UNIVERSIDADE DO ESTADO DO AMAZONAS - UEA

ESCOLA SUPERIOR DE TECNOLOGIA - EST

LENNON BRANDÃO FREITAS DO NASCIMENTO

DESENVOLVIMENTO DE SOLUÇÃO DE INTELIGÊNCIA ARTIFICIAL APLICADA À IMPLEMENTAÇÃO DE *SMART BUILDINGS* COM BASE NO *FRAMEWORK* SMARTLVGRID

Manaus - Amazonas

2019

LENNON BRANDÃO FREITAS DO NASCIMENTO

DESENVOLVIMENTO DE SOLUÇÃO DE INTELIGÊNCIA ARTIFICIAL APLICADA À IMPLEMENTAÇÃO DE *SMART BUILDINGS* COM BASE NO *FRAMEWORK* SMARTLVGRID

Projeto de pesquisa desenvolvido durante a disciplina de Trabalho de Conclusão de Curso II, apresentado à banca avaliadora do curso de Engenharia Elétrica da Escola Superior de Tecnologia da Universidade do Estado do Amazonas, como pré-requisito para a obtenção do título de Bacharel em Engenharia Elétrica.

Orientador: Prof. Dsc. Raimundo Cláudio Souza Gomes

Manaus - Amazonas

2019

CIP – Catalogação na Publicação

Do Nascimento, Lennon Brandão Freitas

Desenvolvimento de solução de inteligência artificial aplicada a implementação de *smart buildings* com base no *framework SmartLVGrid* / Lennon Brandão Freitas do Nascimento; [orientado por] Raimundo Cláudio Souza Gomes. – Manaus: 2019.
44 p.: il.

Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica). Universidade do Estado do Amazonas, 2019.

1. Sistemas Inteligentes 2. *Smart Grids* 3. *Smart Buildings* 4. *Smart Lighting*
I. Cláudio Souza Gomes, Raimundo.

LENNON BRANDÃO FREITAS DO NASCIMENTO

DESENVOLVIMENTO DE SOLUÇÃO DE INTELIGÊNCIA ARTIFICIAL APLICADA A IMPLEMENTAÇÃO DE SMART BUILDINGS COM BASE NO FRAMEWORK SMARTLVGRID

Pesquisa desenvolvida durante a disciplina de Trabalho de Conclusão de Curso II e apresentada à banca avaliadora do Curso de Engenharia Elétrica da Escola Superior de Tecnologia da Universidade do Estado do Amazonas, como pré-requisito para a obtenção do título de Engenheiro Eletricista.

Nota obtida: _____ (_____)

Aprovada em _____/_____/_____.

Área de concentração: Sistemas Inteligentes

BANCA EXAMINADORA

_____

Orientador: Raimundo Cláudio Souza Gomes, DSc.

_____

Avaliador: Carlos Maurício Serodio Figueiredo, DSc.

_____

Avaliador: Fábio de Sousa Cardoso, DSc.

Manaus, 2019

Este trabalho é dedicado à minha família que não mediu esforços até aqui e tornou essa construção possível.

AGRADECIMENTOS

A minha maior gratidão é doada a Deus que sempre me supriu com discernimento e entendimento nos momentos de decisão e dia após dia me agracia com sentido para vida.

Agradeço aos meus pais Sebastião Rodrigues do Nascimento e Maristela Brandão Freitas do Nascimento e aos meus irmãos Valdemar Victor de Oliveira Brandão Freitas e Leonardo Brandão Freitas do Nascimento pelo apoio apoio incondicional e constante nos momentos de aprendizados e conquista.

A todos os meus familiares que de forma direta e indireta foram suporte, em especial minha companheira de caminhada pela compreensão e apoio que foram fundamentais para que este trabalho fosse realizado. Grande gratidão.

Agradeço aos professores que foram incentivo para o meu início na caminhada científica. Em especial ao Professor Raimundo Cláudio Souza Gomes que desde o início da construção deste trabalho, mostrou-se solícito e sempre disposto em compartilhar conhecimento e toda sua notória experiência.

Aos meus amigos do Laboratório de Sistemas Embarcados - LSE que foram imprescindíveis para minha construção profissional. Particularmente aos meus amigos Rubens de Andrade Fernandes e Samuel Bruno Torres Rego que foram primordiais no meu desenvolvimento além de fornecerem uma relação de cooperação notável. Grato a vocês.

"Finis vitae cum virtute degendae est,
ut quis Numini Divino assimiletur."
- "O objetivo da vida virutosa é
tornar-se semelhante a Deus."
São Gregório de Nissa

RESUMO

Neste trabalho, foi realizado o desenvolvimento de algoritmos de *firmware* e *hardware* além de uma aplicação de inteligência artificial para aprendizado do padrão do usuário no que diz respeito a nível de luminosidade desejado. As implementações acima tinham como elemento norteador o *framework SmartLVGrid* que consiste em um meta modelo que realiza a convergência dos circuitos de baixa tensão passivos em uma *Smart Grid*. O modelo sistemático foi utilizado para a convergência ao paradigma *Smart Building*. Conceitos inerentes ao *framework* foram utilizados como comunicação sistemática entre elementos que compõe o sistema bem como o *retrofit*. A adaptação do *framework* para convergência *Smart Building* será feita no Laboratório de Sistemas Embarcados localizado no HUB inovação e tecnologia que, por sua vez, situa-se na Escola Superior de Tecnologia. O *firmware* perpassa pela implementação da conexão em rede utilizando protocolo MQTT indicado para aplicações do gênero, além de favorecer a aquisição de dados do ambiente através de implementação de plataforma de sensoriamento que foram simulados. Por fim, o modelo de inteligência artificial colherá as informações disponibilizadas do ambiente e do usuário para realizar o aprendizado de padrão e tornar a regulação de luminosidade de forma inteligente e autônoma gerando *Smart Lighting*. Após a implementação e coleta do resultado, almeja-se alcançar a convergência *Smart Building* através do *SmartLVGrid* com desenvolvimento de inteligência artificial.

Palavras-chave:Inteligência Artificial; Aprendizado de padrão; SmartLVGrid; retrofit; Smart Building; Smart Lighting.

ABSTRACT

In this document, the development of firmware and hardware algorithms was performed, as well as an artificial intelligence application to learn the user's standard regarding the desired light level. The above implementations had as their guiding element the SmartLVGrid framework which consists of a meta model that converges passive low voltage circuits into a Smart Grid. The systematic model was used for the convergence to the Smart Building paradigm. Concepts inherent to the framework were used as systematic communication between elements that make up the system as well as the *retrofit*. The adaptation of the framework for convergence Smart Building will be made at the Embedded Systems Laboratory located at HUB innovation and technology, which, in turn, is located at the School of Technology. The firmware goes through the implementation of the network connection using MQTT protocol indicated for such applications, besides favoring the acquisition of environment data through the implementation of sensing platform. Finally, the artificial intelligence model will gather the available information from the environment and the user to learn and make the brightness regulation intelligent and autonomous. After the implementation and collection of the result, the goal is to achieve convergence Smart Building through SmartLVGrid with development of artificial intelligence.

Keywords: Artificial intelligence; Pattern Learning; SmartLVGrid; retrofit; Smart Building; Smart Lighting.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ACU	*Automation and Control Unit*
A/D	*Analog-to-Digital*
BI	*Business Inteligence*
CI	Circuito integrado
CIN	*Coupling and Interaction Nodes*
CL	*Central Level*
CSFs	*Computational Support Functions*
D/A	*Digital-to-Analog*
DC	*Direct Current*
DRFs	*Domain Retrofitting Functions*
EEPROM	*Electrically Erasable Programmable Read-Only Memory*
GPIO	*General Purpose Input-Output*
HVAC	*Heating,Ventilation, Air Conditioning*
IDE	*Integrated Development Environment*
IoT	*Internet of Things*
IP	*Internet Protocol*
ISFs	*Inter-Domain Support Functions*
JSON	*JavaScript Object Notation*
LL	*Local Level*
LAN	*Local Area Network*
LANI	*Local Area Network Interface*
LED	*Light-emitting Diode*
LSE	Laboratório de Sistemas Embarcados
MAN	*Metropolitan Area Network*

MANI	*Metropolitan Area Network Interface*
MQTT	*Message Queue Telemetry Transport*
OPs	*Operational Primitives*
PoI	*Points of Interface*
RAM	*Random Access Memory*
SCC	*Supervisor and Control Center*
SN	*Service Nodes*
SMD	*Surface Mounting Device*
SPI	*Serial Peripheral Interface*
SSID	*Service Set Identifier*
UART	*Universal Asynchronous Receiver/Transmitter*
USB	*Universal Serial Bus*
WAN	*Wide Area Network*

SUMÁRIO

INTRODUÇÃO

A sociedade atual passa por constantes avanços tecnológico. Nesse contexto, as inovações tecnológicas e as informações por elas disponibilizadas tornaram-se elementos inerentes à vida da maior parte da população mundial. Essa realidade de contínua troca de dados se tornou possível a partir da representação do mundo físico na forma digital. Esse fato culminou na necessidade da interconexão dos diferentes dispositivos geradores de dados realizando uma transformação no modo de vida das pessoas nos dias de hoje.

Nesse contexto, ressalta-se um dos maiores sistemas implantados pelo homem: O sistema de energia elétrica. A contar de sua concepção,esse sistema manteve-se praticamente sem alteração em seu princípio de funcionamento. Todavia, a revolução tecnológica supracitada já causa efeitos sobre esse sistema. Um exemplo disso é o paradigma denominado *Smart Grid* ou Redes Elétricas Inteligentes que se mostra como uma evolução do sistema existente. A necessidade da convergência ao paradigma *Smart Grid* se deve ao fato da passividade do sistema atual que não possui maneiras de atuação sobre si. Através disso, é possível agregar recursos tais como automação, processamento computacional distribuído, comunicação e controle eletrônico que possibilitará um aumento do desempenho do sistema.

Entretanto grande parte dos sistemas existentes atualmente não possui conexão com a internet. Mais da metade deles não se comunicam e não compartilham os dados entre si ou com as tecnologias de armazenamento em nuvem (AL-FUQAHA et al., 2015). A tecnologia que favoreceria a intercomunicação entre esses sistemas seria a Internet das Coisas largamente conhecida como Internet of Things (IoT).

A IoT é um paradigma de comunicação em evidência atualmente. Ela adquiriu relevância na indústria e na comunidade acadêmica por ser um dos pilares para quarta revolução industrial. (PERERA et al., 2014). O referido paradigma consiste na conexão de objetos comuns à internet. Esses objetos possuem transceptores e protocolos adequados que possibilitam a comunicação digital e intercomunicação entre eles. A partir dessa comunicação entre dispositivos, a IoT proporciona desenvolvimento de aplicações com diferentes finalidades que utilizem quantidades significativas de dados geradas por esses dispositivos para assim disponibilizar aos usuários novos serviços. (A et al., 2017)

A evolução da IoT tem disponibilizado o acesso e o controle de informações através das redes de comunicação presentes nas casas e ambientes corporativos. Esses ambientes, por sua vez, quando dotados de dispositivos que realizam conexões através de uma rede, tornam-se ambientes inteligentes capazes de se adaptar e interagir de acordo com as necessidades do usuário. Estas são as chamadas *Smart Technologies* (KAZAK; BUCHATSKIY, 2018).

O processo de convergência visa a realização da transformação dos sistema com o intuito de avanço tecnológico que muitas vezes ocorre de forma intencional ou não. Um exemplo disso é a convergência digital. Essa convergência iniciou a digitalização do mundo físico trazendo consigo novos paradigmas. O paradigma *Smart* é um dos desdobramentos da digitalização.

Sem um referencial futuro e definição de tempo para ciclo de mudança, a convergência digital se torna um fenômeno típico tecnológico e ainda propaga esse traço para as tecnologias que utilizam dos seus conceitos. Contudo, é viável estipular processos de convergências intencionais para implementação dessas tecnologias. A finalidade da definição desses processos é para que haja uma transição de menor custo relativo e impactos em estruturas pré-existentes, denominadas nesse trabalho de legado, na implementação dessas tecnologias.

A técnica de *retrofit* consiste na transformação de dispositivos que não possuem capacidade de comunicação e que são obsoletos em elementos capazes de se conectar em rede e disponibilizar novos serviços aos usuários. Essa técnica é usada no processo de convergência em *Smart Buildings*. Todavia existem poucos desenvolvimentos práticos que dão alicerce ao paradigma *Smart*.

O *framework* SmartLVGrid (*Smart Low Voltage Grid*) é um modelo de referência aplicado à convergência *Smart Grid* de sistemas legados de distribuição de energia elétrica em baixa tensão. Combinando o conceito de *retrofit* à estratégias de engenharia de sistemas, esse *framework* possibilita uma transição de baixo custo de uma planta passiva a uma rede automatizada, visto que adiciona ao sistema legado recursos de processamento, controle e comunicação sem requerer para isso a substituição de toda ou parte da infraestrutura atual.

Posto isso, o conceito de *Smart Buildings* será trabalhado com enfoque restrito ao desenvolvimento da camada de aplicação do *framework* SmartLVGrid, utilizando inteligência artificial aplicada ao controle e operação de uma planta dotada do *retrofit* de luminárias LED, tal que resulte em um sistema inteligente de iluminação conforme fundamentado no *framework* referenciado.

TEMA

Desenvolvimento de solução de inteligência artificial aplicada a implementação de *Smart Buildings* com base no *framework* SmartLVGrid.

PROBLEMA DE PESQUISA

Atualmente os prédios possuem em suas instalações dispositivos que podem ser denominados passivos, ou seja, possuem uma planta incapaz de manter comunicação em rede, atuar de forma inteligente e oferecer serviços aos seus usuários. Um dos motivos dessa ocorrência se deve ao período de transição de tecnologias onde a convergência digital é protagonista do início dos sistemas inteligentes e autônomos.

A convergência *Smart Buildings*, conforme citado anteriormente, possui poucas implementações no aspecto prático. Essa escassez de soluções ocasiona uma inacessibilidade às inovações. Em se tratando de iluminação, existe o conceito da iluminação inteligente, ou do

inglês *smart lighting*. Além disso, existem também grandes desafios para IoT nessa área, visto a quantidade limitada de tecnologias desenvolvidas. Ademais, o custo-benefício das soluções existentes se torna uma barreira para transição de uma iluminação passiva para iluminação controlável e comunicável.

Dessa forma, esses sistemas que ainda não são dotados de recursos de comunicação e atuação levam ao desperdício de informações geradas por luminárias além de não possibilitar a convergência *Smart Building*. Assim sendo, há uma necessidade da coleta e utilização direcionada dos dados disponibilizados pelas plantas de iluminação para dispor ao usuário um sistema autônomo e inteligente.

HIPÓTESE

Ao empregar o *framework* SmartLVGrid, adaptado para convergência *Smart Building*, é possível converter os dispositivos passivos inseridos nas instalações prediais em um sistema dotado de recursos de inteligência artificial sem alterar as características do sistema existente, o sistema legado. A finalidade de não alteração das características do sistema existente é a transição com melhor custo-benefício. Além disso, é viável adaptar um dispositivo que não dispõe de comunicação e controle em um elemento com essas características e que atuam de forma autônoma e inteligente de acordo com os padrões do usuário. Isso possibilitará a inserção desses dispositivos em uma rede *IoT* dotada de inteligência artificial e ensejará o aproveitamento das informações geradas pelas luminárias.

JUSTIFICATIVA

Ao estudar o *framework* SmartLVGrid, é notório que os conceitos fundamentados em seu escopo podem ser adaptado, haja visto a abordagem sistemática proposta como solução. Assim sendo, percebeu-se a possibilidade do uso desse *framework* como um meio de chegada à convergência *Smart Building* através da transformação de dispositivos comuns em *Smart technologies* (KAZAK; BUCHATSKIY, 2018). Ademais, vale ressaltar que essa transformação citada tem como um de seus critérios o custo de implementação. Sendo assim, existe a possibilidade de aproveitamento da solução em produtos que auxiliem no processo de convergência do paradigma *Smart*.

A implementação de inteligência artificial nesse sistema proposto mostra a possibilidade de integração de estruturas legadas ao contexto de internet das coisas em processos de automatização de ambientes, a fim de que estes operem de forma eficiente e inteligente nos ambientes prediais, especificamente em uma sala localizada no centro de pesquisa HUB tecnologia e inovação situado na Universidade do Estado do Amazonas, visando a preservação da estrutura pré-existente, sem alteração total de seus elementos.

Nesse processo serão aplicados conhecimentos específicos de matérias aprendidas em sala de aula, em particular: Circuito Elétricos I e II, Eletrônica Analógica I, II e III, Eletrônica Digital I e II, Microcontroladores, Linguagem de Programação I e II e Redes de Computadores I.

OBJETIVOS

Objetivo geral

O objetivo deste trabalho é desenvolver uma aplicação de inteligência artificial, inserida na camada de supervisao do SmartLVGrid, capaz de aprender padrões do usuário e ser aplicada em união com *retrofit* de luminárias LED, ensejando a convergência ao paradigma *"Smart"* para sistemas de iluminação aliado à eficiência energética.

Objetivos específicos

Os objetivos específicos deste trabalho são descritos abaixo:

a) expor os conceitos fundamentais do paradigma *Smart Building* e do *framework* SmartLVGrid;

b) estudar técnicas de inteligência artificial que atuem de maneira satisfatória no controle inteligente de sistemas de iluminação LED;

c) projetar e implementar sensoriamento para aquisição de parâmetros que fornecerão as informações para o controle do sistema;

d) projetar e implementar aplicação de inteligência artifical ao *retrofit* de acordo com a plataforma SmartLVGrid;

e) realizar análises dos resultados obtidos no processo de convergência *Smart Building* com o uso da plataforma SmartLVGrid.

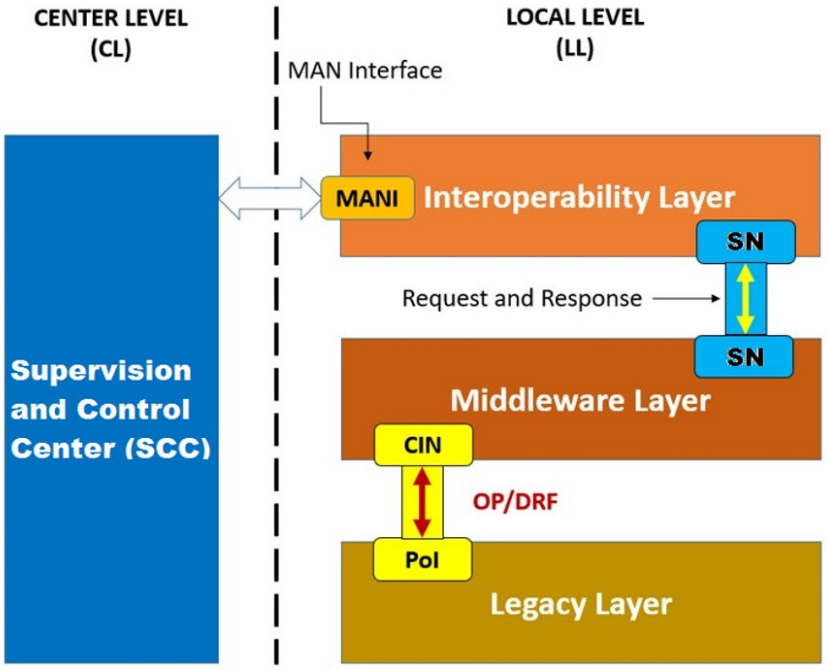
1 REFERENCIAL TEÓRICO

Neste capítulo, serão expostos os conceitos necessários para compreensão do projeto. Inicialmente será abordada a definição do paradigma SmartLVGrid e suas particularidades que foram adaptadas para implementação do projeto. Em seguida será mostrado o conceito do paradigma *Smart* com um enfoque especial para convergência *Smart Building*. Após isso serão apresentados os conceitos necessários para a implementação de *hardware* de sensoriamento que será necessário para aquisição de dados. Então será mostrado conceitos básicos a respeito dos LED's que serão objetos que receberão a atuação do modelo proposto. Por fim será evidenciado aspectos teóricos de técnicas de inteligência artificial que correspondem de melhor forma junto à aplicação.

1.1 O PARADIGMA *SMARTLVGRID*

O *framework* SmartLVGrid é um modelo que descreve uma estratégia, uma estrutura e um conjunto de protocolos aplicados à convergência Smart Grid da rede de distribuição de baixa tensão a partir da adaptação dos circuitos legados que a compõem. Assim, o método de *retrofit* é parte integrante desta plataforma e corresponde à estratégia proposta por ela (GOMES et al., 2017). Com base em uma representação em camadas, a Figura 1 ilustra a pilha SmartLVGrid.

Figura 1 – A pilha SmartLVGrid



Fonte: (GOMES et al., 2017)

Ao observar a Figura 1, nota-se a forma sistemática que está disposta a topologia do *framework*. Partindo dos aspectos macros da figura acima, verifica-se a existência de dois níveis:

Nível Central e Nível Local, denominados no inglês como *Center Level (CL) e Local Level (LL)* respectivamente. O Nível Central localiza-se próximo à concessionária de energia elétrica enquanto o Nível Local situa-se próximo ao consumidor final. Como o *CL e LL* estão alocados em diferentes regiões, de forma natural há uma barreira geográfica que os separam. Com a intenção de transpor esse problema, o SmartLVGrid propõe formas para comunicação dos referidos níveis. O uso de Interface de Rede Metropolitana ou *Metropolitan Area Network Interfaces (MANI)* serve como solução que garante a comunicação entre eles.

Em relação aos aspectos específicos, interno ao Nível Local existem pontos considerados estratégicos pelo *framework*. Esses pontos são denominados como pontos de interface, do inglês *Points of Inteface (PoI)*. Eles estão sob a estrutura legada e tem como intuito ensejar a conexão do sistema existente com as camadas da pilha SmartLVGrid. Com isso, será possível o fluxo de informações e as atuações propostas pelo conceito.

De forma similar, na Camada de *Middleware* também há pontos que farão a conexão entre as camadas. A diferença é que nesse caso não são pontos físicos, mas sim nós virtuais. Como observado na Figura 1 o Nó de Acoplamento e Interação ou *Coupling and Interaction Node (CIN)* é vinculado ao ponto de interface (PoI) da camada legada visando a realização de microprocessos classificados como Funções de *Retrofitting* de Domínio, referenciado no inglês como *Domain Retrofitting Functions (DRFs)* ou Primitivas Operacionais *(OPs)*. As *OPs* são procedimentos que são executados manualmente por operadores de campo no sistema atual, porém com adaptação implementada por meio do *CIN* passam a ser executados por controle automático. A Tabela 1 apresenta *OPs* relevantes para este trabalho descritas no *framework* SmartLVGrid.

Tabela 1 – Os microprocessos, primitivas operacionais, executadas na pilha SmartLVGrid.

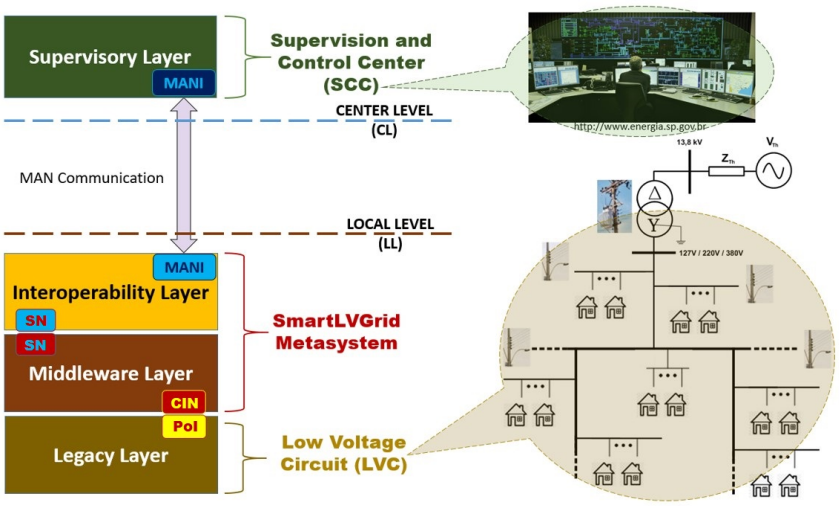
	Classes de Primitivas Operacionais	Descrição
1.	Funções de *Retrofitting* de Domínio (DRFs)	
	1.1. Sensoriamento (Detecção);	Detecção de faltas, distúrbios, ameaças, etc.
	1.2. Medição;	Capacidade de ler e quantificar quantidades físicas, por exemplo, temperatura e variáveis elétricas.
	1.3. On/OFF;	Capacidade de conectar ou desconectar um circuito do seu alimentador.
	1.4. Comutação de fase.	Capacidade de alterar a fonte (ou fase) de uma carga.

Fonte: (GOMES et al., 2017)

De outra forma, os Nós de Serviços *(SN)* são unidades lógicas que fazem a relação entre as camadas de Interoperabilidade e *Middleware*.

Por isso, o principal objetivo da plataforma SmartLVGrid é estipular um modelo para a conversão do sistema atual passivo em um sistema que dispõe de recursos de comunicação e automação que ensejem a supervisão e o controle remoto da planta de distribuição por parte do centro de operações da distribuição e, em alguns casos, a operação local e autônoma do mesmo. A representação em camadas observada na Figura 2 ilustra a arquitetura do sistema que o modelo SmartLVGrid faz na composição com o sistema de distribuição de baixa tensão.

Figura 2 – A pilha SmartLVGrid e os elementos de composição do sistema de distribuição de energia elétrica



Fonte: (GOMES et al., 2017)

Nos sistemas de energia elétrica tradicionais as estratégias que envolvem a captura e a análise de dados para processos de tomada de decisão são aplicadas unicamente na central de operação e distribuição de energia. Partem dessa central todas as informações sobre status de funcionamento do sistema e atuações necessárias a serem desempenhadas pelos operadores de campo. Como alternativa à centralização de *Business Inteligence(BI)*, o modelo SmartLVGrid compartilha o BI com os circuitos de baixa tensão que passaram pelo *retrofit*, visto que a partir disso tornam-se dotados de capacidade de processamento e automação, dessa forma oportunizando uma gama de funcionalidades controláveis localmente (GOMES et al., 2017).

1.1.1 A camada de *Middleware* do *framework* SmartLVGrid

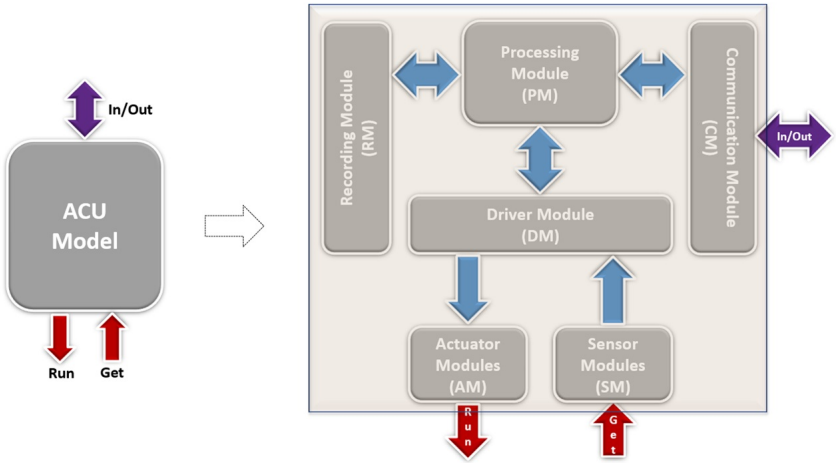
A camada legada na pilha SmartLVGrid diz respeito à toda estrutura que compõe o sistema existente no nível local *(LL)*. A camada que realiza a interface com a camada legada denomina-se camada de *Midleware* como observado na Figura 1. A camada de *Middleware* realiza processos no nível mais baixo da plataforma, realizando o interfaceamento direto através dos nós de acoplamento *(CIN)* que implementam as primitivas operacionais *(OPs)* através da

automação ou controle externo e podem ser executados por meio do sistema supervisório ou de forma autônoma.

É importante salientar que os nós de acoplamento e interação e a camada de *Middleware* são abstrações da referida plataforma. Desta forma, a implementação física do modelo está direcionada ao dispositivo responsável pelas DRFs que são embarcados que possuem em sua composição sensores e atuadores de acordo com suas aplicações.

Devido à necessidade de implementação das primitivas operacionais (OP), faz-se crucial um sistema microprocessado nessa plataforma. O autor denominou esse sistema microprocessado de Unidade de Controle e Automação ou *Automation and Comunication Unit* (ACU). A funcionalidade de cada ACU dependerá das ações (OPs) a serem executadas e do tipo de estrutura legada que a receberá como módulo de *retrofit*. A Figura 3 fornece uma representação do diagrama de blocos do modelo genérico de ACU.

Figura 3 – Estrutura e composição da ACU



Fonte: (GOMES et al., 2017)

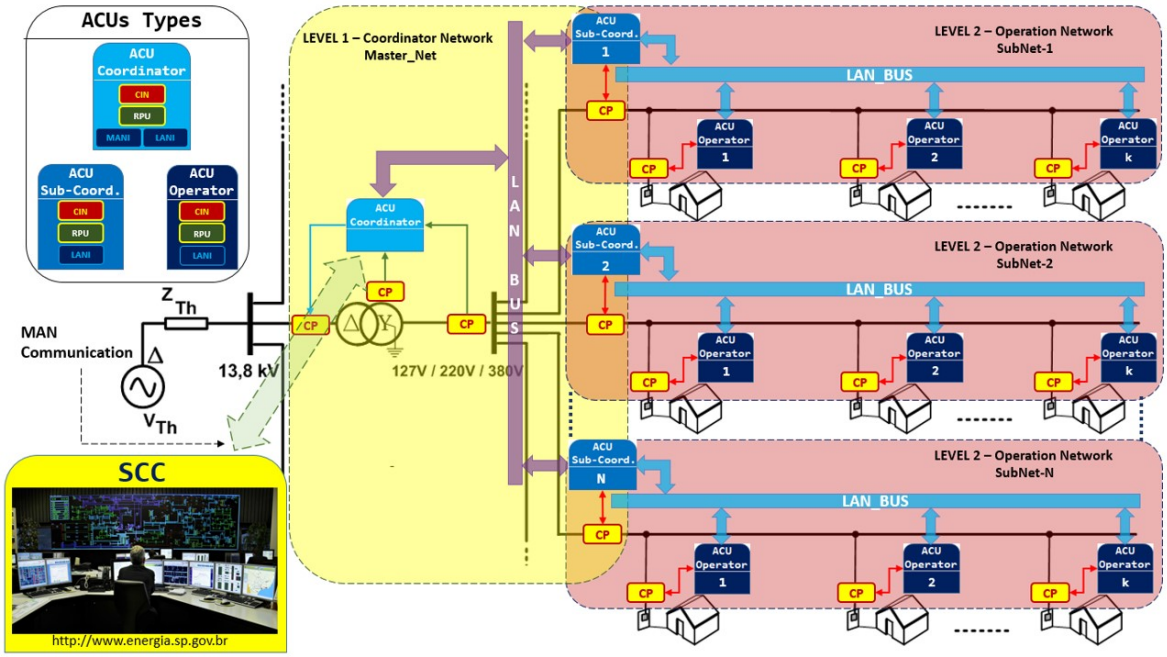
A Figura 3, ilustra uma modelagem de ACU através de um diagrama de blocos. Nesse diagrama, são observadas as portas: *In/Out, Get e Run*; assim como os componentes internos que mostram que a ACU possui a possibilidade de atuação, sensoriamento, armazenamento, processamento e comunicação. Essa comunicação pode ocorrer em uma escala local caracterizando uma *Local Area Network (LAN)* ou uma *Metropolitan Area Network(MAN)*. Esses componentes internos dependem tanto do domínio de associação e nível de controle/monitoramento quanto da função designada a ela na camada de interopelabilidade.

Em relação às funções a serem desempenhadas pelas portas supracitadas, a porta *Get* realiza a leitura de variáveis definidas de acordo com a localidade de sua instalação, com a finalidade de medição dos respectivos valores e detecção de violações dos limites de operação ou quaisquer eventos em disparidade com o esperado, enquanto a porta *Run* é utilizada para atuação no sistema (GOMES et al., 2017).

1.1.2 A camada de interoperabilidade do *framework* SmartLVGrid

Para a definição da camada de interoperabilidade, é crucial recordar que as ACUs são dispositivos que estão espalhados pelo sistema com funcionalidades que estão de acordo com o local onde estão instaladas. Por isso, há ACUs com diferentes funções e que devem ser dispostas de forma organizada para correto funcionamento do sistema. Visando essa organização a camada de interoperabilidade estipula as regras, critérios e infraestrutura essencial para realizar as conexões e estruturar de forma hierárquica as ACUs. Dispostas dessa forma, são realizados os serviços e interações entre objetos propostos pela plataforma de forma eficiente. A figura 4 ilustra a forma que ocorre essa estruturação.

Figura 4 – Arquitetura hierárquica de ACUs em sistema de distribuição de baixa tensão

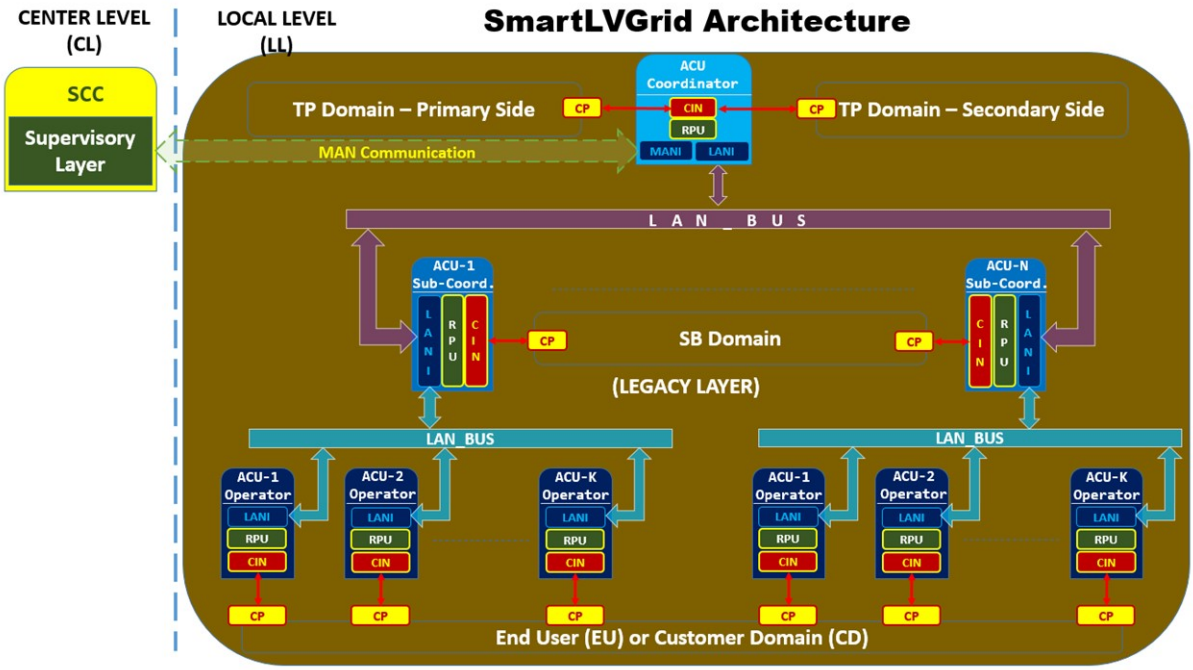


Fonte: (GOMES et al., 2017)

Posto isso, o modelo estipula redes de duas naturezas: puramente operacional e de coordenação. Como é observado na figura 4, a rede puramente operacional é composta pelos operadores ou *operators* e seu respectivo subcoordenador ou *sub-coordinator*. Em razão da natureza operacional essa rede é também denominada de Rede Escrava, do inglês *Slave Net*. Essa rede é a responsável pela atuação direta no sistema por meio das primitivas operacionais. O coordenador, ou *coordinator*, em conjunto com os subcoordenadores, formam outra rede cuja finalidade é a coordenação e, por essa razão é chamada de Rede Mestre ou *Master Net*. A Mestre monitora e supervisiona todos os outros elementos do sistema. A Rede Escrava localiza-se hierarquicamente abaixo da Rede Mestre, sendo a primeira subordinada à segunda. Vale ressaltar que apesar de existir uma rede hierárquica no sistema, os *operators* podem atuar com seus serviços de forma independente. Entretanto, podem executar operações de controle de acordo com a necessidade de hierarquias maiores.

A respeito da comunicação entre os elementos do sistema, ao observar a localização das ACUs do tipo *operators*, percebe-se que são requeridos módulos de comunicação com uma única porta LAN para que possam acessar a *Net Slave*. Quanto as ACUs *sub-coordinators* requerem duas portas LANs para acesso a *SlaverNet* e outra para *Master Net*. A ACU do tipo Coordenador Central requer dois tipos diferentes de módulos de comunicação, um do tipo Rede Metropolitana (MAN) para acesso ao Barramento de Intercomunicação e outro do tipo LAN para acesso à *Master Net*. A estruturação dessa comunicação pode ser observada abaixo na figura 5.

Figura 5 – Arquitetura hierárquica de ACUs com os barramentos de comunicação



Fonte: (GOMES et al., 2017)

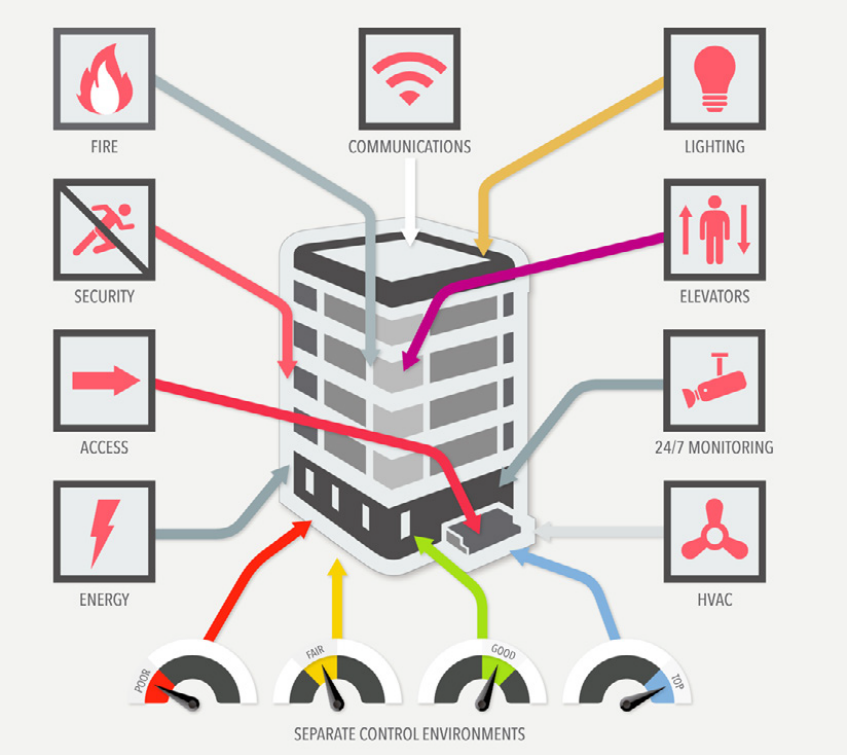
1.2 O PARADIGMA *SMART BUILDING*

Smart Building, ou edifício inteligente, consiste na integração de um sistema em certa coordenação com a finalidade de gerenciamento dos recursos com eficiência. Esse conceito passou a ser discutido por volta do ano de 1980. Vale citar o livro *Intelligent buildings: design, management and operation* (CLEMENTS-CROOME; CROOME, 2004) cita o autor Caffrey que buscou definir o conceito conforme o texto abaixo:

> Um edifício inteligente é aquele que proporciona uma economia produtiva e econômica em ambientes através da otimização dos seus quatro elementos básicos: estrutura, sistemas, serviços e gestão, e as inter-relações entre eles. Edifícios Inteligentes auxiliam os proprietários de edifícios, gerentes de propriedade e ocupantes a realizar suas metas nas áreas de custo, gestão de energia, conforto, conveniência, segurança, flexibilidade e comercialização a prazo (Caffrey 1985, Intelligent Buildings Institute, Washington DC).

Ao observar essa definição, percebe-se que as abordagens atuais do paradigma *Smart Building* seguiram a linha do autor citado acima. O cerne desse conceito de *Smart Building* usa a automação em sua fase operacional. Edifícios conceituados como *Smart* comumente atuam nas seguintes áreas de suas instalações: aquecimento, ventilação e ar condicionado. Este conjunto de instalações é conhecido pela abreviatura HVAC que significa *Heating, Ventilation and Air conditioner*. Todavia, não se limitam à esses elementos. Podem englobar a iluminação, sistemas de segurança entre outros. Na figura 6 pode ser observado o possível escopo de atuação. A intervenção nessas áreas das instalações promovem a eficiência energética, além da interface de comunicação e transporte vertical das informações que são produzidas pelo edifício (EASTMAN et al., 2011).

Figura 6 – Elementos de uma *Smart Building*



Fonte: (SMARTBUILDING, 2018)

O número de edifícios que utilizam este conceito aumentou em todo o mundo nos últimos anos como uma estratégia para mitigar consumo de energia, questões ambientais e mudanças climáticas (CLEMENTS-CROOME, 2011; NGUYEN; AIELLO, 2013).

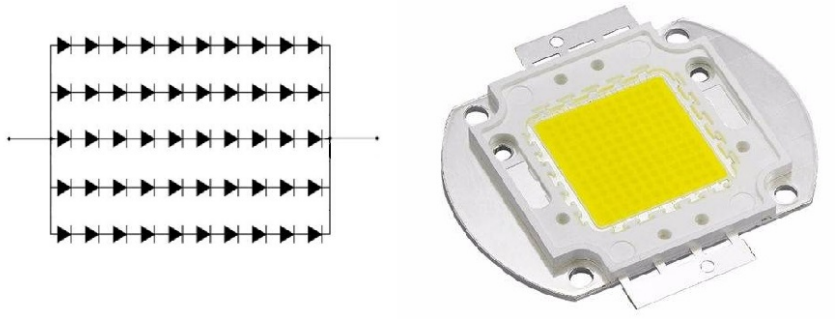
1.3 TÉCNICA DE ILUMINAÇÃO A LED

O LED é um componente eletrônico semicondutor, ou seja, um diodo emissor de luz LED (*Light Emitting Diode*) , mesma tecnologia utilizada nos chips dos computadores, que tem a propriedade de transformar energia elétrica em luz. A emissão espontânea de luz devido à

recombinação radiativa de dentro da estrutura de diodo é conhecido como eletroluminescência. O LED (*Light Emitting Diode*) é um componente eletrônico semicondutor que possui esse efeito. Diferentemente de outras fontes de luz artificiais, as lâmpadas de estado sólido não necessitam de filamentos, eletrodos ou tubos de descarga e são caracterizadas pelo LED (*Light Emitting Diode*) (FERREIRA; TOMIOKA, 2013). O funcionamento se dá a partir da inserção de uma fonte de tensão nos terminais de uma junção p-n extinguindo a banda proibida possibilitando a circulação de elétrons. Sabe-se que os elétrons possuem maior energia do que as lacunas, assim, quando há a união elétron-lacuna há a liberação de energia na forma de radiação luminosa(RANGEL; SILVA; GUEDE, 2009).

Ressalta-se que o espectro luminoso do LED não emite raios ultra-violeta, que são prejudiciais a visão e a pele humana, além de que não emite calor. Conhecer e aplicar corretamente a tecnologia LED é um dos desafios atuais dos pesquisadores das áreas de luminotécnica (DONG et al., 2017) e qualidade de energia (MONTEIRO et al., 2014). Já são diversas as aplicações da iluminação LED e entre elas destacam-se: iluminação de subestações e outros sistemas de energia, suporte na produção e iluminação pública(KANSARA, 2017)(PEREIRA et al., 2015). Na figura abaixo, pode ser observado o arranjo típico de uma luminária LED.

Figura 7 – Arranjo de luminária LED e exemplo de LED



Fonte: (FERNANDES; GUIMARÃES, 2018)

Apesar de todos os benefícios supracitados em relação à iluminação a LED, sistemas que realizam o controle para esses dispositivos apresentam índices de qualidade de energia considerados baixos além da eficiência em níveis abaixo do considerado desejável. O controle de tais sistemas acarretam na perda de eficiência qualidade. Com isso, o sistema elétrico de energia acaba sofrendo com os efeitos prejudiciais advindos de sistemas dessa natureza (MONTEIRO et al., 2014).

1.3.1 Controle de intensidade luminosa em LEDs

Os sistemas de iluminação atuais possuem em sua composição as lâmpadas LED. Essas, devido ao material semicondutor que as compõem, precisam ser alimentadas por um sinal DC.

Em vista disso, faz-se crucial o uso de conversores para o acionamento dessas luminárias. Para o controle de fluxo luminoso, é necessário o uso de *choppers* DC-DC, também denominados *drivers* de LED. O projeto desses conversores devem estar de acordo com as especificações dos LEDs presentes na luminária em questão. Na figura 8 é mostrado um driver de LED do mercado atualmente.

Figura 8 – Driver de LED do mercado

Fonte: (ILUMININ, 2018)

O fluxo luminoso emitido por uma luminária não deve sofrer oscilações. As oscilações de fluxo luminoso, além de informar o mal funcionamento do dispositivo, causa problemas relacionados a saúde do ser humano. O ideal é que se mantenha um fluxo luminoso constante em um ambiente. Para que isso ocorra nas luminárias LED, a corrente média que passa por ela deve permanecer constante. Todavia, o arranjo de LED nas luminárias é um fator que influencia na medida de controle a ser tomada para o controle do fluxo de energia. A disposição de LEDs em série enseja o controle de corrente de forma mais eficiente, pois a partir disso é possível a regulação do fluxo luminoso de um conjunto de LEDs com apenas uma corrente em uma única *string*.

1.4 REDE DE SENSORES

Os seres humanos naturalmente desempenham o papel de sensoriamento do mundo ao seu redor. Os cinco sentidos inerentes ao homem não agem de forma independente. Em vez disso, um sentido complementa o outro e são processados de forma conjunta, ensejando ao cérebro humano a tomada de decisão de forma inteligente. O cérebro humano frequentemente utiliza de informações disponibilizadas pelos sentidos como dados de entrada para validar um evento e ainda compensar quando há informações incompletas no ambiente.

De forma análoga, os sensores desempenham o papel de medição de alguma grandeza física e conversão dessa leitura em um representação digital. Essas informações digitalizadas são enviadas para outros dispositivos que, por sua vez, transformarão em dados úteis e servirão de insumos para dispositivos inteligentes ou para humanos.

Um sensor é baseado na identificação de uma lei física que relaciona duas variáveis de naturezas diferentes. Exemplos disso são: temperatura e resistência, velocidade e tensão. Tendo isso em vista, existe uma infinidade de tipos de sensores, de acordo com suas respectivas aplicações. Há diversas maneiras de classificação de sensores em diferentes categorias. Uma dessas formas é ilustrada na figura a seguir (HANES et al., 2017; BAHGA; MADISETTI, 2014).

Figura 9 – Tipos de sensores

Tipos de sensores	Descrição	Exemplos
Posição	Um sensor de posição mede a posição de um objeto; a posição medida pode ser em valores absolutos(posição absoluta do sensor) ou em valores relativos (sensor de deslocamento). Sensor de posição pode ser linear, angular ou multi-eixos.	Potenciômetro, Clinômetro, Sensor de proximidade
Presença e Movimento	Sensor de presença detecta a ocupação de um ambiente por pessoas e animais, enquanto o sensor de movimento detecta as respectivas movimentações de pessoas e objetos. A diferença é que o sensor de presença gera sinal mesmo se a pessoa fica parada no ambiente enquanto o outro só gera sinal a partir do movimento.	Olho elétrico, radar
Velocidade e Aceleração	Sensor de velocidade deve ser linear ou angular, indicando o quão rápido um objeto se move em uma linha reta ou o quão rápido rotaciona. O sensor de aceleração mede a mudança de velocidade.	Acelerômetro, giroscópio
Pressão	Sensores de pressão são relacionados com a força física aplicada por unidade de área.	Barômetro, piezômetro
Umidade	Sensores de Umidade detectam a Umidade (montante de vapor) no ar ou em uma massa. Umidade pode ser medida de várias maneiras: Umidade absoluta, relativa, etc.	Higrômetro, Humistor
Luz	Sensor de luz detecta a presença de luz visível ou invisível.	Sensor Infravermelho, fotodetector
Temperatura	Sensor de temperatura mede o montante de calor está presente em um sistema. Pode ser basicamente de dois tipos: Contato ou não contato.	Termômetro, Calorímetro

Fonte: (HANES et al., 2017)

Sensores possuem todas as formas e tamanhos, conforme apresentado na figura acima. Eles são desenvolvidos para medição de diversas condições físicas. Sensores organizados em rede são considerados como tecnologia emergente que impactará a economia global e moldará o futuro. A proliferação deles será o principal responsável por esse fenômeno (HANES et al., 2017).

1.5 INTELIGÊNCIA ARTIFICIAL E APRENDIZADO DE MÁQUINA

Na computação, problemas são solucionados basicamente através de algoritmos que especificam exatamente os procedimentos para resolução do mesmo. Entretanto, existem problemas que requerem procedimentos de solução que são complexos para cognição humana. Um exemplo disso consiste no reconhecimento facial ou da voz. Apesar do cérebro humano realizar essa atividade constantemente por meio do reconhecimento de padrões, é muito complexa a descrição de procedimentos em forma de algoritmos para tais situações (FACELI et al., 2011).

Nas últimas décadas, com a crescente complexidade dos problemas a serem tratados computacionalmente e diante do grande volume de dados constantemente gerados por diferentes setores, tornou-se clara a necessidade de métodos e técnicas computacionais mais sofisticados, para soluções autônomas e sem intervenções humanas. Com essa finalidade, essas técnicas teriam a incumbência de criar, partindo de experiências passadas, uma hipótese ou função, capaz de solucionar o problema em questão. Esse processo de indução de uma hipótese partindo de experiências passadas denomina-se Aprendizado de Máquina (AM), largamente conhecido como *Machine Learn* (FACELI et al., 2011).

O aprendizado de máquina pode ser aplicado a uma ampla gama de problemas de negócios, desde detecção de fraudes, a segmentação de clientes e recomendação de produtos, informações de monitoramento industrial, análise de sentimentos e diagnóstico médico. Pode assumir problemas que não podem ser gerenciados manualmente devido à grande quantidade de dados que devem ser processados. Quando aplicado a grandes conjuntos de dados, o AM pode encontrar relacionamentos tão sutis que seriam improváveis de serem encontrados pelo ser humano. E quando muitos desses relacionamentos sutis são combinados, eles se tornam fortes preditores (BRINK et al., 2017). Partindo disso, o aprendizado de máquina é o estudo ordenado de algoritmos e sistemas que melhoram conhecimento ou desempenho com a experiência (FLACH, 2012).

A possibilidade de aplicação de AM se dá em uma infinidade de tarefas que podem ser dispostas de acordo com diferentes critérios. Os critérios dessas tarefas a serem desempenhadas pelo algoritmo estão fortemente ligados com o paradigma de aprendizado a ser selecionado. De acordo com esses critérios, as tarefas de AM podem ser divididas em: Preditivas e Descritivas (FACELI et al., 2011).

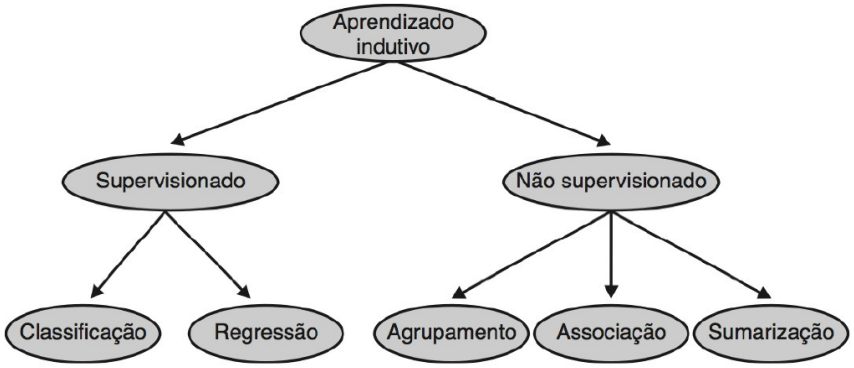
As tarefas preditivas são utilizada para induzir modelos ou teorias (como árvores de decisão ou conjunto de regras) a partir de dados classificados em conjuntos (FÜRNKRANZ; GAMBERGER; LAVRAČ, 2012). Elas utilizam modelos que submetem os dados de entrada à um treinamento que aprende a rotular uma determinada saída com base no conhecimento adquirido. Assim sendo, os conjuntos de dados utilizados neste tipo de tarefa possuem atributos de entrada e saída. Esses algoritmos seguem o paradigma de aprendizado supervisionado (FACELI et al., 2011).

Em tarefas de descrição, as técnicas de AM realizam uma análise exploratória dos dado

contidos no conjunto com a finalidade de descrever os mesmos. Em contraponto com a preditiva, a base de dados não possui atributos de saída e seguem um paradigma de aprendizado chamado aprendizado não supervisionado. As técnicas utilizadas neste paradigma utilizam regras para agrupar os dados ou encontrar associações que relacionam os atributos de entrada (BRINK et al., 2017).

A figura 10 ilustra como estão organizados os tipos de tarefas de aprendizado. No centro mostra o aprendizado indutivo que consiste na generalização a partir de dados passados. Migrando para as demais camadas , tem-se os tipos de aprendizado e em seguida as tarefas por elas desempenhadas.

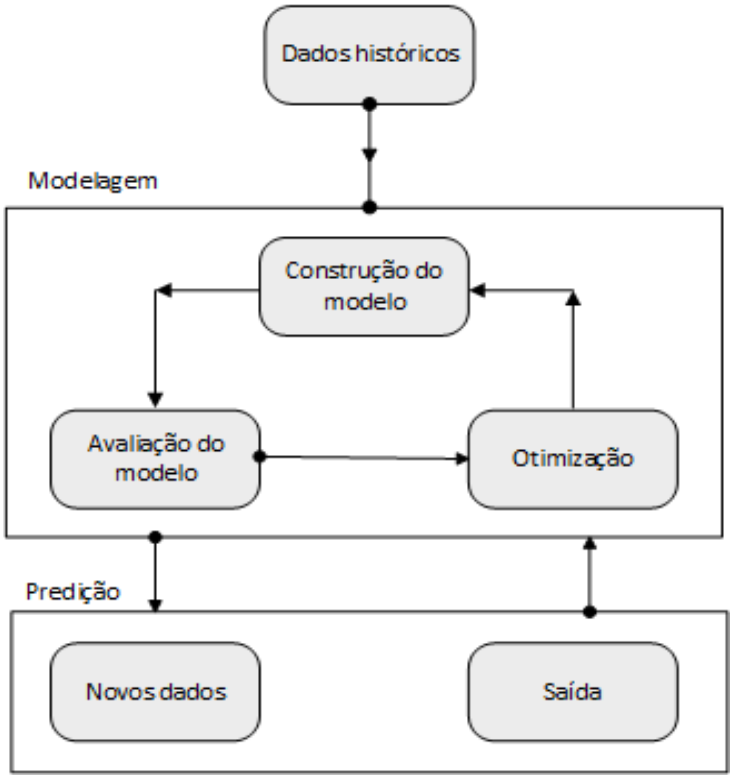
Figura 10 – Hierarquia de aprendizado.



Fonte: (FACELI et al., 2011)

O fluxo de trabalho do AM tem cinco componentes principais: preparação de dados, construção de modelo, avaliação, otimização e previsões em novos dados. A aplicação desses passos tem uma ordem inerente, mas a maioria das Aplicações de aprendizado de máquina exigem revisitar cada etapa várias vezes em um processo iterativo (BRINK et al., 2017). A partir dos dados históricos de entrada é possível construir um modelo usando um algoritmo de AM. Então é preciso avaliar o desempenho do modelo e otimizar precisão e escalabilidade para se adequar aos requisitos do problema. Por fim, com o modelo final é possível realizar previsões sobre novos dados. Essa metodologia pode ser visualizada na figura 11.

Figura 11 – Fluxo de trabalho do aprendizado de máquina



Fonte: Próprio Autor

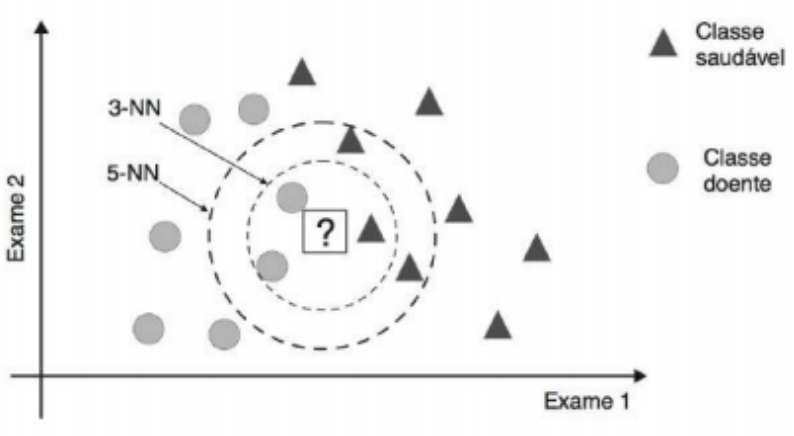
1.5.1 Vizinhos mais próximos

O algoritmo de Vizinhos mais Próximos, do inglês Nearest Neighbours (NN), é um método de AM que compara a distância de dados similares que tendem a estar concentrados em uma mesma região no espaço de entrada (FACELI et al., 2011). Faz-se necessário obter as posições dos pontos de dados dentro do espaço de entrada. Após isso, são encontrados quais dados de treinamento estão localizados próximos a ele. Isso requer o cálculo da distância de cada ponto de dados no conjunto de treinamento. É possível então identificar os "k"vizinhos mais próximos ao ponto de teste, e em seguida, definir a classe de ponto que serão classificados como vizinhos mais próximos. A distância entre todos os pares de pontos é dada comumente pela distância euclidiana, denotada por d, representada na equação abaixo:

$$d(x_i, x_j) = \sqrt{\sum_{l=1}^{d} (x_i^l - x_j^l)^2} \tag{1}$$

A escolha de "k"não é trivial, o que torna esse método sensível ao ruído, extenso e de menor precisão conforme os pontos se distanciam (MARSLAND, 2011). Isso pode ser observado na figura 12.

Figura 12 – Influência do valor de K no algoritmo *k-NN*



Fonte: (FACELI et al., 2011)

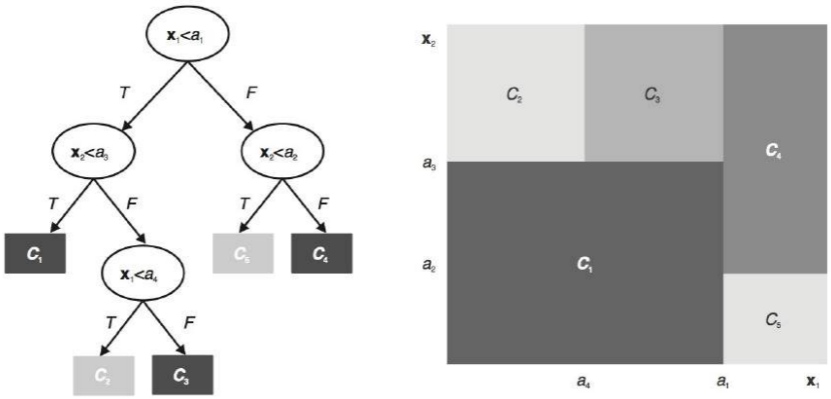
Na figura 12, considera-se um conjunto de dados em que objetos são indivíduos classificados em saudáveis ou doentes, e o espaço de entrada é apresentado como dois atributos que representam o resultado de dois exames. O ponto "?"simboliza o ponto de teste ou o indivíduo a ser classificado. Para k = 3, o objeto seria classificado como "doente", enquanto para k = 5 o objeto de teste seria classificado como "saudável". Para contorno dessas dificuldades outras medidas são tomadas como associar um peso à contribuição de cada vizinho (FACELI et al., 2011).

1.5.2 Árvore de Decisão

A técnica de Árvore de decisão, do inglês *Decision Tree*, soluciona problemas de tomada de decisão utilizando a estratégia de em partes menores para assim atingir a meta. Os dados de um problema são divididos recursivamente em subconjuntos baseados em testes. As soluções desses subconjuntos podem ser combinadas, na forma de uma árvore para assim resolver o problema como um todo. A figura 13 ilustra uma árvore de decisão e sua respectiva representação no espaço definido pelos atributos x_1 e x_2. Cada ponto de decisão ou nó da árvore corresponde à uma região nesse espaço. As regiões são mutuamente excludentes (FACELI et al., 2011; KUBAT, 2017).

Para a construção de uma árvore de decisão, é necessária a definição de um componente importante: a entropia da informação. A entropia consiste em medir a aleatoriedade ou dificuldade de predizer o atributo alvo. A medida em que cresce a incerteza de experimento aleatório, maior é a informação que se obtém ao observar a sua ocorrência. Vale lembrar que quanto mais nós são criados, cada subconjunto será caracterizado pelas próprias probabilidades de decisão. Essa entropia $H(T)$ pode ser medida a partir da Equação a seguir onde p_i é probabilidade do evento da variável aleatória T e logaritmo na base 2 para caracterização de escolha binária.

Figura 13 – Árvore de decisão e regiões de decisão



Fonte: (FACELI et al., 2011)

$$H(T) = -\sum_{i=0} p_i \times log_2 p_i \tag{2}$$

Existem variados algoritmos para Árvore de Decisão dependendo de sua aplicação. O C4.5 é um algoritmo que utiliza uma abordagem recursiva de particionamento do conjunto de dados. Além desse, o CART também é bastante utilizado, capaz de construir somente árvores do tipo binária (MARSLAND, 2011).

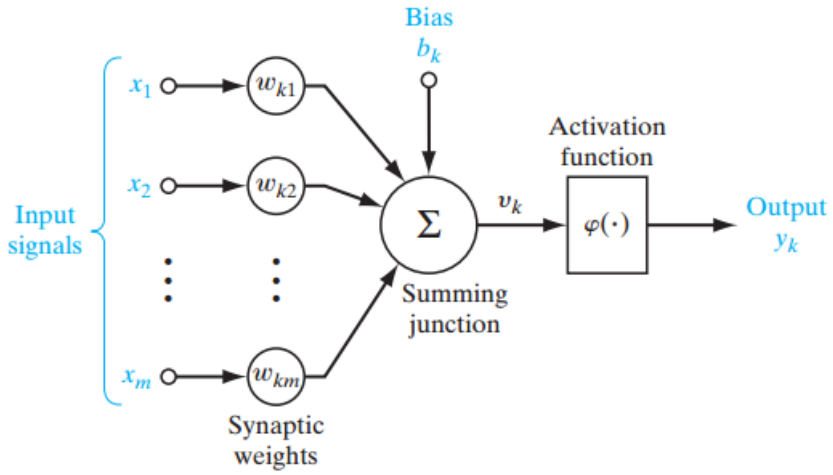
1.5.3 Redes Neurais Artificiais

Rede neural artificial (RNA) consiste na tentativa de modelagem das capacidades de processamento de informações dos sistemas nervosos existentes nos animais. O cérebro humano possui em sua composição de 10 a 500 bilhões de células interconectadas. Essas células nervosas também denominada de neurônios quando agrupadas formam um arranjo complexo que lida com sinais de entrada de muitas maneiras diferentes. Todavia, os neurônios animais possuem uma menor velocidade em relação às portas lógicas eletrônicas. Estes podem atingir tempos de comutação de alguns nanossegundos, enquanto os neurônios precisam de vários milissegundos para reagir a um estímulo. No entanto, o cérebro é capaz de resolver problemas com os quais nenhum computador digital pode lidar eficientemente (ROJAS, 2013; FACELI et al., 2011).

De forma análoga ao sistema nervoso, as RNAs são sistemas computacionais compostos de unidades de processamento simples, densamente interconectados e dispostas em uma ou mais camadas. A partir da forma em que estão dispostas as camadas são formadas diferentes arquiteturas que, por sua vez, possuem suas devidas conexões que atuam como pesos responsáveis por ponderar a entrada recebida por cada neurônio da rede. Esses pesos podem assumir valores positivos ou negativos de acordo com a natureza de excitação ou inibição do comportamento.

Esses valores são ajustados no processo de treinamento, também denominado aprendizado, e codificam o conhecimento adquirido pela rede a fim de alcançar o objetivo pretendido.

Figura 14 – Neurônio artificial



Fonte: (HAYKIN et al., 2009)

Na figura 14 observa-se que um sinal x_j na entrada de sinapse m conectado ao neurônio k é multiplicado pelo peso sináptico w_{kj}. Após isso, um somador age sobre os sinais de entrada, ponderados pelas respectivas forças sinápticas do neurônio constituindo uma combinação linear. Após o somador, tem-se a função de ativação para limitar a amplitude da saída do neurônio. A função de ativação também é conhecida como uma função de esmagamento, na qual limita a faixa de amplitude permitida do sinal de saída para algum valor finito. O modelo apresentado também inclui uma polarização aplicada externamente, denotada por b_k. O valor de b_k tem o efeito de aumentar ou diminuir a entrada líquida da função de ativação, dependendo se é positiva ou negativa, respectivamente. Esse modelo pode ser expresso matematicamente conforme as equações abaixo:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{3}$$

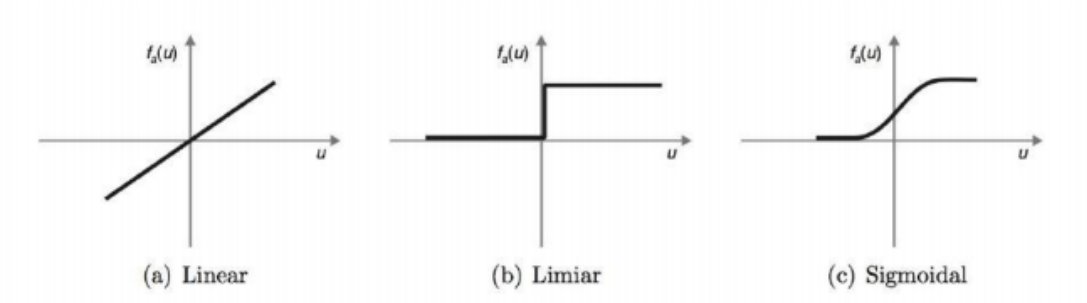
e

$$y_k = \varphi(u_k + b_k) \tag{4}$$

onde $x_1, x_2, ..., x_m$ são os sinais de entrada; $w_{k1}, w_{k2}, ..., w_{km}$ são os respectivos pesos sinápticos do neurônio k; b_k, atenua ou amplifica a função de ativação. φ é a função de ativação e y_k é o sinal de saída do neurônio (HAYKIN et al., 2009).

A função de ativação fornece a resposta de um neurônio para uma dada entrada. Esta função precisa ser monotônica e contínua, podendo comumente ser as funções linear, sigmoide ou limiar. As suas representações podem ser observadas na figura 15.

Figura 15 – Funções de ativação

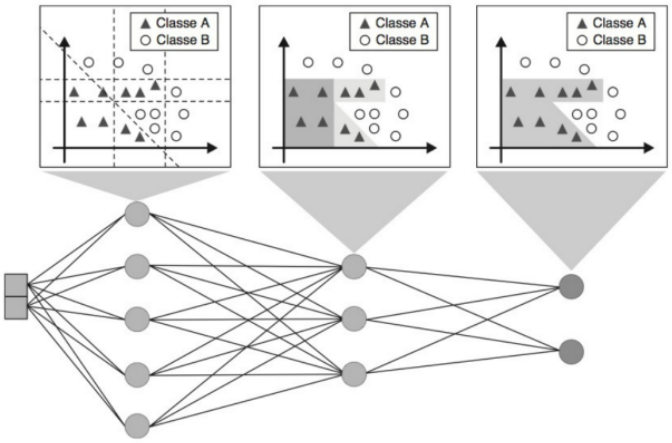


Fonte: (FACELI et al., 2011)

Em uma Rede Neural Artificial, os neurônios podem estar organizados em mais de uma camada. Quando a disposição está em configuração com uma quantidade maior que uma camada, a entrada de um neurônio pode ser a saída de um neurônio localizado em uma camada anterior. A partir disso, cada neurônio desempenha o papel de resolução de problemas linearmente separáveis e em conjunto são capazes de resolver problemas de elevada complexidade (BRAGA; FERREIRA; LUDERMIR, 2007; HAYKIN et al., 2009).

O processo de aprendizado é a capacidade de aprender a partir de dados passados. Esse processo nas RNAs ajusta os pesos aplicando um conjunto de exemplos de treinamento rotulados. A cada padrão de entrada submetido à rede, compara-se a resposta desejada com a resposta calculada, ajustando-se os pesos das conexões para minimizar o erro. A figura 16 ilustra o desempenho de cada neurônio para definição das fronteiras de decisão que ensejarão a RNA realizar a classificação de novos exemplos (HAYKIN et al., 2009; FACELI et al., 2011).

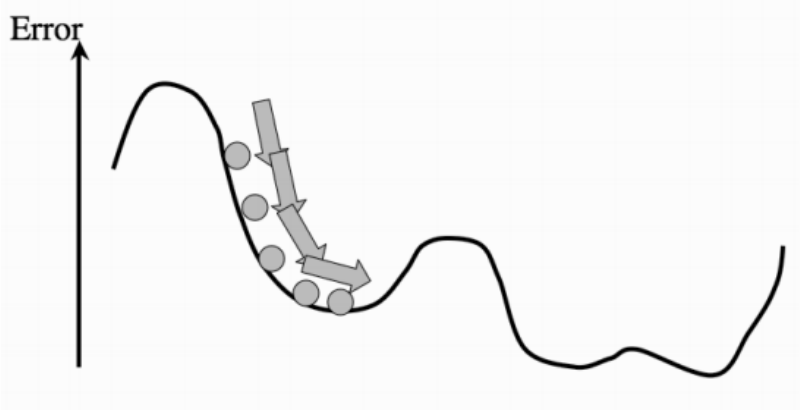
Figura 16 – Papel de cada neurônio das diferentes camadas de uma rede multicamadas



Fonte: (FACELI et al., 2011)

Para realização do aprendizado citado acima, comumente utiliza-se o algoritmo denominado *Back-propagation*. Esse método procura o mínimo da função de erro no espaço de ponderação usando o método de gradiente descendente. A combinação de pesos que minimiza a função de erro, conforme ilustra a figura 17, é considerada uma solução do problema de aprendizagem. Esse método requer o cálculo do gradiente da função de erro em cada etapa de iteração. Os ciclos de apresentação dos dados de treinamento e eventuais ajustes de pesos no *back-propagation* são iterados até que seja atingido um critério de parada como, por exemplo, um número máximo de ciclos ou uma taxa máxima de erro.

Figura 17 – Os pesos das redes são treinadas a fim de encontrar o locais de mínimos.



Fonte: (MARSLAND, 2011)

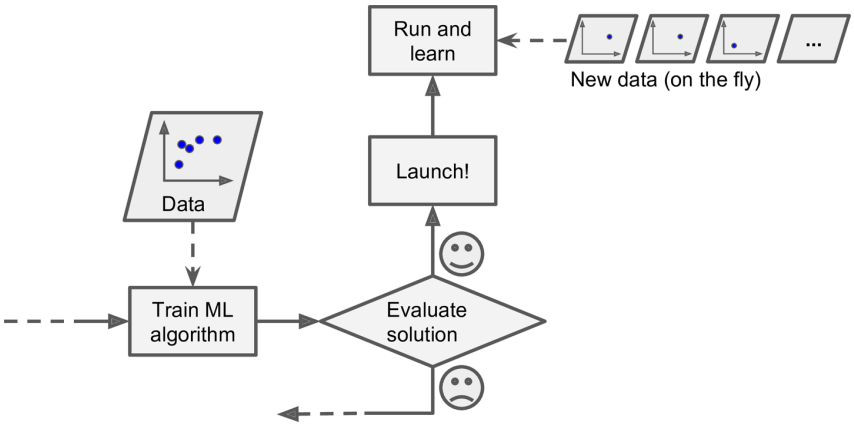
1.5.4 Aprendizado Incremental

Atualmente, praticamente todas as informações são coletadas e armazenadas em formato digital. Empresas como a Google recebe 3,5 bilhões de consultas de pesquisa; Amazon vende cerca de 13 milhões de itens no mundo todo. Todos os tipos de informações do cliente, dados transacionais brutos também como comportamento de clique individual, é coletado para fornecer serviços como recomendações personalizadas. Superar esse estado de coisas requer uma mudança de paradigma para o aprendizado de máquina e processamento sequencial de dados no esquema de streaming. Os Algoritmos incrementais se encaixam nessa problemática, pois incorporam continuamente informações em seu modelo e tradicionalmente buscam tempo e espaço mínimos de processamento. Devido à sua capacidade de processamento contínuo em larga escala e em tempo real, eles recentemente ganharam mais atenção, particularmente no contexto de Big Data (LOSING; HAMMER; WERSING, 2018).

Algoritmos incrementais também são muito adequados para aprender além da fase de produção, o que permite que os dispositivos se adaptem aos hábitos individuais do cliente e ambientes. Isso é particularmente interessante para produtos domésticos inteligentes. Nesse caso,

o principal desafio não é o processamento em larga escala, mas sim contínuo e aprendizado eficiente com poucos dados. A figura 18 ilustra o processo de aprendizado dessa natureza.

Figura 18 – Aprendizado incremental.



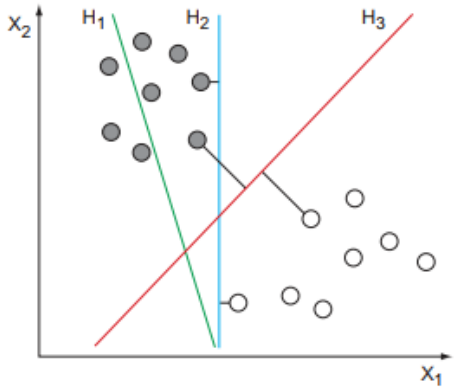
Fonte: (GÉRON, 2019)

O aprendizado incremental é ótimo para sistemas que recebem dados como um fluxo contínuo e precisa se adaptar às mudanças de forma rápida ou autônoma. Também é recomendado quando os recursos computacionais são limitados, visto que quando esse método aprende sobre novas instâncias de dados, ele não precisa mais deles a menos que haja a intenção de reverter para um estado anterior. Isso possibilita uma economia em quantidade de espaço. Os algoritmos de aprendizado incremental também podem ser usados para treinar sistemas em enormes conjuntos de dados que não pode caber na memória principal de uma máquina. Essa forma de implementação é chamada de aprendizado fora do núcleo. O algoritmo carrega parte dos dados, executa uma etapa de treinamento nesses dados e repete o processo até que seja executado em todos os dados (GÉRON, 2019).

1.5.4.1 Máquina de vetores de suporte incremental

O algoritmo da máquina de vetores de suporte (SVM) é uma escolha popular para problemas lineares e não lineares. Possui algumas propriedades teóricas e práticas interessantes que o torna útil em muitos cenários. A ideia principal por trás do algoritmo é para encontrar vetores que separam as classes de maneira óptima. Em vez de medir a distância de todos os pontos, os SVMs tentam encontrar a maior margem (*maximal margin classifier*) entre os pontos de cada lado da linha de decisão. Essa técnica propõe que não há razão para se preocupar com pontos que estão bem dentro do limite, apenas aqueles que são próximos. Na figura 19 a seguir, é possível notar que as linhas H_1 e H_2 são vetores com separação inadequada. O vetor H_3 encontrado é a linha óptima para o sistema (BRINK et al., 2017).

Figura 19 – Fronteira de decisão de uma máquina de vetores de suporte.



Fonte: (BRINK et al., 2017)

Dados linearmente separáveis admitem infinitamente muitos limites de decisão que separam em classes, mas intuitivamente algumas delas são melhores que outras. Máquinas de vetores de suporte foram definidas por três etapas incrementais. Primeiro, Vapnik e Lerner (1963) propor a construção do Hiperplano Ideal, ou seja, o classificador linear que separa os exemplos de treinamento com a maior margem. Guyon, Boser e Vapnik (1993) propõem construir o Hiperplano Ideal no espaço de recurso induzido por uma função do kernel. Finalmente, Cortes e Vapnik (1995) mostra que os problemas com ruídos são melhor abordados, permitindo que alguns exemplos violem a condição de margem.

1.5.5 Métricas de erro

Depois de ajustar um modelo de aprendizado de máquina, o próximo passo é avaliar a precisão de esse modelo. Antes de poder usar um modelo, é necessário saber o quão bom ele é na previsão de novos dados. Seo desempenho preditivo é muito bom, comprova-se que o modelo pode ser implantado para analisar novos dados.

1.5.5.1 Métricas para classificação

Uma forma de medir desempenho de um classificador f_e é a taxa de erro, ou seja classificações incorretas, conforme a equação 1.5.5.1, em que $I(a) = 1$ se a é verdadeiro é **0** em caso contrário. Para exemplo, um conjunto de dados que contém **n** objetos, em que serão feitas as avaliação. A taxa de erro corresponde à proporção de exemplos desse conjunto classificado de forma errada por f_e e é confirmada pela comparação da classe conhecida de x_i, y_i com a classe predita, $f_e(x_i)$ conforme equação abaixo:

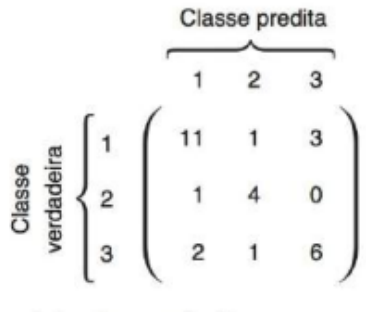
$$err(f_e) = \frac{1}{n}\sum_{i=1}^{n} I(y_i \neq f_e(x_i)) \tag{5}$$

A taxa de erro varia entre 0 e 1. O complemento dessa taxa corresponde a taxa de acerto ou acurácia do classificador:

$$ac(f_e) = 1 - err(f_e) \tag{6}$$

Para a acurácia, valores próximos de 1 são considerados melhores enquanto para o erro, os valores próximos de 0 equivalem à um modelo satisfatório. Uma outra forma de avaliação de um classificador consiste na utilização de uma matriz de confusão. Essa Matriz expõe o número de avaliação corretas e incorretas em cada classe. Da um um conjunto de dados, as linhas dessa matriz informam as classes verdadeiras enquanto as colunas como as classes preditas pelo classificador. Sendo assim, cada elemento m_{ij} de uma matriz de confusão apresenta o número de exemplos da classe i classificados como pertencentes à classe j. Nessa configuração vale ressaltar que a diagonal apresenta os acertos do classificador. Um exemplo dessa matriz pode ser observado na figura 20 (FACELI et al., 2011).

Figura 20 – Matriz de confusão com três classes



Fonte: (FACELI et al., 2011)

1.5.5.2 Métricas para regressão

Em se tratando de modelos de regressão não é factível a avaliação assertiva. É improvável que uma previsão numérica seja exatamente correta, mas pode ser perto ou longe do valor correto. Isso também é uma consequência da natureza do que significa ser um valor correto, pois geralmente medidas numéricas extraídas de uma distribuição são consideradas com um grau de incerteza conhecido como erro. Esta seção apresenta duas métricas simples para medir o desempenho da regressão: o erro médio quadrático do inglês *Mean Square error (MSE)* e distância absoluta média do ingles *Mean Absolute Distance (MAD)*. Elas são definidas conforme as equações 1.5.5.2 e 1.5.5.2.

$$MSE(f_e) = \frac{1}{n}\sum_{i=1}^{n}(y_i - f_e(x_i))^2 \tag{7}$$

$$MAD(f_e) = \frac{1}{n}\sum_{i=1}^{n}|(y_i - f_e(x_i))| \tag{8}$$

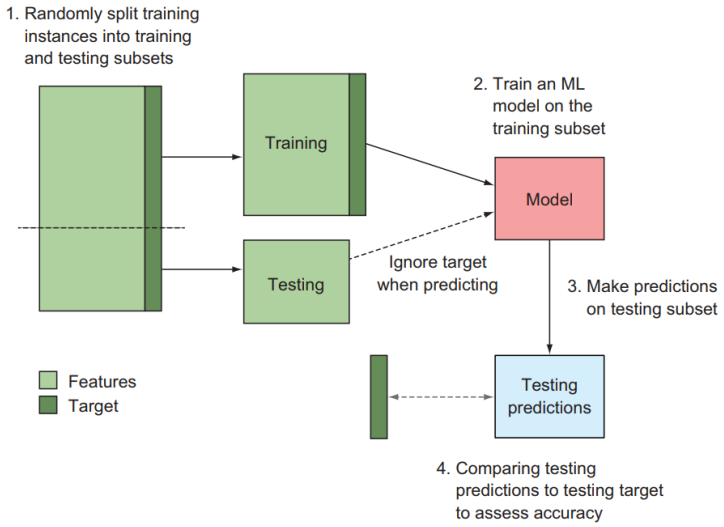
O MSE e MAD sempre assumem valores positivos. Além disso, quanto mais próximos de zero eles são, melhores são a aproximações dos rótulos verdadeiros dos objetos (FACELI et al., 2011; BRINK et al., 2017).

1.5.5.3 Métodos de amostragem - Validação cruzada

O cálculo do desempenho preditivo apenas em termos de taxa de acerto ou de erro baseado nos mesmos objetos empregados em seu treinamento, pode resultar em estimativas otimistas. Isso pode ocorrer devido às tentativas de melhora por parte do algoritmo durante a fase indutiva. Quando a taxa de erro ou acerto é obtida dessa forma de avaliação, ela recebe o nome de aparente. Para que o modelo tenha um melhor desempenho, deve-se então utilizar métodos de amostragem para que as estimativas de desempenho sejam confiáveis. (FACELI et al., 2011). A maneira mais fácil de contornar isso é usar subconjuntos separados de treinamento e teste. Apenas o subconjunto de treinamento é utilizado para se ajustar ao modelo, e apenas o subconjunto de testes para avaliar a precisão do modelo. Essa abordagem é chamada de *holdout method*, porque um subconjunto aleatório dos dados de treinamento são mantidos fora do processo de treinamento. Profissionais geralmente deixam de fora 20 a 40% dos dados como subconjunto de teste. A figura 21 mostra o fluxo algorítmico básico do *holdout method*.

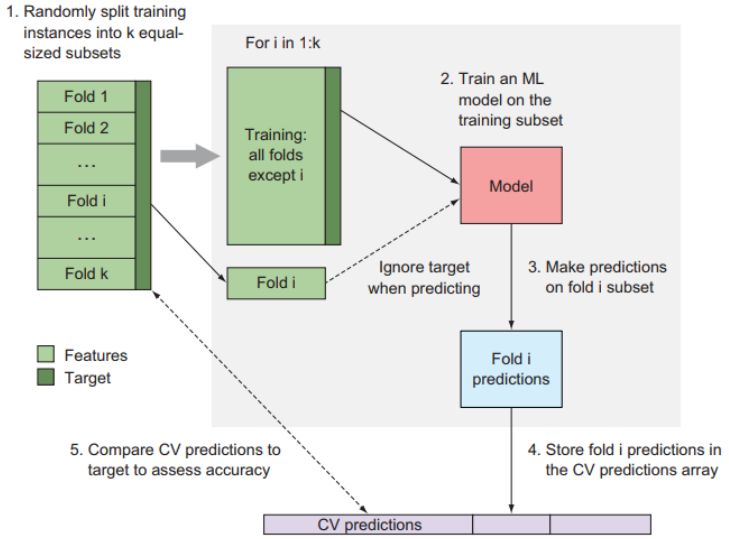
Uma abordagem melhor, porém computacionalmente mais pesada, para a validação cruzada é a validação cruzada *k-fold*. Como o *Holdout method*, a validação cruzada com *k-fold* depende da quantidade de subconjuntos dos dados de treinamento durante o processo de aprendizado. A principal diferença é que ele começa dividindo aleatoriamente os dados em r subconjuntos separados, (as opções típicas para k são 5, 10 ou 20). Para cada subconjunto, um modelo é treinado utilizando todo o *dataset* com exceção os seus dados e é subsequentemente usado para gerar previsões para o dados subconjunto não utilizado. Depois que todos os k subconjuntos são percorridas, as previsões para cada um são agregadas e comparados com a variável alvo real para avaliar a precisão. A figura 22 ilustra esse método.

Figura 21 – Fluxo do *holdout method* de validação cruzada



Fonte: (BRINK et al., 2017)

Figura 22 – fluxo do *k-fold* método de validação cruzada

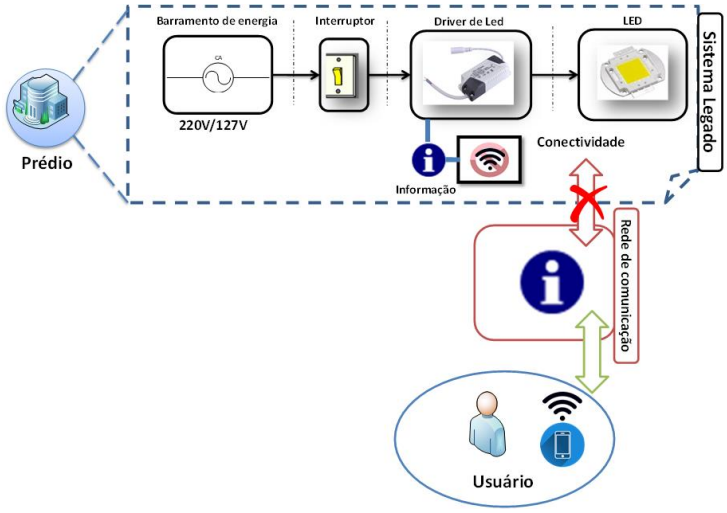


Fonte: (BRINK et al., 2017)

2 DESCRIÇÃO DO SISTEMA PROPOSTO

A maioria dos prédios atuais não possui uma infraestrutura adequada para que ocorra uma convergência para o modelo *Smart Building*. É notório que neles ainda o usuário consiga se comunicar com muitos dos dispositivos que estão instalados. Isso ocorre devido ao período de transição corrente entre as tecnologias que não possuem poder de se conectar em rede, para dispositivos totalmente integrado. A partir disso, pode ser observada na figura 23 qual é a arquitetura em se tratando de comunicação com sistemas de iluminação dos prédios atuais.

Figura 23 – Sistemas prediais atuais



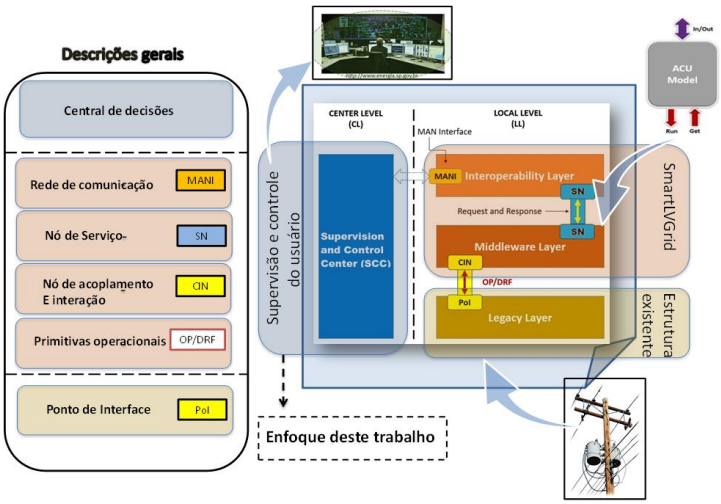
Fonte: Próprio autor

A figura 23 ilustra o modelo instalado atualmente para sistemas de iluminação predial. Nessa topologia existe uma barreira entre o usuário e os dispositivos instalados. Esse impedimento se impede ao fato de os dispositivos não possuírem conexão com a rede, além da falta de infraestrutura para comportar vários elementos conectados. Apesar da possibilidade de conexão por parte do usuários com seus dispositivos portáteis como celulares ou notebooks, ele finda impossibilitado de atuação no sistema instalado. Vale ressaltar também o termo "sistema legado"que consiste no sistema já existente, conforme citado anteriormente.

2.1 PARALELISMO: *SMART BUILDING* PROPOSTO X *SMARTLVGRID*

O paradigma *SmartLVGrid* possui como principal meio para aplicação o sistema elétrico de energia de baixa tensão. Todavia, assume em sua topologia um formato sistemático em aspectos de transição de tecnologia, bem como a intercomunicação entre os elementos do sistema e por fim a atuação do usuário nos dispositivos de forma a viabilizar um aumento de eficiência na operação, manutenção e supervisão integral. Para mais uma vez ilustrar toda essa infraestrutura disponibilizada por esse *framework*, a figura 24 pode ser observada.

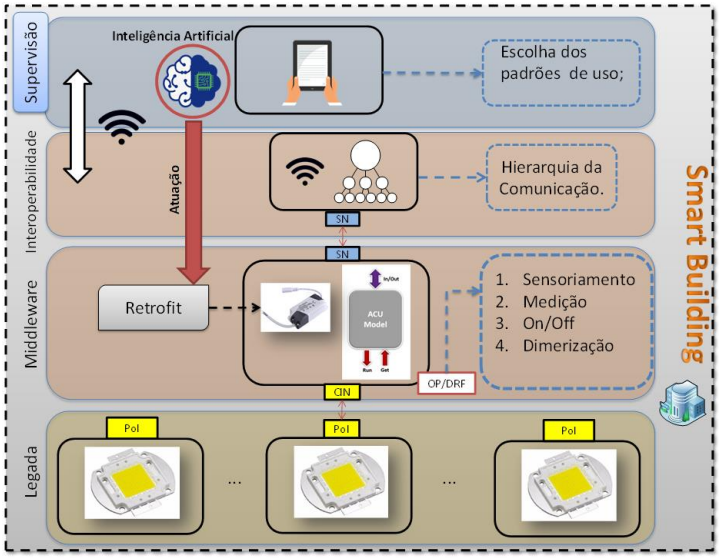
Figura 24 – Visão em camadas *SmartLVGrid*.



Fonte: Próprio autor

Na figura 24 pode ser observado a organização sistêmica supracitada. Desde a camada baixa que consiste na infraestrutura existente, passando pela implementação das camadas de *middleware* e *interoperabilidade* responsáveis pela integração e comunicação do sistema existente e por fim a camada de supervisório onde o sistema é observado e também sofre atuação por parte do usuário. Analogamente, a arquitetura *Smart Building* proposta perpassa todas essas camadas preconizadas no *framework*, iniciando na infraestrutura legada, passando pelas camadas de *middleware* e *interoperabilidade*, até chegar na camada de supervisório, conforme ilustrado na figura 25.

Figura 25 – *Smart Building* proposto.



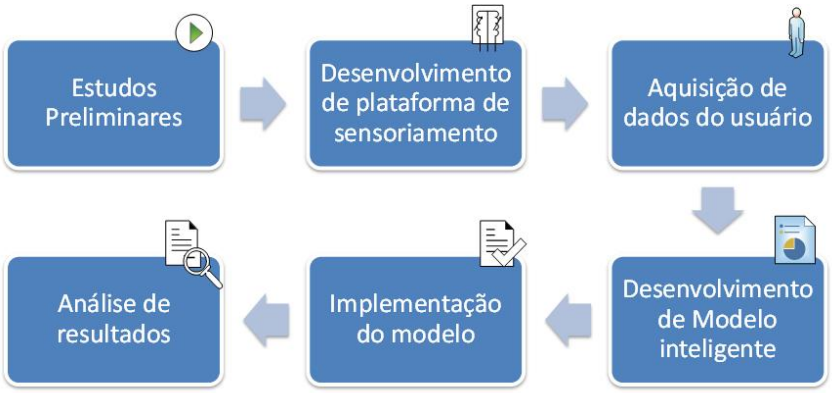
Fonte: Próprio autor

3 MATERIAIS E MÉTODOS

O *framework* SmartLVGrid traz em sua composição a camada de *Middleware* e de Interoperabilidade. Esse modelo em camadas e o conceito de *retrofit* abordado anteriormente serão utilizados como princípio base para a convergência *Smart Building*.

Partindo de um cenário em que luminárias LEDs já passaram por um processo de *retrofit*, as DRFs, ISFs, e CSFs preconizadas no *framework* serão desempenhadas nesses dispositivos. Essas ações estão descritas na tabela 1, as chamadas Primitivas Operacionais. A inteligência artificial atuará como um meio de tornar o sistema autônomo para realização dessas Primitivas Operacionais. Esse processo se dará de acordo como ilustrado por meio do diagrama de blocos na Figura 26.

Figura 26 – Caminho para condução do projeto.



Fonte: Próprio autor

Este sistema possuirá dados emulados oriundos de ambiente baseado em uma sala do Laboratório de Sistemas Embarcados, LSE, do centro de pesquisa HUB- Tecnologia e Inovação, localizado na Escola Superior de Tecnologia da Universidade do Estado do Amazonas.

Neste trabalho, será realizada uma pesquisa aplicada em meio a exploração sobre o material bibliográfico e laboratorial. Para atingir os objetivos propostos no escopo deste trabalho, a condução das atividades obedecerá a metodologia apresentada a seguir, composta dos seguintes passos:

a) consolidação de referencial teórico sobre SmartLVGrid, Sensoriamento e inteligência artificial e aplicações;

b) estudo de ferramentas e bibliotecas para elaboração e execução de projetos de Aprendizado de máquina, incluindo Python, Sci-kit Learn, Pandas, Numpy, dentre outros;

c) estudo de parâmetros que devem ser monitorados e controlados pela inteligência artificial;

d) após identificação dos parâmetros acima, elaborar esquemático e *Layout* de plataforma de sensoriamento;

e) implementações práticas;

f) emular dados padrões do usuário do sistema;

g) descrever o problema como uma tarefa de Aprendizado de Máquina, estabelecendo os atributos, tipo de tarefa, forma de avaliação e métricas de desempenho;

h) realizar o *data mining*, ou tratamento dos dados obtidos;

i) treinar os modelos com os exemplos da base de dados;

j) testar os modelos e coletar métricas de desempenho;

k) análise e conclusões dos resultados obtidos no modelo.

3.1 BIBLIOTECA *SCIKIT-LEARN*

A biblioteca *scikit-learn* é uma biblioteca de aprendizado de máquina *Open source* para a linguagem de programação *Python*. Ela inclui vários algoritmos de classificação, regressão e agrupamento incluindo máquinas de vetores de suporte, florestas aleatórias, *gradient boosting*, *k-means* e *DBSCAN*, e é projetada para interagir com as bibliotecas *Python* numéricas e científicas como *NumPy* e *SciPy* (PEDREGOSA et al., 2011).

O projeto *scikit-learn* começou como *scikits.learn*, um projeto do *Google Summer of Code* de David Cournapeau. É uma extensão de terceiros desenvolvida e distribuída separadamente para o SciPy. A base de código original foi reescrita posteriormente por outros desenvolvedores. Em 2010, Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort e Vincent Michel, todos do Instituto Francês de Pesquisa em Ciência da Computação e Automação em Rocquencourt, França, assumiram a liderança do projeto e fizeram o primeiro lançamento público em 1º de fevereiro de 2010. Dos vários *scikits*, o *scikit-learn* e o *scikit-image* foram descritos como "bem conservados e populares"em novembro de 2012. O *Scikit-learn* é uma das bibliotecas de aprendizado de máquina mais populares no GitHub. A figura 27 mostra a logomarca da biblioteca (PEDREGOSA et al., 2011).

O *Scikit-learn* vem com muitos recursos. Alguns destes recursos estão listados abaixo:

- **Algoritmos de aprendizado supervisionado**: a partir de modelos lineares generalizados (por exemplo, regressão linear), máquinas de vetores de suporte (*SVM*), árvores de decisão e métodos bayesianos - todos eles fazem parte da caixa de ferramentas do *scikit-learn*. A disseminação de algoritmos é uma das grandes razões para o alto uso do *scikit-learn*;

- **Algoritmos de aprendizado não supervisionado**: Novamente, existe uma grande variedade de algoritmos na oferta começando pelo agrupamento, análise fatorial, análise de componentes principais e redes neurais não supervisionadas;

Figura 27 – Biblioteca *Scikit-Learn*.

Fonte: (PEDREGOSA et al., 2011)

- **Validação cruzada**: contém métodos para avaliar a precisão de modelos supervisionados em dados não vistos;
- **Bases de dados experimentais**: dados para que iniciantes possam realizar experimentações. Contém vários conjuntos de dados acadêmicos (por exemplo, conjunto de dados IRIS, conjunto de dados de preços da Boston House). Tê-los disponíveis auxilia no aprendizado.

A biblioteca é construída sobre o SciPy (Scientific Python) que deve ser instalado antes da utilização usar o scikit-learn. Essa pilha que inclui:

1. *NumPy*: pacote de matriz n-dimensional base
2. *SciPy*: Biblioteca fundamental para computação científica;
3. *Matplotlib*: plotagem 2D / 3D abrangente;
4. *IPython*: console interativo aprimorado;
5. *Pandas*: Estruturas e análise de dados.

A visão para a biblioteca é um nível de robustez e suporte necessário para uso em sistemas de produção. Isso significa um foco profundo em preocupações como facilidade de uso, qualidade do código, colaboração, documentação e desempenho.(PEDREGOSA et al., 2011)

3.2 BIBLIOTECA *NUMPY*

O *NumPy* é o pacote fundamental para a computação científica em *Python*. É uma biblioteca *Python* que fornece um objeto de matriz multidimensional, vários objetos derivados (como matrizes e matrizes mascaradas) e uma variedade de rotinas para operações rápidas em matrizes, incluindo matemática, lógica, manipulação de formas, classificação, seleção, E

/ S , transformadas discretas de Fourier, álgebra linear básica, operações estatísticas básicas, simulação aleatória e muito mais (NUMPY, 2017).

No centro do pacote *NumPy*, está o objeto *ndarray*. Isso encapsula matrizes n-dimensionais de tipos de dados homogêneos, com muitas operações sendo executadas em código compilado para desempenho. Existem várias diferenças importantes entre as matrizes *NumPy* e as sequências padrão do *Python*:

- As matrizes *NumPy* têm um tamanho fixo na criação, ao contrário das listas *Python* (que podem crescer dinamicamente). Alterar o tamanho de um *ndarray* criará uma nova matriz e excluirá o original;
- Todos os elementos em uma matriz *NumPy* precisam do mesmo tipo de dados e, portanto, terão o mesmo tamanho na memória. A exceção: pode-se ter matrizes de objetos (*Python*, incluindo *NumPy*), permitindo matrizes de diferentes tamanhos de elementos;
- As matrizes *NumPy* facilitam operações matemáticas avançadas e outros tipos de operações em um grande número de dados. Normalmente, essas operações são executadas de forma mais eficiente e com menos código do que é possível usando as sequências internas do *Python*;
- Uma infinidade crescente de pacotes científicos e matemáticos baseados em *Python* está usando matrizes *NumPy*; embora normalmente suportem entrada de sequência *Python*, eles convertem essa entrada em matrizes *NumPy* antes do processamento e geralmente geram matrizes *NumPy*. Em outras palavras, para usar eficientemente grande parte (talvez até a maioria) do software científico / matemático de *Python* de hoje, apenas saber como usar os tipos de sequência internos do *Python* é insuficiente - é preciso também saber como usar as matrizes *NumPy*.

O *NumPy* suporta totalmente uma abordagem orientada a objetos, iniciando, mais uma vez, com o *ndarray*. Por exemplo, *ndarray* é uma classe, possuindo vários métodos e atributos. Muitos de seus métodos espelham funções no *NumPy namespace* mais externo, dando ao programador total liberdade para codificar de acordo com o paradigma que preferir e / ou o que parecer mais apropriado para a tarefa em questão (NUMPY, 2017).

3.3 BIBLIOTECA PANDAS

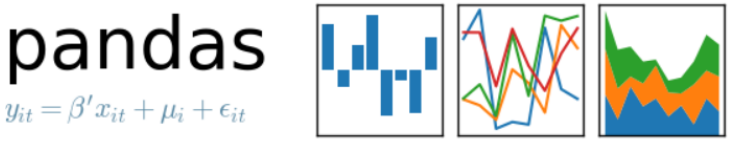
O pandas é uma biblioteca de software criada para a linguagem de programação Python para manipulação e análise de dados. Em particular, oferece estruturas e operações de dados para manipulação de tabelas numéricas e séries temporais. É um software livre lançado sob a licença BSD de três cláusulas (ARTIGO, 2019). O nome é derivado do termo "dados em painel", um termo econométrico para conjuntos de dados que incluem observações durante vários períodos

de tempo para os mesmos indivíduos. Essa biblioteca apresenta as seguintes possibilidades ao programados:

- Objeto DataFrame para manipulação de dados com indexação integrada;
- Ferramentas para ler e gravar dados entre estruturas de dados na memória e diferentes formatos de arquivo;
- Alinhamento de dados e tratamento integrado de dados ausentes;
- Remodelagem e dinamização de conjuntos de dados;
- Faturamento baseado em etiquetas, indexação sofisticada e subconjunto de grandes conjuntos de dados;
- Inserção e exclusão de colunas da estrutura de dados;
- Agrupe por mecanismo, permitindo operações de combinação de aplicação e divisão em conjuntos de dados;
- Conjunto de dados mesclando e ingressando;
- Indexação de eixos hierárquicos para trabalhar com dados de alta dimensão em uma estrutura de dados de menor dimensão;
- Funcionalidade de séries temporais: geração de intervalo de datas [4] e conversão de frequência, estatísticas de janela móvel, regressões lineares de janela móvel, mudança de data e atraso;
- Ferramentas para filtragem de dados.

A biblioteca é altamente otimizada para desempenho. É usado principalmente para Machine Learning na forma de *DataFrame*. O Pandas permite importar dados de vários formatos de arquivo, como csv, excel etc. Permite também várias operações de manipulação de dados, como agrupamento, junção, mesclagem, fusão, concatenação, além de recursos de limpeza de dados, como preenchimento, substituição ou imputação de valores nulos. A imagem que representa a biblioteca e consta no site e pode ser visualizada na figura 28 (ARTIGO, 2019).

Figura 28 – Biblioteca *Pandas*.

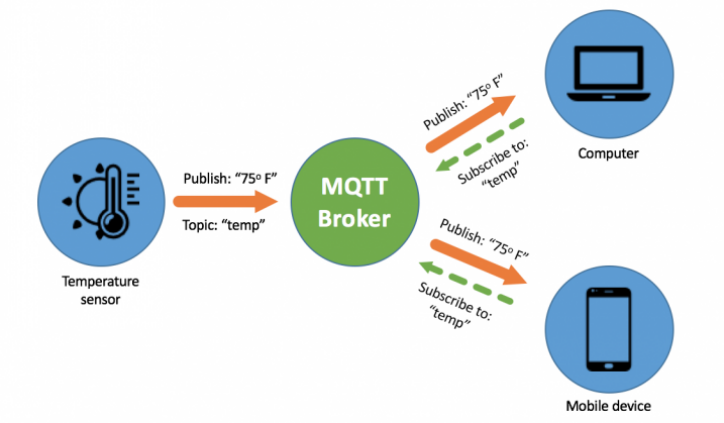


Fonte:(PANDAS, 2019)

3.4 MQTT E BIBLIOTECA PAHO-MQTT

MQTT significa *Message Queue Telemetry Transport*. É um protocolo de mensagens de publicação / assinatura, extremamente simples e leve, projetado para dispositivos restritos e redes de baixa largura de banda, alta latência ou não confiáveis. Os princípios de design são minimizar a largura de banda da rede e os requisitos de recursos do dispositivo, além de tentar garantir a confiabilidade e um certo grau de garantia de entrega. Esses princípios também tornam o protocolo ideal para o mundo emergente de "máquina para máquina" (M2M) ou "Internet das Coisas" de dispositivos conectados e para aplicativos móveis onde a largura de banda e a energia da bateria são muito importantes. A figura 29 ilustra a topologia desse protocolo de comunicação.

Figura 29 – *Message Queue Telemetry Transport* (MQTT)



Fonte: (MQTT, 2019)

A figura acima mostra a composição básica do protocolo MQTT formada por *Publisher*, *Subscriber* e *Broker*. O *broker* é responsável por receber, e disparar as mensagens recebidas dos *publishers* para os *subscribers*. O *publisher* é responsável por se ligar ao *broker* e publicar as mensagens. Já o *subscriber* é responsável por se ligar ao *broker* e receber as mensagens que ele tiver interesse.

Para implementação do MQTT, foi utilizada a biblioteca *Paho-MQTT* em python. O projeto Paho foi criado para fornecer implementações de código-fonte aberto escalonáveis de protocolos de mensagens abertos e padrão voltados para aplicativos novos, existentes e emergentes para Machine-to-Machine (M2M) e Internet of Things (IoT). A figura 30 mostra a logomarca da biblioteca (MQTT, 2019).

Esta biblioteca fornece uma classe de cliente que permite que os aplicativos se conectem a um *broker* MQTT para publicar mensagens e se inscrever em tópicos e receber mensagens publicadas. Ele também fornece algumas funções auxiliares para tornar a publicação de mensagens únicas em um servidor MQTT muito simples.

Figura 30 – Biblioteca *Paho*-MQTT

Fonte: (PAHO, 2019)

Paho reflete as restrições físicas e de custo inerentes à conectividade do dispositivo. Os objetivos incluem níveis efetivos de dissociação entre dispositivos e aplicativos, projetados para manter os mercados abertos e incentivar o rápido crescimento de aplicativos e middleware escalonáveis para Web e Enterprise. O Paho começou inicialmente com as implementações do cliente de publicação / assinatura do MQTT para uso em plataformas incorporadas e, no futuro, trará suporte ao servidor correspondente, conforme determinado pela comunidade (PAHO, 2019).

3.5 BIBLIOTECA CREME

Creme é uma biblioteca para aprendizado de máquina online, também conhecido como aprendizado incremental. O aprendizado online é um regime de aprendizado de máquina em que um modelo aprende uma observação de cada vez. Isso contrasta com o aprendizado em lote tradicional, onde todos os dados são processados de uma só vez. O aprendizado incremental é desejável quando os dados são grandes demais para caber na memória ou simplesmente quando você deseja manipular dados de *streaming*. Além de muitos algoritmos de aprendizado de máquina online, o creme fornece utilitários para extrair recursos de um fluxo de dados. A figura 31 mostra a logomarca da biblioteca

Figura 31 – Biblioteca *Creme*

Fonte: (CREME, 2019)

Essa biblioteca possui como principais características.

- Implementa vários algoritmos populares para classificação, regressão, seleção de recursos e pré-processamento de recursos;
- Possui uma API semelhante ao scikit-learn;
- E facilita a realização de aprendizado online / incremental.

3.6 ESP32 - WROOM-32

ESP32-WROOM-32 é um microcontrolador poderosos, genéricos Wi-Fi + BT + BLE módulos direcionados a uma ampla variedade de aplicações, desde redes de sensores de baixa potência até as tarefas mais exigentes, como codificação de voz, *streaming* de música e decodificação de MP3. A tabela 2 ilustra algumas características deste microcontrolador

Tabela 2 – Características do ESP32-WROOM-32.

Módulo	Esp32-Wroom-32
Core;	ESP32-D0WD
Flash SPI;	32 Mbits, 3.3V
Crystal;	40 MHz
Antena.	Antena onboard
Dimensões (mm).	(18,00)x(25.50)x(3,10)

Fonte: (ESPRESSIF, 2019)

No centro dos dois módulos está o chip ESP32-D0WD, que pertence à série ESP32 de chips. O chip incorporado é projetado para ser escalável e adaptável. Existem dois núcleos de CPU que podem ser controlados individualmente, e a frequência do clock da CPU é ajustável de 80 MHz a 240 MHz. O usuário também pode desligar a CPU e utilizar o coprocessador de baixa potência para monitorar constantemente os periféricos quanto a alterações ou cruzamento de limiares. O ESP32 integra um rico conjunto de periféricos, variando de sensores de toque capacitivo, sensores Hall, interface SD card, Ethernet, SPI de alta velocidade, UART, I²S e I²C. A figura 32 mostra o microcontrolador.

A integração do Bluetooth, Bluetooth LE e Wi-Fi garante que uma ampla variedade de aplicativos possa ser direcionada, e que o módulo é completo: o uso de Wi-Fi permite um grande alcance físico e conexão direta à Internet através de um roteador Wi-Fi, enquanto estiver usando o Bluetooth, o usuário poderá se conectar convenientemente ao telefone ou transmitir

Figura 32 – *Esp32-wroom-32*

Fonte: Próprio autor

mensagens de baixa energia para sua detecção. A corrente de suspensão do chip ESP32 é inferior a 5 µA, tornando-o adequado para aplicações eletrônicas vestíveis e alimentadas por bateria. O módulo suporta uma taxa de dados de até 150 Mbps, e 20 dBm de potência de saída na antena para garantir a maior faixa física. Como tal, o módulo oferece especificações líderes do setor e o melhor desempenho para integração eletrônica, alcance, consumo de energia, e conectividade. Essa característica de conectividade influencia de forma significativa no consumo de energia do módulo na tabela 3 podem ser observadas as condições elétricas de operação recomendada para o microcontrolador.

Tabela 3 – Condição de operação recomendada-ESP32-Wroom-32.

Símbolo	Parâmetro	Min.	Típico	Máx.
VDD3	Tensão de alimentação(V)	3,0	3,3	3,6
I_{VDD}	Corrente requerida(A)	0,5	-	-
T	Temperatura de operação (°C)	-40	-	85

Fonte: (ESPRESSIF, 2019)

3.7 SENSOR DE ILUMINAÇÃO ECP LS150P DIGITAL INFRA VERMELHO

Esse modelo de sensor detecta objetos por meio da reflexão de raios infravermelhos. O sensor, lança uma luz invisível e no momento em que o material invade seu espaço de alcance, ele estima as partículas de luz que foram redirecionadas pelo objeto que adentrou o local, desta maneira, ligando um circuito elétrico. Através do cálculo de frequência do sinal alcançado é possível verificar sua distância da luz retratada pelo objeto (ELéTRICA, 2019). O sensor utilizado

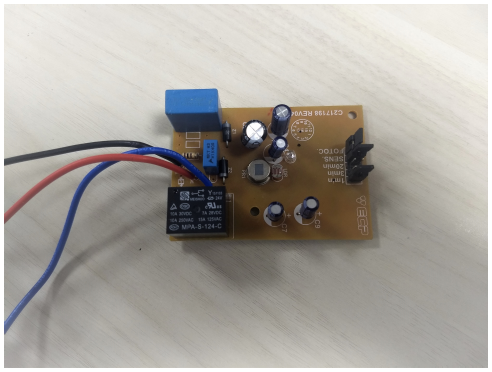
foi da marca ECP ilustrado abaixo na figura 33

Figura 33 – *Sensor de Iluminação Ecp Ls 150° Luz Automática Presença*

Fonte: (ECP, 2019)

O sensor ECP Ls150P Digital possui sensores de presença em seu interior e possui três cabos de saída necessários para sua instalação. Esses fios devem ser ligados na fase e neutro do sistema de distribuição de energia e um terceiro cabo de retorno que deve ser ligado diretamente na carga. A figura 34 mostra a composição interna do sensor.

Figura 34 – Circuito interno ao sensor *Ecp Ls150P*

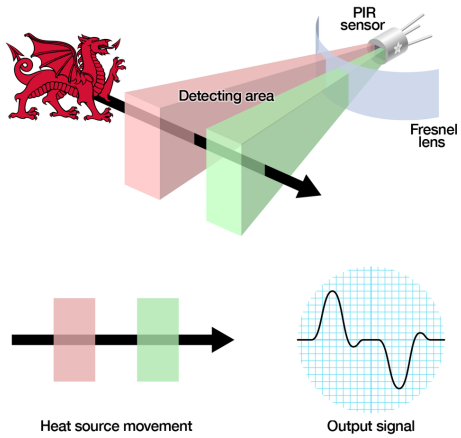


Fonte: Próprio autor

Os sensores PIR possuem um princípio de funcionamento mais elaborado do que muitos dos outros sensores, como fotocélulas e comutadores de inclinação, pois possuem várias variáveis que afetam a entrada e a saída. O seu funcionamento pode ser analisado observando a figura 35.

O próprio sensor PIR possui dois slots, cada slot é feito de um material especial que é sensível ao IR. Quando o sensor está ocioso, os dois slots detectam a mesma quantidade de IR, a quantidade ambiente irradiada da sala ou das paredes ou do exterior. Quando um corpo quente como um humano ou um animal passa, ele primeiro intercepta metade do sensor PIR, o que causa uma mudança diferencial positiva entre as duas metades. Quando o corpo quente sai da área de detecção, acontece o inverso, pelo qual o sensor gera uma alteração diferencial negativa. Esses pulsos de mudança são o que é detectado (ADAFRUIT, 2019).

Figura 35 – Funcionamento de Sensor de presença infravermelho

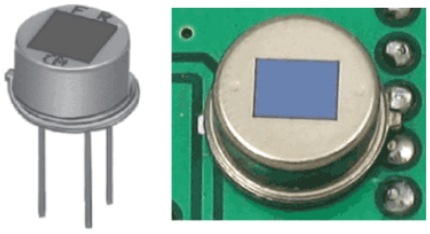


Fonte: (ADAFRUIT, 2019)

O sensor de proximidade é um sensor adequado para determinar a distância e a presença de um objeto que esteja próximo, sem que tenha contato direto. Ao detectar alguma presença ele aciona um circuito elétrico, que ativa um mecanismo, seja ele alarme sonoro, abrir uma porta entre outros.

O próprio sensor de infravermelho está alojado em uma lata de metal hermeticamente fechada para melhorar a imunidade a ruídos, temperatura e umidade localizado internamente à carcaça mostrada na figura 33 . Há uma janela feita de material transmissivo por infravermelho que protege o elemento sensor. Atrás da janela estão os dois sensores balanceados. Essas características são observadas na figura 36.

Figura 36 – *Sensor de presença infravermelho*



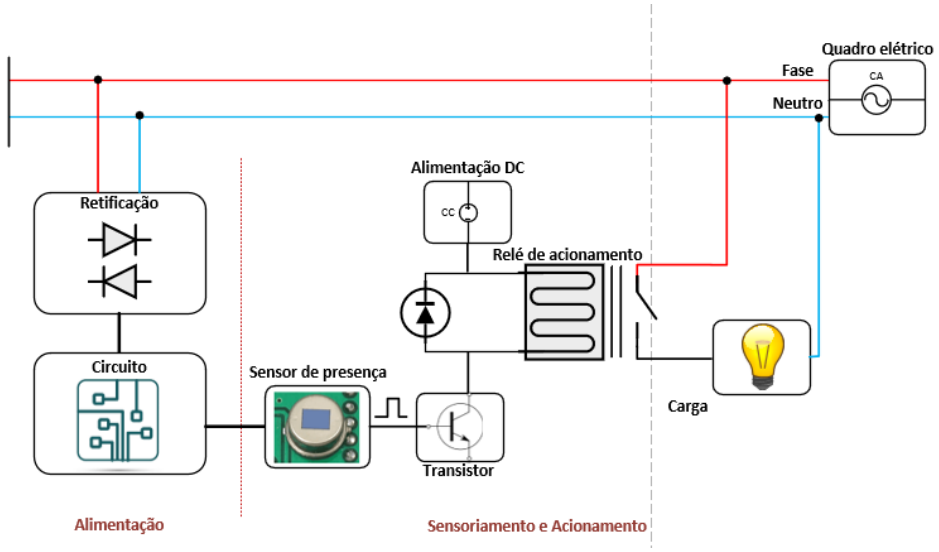
Fonte: (ADAFRUIT, 2019)

Os sensores de presença são comumente utilizados nas instalações elétricas residenciais, comerciais e prediais, principalmente para acionamento de lâmpadas ou portas, por exemplo. Sensores de presença são basicamente dispositivos capazes de detectar movimentos de determinados corpos.

Quando o sensor detecta algum movimento um relé é acionado, fechando um contato normalmente aberto, permitindo a passagem da corrente elétrica e acionando um componente, que neste caso é uma lâmpada.

Tendo isso em vista, o ECP Ls150P Digital se utiliza dessas metodologias citadas logo acima para seu funcionamento. Para ilustrar esse funcionamento, a figura 37 mostra um diagrama.

Figura 37 – Funcionamento sensor de presença infravermelho



Fonte:Próprio autor

Como ilustrado acima, o ECP Ls150P Digital possui em sua entrada um sistema de retificação que é responsável pelo fornecimento de corrente contínua para alimentação do circuito lógico. Após alimentado, o circuito é capaz de detectar a presença de pessoas ou calor através do sensor de presença infravermelho que possui internamente. Essa detecção gera um nível lógico para base de um transistor que funciona como uma chave. Após o nível lógico ser injetado na base desse transistor, o relé fecha o contato e faz com que o cabo denominado carga citado anteriormente receba alimentação da fase. De forma resumida, quando há a detecção de presença, a saída para carga recebe alimentação.

3.8 MANUFATURA PLACA DE CIRCUITO IMPRESSO

O protótipo de uma Placa de Circuito Impresso (PCI) é uma das peças e ferramentas mais importantes para a indústria da eletrônica, fabricação e montagem de componentes. O protótipo de PCI é extremamente importante desde o projeto até o produto eletrônico acabado. Ele permite o teste de um determinado projeto para verificação do funcionamento. Desta forma, pode-se depurar e corrigir qualquer erro. A produção de protótipos em PCI são fundamentais para que se evite uma produção em massa de produtos desenvolvidos com erros técnicos. Essa prática, portanto assume posição de destaque no desenvolvimento de circuitos eletrônicos. Um material fundamental para a fabricação do circuito impresso consiste de uma placa isolante que pode ser de fenolite, fibra de vidro, fibra de poliéster, filme de poliéster, filmes específicos à base de diversos polímeros, etc, que possuem a superfície com uma, duas ou mais faces, revestida por fina película de cobre, que quando fresadas, formam as trilhas condutoras que representam o

circuito onde serão soldados e interligados os componentes eletrônicos. A placa isolante utilizada foi a de fenolite conforme ilustrado na figura 38 abaixo.

Figura 38 – Placa de fenolite cobreada

Fonte:(MAKERS, 2019)

Para a fabricação dos protótipos para este trabalho, foi utilizada a prototipadora Protomat M60 da fabricante LPKF que consiste em uma unidade de fresagem e furadeira que realiza a plotagem de placa de circuito que pode ser usada para produzir protótipos de PCBs e filmes de rotogravura e para gravar alumínio ou plástico. Esse equipamento pode ser visualizado na figura 39.

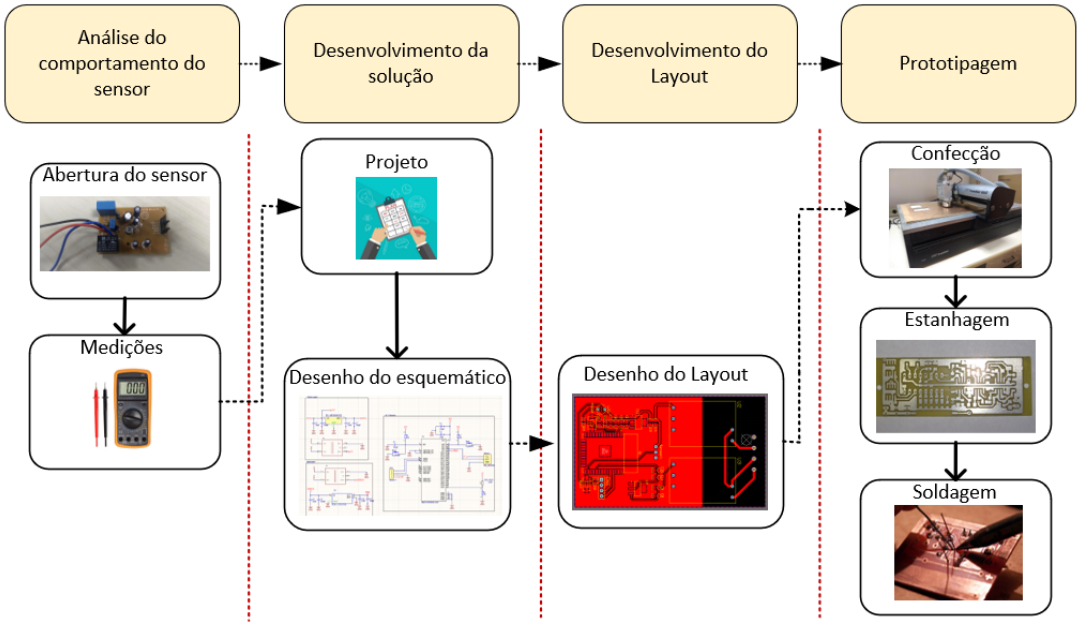
Figura 39 – Prototipadora utilizada na manufatura

Fonte: Próprio Autor

3.8.1 Processo de Manufatura

O processo de Manufatura de placas eletrônicas requer uma série de etapas. Ela passa pelo desenvolvimento do esquemático do circuito seguindo um guia de requisitos do projeto a fim de atender todos os objetivos do protótipo desejado. Após o desenvolvimento e verificação do esquemático de acordo com os critérios a serem atendidos, deve ser realizado a produção do *layout* da placa para que ela então possa ser fabricada. Após a produção do *layout*, ela então pode ir para produção, passando pelo processo de confecção e montagem. Neste trabalho o processo de manufatura para a placa passou por uma etapa a mais. Visto que o protótipo a ser desenvolvido está no contexto de *retrofit*, antes de todas essas etapas, foi necessário a análise do comportamento do sensor que receberia o *retrofit*. Tendo isso em vista, a figura 40 mostra o processo de manufatura prevendo a análise do comportamento do sensor a receber o *retrofit*.

Figura 40 – Processo da manufatura



Fonte: Próprio Autor

Como pode ser observado, o desenvolvimento do protótipo passou pelas seguintes etapas: análise do comportamento do sensor, desenvolvimento do esquemático, desenvolvimento do *layout* e a prototipagem. Vale ressaltar que a análise do sensor e prototipagem precisam passar por subetapas necessárias como a desmontagem e medição para entendimento de funcionamento e, acerca da prototipagem, é necessário confeccionar a placa, estanhar para evitar a oxidação e mau funcionamento bem como a montagem dos componentes.

3.9 FUNCIONAMENTO DO *ACU-PI*

O *Automation and Comunication Unit - Presence and Illumination* (ACU-PI) corresponde a uma unidade de automação e comunicação preconizada no *framework SmartLVGrid*. Sua composição perpassa por elementos de *hardware* como sua estrutura, componentes que o constrói e instalação do mesmo e elementos de *firmware* que faz a descrição do que será feito no *hardware* como a leitura de *Analog to digital converter* (ADC), conexão com a rede como o Wi-Fi e utilização de protocolos como MQTT (*Message Qeue Telemetry Transport*). Todo esse escopo será abordado nessa seção.

3.9.1 *Hardware* - ACU-PI

Esse dispositivo, fabricado conforme o processo supracitado, consiste em um módulo que realiza o *retrofit* do sensor de presença do ECP Ls150P Digital já instalado na infraestrutura predial existente. Ele se utiliza das saídas já existente nesse sensor para realizar a interface. As saídas desse sensor funcionam como PoI ou pontos de interface citados no *framework*. O ACU-PI é composto basicamente pelos componentes expostos na tabela 4.

Tabela 4 – Composição do ACU-PI.

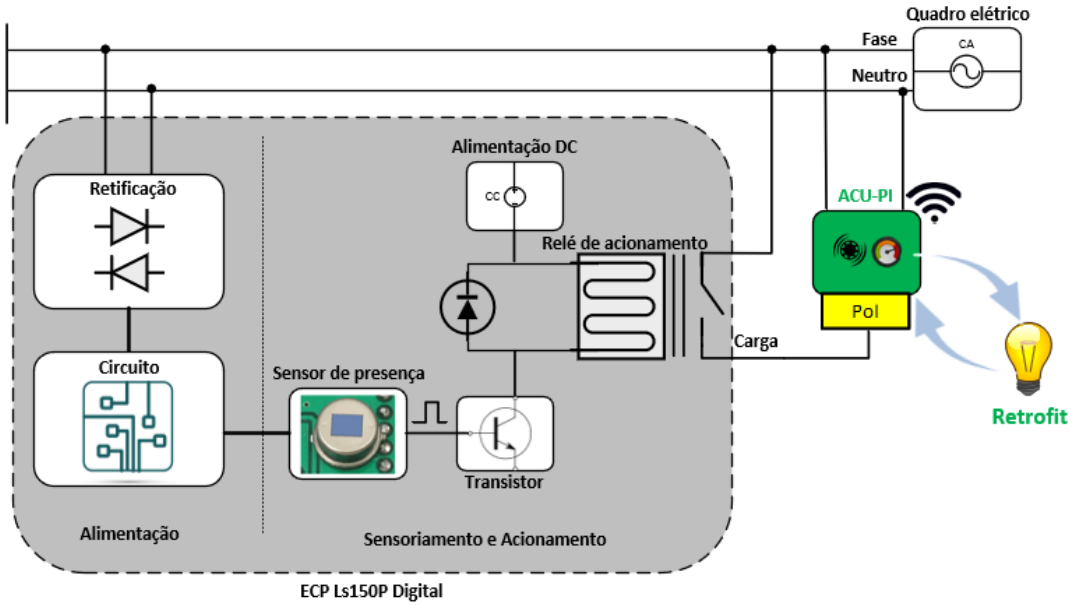
Componentes	Descrição
ESP WROOM 32	Responsável por leitura do sensor e comunicação com rede.
Fonte de alimentação	Fornecimento de energia e detecção de presença
LDR	Responsável pelo sensoriamento do nível de iluminação

Fonte: Próprio autor

Para o aproveitamento do sensor existente, verificou-se que quando havia a detecção de presença, a saída era alimentada em 127V. Com isso essa saída se torna um ponto de interface em potencial para o *retrofit*. Tendo isso em vista, o ACU-PI foi desenvolvido para ser acoplado nessa saída, que será monitorada. Caso haja 127V nela, significa que o sensor está captando presença. Sendo assim, a sua instalação pode ser observada na figura 41.

Esse dispositivo tem como principal objetivo captar os sinais de detecção de presença advindos do sensor de presença, além de realizar a medição do nível de luminosidade com o auxílio de um LDR. Logo, o ACU-PI possui função de sensoriamento desses parâmetros e disponibilização desses dados em rede. Ele se utiliza do modelo ilustrado na figura 3, onde fará uso da porta *get* para aquisição do valores desejados pelo sistema. Vale ressaltar também que ele

Figura 41 – Instalação do módulo ACU-PI



Fonte: Próprio Autor

não faz o uso da porta *run*, visto que não atua sobre o sistema. As primitivas operacionais do ACU-PI são expostas na tabela 5

Tabela 5 – Primitivas operacionais do ACU-PI.

	Classes de Primitivas Operacionais	Descrição
1.	Funções de *Retrofitting* de Domínio (DRFs)	
	1.1. Sensoriamento (Detecção);	Detecção de presença,
	1.2. Medição;	Medição do nível de luminosidade no ambiente

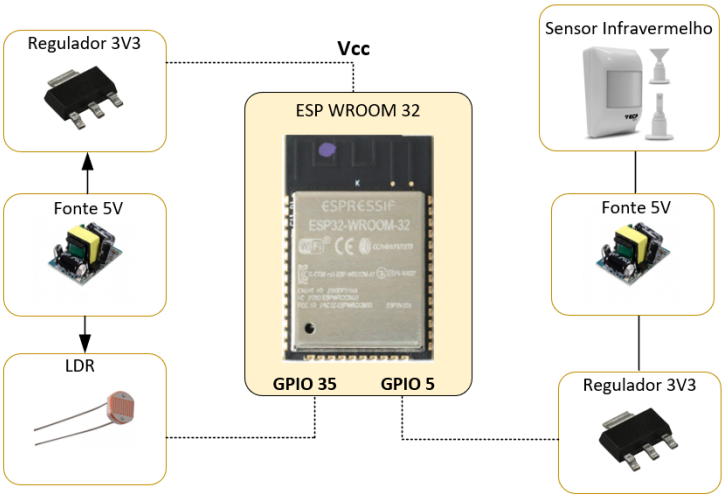
Fonte: Próprio autor

3.9.2 *Firmware* - ACU-PI

O *firmware* é um *software* usado para operar dispositivos, máquinas, equipamentos, veículos e infraestruturas. Por exemplo, o *firmware* embutido em uma máquina de lavar pode incluir algoritmos para o uso de elementos mecânicos da máquina para lavar com eficiência diferentes tipos de roupas sob diferentes condições de carga. Com o ACU-PI não é diferente. Após a manufatura do *hardware*, faz-se necessário o desenvolvimento do *firmware* que realizará o controle das ações a serem realizadas. O ACU-PI tem como principais funcionalidades as primitivas operacionais mostradas na tabela 5. Para que as primitivas operacionais sejam satisfeitas, é necessário que o microcontrolador realize as leituras dos sensores de forma correta em suas

portas. Para isso, a figura 42 ilustra como estão as ligações implementadas no *hardware* com o microcontrolador.

Figura 42 – Informações de *hardware* para o *firmware* do *ACU-PI*



Fonte: Próprio Autor

No primeiro momento, as *General Purpose In and Out* (GPIOs) que são as entradas e saídas de uso geral, devem ser escolhidas para que seja feita a programação responsável pela leitura ou atuação na mesma. Na ocasião do ACU-PI, as primitivas operacionais consistem apenas em *gets*, ou seja, nas GPIOs 35 e 5, mostradas na figura 42, serão realizadas apenas leituras dos parâmetros de presença e nível de iluminação. Os métodos referentes às primitivas implementados em C++ são expostos na figura 43.

Figura 43 – Métodos referentes às primitivas operacionais do *ACU-PI*

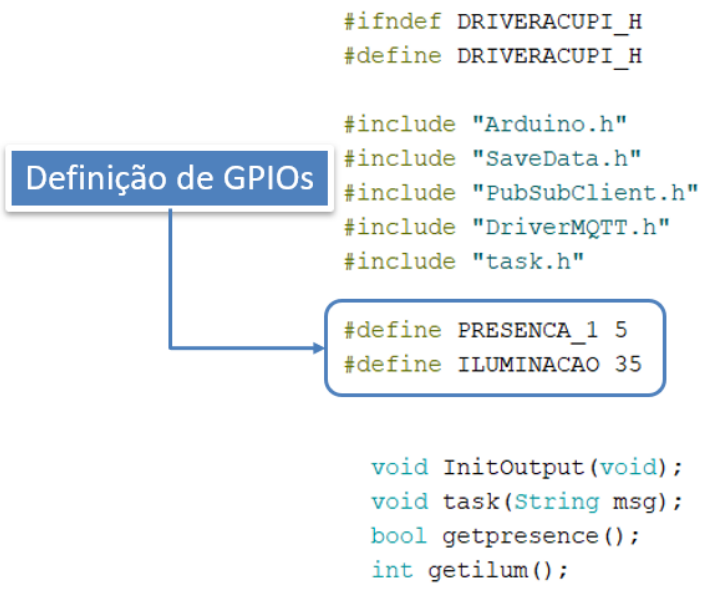
Primitivas Operacionais

```
bool getpresence()
{
  bool presence;
  if (digitalRead(PRESENCA_1)) presence = true;
  else presence = false;

#ifdef DEBUG
  Serial.print(presence);
#endif

  return presence;
}

int getilum()
{
  int ldr;
  ldr = analogRead(ILUMINACAO);
  Serial.print(ldr);
  return ldr;
}
```

Fonte: Próprio Autor

De forma resumida, os métodos realizam as leituras das GPIOs. No caso do método *getpresence()*, será feita uma leitura digital na GPIO 5, o que terá como resposta uma lógica booleana, ou seja, retorno com valor verdadeiro ou falso para verificação de presença no ambiente. Já o método *getilum()* realizará uma leitura utilizando um canal ADC de 12 bits. Isso faz com que a leitura do nível de iluminação varie entre 0 e 4096, visto que esse é o valor de fundo de escala para ADC com essa resolução. Esse fundo de escala será atingido apenas quando for gerado o valor 3,3V na GPIO 35, que é o valor máximo de tensão suportado. Na figura 44 pode ser observado que as leituras são realizadas nas variáveis *PRESENCA_1* e *ILUMINACAO*. Os valores que correspondem à essas variáveis são as GPIOs que são declaradas como macors nos arquivos de cabeçalho em conjunto com outras declarações do algoritmo.

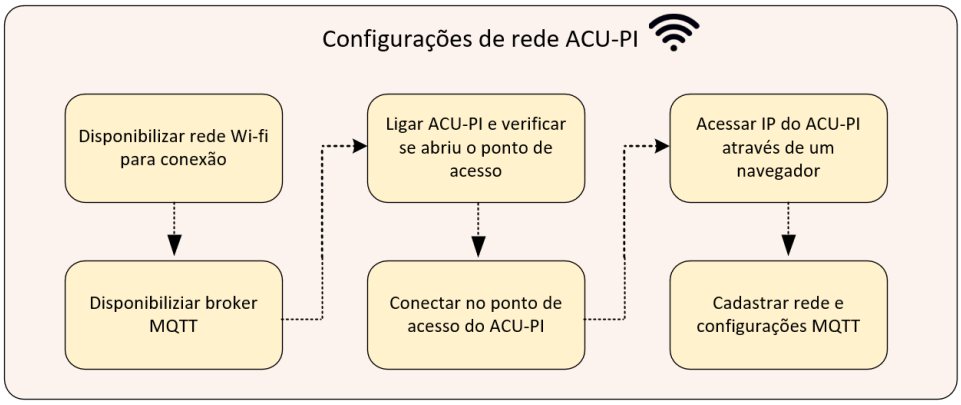
Figura 44 – Arquivo de cabeçalho - *ACU-PI*



Fonte: Próprio Autor

Na figura 44 é mostrado que o arquivo de cabeçalho de formato *header* contém os métodos citados anteriormente além de quais variáveis armazenarão os valores referentes às GPIOs. Além disso, o ACU-PI é dotado da capacidade de comunicação em rede. Quando o dispositivo é inicializado a primeira vez, é gerado um ponto de acesso do mesmo com o seu ID. Após isso, é necessária a conexão nessa rede acessando o endereço de IP disponibilizado pelo ESP 32 para cadastramento dos dados de rede e dos parâmetros para o implementação do MQTT em uma página HTML. Todo esse processo pode ser visualizado na figura 45, que exemplifica a conexão.

Figura 45 – Passos para conexão em rede e MQTT - *ACU-PI*



Fonte: Próprio Autor

Seguindo essa metodologia, será desenvolvido um Driver para conexão Wi-fi e MQTT do dispositivo ACU-PI. As etapas acima serão desenvolvidas em funções de forma separada, a fim de que a conexão em rede seja satisfeita. A figura 46 abaixo ilustra parte da estrutura em algoritmo desse desenvolvimento.

Figura 46 – Driver Wi-fi e MQTT - *ACU-PI*

```
#define SSID_AP
#define psswrd_AP
```
Definição do SSID do ponto de acesso e senha

```
void initWifi();
bool connectWifi(String ssid, String psswrd);
void openAP();
bool CheckWifiConnection();
String getSSID();
String getPASS();
```
Coleta o SSID e Senha do Wi-Fi cadastrado

```
void setNetParametters(String ssid, String psswrd, String broker, String id);
String getDevID();
String getBroker();
```
Coleta o ID e o broker a ser conectado

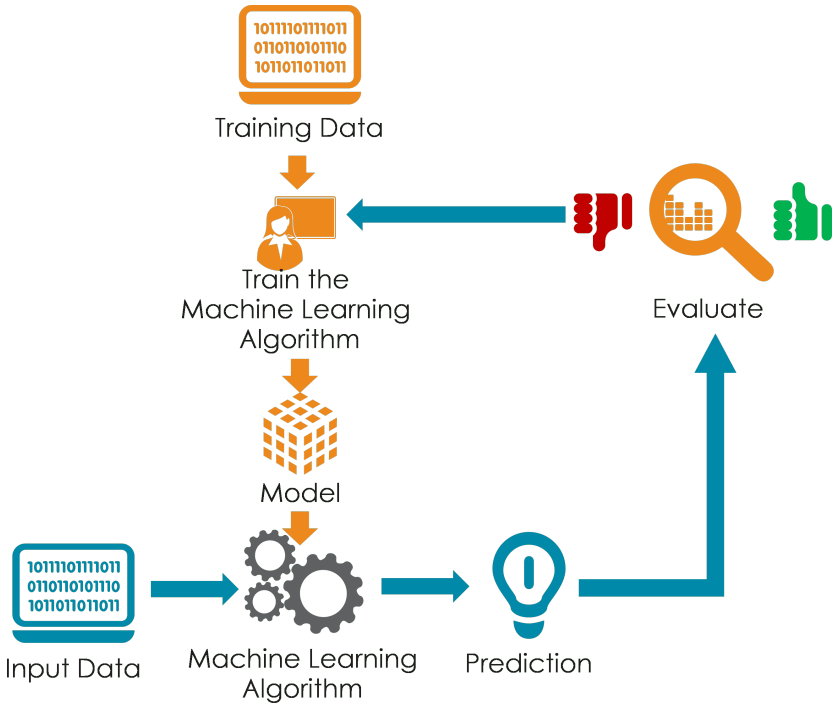
Fonte: Próprio Autor

Na figura acima, verifica-se a apresentação das funções de que caracterizam a criação do ponto de acesso do dispositivo ACU-PI, a coleta do SSID e senha do Wi-fi cadastrado além das funções de retorno do ID, que corresponde a identificação do dispositivo na rede MQTT, e o endereço de IP do *broker*.

3.10 MODELO DE INTELIGÊNCIA ARTIFICIAL

O fluxo de trabalho do desenvolvimento de um modelo de inteligência artificial passa pelas etapas de obtenção de dados de variáveis, avaliação de quais variáveis tem forte correlação com a resolução do problema e tratamento de dados, etapa de treinamento do modelo onde os dados coletados servirão como banco de dados inicial. Usualmente para que seja feita uma avaliação do treinamento, no *machine learn* tradicional uma das formas de trabalho é separar 70% dos dados para treinamento e 30% para teste e avaliação inicial do modelo. A figura 47 ilustra o processo tradicional do aprendizado de máquina.

Figura 47 – Processo tradicional de desenvolvimento de modelos de *machine learn*.



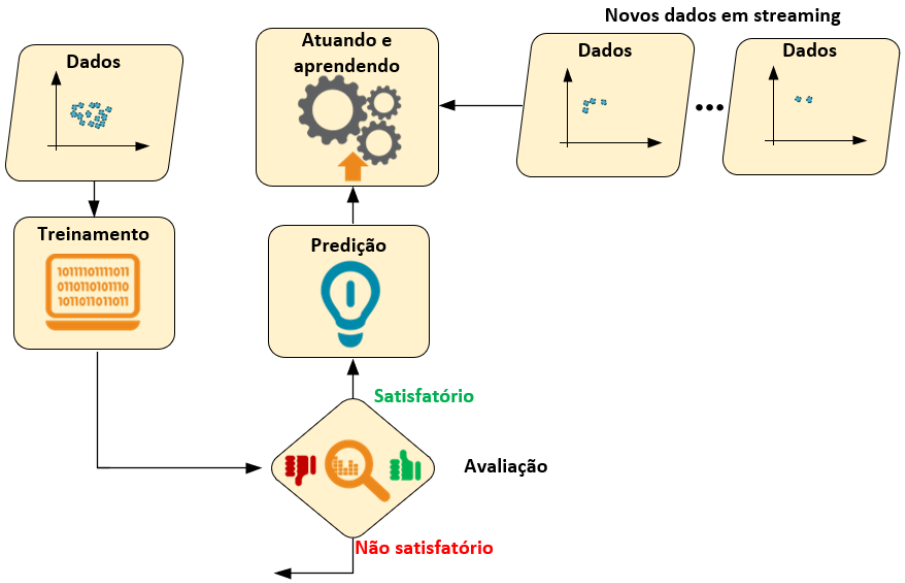
Fonte: (INTELLIPAAT, 2019)

Na figura acima são observadas as etapas supracitadas. Primeiramente, dados de treinamento passam pelo treinamento de acordo com o método de *machine learn* escolhido. Após esse treinamento é gerado um modelo que pode ser testado com uma parte do banco de dados, inicialmente alocado para teste, ou receber dados de entrada para realização das predições. Esses dados de entrada passarão pelo modelo salvo e farão as predições que serão avaliadas. Após a avaliação, verifica-se se algum hiperparâmetro referente à técnica utilizada precisa ser mudado. Além disso, o desenvolvedor avalia o tratamento de dados também conhecido como *data mining* no *dataset*, com o intuito de obter um melhor filtro ensejando o melhor desempenho do modelo.

No método tradicional, o aprendizado é feito em lote. Ou seja, um banco de dados é utilizado inteiramente para realizar os treinamentos e testes. Esse banco de dados fica estático não gerando uma dinâmica ao modelo. Ao realizar o processo de aprendizagem e chegar em

seu ponto óptimo de acordo com as avaliações do desenvolvedor, ele não passará pelo processo de aprendizagem novamente. Isso é prejudicial para casos em que há um contínuo fluxo de informações que precisam ser aprendidas. Para contornar isso, neste trabalho será utilizado o aprendizado incremental também conhecido como *online learning* ou *incremental learning*. Confome já citado no capítulo de referencial teórico, essa técnica aprende conforme as instâncias de dados vão alimentando o sistema ou após receber pequenos subgrupos para aprendizado denominados mini-lotes. A figura 48 ilustra o processo de implementação do modelo utilizando aprendizado incremental.

Figura 48 – Processo de aprendizado incremental *machine learn*.



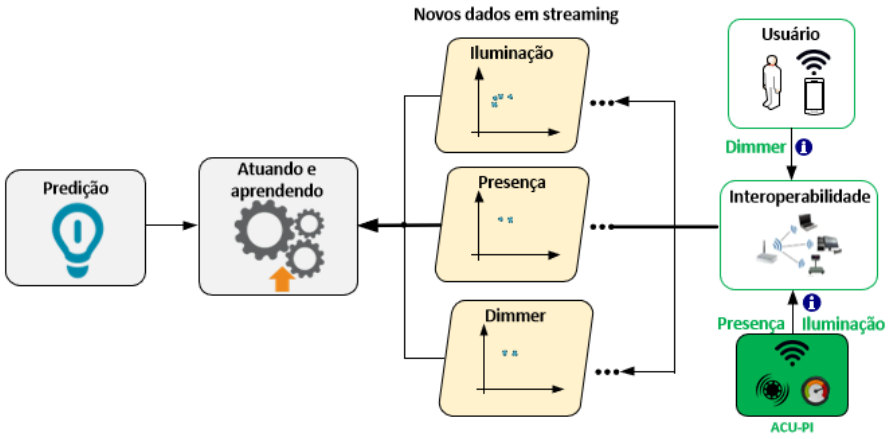
Fonte: Próprio autor

Conforme observardo na figura 48, o aprendizado incremental realiza o treinamento inicial com uma base de dados. Após isso, há a avaliação do modelo e então, com a disponibilização de novos dados através de uma *streaming* de dados, é possível que o processo de aprendizado vá incrementando a fim de que esse modelo melhore seu desempenho dinamicamente.

3.10.1 Fluxo de dados do modelo

Neste trabalho, os dados disponibilizados pelo ACU-PI de iluminação e presença estão chegando constantemente ao modelo. Visto que o objetivo é realizar o aprendizado do padrão do usuário, o nível de iluminação configurado por ele chegará ao modelo também. A camada de interoperabilidade será a responsável por encaminhar esses dados. A figura 49 abaixo já mostra o processo de aprendizado levando em consideração a aplicação em questão.

Figura 49 – Processo de aprendizado incremental para o controle de iluminação.



Fonte: Próprio autor

Por meio da figura 49 é possível verificar que as informações disponibilizadas pelo ACU-PI de iluminação e presença, assim como as informações de dimmerização, advindas do usuário passam pela camada de interoperabilidade até que cheguem ao modelo. Essa troca de informações é feita via protocolo MQTT, como já descrito na seção **"*Firmware* - ACU-PI"**. No lado da máquina que irá conter o modelo, foi desenvolvido um serviço para a gravação dessas informações em um arquivo CSV (*Comma Separated Values*) que consiste em um arquivo com valores separados por vírgula. Essas informações são salvas por meio do serviço ilustrado na figura 50 abaixo, implementado na linguagem *Python 3.7*.

Figura 50 – Serviço para escrita de informação em arquivo CSV.

```
#Abre o arquivo csv e realiza um append da última linha dado recebida
    with open('Dataset.csv', 'a') as csvFile:
        nome_chave = ['Hour', 'Presence', 'Ligthing', 'Dimmer2','Dimmer3']
        writer = csv.DictWriter(csvFile,fieldnames=nome_chave)
        writer.writerow(data_input)
    csvFile.close()

dataset = pd.read_csv('Dataset.csv')
```

Fonte: Próprio autor

Esse formato pode ser visualizado ao abrir o arquivo no formato ".CSV"em um bloco de notas ou aplicativos de visualização de planilhas. Para a visualização desses dados via *Python*, foi utilizada a biblioteca Pandas conforme citado anteriormente. No momento do manuseio e tratamento dos dados, as bibliotecas Numpy e Pandas tornam-se imprescindíveis. O processo consiste na visualização dos dados através de *Dataframes*, que são objetos do Pandas. Quando há inconsistência dos dados, é feito uma manipulação dos mesmos através do Numpy: retirada de linha com um parâmetro faltante, ajustes nas variáveis dentre outras coisas. Por meio dessas operações, é possível deixar o *dataset* inicial compatível com o esperado para que assim seja

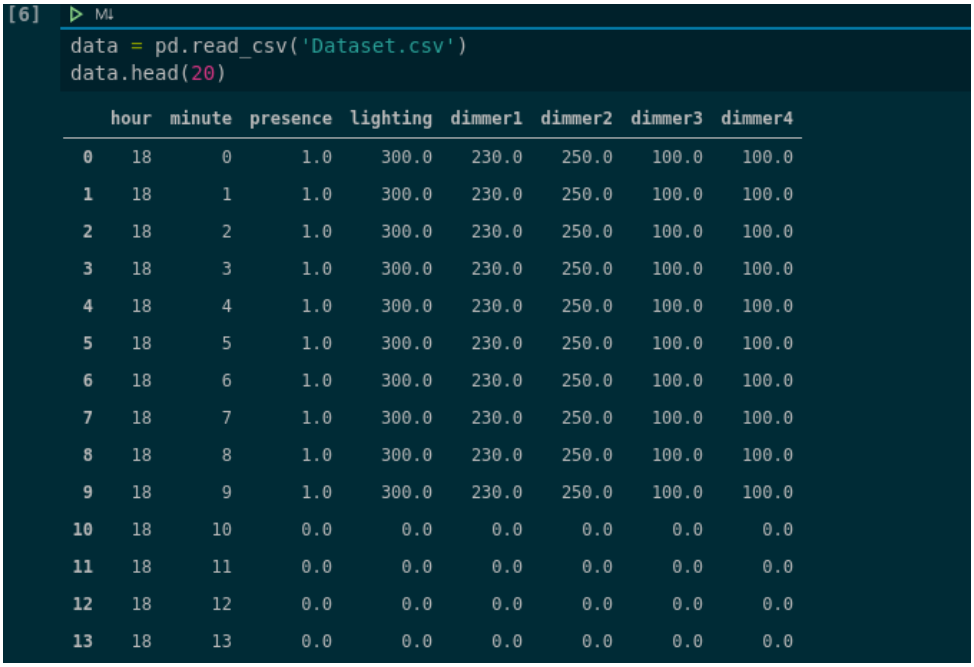
iniciado o treinamento e teste do modelo. Para a ilustração disso, a figura 51 mostra visualização das informações quando aberto em um bloco de notas e na figura 52 é ilustrado quando aberto em um Dataframe da biblioteca Pandas em *Python*.

Figura 51 – Visualização dos dados no bloco de notas.

```
hour,minute,presence,lighting,dimmer1, dimmer2, dimmer3, dimmer4
18,00,1,300,230,250,100,100
18,01,1,300,230,250,100,100
18,02,1,300,230,250,100,100
18,03,1,300,230,250,100,100
18,04,1,300,230,250,100,100
18,05,1,300,230,250,100,100
18,06,1,300,230,250,100,100
18,07,1,300,230,250,100,100
18,08,1,300,230,250,100,100
18,09,1,300,230,250,100,100
18,10,0,0,0,0,0,0
18,11,0,0,0,0,0,0
18,12,0,0,0,0,0,0
18,13,0,0,0,0,0,0
18,14,0,0,0,0,0,0
18,15,0,0,0,0,0,0
18,16,0,0,0,0,0,0
18,17,0,0,0,0,0,0
18,18,0,0,0,0,0,0
18,19,0,0,0,0,0,0
18,20,0,0,0,0,0,0
18,21,0,0,0,0,0,0
18,22,0,0,0,0,0,0
```

Fonte: Próprio autor

Figura 52 – Visualização dos dados no Pandas.



```
[6]
data = pd.read_csv('Dataset.csv')
data.head(20)
```

	hour	minute	presence	lighting	dimmer1	dimmer2	dimmer3	dimmer4
0	18	0	1.0	300.0	230.0	250.0	100.0	100.0
1	18	1	1.0	300.0	230.0	250.0	100.0	100.0
2	18	2	1.0	300.0	230.0	250.0	100.0	100.0
3	18	3	1.0	300.0	230.0	250.0	100.0	100.0
4	18	4	1.0	300.0	230.0	250.0	100.0	100.0
5	18	5	1.0	300.0	230.0	250.0	100.0	100.0
6	18	6	1.0	300.0	230.0	250.0	100.0	100.0
7	18	7	1.0	300.0	230.0	250.0	100.0	100.0
8	18	8	1.0	300.0	230.0	250.0	100.0	100.0
9	18	9	1.0	300.0	230.0	250.0	100.0	100.0
10	18	10	0.0	0.0	0.0	0.0	0.0	0.0
11	18	11	0.0	0.0	0.0	0.0	0.0	0.0
12	18	12	0.0	0.0	0.0	0.0	0.0	0.0
13	18	13	0.0	0.0	0.0	0.0	0.0	0.0

Fonte: Próprio autor

Com esses dados disponibilizados ao modelo de forma contínua, é possível realizar o treinamento do modelo constantemente.

3.10.2 Método para avaliação do modelo

O modelo de inteligência artificial possui como saída a regulação do nível de iluminação a ser enviado para as luminárias. Esse nível deve ser de acordo com o padrão aprendido do usuário. Pelo fato de que a interface entre o usuário e o sistema está em desenvolvimento, é possível apenas simular o padrão de uso do mesmo. A partir disso, foi realizada a simulação do gosto do usuário de acordo com o horário do dia que ele está usando o ambiente em questão.

Para o controle de iluminação, o usuário simulado pode escolher o nível de iluminação na faixa de valores entre 0 e 255, conforme já citado anteriormente. Acerca do padrão do usuário, ele utiliza três faixas de iluminação que correspondem à baixa, média e alta iluminação. Essas faixas foram definidas para que o comportamento seja de fato padronizado e permita o aprendizado por parte do modelo e é mostrada na tabela 6.

Tabela 6 – Faixas de iluminação.

Faixa de iluminação	Nível correspondente à faixa
Iluminação baixa	0 - 85
Iluminação média	86 - 170
Iluminação alta	171 - 255

Fonte: Próprio autor

As faixas mostradas na tabela 6 são fundamentais para o estabelecimento do padrão. O usuário simulado tem como preferência a iluminação baixa ou média pela parte da manhã, após o almoço ele usualmente utiliza baixa ou média, pela parte da tarde ele prefere valores médios para iluminação e, por fim, a noite ele usa a iluminação em valores correspondentes à níveis altos. A tabela 7 mostra o padrão utilizado por ele de acordo com o período do dia.

Tabela 7 – Faixas de iluminação.

Período do dia	Faixa de iluminação	Nível correspondente à faixa
Manhã	Baixa/Média	0 - 170
Tarde	Baixa/Média	0 - 170
Noite	Alta	171 - 255

Fonte: Próprio autor

Vale ressaltar que pela parte da manhã, apesar de possuir duas faixas de iluminação,

o usuário utiliza com mais frequência valores correspondentes ao nível baixo de iluminação, além de que pela parte da tarde a maior frequência é o nível médio de iluminação. Foram realizas as simulações correspondente à sete dias de uso de acordo com o citado acima. Os dados disponibilizados ao modelo para a simulação foram: hora, minuto, presença, onde a variável objetivo foi o nível de iluminação. A figura 53 mostra o arquivo CSV enviado para a simulação.

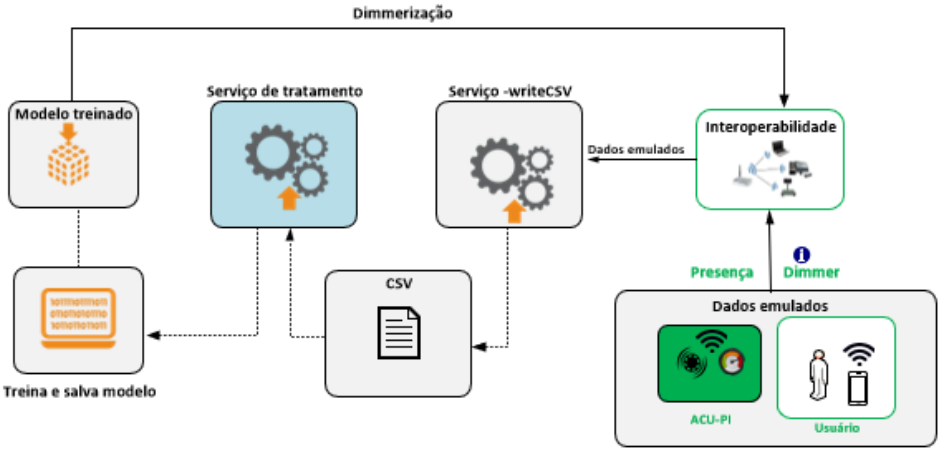
Figura 53 – Dados disponibilizados ao modelo.

```
Hora,Minuto,Presença,Dimmer1
00,00,0,0
00,01,0,0
00,02,0,0
00,03,0,0
00,04,0,0
00,05,0,0
00,06,0,0
00,07,0,0
00,08,0,0
00,09,0,0
00,10,0,0
```

Fonte: Próprio autor

O sistema emulado possui um fluxo de informações que permite a avaliação de fato do modelo. Esse fluxo de informações pode ser visto na figura 54.

Figura 54 – Simulação do sistema.



emulado$_2$.*png*

Fonte: Próprio autor

Dessa forma o sistema utiliza os dados emulados, todavia possui como saídas os valores de iluminação conforme proposto.

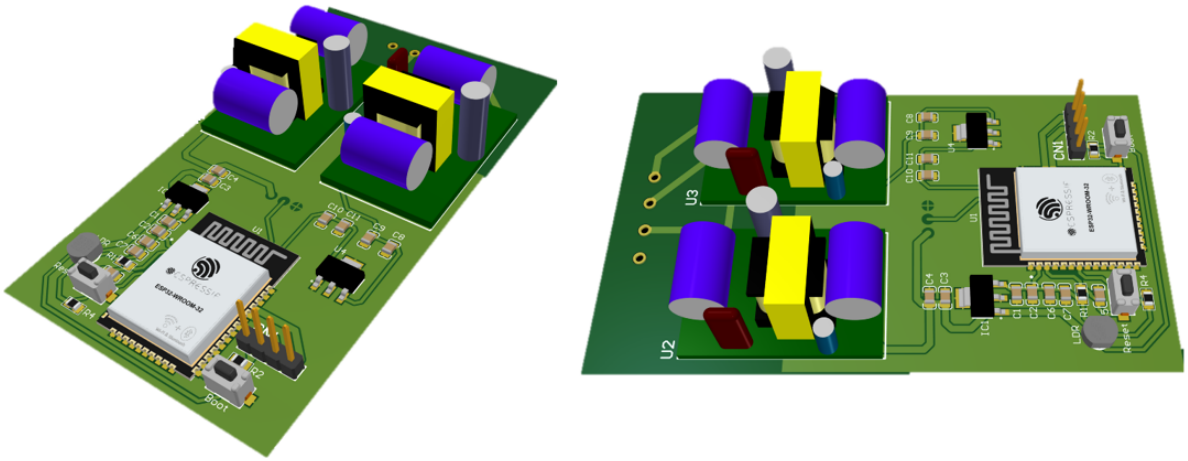
4 RESULTADOS E DISCUSSÕES

A proposta do *framework SmartLVGrid*, que abarca conceitos de *retrofit* e comunicação sistemática, serviu de base para essa implementação. A partir desse fundamento, foi implementado uma plataforma de *hardware* para o sensor de presença convencional baseado no conceito de *retrofit*. O *firmware* desse dispositivo que ensejou a aquisição dos dados provenientes do sensor bem como de um sensor de iluminação nele inserido. E, por fim, a implementação de um modelo de inteligência artificial capaz de realizar o aprendizado das preferências do usuário na hora da escolha do nível de luminosidade do seu ambiente. Essa inteligência atua diretamente em luminárias LED que de antemão já haviam passado pelo processo de *retrofit*. Tendo isso em vista, esse capítulo exporá os resultados obtidos conforme expostos no capítulo **Objetivos**.

4.1 ACU-PI - PLATAFORMA DE SENSORIAMENTO

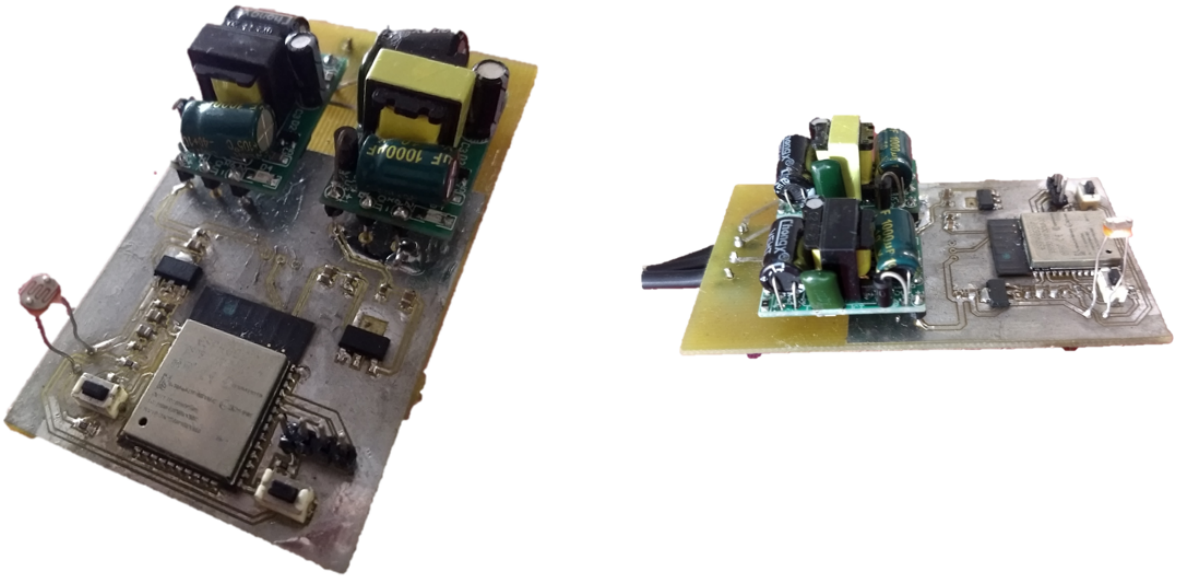
Após a realização projeto proposto a fim de elaborar a plataforma de sensoriamento para favorecimento da convergência *Smart Building*, foi realizada a confeção da PCI para realização do *retrofit*, conforme preconizado no *SmartLVGrid*. O valor de tensão na entrada do ESP para a leitura do LDR deve ter como fundo de escala 3,3V. Visto que a resistência do LDR diminui com a incidência de luz, o resistor em série foi calculado de forma que a queda de tensão nele não ultrapasse 3,3V para que a porta do esp não queime. Para isso, foi realizada a medição do LDR quando havia uma incidência de luz intensa. O valor da resistência encontrada foi em torno de 1 kOhm. Em posse disso foi feita a escolha da resistência em série com LDR. Em adição, *headers* foram externados para que a comunicação serial fosse feita atravé de um conversor USB-SERIAL. A figura 55 mostra os resultados do projeto em uma visualização em três dimensões de duas perspectivas e, em seguida, na figura 56 é exposta uma foto com a placa devidamente montada.

Figura 55 – Visualização em três dimensões da plataforma de sensoriamento.



Fonte: Próprio autor

Figura 56 – Plataforma de sensoriamento montada.



Fonte: Próprio autor

Ao observar as figuras 55 e 56, percebe-se que os componentes expostos na metodologia, apresentados na tabela 5, estão presentes caracterizando a montagem de acordo com a proposta. Para que a plataforma de sensoriamento atingisse o seu objetivo de disponibilizar os dados de presença e iluminação ambiente para a aplicação de inteligência artificial, fez-se necessário a conexão desses dados em rede. Isso porque a comunicação com a camada de interoperabilidade é primordial para a garantia da convergência *Smart Building* através da *framework* em questão. Para isso foi utilizado o protocolo de comunicação MQTT. A figura 45 mostrou a metodologia utilizada na implementação dos algoritmos de *firmware* para realizar essa conexão. Como

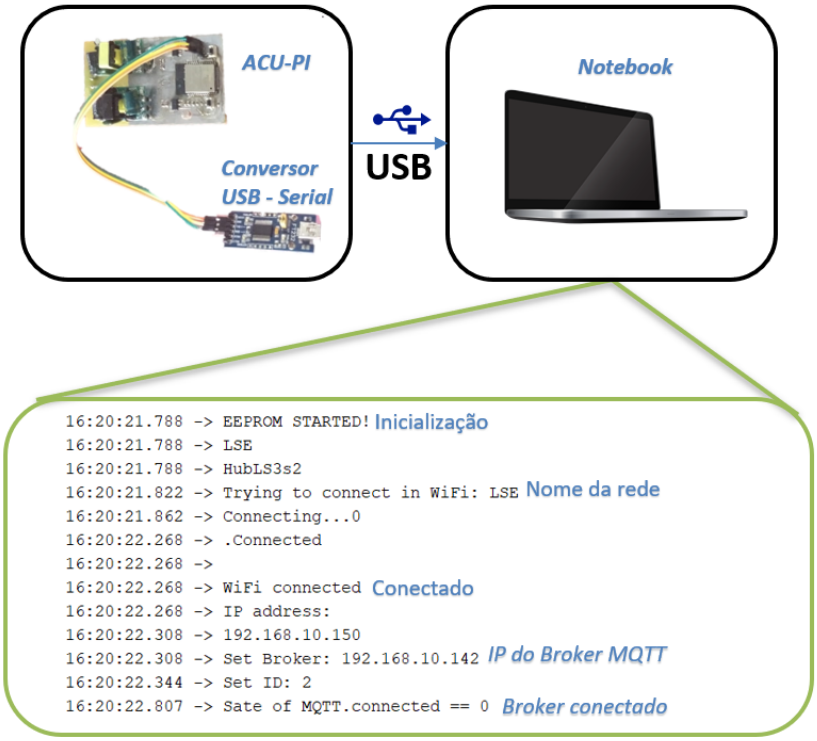
resultado da implementação de *firmware* dessa plataforma de sensoriamento, a figura 57 ilustra a forma em que o dispositivo pode ser cadastrado na rede e iniciar sua comunicação via MQTT.

Figura 57 – Resultado do processo de cadastro de rede implementado no *firmware*.

Fonte: Próprio autor

Como verifica-se na figura acima, o desenvolvimento do *firmware* gerou um ponto de acesso que deve ser conectado para que comunicação seja feita. Após o acesso, foi gerada uma página para cadastro dos parâmetros já citados. A figura 58 mostra quando a conexão com a rede e o *broker* foi realizada corretamente.

Figura 58 – Resultado da conexão à rede e ao MQTT



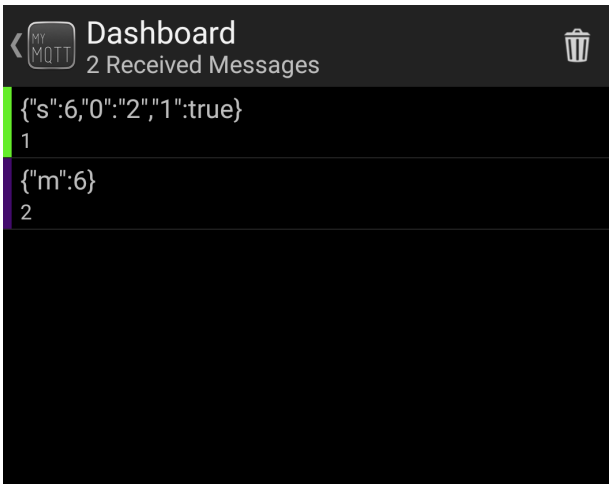
Fonte: Próprio autor

É possível notar na figura acima que o objetivo de conexão com a rede por parte do ACU-PI foi atingido através dos *logs* provenientes do dispositivo através de um barramento serial. No *log* é possível verificar a confirmação de conexão com a rede *Wi-fi* e a rede MQTT.

4.1.1 Verificação das Primitivas operacionais do ACU-PI.

Após a implementação do *hardware* e *firmware* do dispositivo ACU-PI, foi feita a verificação do das primitivas operacionais do dispositivo. As primitivas operacionais do ACU-PI, já mostradas na tabela 5, utilizam apenas a porta *get* de acordo com o preconizado no *framework*. No caso do ACU-PI, é feita a aquisição do sensoriamento de presença e do nível de iluminação no ambiente. A figura 59 mostra a resposta do sensor através do protocolo MQTT. A comunicação foi feita por meio do aplicativo *MyMQTT* que realiza o envio e a recepção da mensagem em determinado tópico.

Figura 59 – ACU-PI - DRFs - Presença.

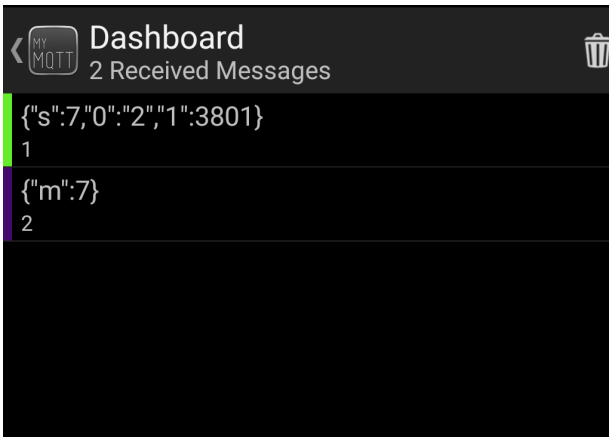


Fonte: Próprio autor

A figura 59 mostra a resposta do dispositivo à uma mensagem de requisição da informação de presença. Os valores permitidos para essa primitiva operacional são boolianos e correspondem a detecção de presença do usuário ou ausência dele respectivamente.

Em complemento, a figura 60 mostra a resposta à mensagem de requisição da informação de nível de luminosidade no ambiente. A faixa de valor permitido vai de 0 à 4096, visto que o ADC do microcontolador é de 12 bits.

Figura 60 – ACU-PI - DRFs - Iluminação.



Fonte: Próprio autor

Como pode ser notado na figura 60 os níveis de iluminação obtidos foram obtidos como baixos níveis de iluminação, o que corresponde a baixos valores da faixa de 0 a 4096, e altos níveis de iluminação, que correspondem a valores próximos ao fundo de escala dessa faixa.

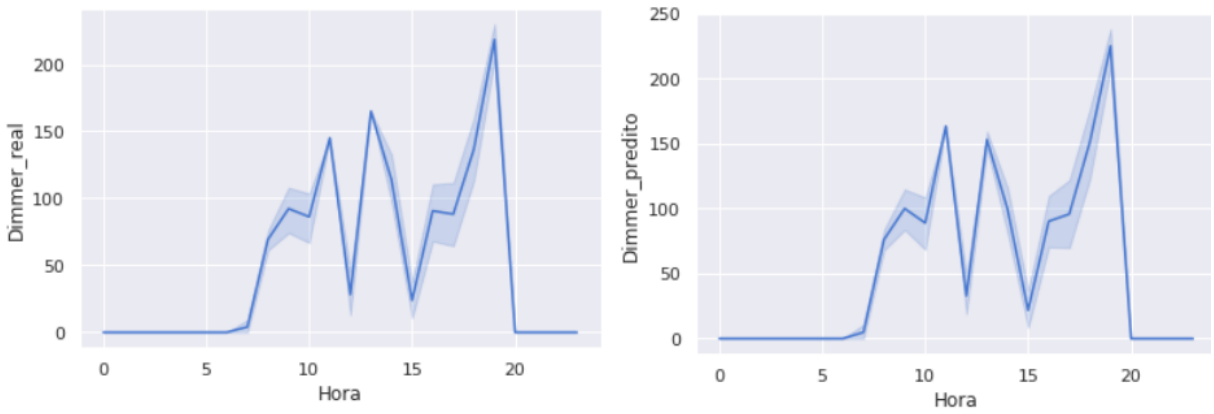
Com isso, ao analisar os resultados das primitivas operacionais, que correspondem a aquisição de dados de presença ou ausência do usuário, bem como o nível de iluminação do

local onde a plataforma de sensoriamento está instalada, verificou-se que o dispositivo ACU-PI atingiu o objetivo.

4.2 ANÁLISE DO MODELO DE INTELIGÊNCIA ARTIFICIAL

O modelo de inteligência artificial possuiu vários estágios de treinamento. Como citado anteriormente, foram simulados sete dias de padrão de uso do nível de iluminação. Os treinamentos foram realizados utilizando o a técnina de regressão por meio do K-NN, que realiza a comparação dos dados próximos para a realização da predição. Esse treinamento foi realizado ao fim de cada dia, para que a predição do próximo dia fosse realizada. A figura 61 mostra um gráfico que ilustra como foi a predição do segundo dia após o treinamento com os dados do primeiro dia.

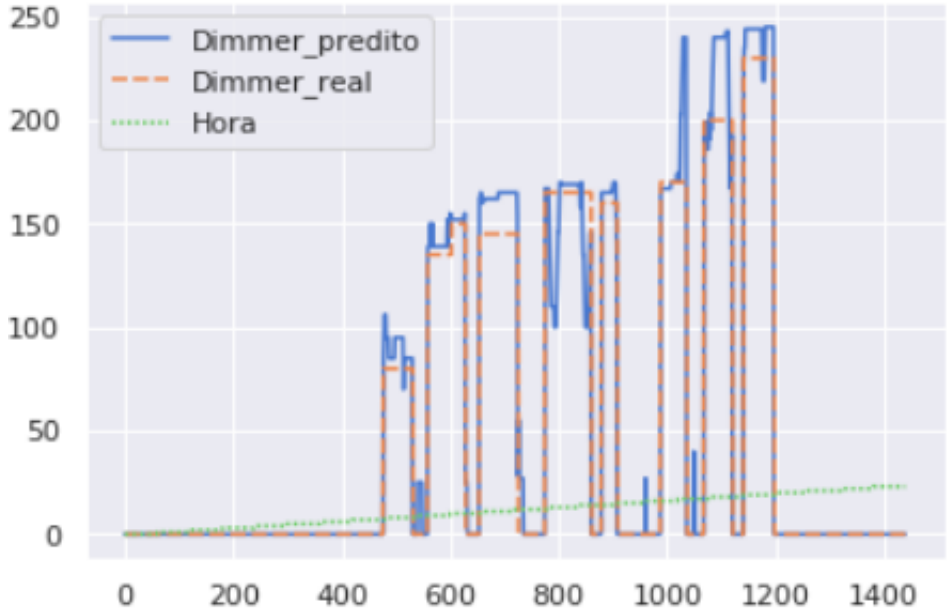
Figura 61 – Valores dimmerizados e preditos para o segundo dia



Fonte: Próprio autor

A figura 61 mostra os dados dos valores reais no *dataset* do segundo dia, em comparação com os valores preditos nesse mesmo dia. Percebe-se que com apenas um dia para o treinamento, o desempenho ainda é bem aquém do desejado. Para a visualização de como a predição está longe do desejado, a figura 62 pode ser observada.

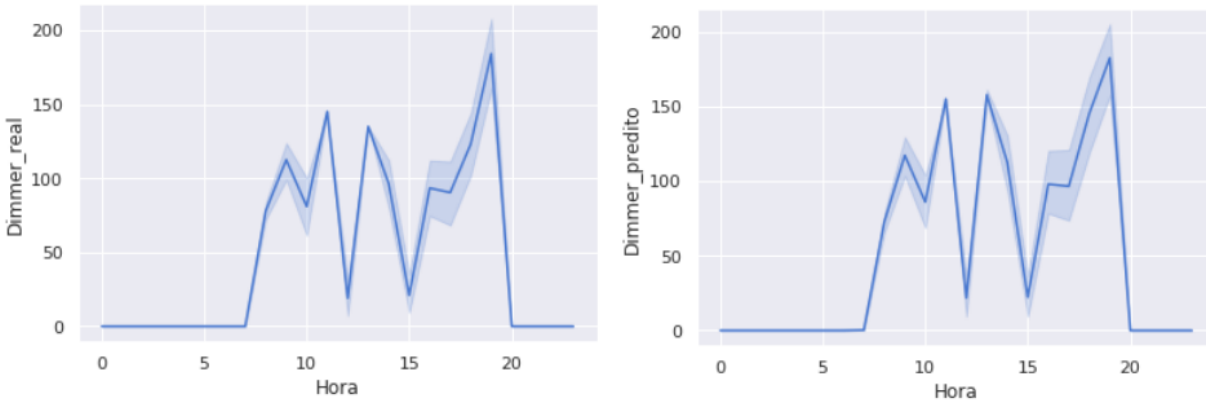
Figura 62 – Valores dimmerizados e preditos para o segundo dia



Fonte: Próprio autor

Na figura 62 pode ser visto que com o acréscimo de horas o nível de dimmerização real por parte do usuário aumenta por seguir o padrão já mencionado e em muitos pontos o modelo não consegue predizer de forma satisfatória. Para superar esse problema, foram realizados treinamentos ao final de cada dia para a melhoria e o aprendizado contínuo do modelo. A figura 63 mostra a predição e o valor real do terceiro dia com dados acumulados para treinamento do primeiro e do segundo dia.

Figura 63 – Valores dimmerizados e preditos para o terceiro dia

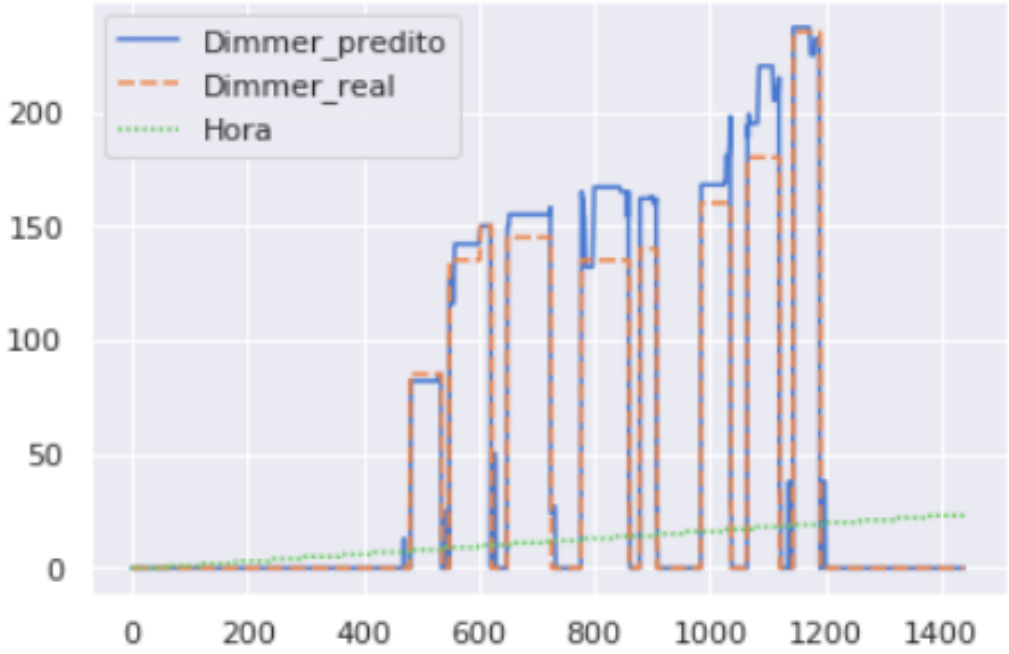


Fonte: Próprio autor

Verifica-se uma melhoria no desempenho da predição em relação ao dia anterior, pois o

modelo dessa vez dispõe de mais dados para o aprendizado. Vale ressaltar que o acúmulo dessas informações para aprendizado caracteriza o aprendizado incremental. A figura 64 mostra em um mesmo gráfico o comportamento da predição em relação ao valor real.

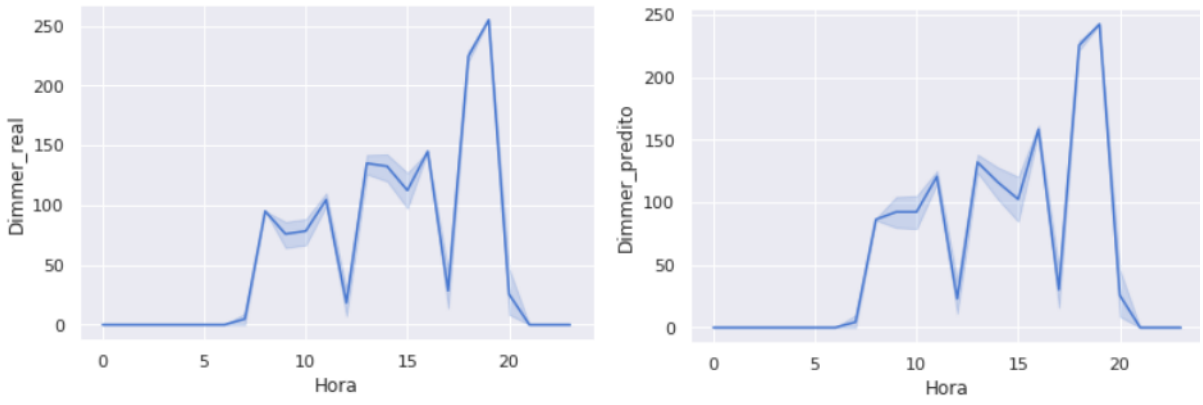
Figura 64 – Comparação dos valores reais e preditos para o terceiro dia



Fonte: Próprio autor

Por fim, com o acréscimo dos dados dos seis dias, foi realizado o treinamento para a avaliação do desempenho do modelo para predição do sétimo dia. A figura 65 como as anteriores mostra lado a lado o desempenho da predição e valor real que o usuário colocaria na luminária.

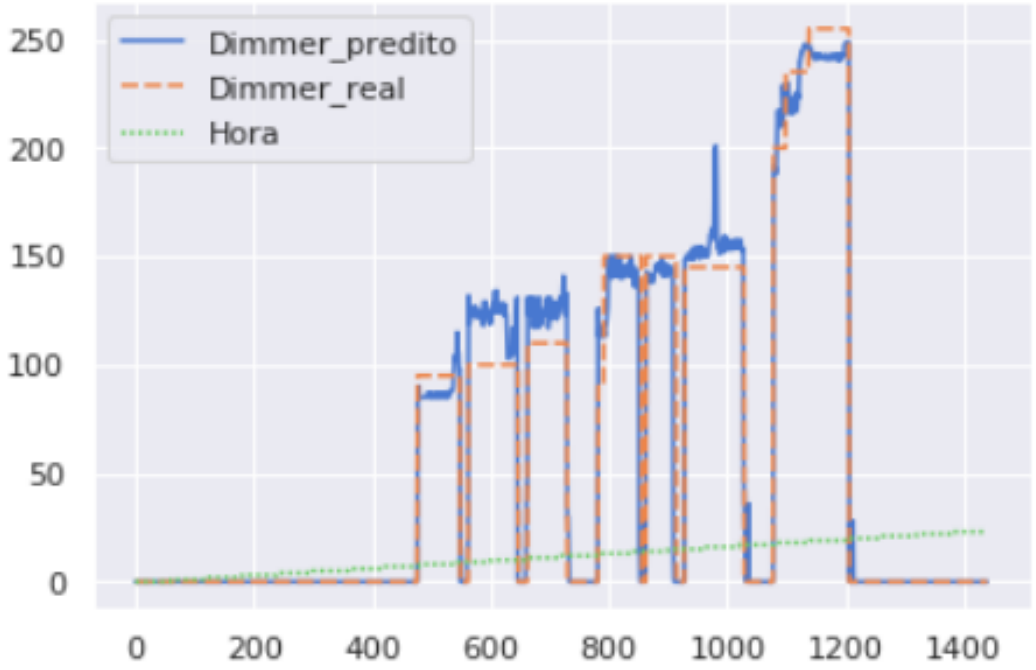
Figura 65 – Valores dimmerizados e preditos para o sétimo dia



Fonte: Próprio autor

Para uma visualização melhor do desempenho, da mesma forma exposta na figura 64, a figura 66 mostra em um mesmo gráfico o comportamento das predições de acordo com o passar das amostras.

Figura 66 – Comparação dos valores reais e preditos para o terceiro dia



Fonte: Próprio autor

CONCLUSÕES

A partir da análise dos resultados gerados e apresentados neste documento, é possível notar a que a aplicação de inteligência artificial e o *retrofit* de sensoriamento para a plataforma SmartLVGrid foi implementada com êxito. Em se tratando da base para implementação do modelo de inteligência artificial, verificou-se que havia uma lacuna no quesito de coleta e virtualização de dados do ambiente. Esse obstáculo foi significante, visto que há uma necessidade de coleta de informações para implementações do gênero. Todavia, na planta predial em questão, haviam sensores de presença desprovidos de conexão em rede. Essa passividade relacionada à conectividade, impossibilitava a aquisição de dados importantes que seriam necessários para implementação do modelo de inteligência artificial. Ao observar isso, foi verificada necessidade de tornar esse sensor um dispositivo provido de conectividade. O uso do *framework* norteou no quesito da busca de um ponto de interface do sistema existente e possibilitou que o conceito de *retrofit* fosse utilizado para aplicação nesses sensores, o que os tornou dotados de conexão em rede. Sendo assim, o *retrofit* foi desempenhado através do ACU-PI de acordo com o preconizado no *framework*.

O dispositivo implementado denominado ACU-PI dotado de *hardware* e *firmware* foi desenvolvido para ser acoplado no ponto de interface que no caso em questão eram os sensores de presenças existentes. Após a instalação do ACU-PI nesse ponto de interface, foram verificadas as funcionalidades do mesmo. Essas funcionalidades são correspondentes às DRFs ou primitivas operacionais expostas no *framework* SmartLVGrid. Essas DRFs implementadas no *firmware*, em conjunto com a conectividade disponibilizada ao sistema, foram validadas com sucesso. Isso foi observado na análise dos testes das funções *getilum()* correspondente a coleta de informação de iluminação do ambiente bem como *getpresence()* correspondente a coleta de informação de presença no local. É importante enfatizar que essas primitivas operacionais se relacionam com a porta *get* do modelo de ACU já prevista no SmartLVGrid, o que corrobora que toda a implementação foi norteada pelo *framework*.

Com a disponibilização de dados ensejada pela implementação do ACU-PI, foi possível iniciar o processo de implementação do modelo de inteligência artificial. Todavia, no meio do processo de implementação, optou-se por realizar a simulação dos dados que seriam oriundos do ACU-PI que seriam especificamente de presença. Ao decorrer da implementação do modelo, percebeu-se a necessidade do correto tratamento dos dados antes da etapa de treinamento e o quanto isso pode influenciar no desempenho final do modelo. Isso foi observado na diferença de desempenho quando o modelo é treinado por uma quantidade menor de dados. Com os resultados foi verificado que o modelo conseguiu predizer os valores de iluminação de acordo com os horários que o usuário se encontra na sala.

REFERÊNCIAS

A, J. et al. Intelligent smart home automation and security system using arduino and wi-fi. *International Journal Of Engineering And Computer Science*, Valley International, mar 2017.

ADAFRUIT. 2019. <https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/how-pirs-work>. Acessado em 28/10/2019.

AL-FUQAHA, A. et al. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials*, Institute of Electrical and Electronics Engineers (IEEE), v. 17, n. 4, p. 2347–2376, 2015.

ARTIGO. 2019. <https://www.learndatasci.com/tutorials/python-pandas-tutorial-complete-introduction-for-beginners/>. Acessado em 25/10/2019.

BAHGA, A.; MADISETTI, V. *Internet of Things: A hands-on approach*. [S.l.]: Vpt, 2014.

BRAGA, A. de P.; FERREIRA, A. C. P. de L.; LUDERMIR, T. B. *Redes neurais artificiais: teoria e aplicações*. [S.l.]: LTC Editora Rio de Janeiro, Brazil:, 2007.

BRINK, H. et al. *Real-world machine learning*. [S.l.]: Manning, 2017.

CLEMENTS-CROOME, D. Sustainable intelligent buildings for people: a review. *Intelligent Buildings International*, Taylor & Francis, v. 3, n. 2, p. 67–86, 2011.

CLEMENTS-CROOME, D.; CROOME, D. J. *Intelligent buildings: design, management and operation*. [S.l.]: Thomas Telford, 2004.

CREME. 2019. <https://pypi.org/project/creme/>. Acessado em 28/10/2019.

DONG, L. et al. The impact of led correlated color temperature on visual performance under mesopic conditions. *IEEE Photonics Journal*, IEEE, v. 9, n. 6, p. 1–16, 2017.

EASTMAN, C. et al. *BIM handbook: A guide to building information modeling for owners, managers, designers, engineers and contractors*. [S.l.]: John Wiley & Sons, 2011.

ECP. 2019. <http://ecp.com.br/produto-detalhes/34?locale=pt-BR>. Acessado em 28/10/2019.

ELéTRICA. 2019. <https://www.mundodaeletrica.com.br/esquema-para-sensor-de-presenca/>. Acessado em 28/10/2019.

ESPRESSIF. *Datasheet ESP32-Wroom 32D*. 2019.

FACELI, K. et al. Artificial intelligence: a machine learning approach. *LTC, Rio de Janeiro*, 2011.

FERNANDES, R.; GUIMARÃES, W. Implementation of a buck converter with hysteresis voltage control applied to led chip array package for street lighting. In: IEEE. *2018 Argentine Conference on Automatic Control (AADECA)*. [S.l.], 2018. p. 1–6.

FERREIRA, A. R.; TOMIOKA, J. Iluminação de estado sólido, economia potencial de energia elétrica para o país. In: *Anais do VIII Workshop de Pós-Graduação e Pesquisa do Centro Paula Souza, São Paulo, São Paulo, Brasil*. [S.l.: s.n.], 2013.

FLACH, P. *Machine learning: the art and science of algorithms that make sense of data*. [S.l.]: Cambridge University Press, 2012.

FÜRNKRANZ, J.; GAMBERGER, D.; LAVRAČ, N. *Foundations of rule learning*. [S.l.]: Springer Science & Business Media, 2012.

GÉRON, A. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. [S.l.]: O'Reilly Media, 2019.

GOMES, R. C. S. et al. Automation meta-system applied to smart grid convergence of low voltage distribution legacy grids. In: IEEE. *2017 IEEE International Conference on Smart Energy Grid Engineering (SEGE)*. [S.l.], 2017. p. 400–413.

HANES, D. et al. *IoT fundamentals: Networking technologies, protocols, and use cases for the internet of things*. [S.l.]: Cisco Press, 2017.

HAYKIN, S. S. et al. *Neural networks and learning machines*. [S.l.]: Pearson education Upper Saddle River, 2009. v. 3.

ILUMININ. 2018. <https://www.iluminim.com.br/drivers?sort=mais_vendidosl>. Acessado em 25/10/2019.

INTELLIPAAT. 2019. <https://intellipaat.com/blog/tutorial/data-science-tutorial/modeling-the-data/>. Acessado em 24/11/2019.

KANSARA, V. K. Solar & led technology for energy efficient substation. In: IEEE. *2017 IEEE Region 10 Symposium (TENSYMP)*. [S.l.], 2017. p. 1–5.

KAZAK, A. N.; BUCHATSKIY, P. Perspectives for smart city technologies in the resort region. In: *2018 IEEE International Conference "Quality Management, Transport and Information Security, Information Technologies"(IT&QM&IS)*. [S.l.]: IEEE, 2018.

KUBAT, M. *An introduction to machine learning*. [S.l.]: Springer, 2017. v. 2.

LOSING, V.; HAMMER, B.; WERSING, H. Incremental on-line learning: A review and comparison of state of the art algorithms. *Neurocomputing*, Elsevier, v. 275, p. 1261–1274, 2018.

MAKERS. 2019. <https://natalmakers.lojaintegrada.com.br/placa-de-fenolite-10cm-x-15cm-face-cobreada-simples>. Acessado em 30/10/2019.

MARSLAND, S. *Machine learning: an algorithmic perspective*. [S.l.]: Chapman and Hall/CRC, 2011.

MONTEIRO, R. V. A. et al. Led tubular lamps and tubular fluorescent: Power quality. In: IEEE. *2014 16th International Conference on Harmonics and Quality of Power (ICHQP)*. [S.l.], 2014. p. 400–404.

MQTT. 2019. <https://pplware.sapo.pt/tutoriais/networking/mqtt-protocolo-de-comunicacao/>. Acessado em 28/10/2019.

NGUYEN, T. A.; AIELLO, M. Energy intelligent buildings based on user activity: A survey. *Energy and buildings*, Elsevier, v. 56, p. 244–257, 2013.

NUMPY. 2017. <https://docs.scipy.org/doc/numpy-1.13.0/user/whatisnumpy.html>. Acessado em 25/10/2019.

PAHO. 2019. <https://www.eclipse.org/paho/>. Acessado em 28/10/2019.

PANDAS. 2019. <https://pandas.pydata.org/>. Acessado em 25/10/2019.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.

PEREIRA, A. et al. Some considerations about led technology in public lighting. In: IEEE. *2015 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*. [S.l.], 2015. p. 561–565.

PERERA, C. et al. A survey on internet of things from industrial market perspective. *IEEE Access*, Institute of Electrical and Electronics Engineers (IEEE), v. 2, p. 1660–1679, 2014.

RANGEL, M. G.; SILVA, P. B.; GUEDE, J. R. A. Led–iluminação de estado sólido. *São José do Campos*, 2009.

ROJAS, R. *Neural networks: a systematic introduction*. [S.l.]: Springer Science & Business Media, 2013.

SMARTBUILDING. 2018. <https://software.intel.com/en-us/articles/relayr-any-to-any-connectivity-transforms-smart-buildings>. Acessado em 22/04/2019.