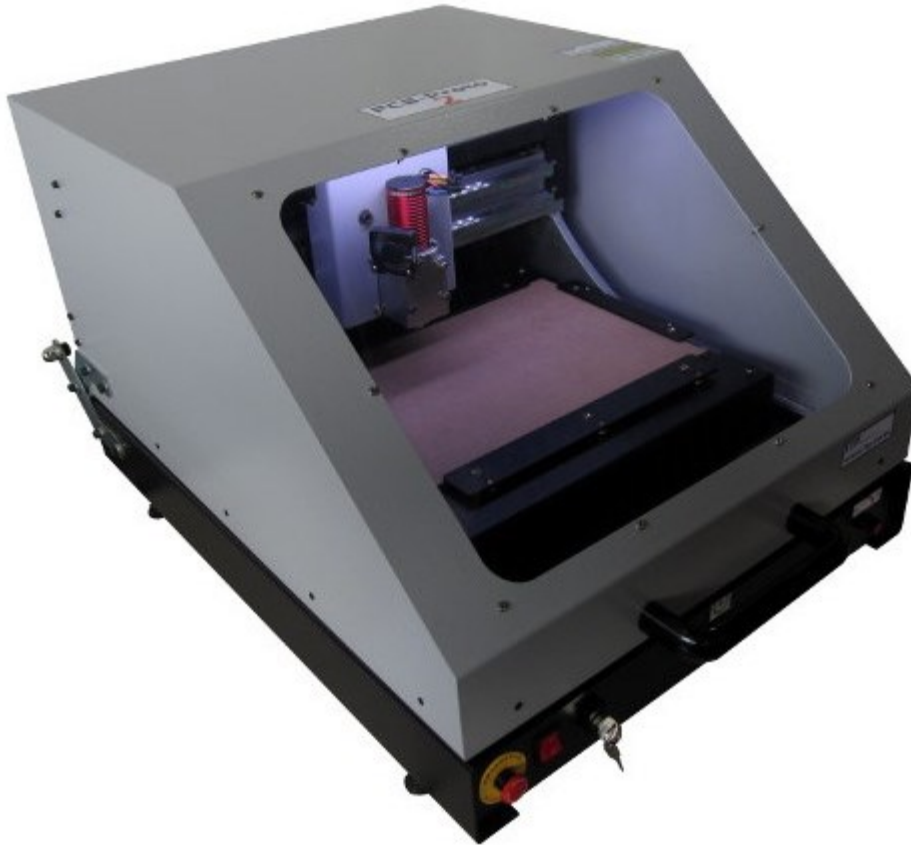


Figura 12: Fresadora CNC de placas de circuito impresso. Fonte: [www.ttp.ind.br](http://www.ttp.ind.br), acessado em 10/08/2021.

Segue características técnicas da fresadora disponíveis no site do fabricante(<http://www.ttp.ind.br/>).

Tabela 7: Características da fresadora de placas de circuito impresso. Fonte: Elaborado pelo autor (2021).

<b>Característica</b>	<b>Valores</b>
<b>Resolução</b>	Resolução de 0,625 $\mu$ m (podendo ser customizado para até 0,25 $\mu$ m)
<b>Diâmetros dos Furos</b>	0,1 mm até 2,5 mm
<b>Precisão de alinhamento geométrico</b>	Melhor que $\pm 0,02$ mm em todo o volume de trabalho
<b>Eixo arvore (spindle)</b>	Motor sem escovas de 38.000 RPM
<b>Velocidade de Trabalho</b>	1200 mm/min
<b>Velocidade de Furação</b>	100 furos/min
<b>Velocidade Máxima</b>	1200 mm/min

<b>Área de Usinagem</b>	305 mm x 220 mm
<b>Produz placas dupla face</b>	Sim
<b>Sistema de segurança</b>	Interruptor de segurança trava o movimento da máquina quando a tampa é aberta.
<b>Sistema de aspiração</b>	Liga automaticamente o sistema de aspiração, quando o spindle começa a girar.
<b>Peso</b>	82Kg
<b>Fabricação</b>	Fabricada no Brasil

Fonte: Elaborado pelo autor (2021).

Para fabricar a placa foi gerado o arquivo Gerber no software CAD e posteriormente o software da fresadora denominado PCB Proto Studio converteu o arquivo Gerber em arquivos tipo Gcode para a orientação da fresadora. Na figura 13 é possível visualizar a placa de circuito pós usinagem com as trilhas e furos completos.

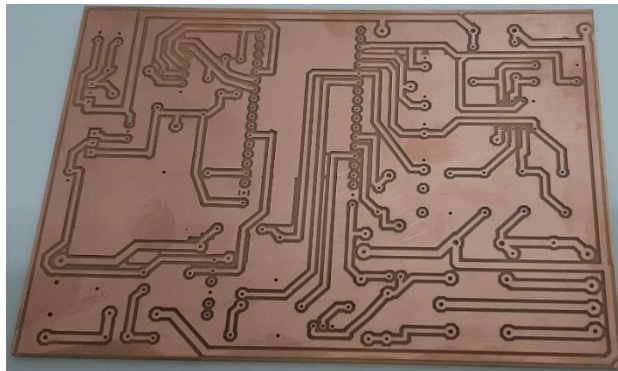


Figura 13: Placa de circuito impresso finalizada. Fonte: Elaborado pelo autor (2021).

Também foi utilizada a placa Raspberry Pi Zero W para execução do modelo treinado. Na figura 14 é possível visualizar o aspecto da mesma.

O processador BCM2835 foi utilizado no projeto pelos seguintes fatores:

- Baixo custo;
- Vasta quantidade de memória RAM;
- Baixo consumo de energia;
- Facilidade de programação em diversas linguagens;

- Processador de alta velocidade operando a 1Ghz.
- Total compatibilidade com modelos do Tensor Flow Lite.
- Várias portas de comunicação como Bluetooth, Wifi, protocolos SPI e I2C.
- Possui diversos pinos de I/O programáveis.

O processador BCM2835 foi utilizado como exemplo, porém é possível a utilização de outros microprocessadores de baixo consumo desde de que sejam compatíveis com o TensorFlow Lite.

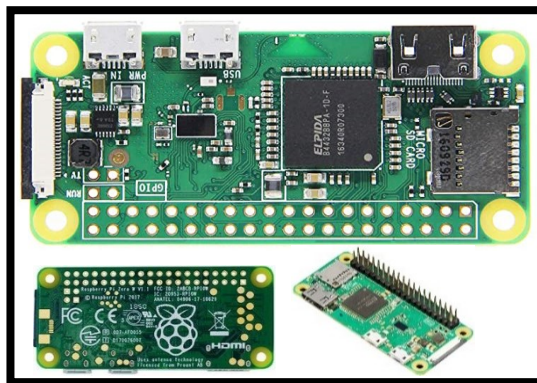


Figura 14: Módulo Raspberry Pi Zero W. Fonte: Elaborado pelo autor (2021).

A figura 15 apresenta o ambiente de desenvolvimento utilizado para a construção do firmware.

```

1  /* Copyright 2019 The TensorFlow Authors. All Rights Reserved.
2
3  Licensed under the Apache License, Version 2.0 (the "License");
4  you may not use this file except in compliance with the License.
5  You may obtain a copy of the License at
6
7  http://www.apache.org/licenses/LICENSE-2.0
8
9  Unless required by applicable law or agreed to in writing, software
10 distributed under the License is distributed on an "AS IS" BASIS,
11 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
12 See the License for the specific language governing permissions and
13 limitations under the License.
14
15 -----*/
16 #include <TensorFlowLite_ESP32.h>
17
18 #include "main_functions.h"
19
20 #include "constants.h"
21 #include "output_handler.h"
22 #include "sine_model_data.h"
23 #include "tensorflow/lite/experimental/micro/kernels/all_ops_resolver.h"
24 #include "tensorflow/lite/experimental/micro/micro_error_reporter.h"
25 #include "tensorflow/lite/experimental/micro/micro_interpreter.h"
26 #include "tensorflow/lite/schema/schema_generated.h"
27 #include "tensorflow/lite/version.h"
28
29 // Globals, used for compatibility with Arduino-style sketches.
30 namespace {
31   tflite::ErrorReporter* error_reporter = nullptr;
32   tflite::Model* model = nullptr;
33   tflite::MicroInterpreter* interpreter = nullptr;
34   TfLiteTensor* input = nullptr;
35   TfLiteTensor* output = nullptr;
36   int inference_count = 0;
37
38   // Create an area of memory to use for input, output, and intermediate arrays.

```

Figura 15: ambiente de desenvolvimento utilizado para a construção do firmware.

Fonte: Elaborado pelo autor (2021).

## 5.0 SISTEMA PROPOSTO.

Uma visão geral do sistema proposto pode ser visto na figura 16.

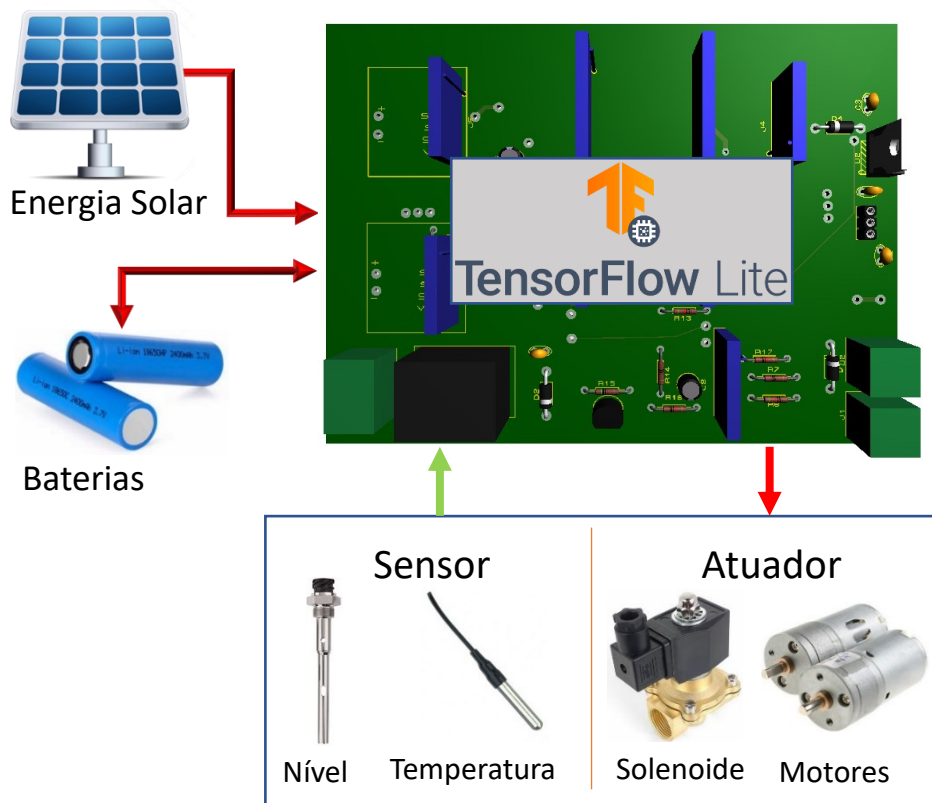


Figura 16: Visão geral dos componentes básicos do projeto. Fonte: Elaborado pelo autor (2021).

Inicialmente foi projetado o circuito eletrônico do protótipo e realizada a simulação em software para verificar o correto funcionamento do circuito, posteriormente foi projetada a placa de circuito impresso. A figura 17 apresenta o circuito eletrônico final do projeto, onde é possível visualizar a distribuição dos sinais e da alimentação do circuito, os reguladores de tensão da placa possuem designação de “5v” para os circuitos alimentados em 5 Volts e “33v” para os circuitos alimentados em 3.3V pelo regulador de tensão, as linhas com denominação “3,3v” se referem a tensão originada do módulo do ESP32.



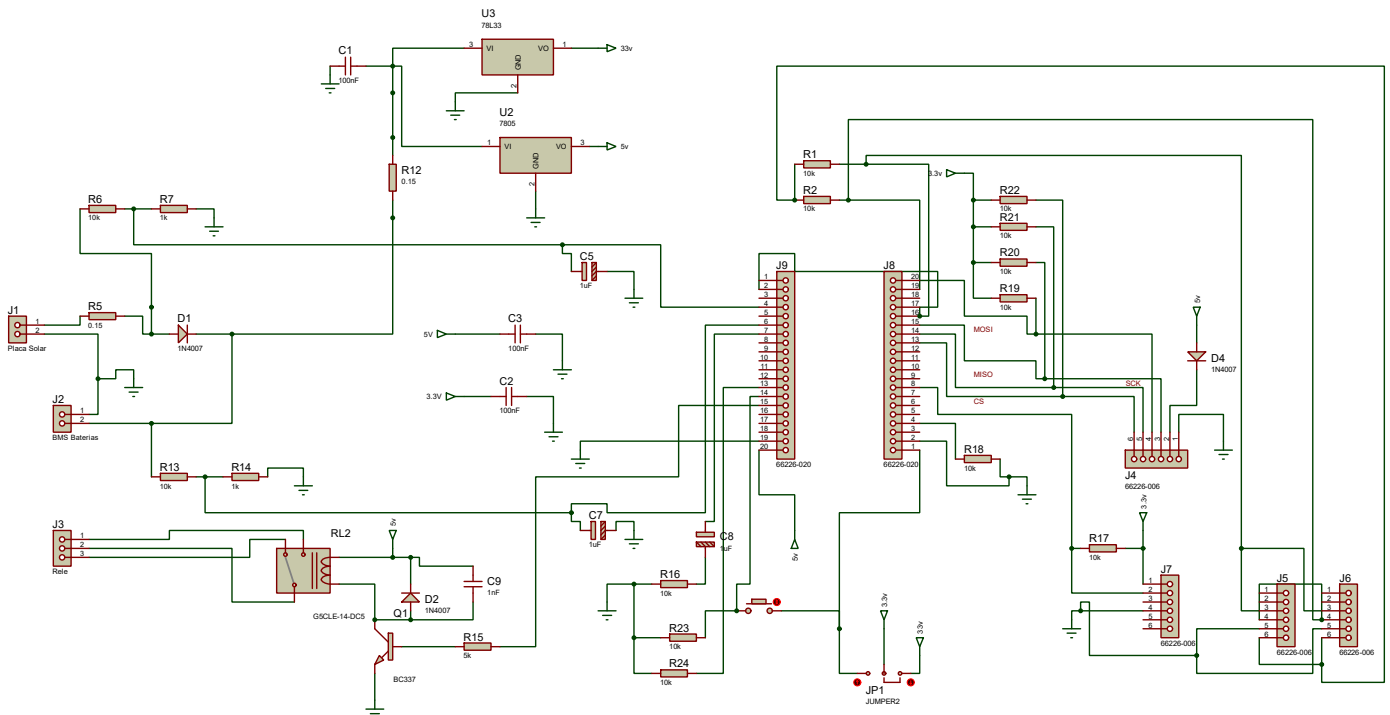


Figura 17: Diagrama do circuito elétrico do projeto. Fonte: Elaborado pelo autor (2021).

Os conectores utilizados no diagrama elétrico possuem as seguintes funções:

- J1: Conexão com a placa fotovoltaica.
- J2: Conexão com o módulo BMS que alimenta as baterias.
- J3: Conexão do relé de acionamento de cargas externas.
- J4: Conexão do módulo de cartão de memória.
- J5: Conexão do módulo de medição de corrente e tensão da placa fotovoltaica.
- J6: Conexão do módulo de medição de corrente e tensão da bateria.
- J7: Conexão do sensor de temperatura e umidade DHT-11
- J8 e J9: Conexão do módulo ESP32.

Ao todo foram desenvolvidos 3 códigos de programação para o correto funcionamento do protótipo, sendo:

1-Código em linguagem C para aquisição e levantamento da base de dados.

2-Código em linguagem Python para pré-processamento, treino e teste da rede neural LSTM.

3-Código em Python para efetuar as inferências no modelo treinado em um dispositivo embarcado de baixo consumo de energia.

Conforme apresentado no gráfico 18 o sistema possui um limite de operação dirigido pelo percentual de carga da bateria, onde no exemplo da figura o sistema encerra as operações quando a carga da bateria for menor que 75% do total, o objetivo do sistema de previsão por redes neurais é permitir que o limite de operação seja flexível conforme a previsão de energia que o sistema irá realizar no modelo LSTM.

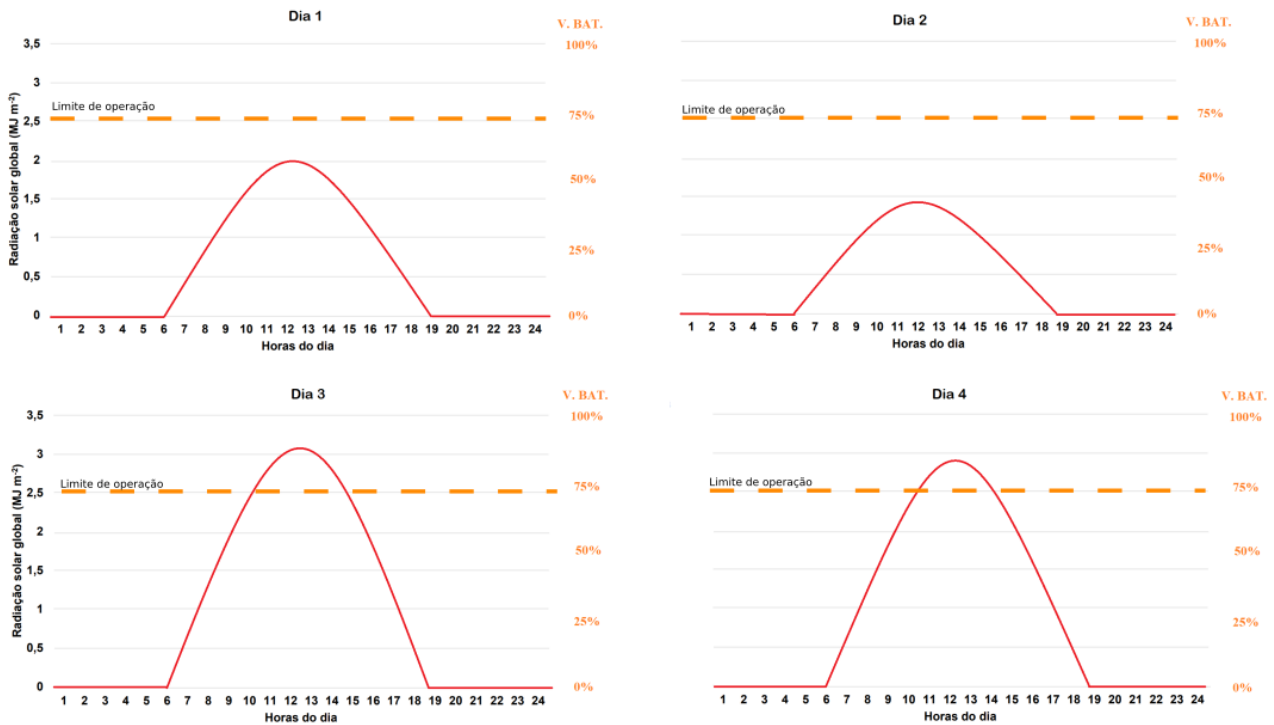


Figura 18: Limite de operação fixo de um sistema de gerenciamento de energia. Fonte: Elaborado pelo autor (2021).

## 6.0 MÉTODOS.

### 6.1 Código utilizado para criar o modelo LSTM

No quadro 1 é possível visualizar a primeira parte do código desenvolvido em linguagem Python para treinamento e teste da rede neural LSTM.

Quadro 1: Primeira parte do código com a declaração das bibliotecas e checagem das versões.

```
#Carregando dependências e bibliotecas.
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, LSTM
import tensorflow as tf
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau,
ModelCheckpoint

#Numpy, Tensorflow and Pandas teste de versão.
print(np.__version__)
print(pd.__version__)
print(tf.__version__)
```

Fonte: Elaborado pelo autor (2021).

No quadro 2 é possível verificar a segunda parte do código onde se encontra a preparação dos dados para treino da rede neural. A criação do modelo LSTM é apresentada no quadro 3 onde também se encontram as funções de callback que permitem otimizar o processo de treinamento e salvam os dados gerados durante o treino.

Quadro 3: Preparação dos dados para treino da rede neural.

```

#Carregando e preparando os dados para as camadas LSTM.
#-----//-----

d1 = pd.read_csv('dadosprevisao_POT_HUM_TEMP_12h.csv')
d1.head()

d1['P_A'] = d1['P_A'].abs()
d1['P_A'] = d1['P_A'] * 10

d1['Time'] = pd.to_datetime(d1['Time'])
df = d1.groupby(pd.Grouper(key='Time', freq='12h')).mean()
df = df.round(2)

df.to_csv(r'arquivo_downsample.csv', index = False, header=True)
d2 = pd.read_csv('arquivo_downsample.csv')

base_training_y = d2.iloc[:, 0:1].values
base_training_x = d2.iloc[:, 0:3].values

predictors = []
real_value = []
for i in range(3, 50):
    predictors.append(base_training_x[i-3:i, 0:3])
    real_value.append(base_training_x[i, 0])

predictors, real_value = np.array(predictors), np.array(real_value)

```

Fonte: Elaborado pelo autor (2021).

Quadro 2: Criação do modelo LSTM e declaração das funções de callback que permitem otimizar o processo de treinamento.

```

#Criação de camadas LSTM e configuração de parâmetros de treinamento.
#-----//-----

nn = Sequential()

nn.add(LSTM(units = 700, return_sequences = True, input_shape = (predictors.shape[1],
3)))
nn.add(Dropout(0.3))

nn.add(LSTM(units = 700, return_sequences = True))
nn.add(Dropout(0.3))

nn.add(Dense(units = 1, activation = 'relu'))

nn.compile(optimizer = 'rmsprop', loss = 'mean_squared_error',
           metrics = ['mean_absolute_error'])

nn.summary()

es = EarlyStopping(monitor = 'loss', min_delta = 1e-10, patience = 100, verbose = 1)
rlr = ReduceLROnPlateau(monitor = 'loss', factor = 0.2, patience = 5, verbose = 1)
mcp = ModelCheckpoint(filepath = 'pesos.h5', monitor = 'loss',
                      save_best_only = True, verbose = 1)
history = nn.fit(predictors, real_value, epochs = 500, batch_size = 1, callbacks =
[es, rlr, mcp])

```

Fonte: Elaborado pelo autor (2021).

No quadro 3 é possível visualizar o código de criação do modelo LSTM e a declaração das funções de callback chamadas es, rlr e mcp que oferecem maior eficiência no processo de treinamento da rede.

Para análise pós treino do modelo foi feita a plotagem do gráfico do erro e da perda que pode ser visto na figura 24 no capítulo de conclusão. O Código ainda contempla a análise do modelo pelo banco de dados de teste, onde o resultado pode ser visto na figura 25 no mesmo capítulo onde se encontra a figura 24, o código para essas funções pode ser vistos no quadro 4.

Quadro 4: Análise da perda e do erro absoluto médio e teste do modelo treinado através da base de dados de teste.

```
#Analisando a perda e o erro absoluto médio
history.history.keys()

plt.plot(history.history['loss'], label='Training loss')
plt.plot(history.history['mean_absolute_error'], label='Validation loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend()

#Executando o teste com o banco de dados de teste.
base_test = pd.read_csv('arquivo_teste.csv')
real_value_test = base_test.iloc[:, 0:1].values

frames = [d2, base_test]
full_base = pd.concat(frames)

entradas = full_base[len(full_base) - len(base_test) - 3:].values

X_teste = []
for i in range(3, 13):
    X_teste.append(entradas[i-3:i, 0:3])
X_teste = np.array(X_teste)

forecasts = nn.predict(X_teste)
forecasts2 = forecasts[:,0]

forecasts.mean()
real_value_test.mean()

plt.plot(real_value_test, color = 'red', label = 'Real Value')
plt.plot(forecasts2, color = 'blue', label = 'Predictions')
plt.title('Prediction of measurements')
plt.xlabel('Time')
plt.ylabel('Expected energy')
plt.legend()
plt.show()
```

Fonte: Elaborado pelo autor (2021).

O código para a conversão do modelo treinado para o padrão tensorflow lite pode ser visto no quadro 5, onde o modelo é salvo e posteriormente convertido para a extensão .tflite. O arquivo .tflite é utilizado pelo interpretador do TensorFlow Lite que é compatível com uma larga gama de dispositivos embarcados.

Quadro 5: Conversão do modelo keras para TensorFlow Lite.

```
# Salvando o modelo keras e convertendo o modelo para o formato tflite.
#-----
# Salva o modelo keras após compilar
nn.save('model_keras1.h5')
model_keras= tf.keras.models.load_model('model_keras1.h5')

# Converter um modelo tf.Keras em um modelo TensorFlow Lite.
converter = tf.lite.TFLiteConverter.from_keras_model(nn)
tflite_model = converter.convert()
# Salve o modelo no disco
open("lstm_tf_lite.tflite", "wb").write(tflite_model)
```

Fonte: Elaborado pelo autor (2021).

## 6.2 Firmware para levantamento da base de dados

O firmware do projeto foi desenvolvido para o módulo ESP32 com o objetivo de coletar os dados dos sensores de tensão e corrente, e temperatura e humidade. As medições foram feitas a cada períodos de um minuto e gravadas no cartão de memória incluído na placa de circuito.

Todo o código fonte foi desenvolvido em linguagem C e possui as seguintes tarefas:

- Ler e filtrar os sinais analógicos dos sensores de corrente e tensão.
- Receber o sinal do sensor de humidade relativa e temperatura.
- Registrar dados no cartão de memória.

A todo o projeto utiliza 11 pinos de entradas e saídas, sendo:

Tabela 8: Pinos de entrada e saída utilizados no ESP-32. Fonte: Elaborado pelo autor (2021).

Função	Número de pinos	Aplicação
comunicação I2C	2 Pinos	Comunicação entre os sensores de corrente e tensão.
comunicação SPI	4 Pinos	Comunicação com o cartão de memória.
Entrada Digital	1 Pino	Sinal do sensor de humidade e temperatura.

Saída Digital	1 Pino	Acionamento do LED vermelho da placa.
Saída Digital	1 Pino	Acionamento do relé da placa.
Entradas Digitais	2 Pinos	Sinais de dois botões de operação.

Fonte: Elaborado pelo autor (2021).

O ambiente utilizado para a criação do código foi o Visual Studio Code, o framework utilizado foi o framework Arduino, embora possa ser utilizado o framework ESP-IDF de autoria da própria empresa fabricante do microcontrolador. O framework Arduino foi escolhido pois já possui as bibliotecas necessárias para realizar as comunicações I2C e SPI diminuindo o tempo de desenvolvimento do firmware de coleta de dados.

No mesmo firmware do ESP32 foi programado o modo Sleep-Mode onde o ESP32 reduz a corrente de consumo enquanto não é necessário gravar os dados no cartão de memória, dessa forma a cada 1 minuto o modo Sleep-Mode é ativado para reduzir a corrente de consumo do ESP32.

A transferência do código compilado para o ESP-32 foi feita por meio de cabo USB. Embora tenha sido testada a transferência por rede Wifi, está é prejudicada pelo aumento de consumo de memória no ESP-32.

Os dados foram armazenados em um cartão tipo Micro SD de fabricante KINGMAX de 256MB em formato texto de extensão .txt. Na figura 20 é possível verificar o formato dos dados salvos no cartão de memória.

Como o objetivo do projeto é o gerenciamento de energia através da previsão de geração da placa solar fotovoltaica os atributos previsores utilizados para treino foram:

- Tensão e corrente geradas pela placa de energia solar;
- Temperatura e humidade relativa do ar;

Como a geração de energia através do sol depende de características ambientais, o sensor de temperatura e humidade oferece dados que contribuem para a maior precisão na previsão de energia disponível na placa de energia solar. Um número maior de dados de entrada podem ser utilizados para o treino do modelo, como por exemplo:

- Direção e velocidade do vento.
- Humidade do solo.

- Época do ano e estação;

Porém o objetivo do projeto não é esgotar as formas de medição de dados ambientais, mas de comprovar a eficiência do modelo LSTM na previsão de energia, dessa forma foram selecionadas as variáveis que possibilitam fácil integração com sistemas microprocessados e microcontrolados como os dados do sensor DHT11.

### 6.3 Software de gerenciamento de energia

O software para o gerenciamento de energia foi desenvolvido na linguagem python no processador BCM2835 da placa Raspberry Pi Zero W.

O software tem as seguintes funções:

- Realizar a leitura dos sensores INA219A, INA219B e do sensor DHT11.
- Registrar as medições das últimas 36 horas.
- Efetuar a inferência no modelo Tensorflow Flow Lite.
- Calcular a potência prevista para acionamento da carga.
- Supervisionar o nível de tensão da bateria.
- Supervisionar a validade das previsões do modelo.

Para simular uma carga foi adicionado ao circuito uma lâmpada de 12V 5Watts, o objetivo do software é permitir o acionamento da lâmpada por um período de tempo mais longo possível com base nas previsões de energia do modelo LSTM.

Para uma maior segurança na operação, o software de gerenciamento de energia verifica se as duas últimas previsões feitas pelo modelo LSTM possuem no mínimo 65% de assertividade, dessa forma evitando que erros de predição comprometam o nível de carga das baterias durante a operação. Além disso o software monitora a tensão das baterias e suspende todas as atividades de fornecimento de energia para a lâmpada caso a tensão mínima de 7.0Volts não esteja presente nas baterias.

O objetivo do algoritmo será transferir o máximo de carga das placas solares para o pack de baterias, e permitir a operação total do protótipo durante o dia e parcial durante a noite.

A figura 19 representa o fluxograma básico da operação do software.



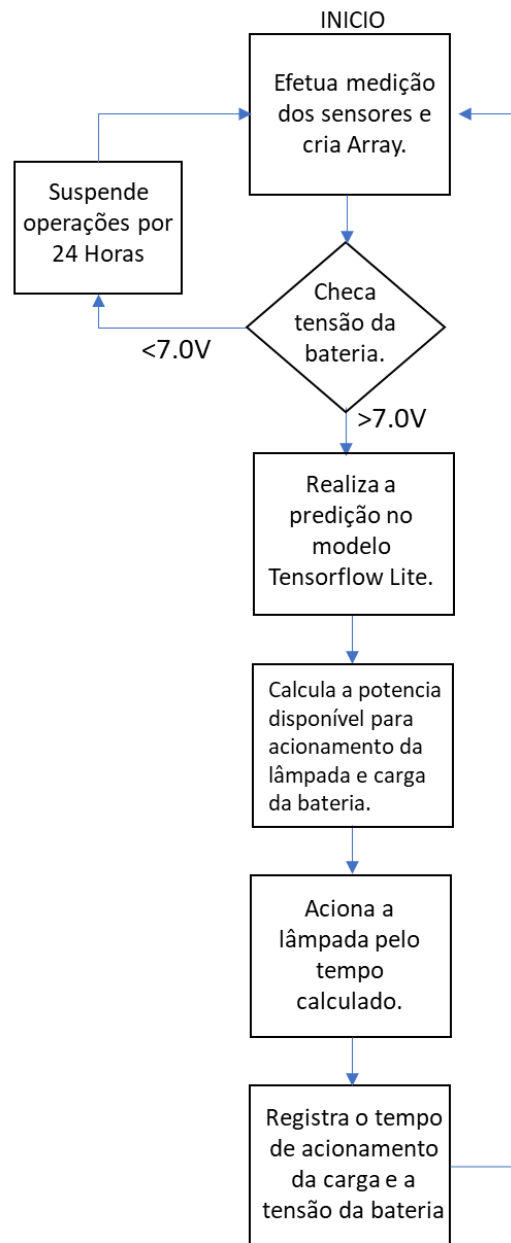


Figura 19: Fluxograma básico do software de operação das previsões. Fonte: Elaborado pelo autor (2021).

Para efeito de validação do código de gerenciamento de energia foi feito um teste do sistema durante um período de 5 dias.

Também foram feitos testes de tensão mínima da bateria onde foi possível verificar que o sistema ainda é funcional quando a bateria apresenta uma tensão mínima de 6.25 Volts, dessa forma ainda é possível obter uma pequena margem de segurança no funcionamento do sistema caso ocorra

algum erro na operação da placa, uma vez que o sistema entra em modo standby quando a tensão diminui abaixo de 7.0Volts. Antes do início dos testes as baterias foram totalmente carregadas apresentando uma tensão de 8.45Volts sem carga e diminuindo para 8.23Volts quando ligada na placa de circuito impresso e com a lâmpada no estado de apagada.

Para a execução do modelo LSTM treinado e convertido foi utilizado o processador BCM2835 da placa Raspberry Pi Zero W com sua saída de vídeo HDMI desativada. Essa configuração permite a diminuição da corrente de consumo da placa para uma corrente de 90 miliamperes. Ainda é possível realizar a diminuição no clock base da placa a fim de se obter uma maior redução no consumo de corrente, porém isso não foi feito pois o consumo da placa já era compatível com o sistema de fornecimento de energia.

Para realizar o treinamento da rede neural foi utilizado um computador desktop com as seguintes configurações:

- -Processador AMD Ryzen 1600 com 6 núcleos físicos e 12 threads operando a 3.4Ghz;
- -16Gb de memória DDR4 operando a 2600Mhz;
- -Placa de vídeo Asus GTX 1060 com 6Gb de memória;
- -Placa mãe Asus Prime Plus B350;

O TensorFlow utilizou o processador Ryzen para treinamento da rede neural LSTM.

Segue abaixo as versões de softwares e bibliotecas utilizadas.

- -Python: 3.8.11;
- -TensorFlow: 2.5.0;
- -Pandas:1.3.1;
- -Numpy: 1.19.2.

## 7.0 RESULTADOS

Os dados utilizados para o treino da rede neural foram gravados no cartão de memória no formato .txt delimitado por vírgulas, um trecho de exemplo dos dados podem ser vistos na tabela 9.

Tabela 9: Trecho dos dados armazenados no cartão de memória.

Data e hora	Temperatura (°C)	Humidade (%Umidade Relativa)	Tensão Placa Solar (Volts)	Corrente Placa Solar (Miliamperes)	Tensão da Bateria (Volts)	Corrente da Bateria (Miliamperes)
23/5/21 15:22	29.5	61	1.06	0	8.31	70.3
23/5/21 15:22	29.5	61	1.06	0	8.21	137.6
23/5/21 15:22	29.4	61	1.06	0	8.21	130.6
23/5/21 15:22	29.4	61	1.04	0	8.2	141.9
23/5/21 15:22	29.5	61	1.02	0	8.17	156.8
23/5/21 15:22	29.5	61	1	0	8.2	161.6
23/5/21 15:22	29.5	61	1.04	0	8.2	138.4
23/5/21 15:22	29.5	61	1.01	0	8.2	138.2
23/5/21 15:22	29.5	61	1.04	0	8.21	135.2
23/5/21 15:22	29.5	61	1.02	0	8.21	133.1
23/5/21 15:22	29.5	61	1.05		8.21	133.1

23/5/21 15:22	29.5	61	1.05	0	8.16	138.3
23/5/21 15:22	29.5	61	1.06	0	8.19	145.9
23/5/21 15:22	29.5	61	1.05	0	8.18	144.6

Fonte: Elaborado pelo autor (2021).

Na figura 20 é possível visualizar o gráfico dos 3 primeiros dias de aquisição de dados, onde V\_A significa a tensão da placa fotovoltaica.

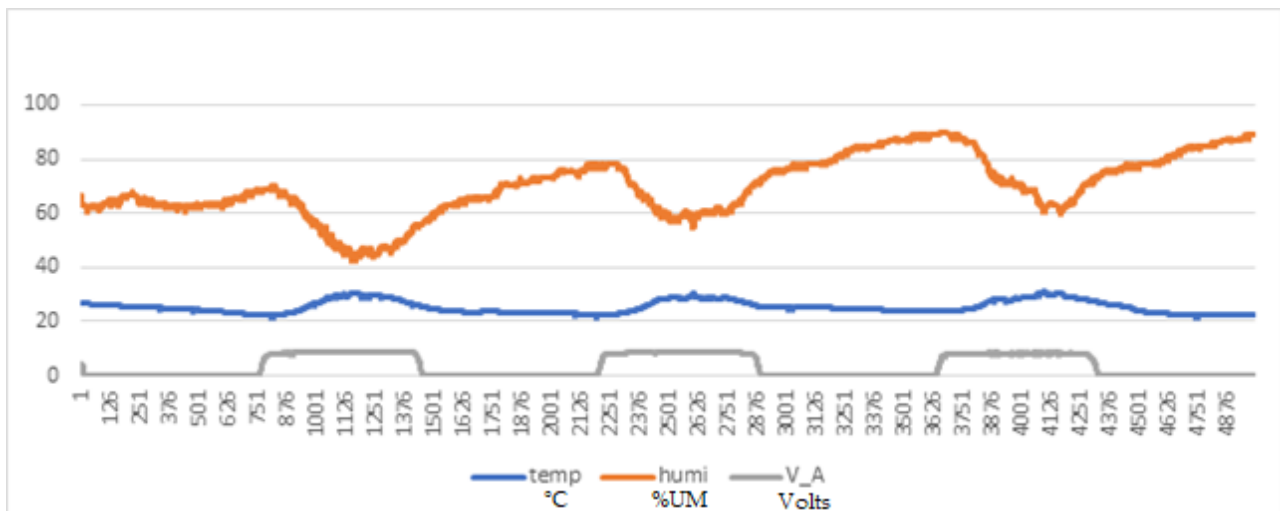


Figura 20: Gráfico com a variação da tensão da placa fotovoltaica(V\_A) juntamente com a temperatura e humidade. Fonte: Elaborado pelo autor (2021).

No próximo gráfico é possível visualizar a corrente(I\_B) em miliamperes e a tensão (V\_B) em volts nas baterias.

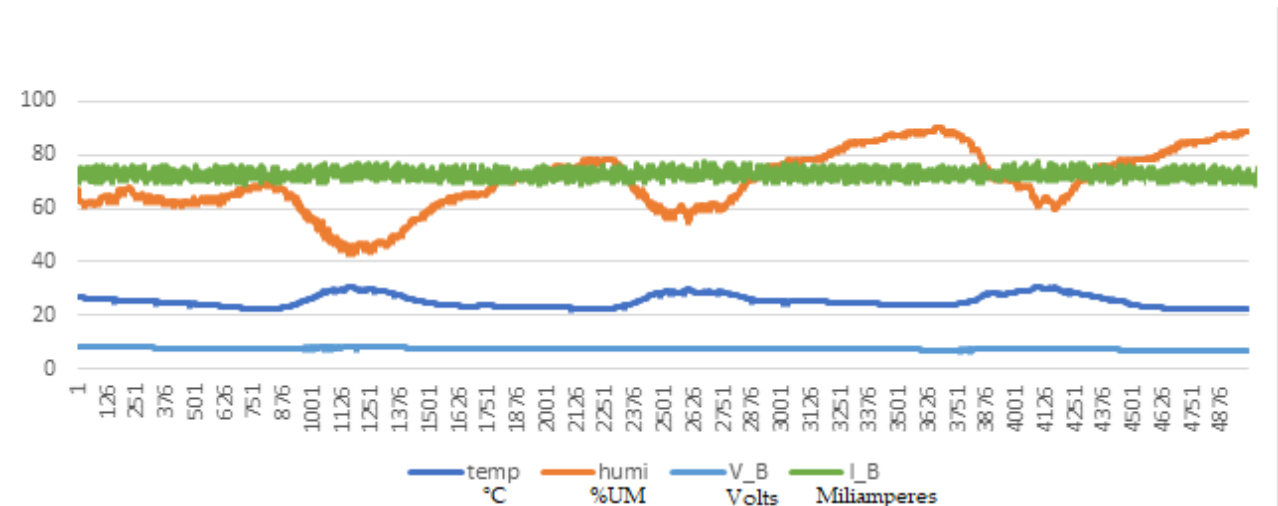


Figura 21: Gráfico com a variação da tensão e corrente da bateria juntamente com a temperatura e humidade. Fonte: Elaborado pelo autor (2021).

Em seguida é possível visualizar a corrente ( $V_I$ ) em miliamperes e a tensão ( $V_A$ ) em volts fornecidas pela placa fotovoltaica para o sistema na figura 22.

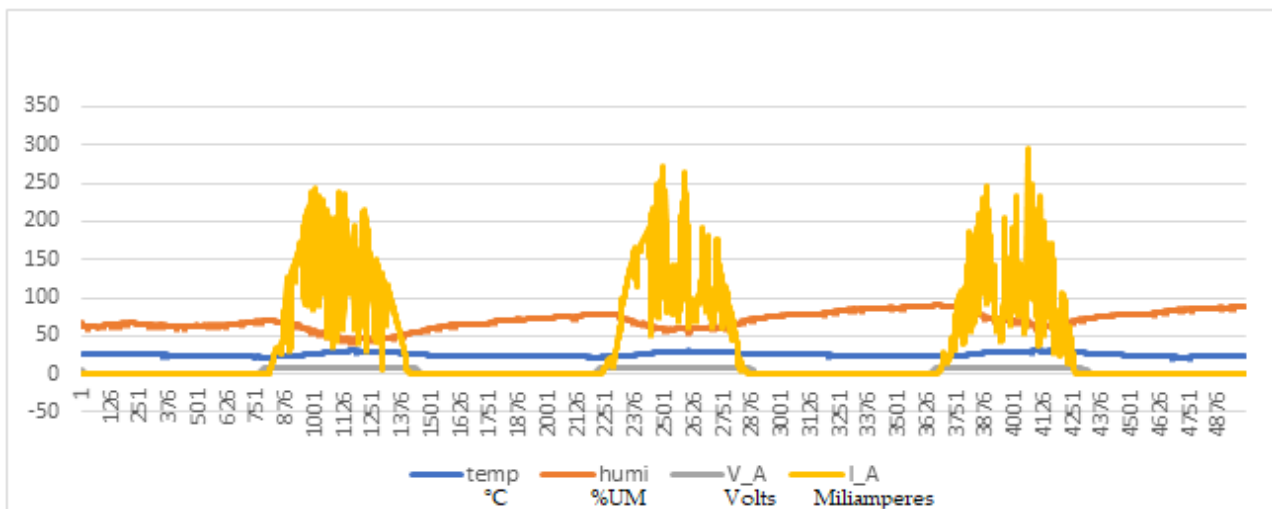


Figura 22: Gráfico com a variação da tensão e corrente da placa fotovoltaica juntamente com a temperatura e humidade. Fonte: Elaborado pelo autor (2021).

Todas as medições dos 3 gráficos acima iniciam em 31/05/2021 as 17:23:42 e finalizam em 25/05/2021 as 09:58:50. Ainda é possível verificar que a humidade reduz durante o dia e se eleva durante a noite, sendo inversamente proporcional a temperatura que reduz durante a noite e se eleva durante o dia. A placa fotovoltaica inicia a produção de energia elétrica por volta das 06:20 da manhã e encerra a produção de energia elétrica por volta das 18:00. A corrente elétrica atinge picos de até

320 miliamperes de corrente e tensão de até 9Volts. Durante o processo de coleta dados a bateria manteve uma tensão entre 7.00V e 8.7V recebendo cargas da placa fotovoltaica durante o dia.

Para testar e validar a placa de circuito a mesma passou por testes individuais em cada um de seus circuitos, a saber:

- Circuito de alimentação de 5v e 3.3v.
- Circuito de comando do relé.
- Circuito do cartão de memória.
- Circuito dos sensores INA219.
- Circuito do sensor DHT11.
- Circuito do LED 5mm vermelho.

A figura 23 apresenta a caixa com os componentes durante o processo de montagem do sistema, bem como a placa de circuito com o ESP-32 utilizado para levantamento dos dados de treino.

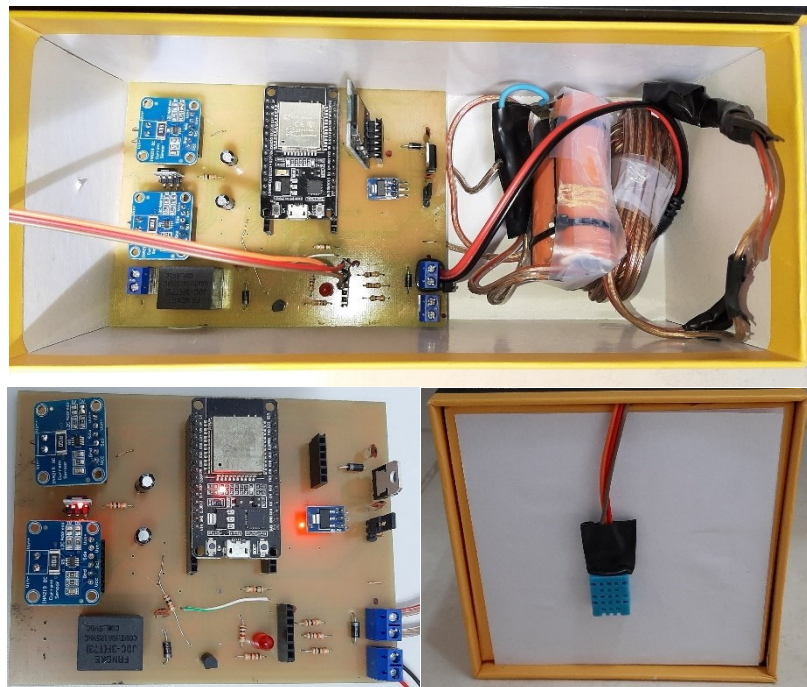


Figura 23: Foto da montagem do conjunto e sensor DHT11 localizado externamente a caixa de proteção. Fonte: Elaborado pelo autor (2021).

A figura 22 ainda mostra o sensor de temperatura e humidade localizado na parte externa da caixa.

O modelo foi criado com base em uma rede neural de duas camadas com cada camada contendo 700 células de memória LSTM. Na figura 24 é possível visualizar o modelo utilizado com maiores detalhes.

```

...: nn.summary()
Model: "sequential_1"

```

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 3, 700)	1971200
dropout (Dropout)	(None, 3, 700)	0
lstm_1 (LSTM)	(None, 3, 700)	3922800
dropout_1 (Dropout)	(None, 3, 700)	0
dense (Dense)	(None, 3, 1)	701

```

=====
Total params: 5,894,701
Trainable params: 5,894,701
Non-trainable params: 0

```

Figura 24: Resumo do modelo LSTM. Fonte: Elaborado pelo autor (2021).

Como apresentado na figura 24 o total de parâmetros do modelo foram 5,894,701. Para maior precisão do modelo treinado foram adicionadas duas camadas do tipo Dropout que permitem que o modelo não fique limitado as variáveis utilizadas durante o treino, e possa avaliar com maior precisão valores diversos oriundos do mundo real.

Afim de se obter o melhor modelo possível nos quesitos precisão e tamanho final do modelo, foram feitos diversos testes, na tabela 10 é possível visualizar as diferentes configurações utilizadas nos testes e o resultado das mesmas.

Tabela 10: Testes efetuados no modelo com 2 camadas LSTM. Fonte: Elaborado pelo autor (2021).

Unidades LSTM 1° camada	Unidades LSTM 2° camada	Mean Absolute Error(MAE)	LOSS	Otimizador
200	50	0,8539	1,4684	RMSPROP
300	50	0,8053	1,3478	
400	50	0,7452	1,2132	
500	50	0,7283	1,1948	
600	50	0,7192	1,1903	
700	50	0,7121	1,1805	
700	200	0,7116	1,1543	
700	400	0,6564	1,0189	
700	700	0,6381	0,9935	

Fonte: Elaborado pelo autor (2021).

Como apresentado na tabela 11 o aumento de camadas LSTM não ofereceu aumento na precisão do modelo.

Tabela 11: Testes efetuados no modelo 3 camadas LSTM.

Unidades LSTM 1° camada	Unidades LSTM 2° camada	Unidades LSTM 3° camada	Mean Absolute Error(MAE)	LOSS	Otimizador
700	700	700	0,7187	1,1866	RMSPROP

Fonte: Elaborado pelo autor (2021).

Após o período de treino foi possível verificar as curvas de perdas LOSS do modelo treinado, a figura 25 apresenta esse gráfico onde é possível verificar que por volta de 50 épocas de treino o modelo não obteve grandes alterações nas perdas LOSS.



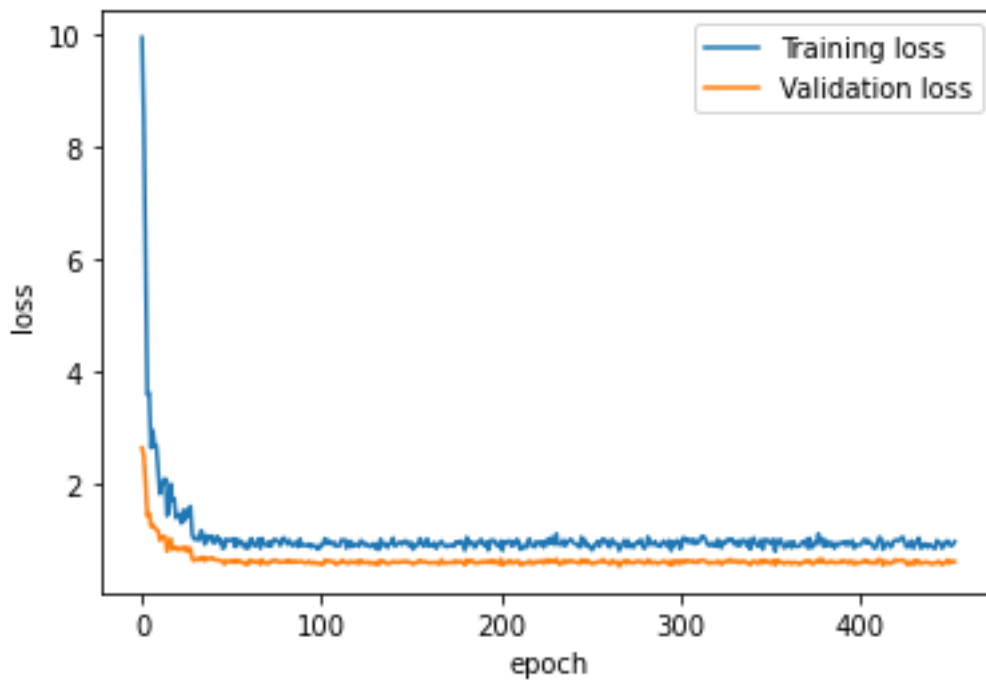


Figura 25: Gráfico de variação da perda loss. Fonte: Elaborado pelo autor (2021).

Após o treino do modelo foi efetuada a análise com a base de dados de teste, o resultado pode ser visto na figura 26. O treino do modelo deve ser o mais otimizado possível, pois ao se realizar inferências com o modelo convertido TensorFlow Lite ocorrem pequenas perdas devido as otimizações realizadas pelo conversor no modelo afim de otimizar o tamanho final do mesmo.

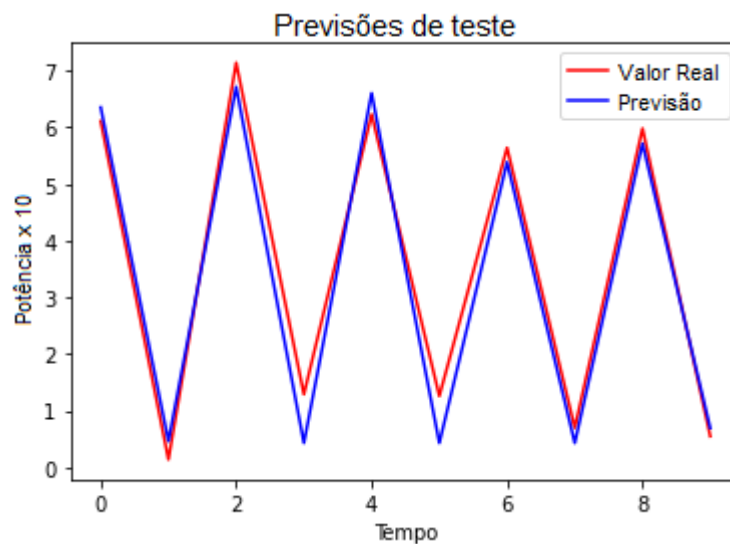


Figura 26: Gráfico de comparação entre dados previstos e dados de teste. Fonte: Elaborado pelo autor (2021).

Para o maior aproveitamento dos dados de previsão e a redução da quantidade de inferências o período de previsão foi definido em 12 horas, ou seja, o modelo irá prever o valor de potência elétrica aproximado para as próximas 12 horas após realizada a inferência. Como a operação de inferência consome energia do processador a adoção desse período possibilitou a redução do consumo, pois se comparado a inferências de menor período como minutos ou horas o consumo de energia e processamento seria maior.

Como o período a ser previsto é de 12 horas, as informações dos atributos previsores também devem estar no mesmo período de tempo, portanto foi necessário realizar o cálculo da média de cada atributo previsor em períodos de 12 horas. Segue os atributos previsores utilizados no treinamento do modelo:

- Tensão da placa de energia solar;
- Corrente da placa de energia solar;
- Temperatura do ar;
- Humidade relativa do ar;

Esses dados foram retirados do cartão de memória, onde as informações foram gravadas em períodos de um minuto durante 30 dias, portanto totalizando 35277 registros. Após adequar a base de dados em períodos de 12 horas, foi obtido um total de 48 registros. Durante os testes foi possível verificar que a quantidade de 50 registros foi o suficiente para o treinamento da rede LSTM, onde 40 registros foram utilizados para o treino do modelo e os outros 10 foram utilizados para teste.

Na figura 27 é possível visualizar o diagrama de blocos da operação de inferência com os atributos previsores.

A rede LSTM trabalha com dados de longo e curto prazo, portanto deve-se realizar a inferência fornecendo ao modelo um número valores anteriores ao momento da inferência. Neste projeto foram utilizados valores de 3 períodos de referência, como cada período possui 12 horas o total foi de 36 horas, ou seja, para que o modelo retorne o valor de potência das próximas 12 horas é necessário fornecer os valores das últimas 36 horas dos atributos previsores.

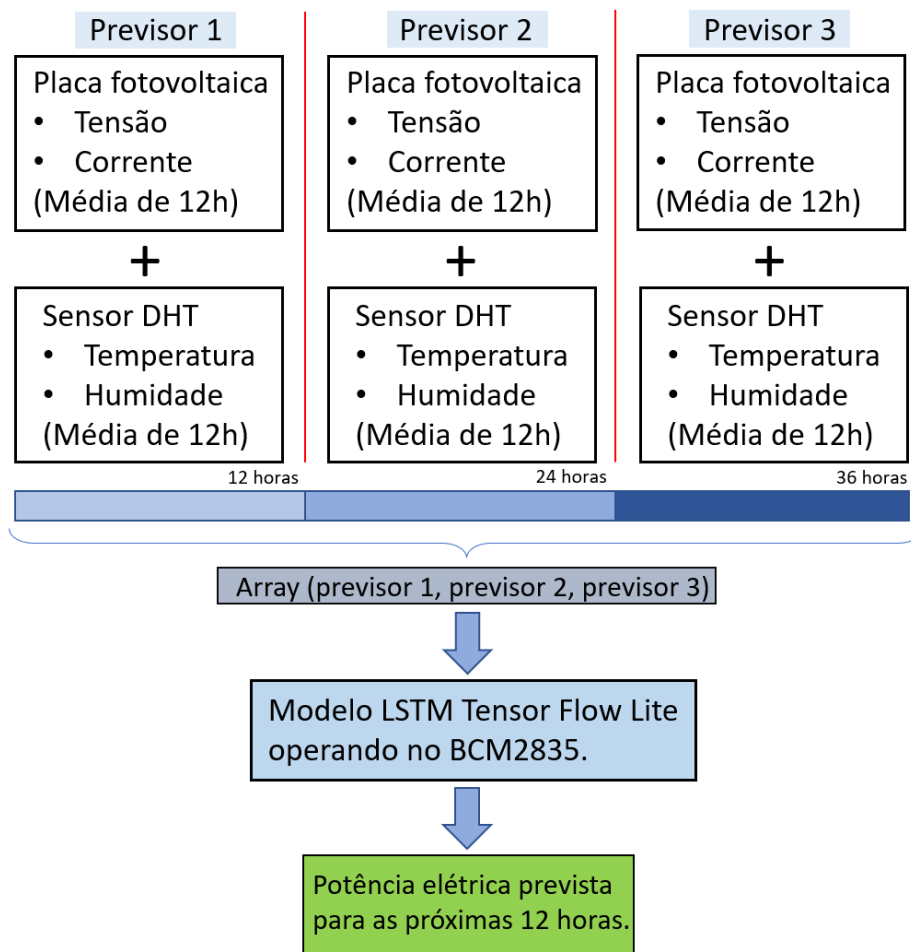


Figura 27: Diagrama de blocos dos atributos previsores. Fonte: Elaborado pelo autor (2021).

Mediante a previsão da energia fornecida pela placa fotovoltaica foi possível gerenciar a carga da bateria a fim de fornecer o máximo de energia para as cargas da placa fotovoltaica. O estado da carga da bateria foi relevante para determinar a energia disponível para períodos noturnos onde a placa de energia solar está inativa.

Conforme citado por BHATTACHARJEE, modelos de inteligência artificial também podem ser utilizados para determinar a carga da bateria. Os modelos de inteligência artificial usam funções não lineares aprendíveis para identificar de forma adaptativa o modelo da bateria a partir das variáveis observáveis da bateria, como tensão terminal, corrente de entrada, temperatura da célula, etc. e, portanto, são capazes de estimar os estados internos da bateria diretamente dos dados. O aprendizado de máquina e os algoritmos de aprendizado profundo se enquadram nessa categoria. Eles são robustos ao ruído, fornecem baixos valores de erro de estimativa e são altamente escaláveis e facilmente implantáveis. Diferentes tipos de redes neurais têm sido usados anteriormente na literatura para fins de estimativa de estado de carga da bateria. (BHATTACHARJEE et al., 2021)

A figura 28 representa as etapas que foram executadas para que o modelo do Tensor Flow Lite pudesse ser gerado com sucesso.

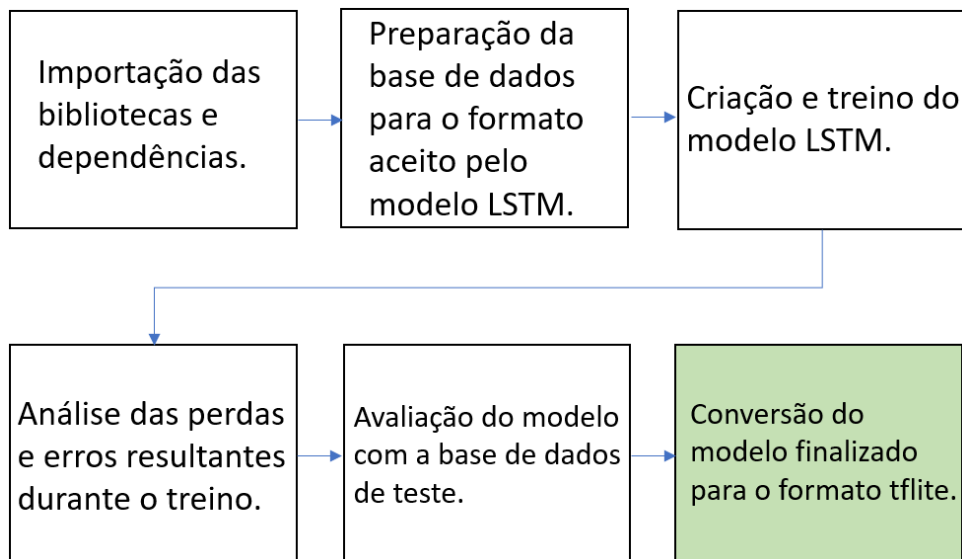


Figura 28: Etapas para criação do modelo TensorFlow Lite. Fonte: Elaborado pelo autor (2021).

Durante o teste operacional a placa necessitou de 36 horas para produzir os dados previsores, após esse período já foi possível efetuar as previsões de energia. A tabela 12 apresenta os dados obtidos no teste, bem como o erro referente a cada previsão.

Tabela 12: Resultados do teste de campo. Fonte: Elaborado pelo autor (2021).

Potência x 10	Temperatura(°C)	Humidade(%UM)	Previsão (Potência x 10)	Valor Real (Potência x 10)	Erro(%)
3.7	23.76	70.38	-	-	-
3.73	27.1	70.01	-	-	-
2.87	24.53	83.97	3.34	3.5	5%
3.5	26.79	68.3	2.8	2.96	6%
2.96	24.83	84.51	3.54	3.12	-12%
3.12	23.81	72.6	3.34	2.97	-11%

Fonte: Elaborado pelo autor (2021).

Os resultados dos testes de campo se aproximam dos testes realizados no modelo, porém com uma leve diferença devido as perdas no modelo convertido para TensorFlow Lite e a própria natureza de variação dos dados que impossibilita a assertividade de 100% das previsões. De qualquer forma os resultados foram sólidos e comprovam a eficiência das redes neurais LSTM na previsão de dados temporais com várias variáveis.

Foi realizado o teste de autonomia de carga e foi verificado que a bateria tem a capacidade de alimentar o sistema e manter a lâmpada acesa por 4 horas e 4 minutos, a lâmpada alimentada por uma tensão entre 7 e 8 Volts consome em média 245 miliamperes (figura 29) e a placa com todos os seus componentes consome cerca de 130 miliamperes, ou seja, a corrente total em plena operação da lâmpada é de 375 miliamperes.



Figura 29: Medição da corrente da lâmpada utilizada como carga. Fonte: Elaborado pelo autor (2021).

Os dados armazenados durante a operação do protótipo ainda podem ser utilizados para futuros treinamentos do modelo LSTM afim de se obter resultados mais precisos nas inferências. O TensorFlow Lite ainda permite a atualização remota do modelo por meio de diversos meios de comunicação como Wifi e Bluetooth ou até mesmo pela internet.

## 8.0 CONCLUSÃO

Sistemas de gerenciamento de energia tradicionais não possuem a capacidade de prever a energia de alimentação disponível para as próximas horas de operação e estão sujeitos ao desperdício de energia ou até mesmo a escassez de energia durante a operação. Dessa forma o presente projeto demonstra um modelo inteligente de previsão de energia elétrica baseado em redes neurais LSTM que é capaz de prever e gerenciar a energia elétrica presente e prevista em um sistema embarcado alimentado por energia solar.

Após finalizar o experimento foi possível concluir que a utilização de redes neurais recorrentes do tipo LSTM pode ser eficiente e viável em aplicações científicas e até mesmo aplicações comerciais.

O tamanho da base de dados utilizada para treino é um fator de grande relevância, sendo um dos principais pontos que afetam a precisão do sistema, embora nesse trabalho tenha se utilizado uma base de dados relativamente pequena, isso já foi suficiente para se obter resultados positivos, principalmente quando o modelo é otimizado através do número de camadas e células LSTM.

Para um futuro próximo será avaliada a funcionalidade do sistema em locais diferentes do local onde foi feita a aquisição dos dados de treino do modelo. Ainda será possível realizar novas aquisições de dados com o objetivo de aumentar o número de dados de treino.

A possibilidade de realizar inferências em um dispositivo embarcado de baixa potência abre um largo campo de aplicações na engenharia, pois permite a utilização de modelos de inteligência artificial que podem realizar inferências sem a necessidade de computadores de alto custo e alto consumo de energia como os encontrados em servidores, desktops e notebooks.

Se comparado aos métodos convencionais de inferência em redes neurais LSTM o presente projeto demonstra a eficiência do modelo reduzido e otimizado para operação em sistemas embarcados através do TensorFlow Lite, dessa forma ampliando as aplicações de redes neurais em equipamentos alimentados a bateria ou que possuem limitações de fornecimento de energia.

## 9.0 REFERÊNCIAS

- KIM, E. et al. Power Guarantee for Electric Systems Using Real-Time Scheduling. **IEEE Transactions on Parallel and Distributed Systems**, v. 31, n. 8, p. 1783–1798, 1 ago. 2020a.
- KIM, S. J. et al. **State of Health Estimation of Li-Ion Batteries Using Multi-Input LSTM with Optimal Sequence Length**. 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE). **Anais...** In: 2020 IEEE 29TH INTERNATIONAL SYMPOSIUM ON INDUSTRIAL ELECTRONICS (ISIE). Delft, Netherlands: IEEE, jun. 2020cDisponível em: <<https://ieeexplore.ieee.org/document/9152544/>>. Acesso em: 13 mar. 2021
- KARIMI, M. et al. **Real-Time Task Scheduling on Intermittently-Powered Batteryless Devices**. IEEE Internet of Things Journal, p. 1–1, 2021.
- ZHANG, R.; LIU, F.; DING, Y. **An Energy Optimization-Based Fast Rerouting Method for Micro-nano Satellite Formation**. 2019 IEEE 2nd International Conference on Electronics Technology (ICET). **Anais...** In: 2019 IEEE 2ND INTERNATIONAL CONFERENCE ON ELECTRONICS TECHNOLOGY (ICET). Chengdu, China: IEEE, maio 2019aDisponível em: <<https://ieeexplore.ieee.org/document/8839498/>>. Acesso em: 13 mar. 2021
- LI, C.; XIAO, F.; FAN, Y. **An Approach to State of Charge Estimation of Lithium-Ion Batteries Based on Recurrent Neural Networks with Gated Recurrent Unit**. **Energies**, v. 12, n. 9, p. 1592, 26 abr. 2019.
- ALSHARIF, M. H.; KIM, S.; KURUOĞLU, N. **Energy Harvesting Techniques for Wireless Sensor Networks/Radio-Frequency Identification: A Review**. **Symmetry**, v. 11, n. 7, p. 865, 3 jul. 2019.
- BHATTACHARJEE, A. et al. **Estimating State of Charge for xEV batteries using 1D Convolutional Neural Networks and Transfer Learning**. IEEE Transactions on Vehicular Technology, p. 1–1, 2021.
- KAIJU, L. et al. **Short-term photovoltaic power prediction based on T-S fuzzy neural network**. 2018 33rd Youth Academic Annual Conference of Chinese Association of Automation (YAC). **Anais...** In: 2018 33RD YOUTH ACADEMIC ANNUAL CONFERENCE OF CHINESE ASSOCIATION OF AUTOMATION (YAC). Nanjing, China: IEEE, maio 2018Disponível em: <<https://ieeexplore.ieee.org/document/8406448/>>. Acesso em: 13 mar. 2021.

NEWELL, D.; DUFFY, M. **Review of Power Conversion and Energy Management for Low-Power, Low-Voltage Energy Harvesting Powered Wireless Sensors**. IEEE Transactions on Power Electronics, v. 34, n. 10, p. 9794–9805, out. 2019.

THAKURTA, V. et al. **Design and Implementation of Power Management Algorithm for a Nano-Satellite**. 2019 IEEE Aerospace Conference. **Anais...** In: 2019 IEEE AEROSPACE CONFERENCE. Big Sky, MT, USA: IEEE, mar. 2019Disponível em: <<https://ieeexplore.ieee.org/document/8741556/>>. Acesso em: 13 mar. 2021.

PARK, K. et al. **LSTM-Based Battery Remaining Useful Life Prediction With Multi-Channel Charging Profiles**. IEEE Access, v. 8, p. 20786–20798, 2020.

FILIOS, G. et al. **A smart energy management power supply unit for low-power IoT systems**. 2020 16th International Conference on Distributed Computing in Sensor Systems (DCOSS). **Anais...** In: 2020 16TH INTERNATIONAL CONFERENCE ON DISTRIBUTED COMPUTING IN SENSOR SYSTEMS (DCOSS). Marina del Rey, CA, USA: IEEE, maio 2020. Disponível em: <<https://ieeexplore.ieee.org/document/9183533/>>. Acesso em: 21 ago. 2021.

PAMULA, V. R.; VAN HOOFF, C.; VERHELST, M. **Analog-and-Algorithm-Assisted Ultra-low Power Biosignal Acquisition Systems**. Cham: Springer International Publishing, 2019.

TEKESTE HABTE, T. et al. **Ultra Low Power ECG Processing System for IoT Devices**. 1st ed. 2019 ed. Cham: Springer International Publishing : Imprint: Springer, 2019.

SIU, C.; INIEWSKI, K. **IoT and low-power wireless: circuits, architectures, and techniques**. Boca Raton, FL: CRC Press/Taylor & Francis Group, 2018.

LIN, L. et al. Power Management in Low-Power MCUs for Energy IoT Applications. **Journal of Sensors**, v. 2020, p. 1–12, 14 dez. 2020.

IHKUBO, A.; LO, S.-C. **Design of Power Saving Schemes for the IoT**. 2019 International Conference on Information Networking (ICOIN). **Anais...** In: 2019 INTERNATIONAL CONFERENCE ON INFORMATION NETWORKING (ICOIN). Kuala Lumpur, Malaysia: IEEE, jan. 2019. Disponível em: <<https://ieeexplore.ieee.org/document/8718179/>>. Acesso em: 21 ago. 2021



AKERKAR, RAJENDRA. **Artificial intelligence for business**. 1st. edition ed. New York, NY: Springer Science+Business Media, 2018.

JOSHI, N.; SAFARI, AN O. M. C. **Hands-On Artificial Intelligence with Java for Beginners**. Birmingham, Inglaterra. Packt Publishing, 2018.

CHOWDHARY, K. R. **FUNDAMENTALS OF ARTIFICIAL INTELLIGENCE**. Place of publication not identified: SPRINGER, INDIA, PRIVATE, 2020.

PAREDES LARROCA, F. et al. **Development of a Fuzzy Logic-Based Solar Charge Controller for Charging Lead–Acid Batteries**. In: NUMMENMAA, J. et al. (Eds.). . *Advances and Applications in Computer Science, Electronics and Industrial Engineering. Advances in Intelligent Systems and Computing*. Cham: Springer International Publishing, 2020. v. 1078p. 168–183.

BELOVA, I. A. et al. **Optimization of Artificial Neural Network Learning for Maximum Power Point Tracking After the Degradation of the Solar Battery**. 2019 20th International Conference of Young Specialists on Micro/Nanotechnologies and Electron Devices (EDM). Anais... In: 2019 20TH INTERNATIONAL CONFERENCE OF YOUNG SPECIALISTS ON MICRO/NANOTECHNOLOGIES AND ELECTRON DEVICES (EDM). jun. 2019

KHUMPROM, P.; YODO, N. **Data-driven Prognostic Model of Li-ion Battery with Deep Learning Algorithm**. 2019 Annual Reliability and Maintainability Symposium (RAMS). Anais... In: 2019 ANNUAL RELIABILITY AND MAINTAINABILITY SYMPOSIUM (RAMS). jan. 2019.

VEERARAGHAVAN, A. et al. **Battery aging estimation with deep learning**. 2017 IEEE Transportation Electrification Conference (ITEC-India). Anais... In: 2017 IEEE TRANSPORTATION ELECTRIFICATION CONFERENCE (ITEC-INDIA). dez. 2017.

SILVA, E. L.; MENEZES, E. M. **Metodologia da Pesquisa e Elaboração de Dissertação**. 4. ed. Florianópolis: UFSC, 2005.

SPARKFUN. **µA7800 SERIES POSITIVE-VOLTAGE REGULATORS**. Disponível em: <<https://www.sparkfun.com/datasheets/Components/LM7805.pdf>>. Acesso em: 09 out. 2021.

ISTIAKE SUNNY, MD. A.; MASWOOD, M. M. S.; ALHARBI, A. G. **Deep Learning-Based Stock Price Prediction Using LSTM and Bi-Directional LSTM Model**. 2020 2nd Novel

Intelligent and Leading Emerging Sciences Conference (NILES). **Anais...** In: 2020 2ND NOVEL INTELLIGENT AND LEADING EMERGING SCIENCES CONFERENCE (NILES). out. 2020.

WAN, R. et al. Multivariate Temporal Convolutional Network: A Deep Neural Networks Approach for Multivariate Time Series Forecasting. **Electronics**, v. 8, n. 8, p. 876, 7 ago. 2019.

SAGHEER, A.; KOTB, M. Time series forecasting of petroleum production using deep LSTM recurrent networks. **Neurocomputing**, v. 323, p. 203–213, jan. 2019.

ALHIRMIZY, S.; QADER, B. **Multivariate Time Series Forecasting with LSTM for Madrid, Spain pollution**. 2019 International Conference on Computing and Information Science and Technology and Their Applications (ICCISTA). **Anais...** In: 2019 INTERNATIONAL CONFERENCE ON COMPUTING AND INFORMATION SCIENCE AND TECHNOLOGY AND THEIR APPLICATIONS (ICCISTA). Kirkuk, Iraq: IEEE, mar. 2019. Disponível em: <<https://ieeexplore.ieee.org/document/8830667/>>. Acesso em: 21 ago. 2021

WARDEN, P.; SITUNAYAKE, D. **TinyML: machine learning with TensorFlow Lite on Arduino and ultra-low-power microcontrollers**. [s.l.: s.n.].

LI, S. et al. **Machine learning algorithm based battery modeling and management method: A Cyber-Physical System perspective**. 2019 3rd Conference on Vehicle Control and Intelligence (CVCI). **Anais...** In: 2019 3RD CONFERENCE ON VEHICLE CONTROL AND INTELLIGENCE (CVCI). set. 2019.

SIAMI-NAMINI, S.; TAVAKOLI, N.; SIAMI NAMIN, A. **A Comparison of ARIMA and LSTM in Forecasting Time Series**. 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA). **Anais...** In: 2018 17TH IEEE INTERNATIONAL CONFERENCE ON MACHINE LEARNING AND APPLICATIONS (ICMLA). Orlando, FL: IEEE, dez. 2018. Disponível em: <<https://ieeexplore.ieee.org/document/8614252/>>. Acesso em: 21 ago. 2021