

**Hugo Su Zhizuo**  
**Victor Rocha Barreira**

**Reconhecimento de atividades anômalas em ambientes de Smart Home**

São Paulo  
2019

**Hugo Su Zhizuo**  
**Victor Rocha Barreira**

**Reconhecimento de atividades anômalas em ambientes de Smart Home**

Trabalho apresentado à Escola  
Politécnica da Universidade de São  
Paulo para obtenção do Título de  
Engenheiro Eletricista com ênfase  
em Computação

São Paulo  
2019

**Hugo Su Zhizuo**  
**Victor Rocha Barreira**

**Reconhecimento de atividades anômalas em ambientes de Smart Home**

Trabalho apresentado à Escola  
Politécnica da Universidade de São  
Paulo para obtenção do Título de  
Engenheiro Eletricista com ênfase  
em Computação

Área de Concentração:  
Reconhecimento de atividades  
humanas

Orientador:  
Prof. Dr. Reginaldo Arakaki  
Co-orientador:  
Eng. Victor Takashi Hayashi

São Paulo  
2019

## **AGRADECIMENTOS**

Nossos sinceros agradecimentos aos familiares, amigos de universidade e aos orientadores por nos oferecer o apoio necessário, e principalmente acreditar em nossa jornada.

Agradecimentos especiais ao nosso co-orientador, Victor Hayashi, que nos acompanhou de perto durante a evolução de nosso projeto em nossas reuniões, e que nos possibilitou a continuidade de seu trabalho desenvolvido para seu projeto de formatura.

## RESUMO

Com o desenvolvimento e popularização dos dispositivos IoT (Internet of Things) nos últimos anos, vemos diversos cenários de aplicações e pesquisa relacionados surgindo de forma nunca antes tão explorados profundamente. Dentre estes vários novos cenários, estão as casas inteligentes (smart homes), que consistem basicamente de uma casa comum equipada com aparelhos ligados a internet.

Pensando nisso, foi desenvolvido um projeto onde dados gerados por dispositivos IoT presentes em uma casa inteligente foram utilizados a fim de fornecer mais segurança aos seus residentes, sem comprometer sua privacidade.

**Palavras-chave** - Casa inteligente. Detecção de anomalias. Reconhecimento de atividades. Padrões sequenciais.

## **ABSTRACT**

With the development and popularization of Internet of Things (IoT) devices in recent years, we see a variety of related application and research scenarios emerging in ways never before explored in depth. Among these various new scenarios are smart homes, which basically consist of a common home equipped with internet devices that communicate with each other.

With this in mind, a project was developed where data generated by IoT devices present in a smart home was used to provide more security to its residents, without compromising their privacy.

**Keywords** - Smart home. Anomaly detection. Activity recognition. Sequential Patterns.

## **Lista de abreviações**

IoT - Internet of Things

RFID - Radio Frequency Identification

LAN - Local Area Network

HAR - Human Activity Recognition

CEP - Complex Event Processing

RF - Requisito Funcional

RNF - Requisito não Funcional

FDP - Função densidade de probabilidade

CEP - Complex Event Processing

## **Lista de figuras**

- Figura 1 - Base analisada [Página 23]
- Figura 2 - Conjunto sequencial de tamanho 1 [Página 23]
- Figura 3 - Conjunto sequencial de tamanho 2 [Página 23]
- Figura 4 - Conjunto sequencial de tamanho 3 [Página 24]
- Figura 5 - Conjunto sequencial de tamanho 4 [Página 24]
- Figura 6 - Padrões sequenciais encontrados [Página 24]
- Figura 7 - Exemplo da geração de conjuntos [Página 25]
- Figura 8 - Diagrama arquitetural da casa inteligente [Página 26]
- Figura 9 - Disposição dos sensores na casa [Página 28]
- Figura 10 - Fluxo de módulos do sistema [Página 32]
- Figura 11 - Diagrama arquitetural do sistema [página 33]
- Figura 12 - Dashboard Tableau [Página 37]
- Figura 13 - Features selecionadas para treinamento de Random Forest [Página 44]
- Figura 14 - Distribuição de probabilidades entre possíveis atividades [Página 45]
- Figura 15 - Arquitetura do algoritmo de série temporal [Página 46]
- Figura 16 - Fórmula para o cálculo de probabilidade [Página 47]
- Figura 17 - Número de alertas em cenário randômico de série temporal [Página 49]
- Figura 18 - Número de alertas no cenário randômico Random Forest [Página 50]
- Figura 19 - Número de alertas em diferentes cenários de série temporal [Página 51]
- Figura 20 - Registro de log para cenário de invasão [Página 52]
- Figura 21 - Número de alertas em cenário de invasão Random Forest [Página 52]
- Figura 22 - Número de alertas em cenário de invasão Random Forest tratado [Página 53]
- Figura 23 - Número de alertas em cenário de visita Random Forest [Página 54]
- Figura 24 - Médias e desvios padrões no comportamento normal de série temporal [Página 55]
- Figura 25 - Avaliação dos sensores acionados em um dia por faixa [Página 56]
- Figura 26 - Número de alertas para o dia de teste [Página 56]
- Figura 27 - Número de alertas no cenário normal Random Forest [Página 58]



## **Lista de tabelas**

Tabela 1 - Comparação das soluções HAR propostas em trabalhos citados [Página 16]

Tabela 2 - Relações temporais de Allen [Página 21]

Tabela 3 - Requisitos funcionais [Página 34]

Tabela 4 - Requisitos não-funcionais [Página 34]

Tabela 5 - Correlação entre cômodos e sensores na base “home.csv” [Página 36]

Tabela 6 - Campos do arquivo de atividades [Página 40]

Tabela 7 - Campos do arquivo de cômodos e sensores [Página 41]

Tabela 8 - Campos do arquivo de correlação de sensores [Página 41]

Tabela 9 - Features adicionais geradas para treinamento de Random Forest [Página 43]

Tabela 10 - Thresholds para níveis de anormalidade [Página 45]

# Sumário

<b>1. Introdução</b>	<b>12</b>
1.1 Motivação	12
1.2 Justificativa	13
1.3 Objetivo	14
1.4 Organização do Trabalho	14
<b>2. Aspectos Conceituais</b>	<b>15</b>
2.1 Trabalhos relacionados	15
2.2 Inteligência artificial	17
2.2.1 Machine Learning	17
2.2.2 Random Forest	18
2.2.2.1 Decision Trees	18
2.3 CEP (Complex Event Processing)	19
2.4 Metodologias de HAR	19
2.5 Séries temporais	19
2.6 Data mining	22
2.6.1 AprioriAll	22
2.6.1.1 Geração de conjuntos	25
<b>3. Tecnologias Utilizadas</b>	<b>26</b>
3.1 Arquitetura do sistema atual	26
3.2 Tipos de sensores	27
3.3 Reconhecimento de atividades (HAR)	28
<b>4. Metodologia do Trabalho</b>	<b>29</b>
4.1 Definição do escopo	29
4.2 Projeto	29
4.3 Cronograma	30
4.3.1 Divisão de tarefas	30
4.4 Evolução do projeto	30
4.5 Testes e resultados previstos	31
4.5.1 Comportamentos normais	31
4.5.2 Comportamentos anormais	31
<b>5. Especificação de Requisitos do Sistema</b>	<b>32</b>
5.1 Metodologia	32
5.2 Arquitetura do sistema	33
5.3 Requisitos do sistema	34
5.3.1 Requisitos funcionais	34
5.3.2 Requisitos não funcionais	34

<b>6. Projeto e implementação</b>	<b>35</b>
6.1 Base de dados	35
6.2 Estudo de comportamento da casa	36
6.3 Segmentação	38
6.4 Classificação	39
6.4.1 Arquivos de configuração	39
6.4.1.1 Arquivo de atividades	39
6.4.1.2 Arquivo de cômodos e sensores	40
6.4.1.3 Arquivo de correlação entre sensores	41
6.4.2 Classificação	42
6.4.2.1 Classificação nível 1	42
6.4.2.2 Classificação nível 2	42
6.5 Verificação de padrões (Random Forest)	42
6.6 Algoritmo de série temporal	46
<b>7. Testes e Avaliação</b>	<b>48</b>
7.1 Comportamentos anormais	48
7.1.1 Teste Randômico	48
7.1.1.1 Série temporal	48
7.1.1.2 Random Forest	50
7.1.2 Teste específico	51
7.1.2.1 Série temporal	51
7.1.2.2 Random Forest	52
7.2 Comportamentos normais	54
7.2.1 Série temporal	54
7.2.2 Random Forest	57
7.3 Avaliação	58
<b>8. Considerações Finais</b>	<b>60</b>
8.1 Conclusões do Projeto de Formatura	60
8.2 Contribuições	61
8.3 Trabalhos futuros	61
8.3.1 Redes neurais e Deep Learning	61
8.3.2 Sistema de notificação ao usuário e ação	62
8.3.3 Sistema de Healthcare	62
<b>Referências</b>	<b>63</b>

# 1. Introdução

O ponto inicial para desenvolvimento do projeto são suas motivações, que somadas com justificativas (argumentos) para criação de algo concreto, acabam por se transformar em objetivos claros. Com isso, todos estes pontos serão citados a seguir.

## 1.1 Motivação

Graças aos avanços tecnológicos na microeletrônica e ao barateamento dos hardwares que possibilitaram a criação de máquinas cada vez menores e mais rápidas. Isso proporcionou a criação de um novo paradigma, onde tudo pode ser conectado através da internet. Surge assim equipamentos de todos os tipos e tamanhos, todos se comunicando e compartilhando informações através de uma conexão local ou de longa distância.

Neste contexto surge o conceito de casa inteligente, que é o tema foco deste trabalho. Uma casa inteligente é construída com a intenção de automatizar trabalhos manuais e dar mais comodidade aos seus residentes, contudo uma casa inteligente pode ir além da automatização de tarefas, tendo um grande potencial não aproveitado na área de segurança residencial que são foco de estudo e desenvolvimento por empresas como a Amazon. Além do conforto e praticidade que é oferecido por uma casa inteligente, também são produzidos dados de uso dos aparelhos como a temperatura do ar condicionado, os horários de entrada e saída, horário de uso das lâmpadas, etc. Esses dados informam muito mais do que apenas se um aparelho está desligado ou ligado, eles dizem respeito ao hábito dos residentes de uma casa, hábito este que é único para cada família.

Com a popularização das casas inteligentes que são casas equipadas com aparelhos conectados à internet de forma a constituir um sistema integrado com o propósito de auxiliar na vida diária de seus residentes. Aumentou-se também o interesse e o investimento nessa área de pesquisa, pois uma casa inteligente gera dados muito ricos, sendo utilizados para pesquisas que vão desde reconhecimento de atividades humanas [\[13\]](#) até a detecção de sintomas de alzheimer [\[15\]](#). Além do fato deste método de coleta de dados ser não intrusivo, uma vez que os aparelhos

IoT (aparelhos que compartilham informações via internet) não interferem na execução das atividades dos residentes nem compromete a privacidade dos residentes, como é o caso de câmeras IP, onde a rotina diária de uma casa pode ser observada por um terceiro caso haja processamento externo, o que justifica o grande interesse dos cientistas e investidores.

## 1.2 Justificativa

Sempre houve uma preocupação por parte das pessoas com a segurança de suas casas ao redor do mundo, fazendo com que coloquem fechaduras mais sofisticadas nas portas, redes elétricas ao redor da casa, câmeras de vigilância, etc. Todas as técnicas e métodos que uma pessoa normalmente utiliza para aumentar a segurança em sua casa têm algo em comum: são artifícios que visam bloquear ou desestimular o criminoso a cometer o crime. Contudo, esse meio não é o mais eficaz contra este tipo de crime, pois ao perceber o nível de segurança de uma casa o criminoso que quer cometer o crime busca soluções para burlar o sistema, colocando em risco os residentes da casa.

Pensando nisso, elaboramos uma solução que dá mais segurança aos residentes usando os dispositivos presentes em uma casa inteligente. Ao invés de trabalhar a segurança da casa de forma ativa, impedindo o criminoso de entrar na casa, trabalhamos com um método passivo de segurança, onde por meio dos dados gerados pelos dispositivos IoT, traçamos um hábito da casa a fim de identificar comportamentos anormais potencialmente perigosos. Esta abordagem se mostra mais eficiente na medida que uma pessoa dificilmente consegue imitar o hábito de outra pessoa, muito menos um criminoso tentando cometer um crime, diferente do que acontece nos sistemas de segurança tradicionais onde o foco está na prevenção e não na detecção, como acontece no nosso sistema, o que torna sistemas tradicionais de segurança vulneráveis a ameaças que partem de dentro da casa, como no caso em que criminosos abordam o morador e entram junto na casa.

## 1.3 Objetivo

O objetivo deste trabalho é projetar e implementar um sistema de monitoramento de casas inteligentes que seja capaz de mapear os dados recebidos de dispositivos IoT em atividades diárias, como comer, dormir e cozinhar, e avaliar essas atividades como sendo normais ou anormais. Este sistema poderia ser utilizado em diversos outros, como sistema de tomada de decisão, onde caso detecte uma atividade anormal uma ação é tomada de acordo com o nível de periculosidade.

## 1.4 Organização do Trabalho

Iniciando com o capítulo 2, são apresentados o conceito de Inteligência Artificial e alguns de seus conceitos relacionados: Machine Learning, Random Forest e Árvores de Decisão, além do conceito de CEP.

No capítulo 3, são descritas as funcionalidades das tecnologias que irão auxiliar no desenvolvimento do projeto.

No capítulo 4, estão todas as decisões de projeto tomadas durante sua concepção, bem como suas justificativas.

O capítulo 5 exemplifica o funcionamento de requisitos funcionais e não-funcionais através de diagramas e tabelas.

O capítulo 6 relata a arquitetura de software projetada para as 2 vertentes de aprendizado de atividades: *Random Forest* (supervisionada) e série temporal (não-supervisionada).

No capítulo 7, são relatados cenários de teste realizados para as 2 vertentes, explicando critérios de escolha para tais cenários, resultados e conclusões retiradas desses exemplos criados.

Por fim, no capítulo 8 é apresentada uma conclusão geral sobre o projeto desenvolvido ao decorrer do ano, assim como contribuições relacionadas e perspectivas de continuidade deste trabalho.

## 2. Aspectos Conceituais

Neste capítulo se encontram alguns fundamentos conceituais necessários para a elaboração do projeto além de um panorama geral de como está a pesquisa nesta área.

### 2.1 Trabalhos relacionados

O reconhecimento da atividade humana tornou-se uma área de pesquisa muito popular. Como resultado, numerosos estudos investigam métodos online e off-line para reconhecimento de atividades [\[5\]](#).

Aplicações bem-sucedidas nesta área incluem análise de comportamento em casa [\[3\]](#), reconhecimento de atividades por vídeo [\[7\]](#) e reconhecimento de gestos [\[8\]](#). Estes trabalhos procuram reconhecer atividades humanas, por meio de sensores IoT, tais como dormir, comer, cozinhar, etc. Existem principalmente dois tipos de HAR (Human Activity Recognition): HAR baseados em vídeo e HAR baseados em sensor. O HAR baseado em vídeo analisa vídeos ou imagens contendo movimentos humanos a partir da câmera, enquanto o HAR baseado em sensor foca nos dados de movimento de sensores inteligentes, como um acelerômetro, giroscópio, *Bluetooth*, sensores de som e assim por diante. Devido ao desenvolvimento próspero da tecnologia de sensores e da computação pervasiva, o HAR baseado em sensor está se tornando mais popular e amplamente utilizado com privacidade bem protegida [\[9\]](#).

Os sensores corporais foram amplamente utilizados em HARs baseados em *deep learning* [\[10\]\[11\]\[12\]](#). Entre esses trabalhos, o acelerômetro é o sensor mais utilizado. O giroscópio e o magnetômetro são também frequentemente usados em conjunto com o acelerômetro. Esses sensores são frequentemente explorados para reconhecer atividades de vida diária e esportes. Em vez de extrair características estatísticas e de frequência dos dados de movimento, o sinal original é usado diretamente como entradas para a rede [\[9\]](#).

Várias literaturas usaram sensores de ambiente para reconhecer atividades diárias e gestos com as mãos [\[13\]\[8\]](#). A maior parte do trabalho foi testada no ambiente de casa inteligente. A implantação de sensores de ambiente também é

difícil. Além disso, sensores de ambiente são facilmente afetados pelo ambiente, e somente certos tipos de atividades podem ser inferidas de forma robusta [9].

Muitos dos projetos nesta área ficam apenas no campo da pesquisa, não se preocupando com os problemas dos usuários, como questões de privacidade e disponibilidade. A seguir na tabela 1 encontra-se uma comparação das soluções propostas nos trabalhos citados anteriormente com a nossa solução.

Tabela 1 - Comparação das soluções HAR propostas em trabalhos citados

<b>Trabalhos</b>	<b>Privacidade</b>	<b>Processamento local</b>	<b>Sistema distribuído</b>
comportamento em casa	Preserva a privacidade direta do usuário	não	não
reconhecimento de atividades por vídeo	não há	não	não
reconhecimento de gestos	-	não	não
HARs baseados em <i>deep learning</i>	usa dataset público	não	não
reconhecimento atividades diárias	usa dataset público	não	não
Autores	sim	sim	sim

Fonte: Autores

A tabela 1 ilustra que as soluções analisadas em sua maioria não oferecem medidas para garantir a privacidade e sigilo dos dados, além de não lidarem com aspectos relativos a disponibilidade, com isso em mente propusemos uma arquitetura de sistema que atende a esses requisitos, pois estes requisitos são na implementação de física de qualquer sistema de segurança.



## 2.2 Inteligência artificial

A expressão inteligência artificial está associada, geralmente, ao desenvolvimento de sistemas especialistas. Estes sistemas baseados em conhecimento, construídos, principalmente, com regras que reproduzem o conhecimento do perito, são utilizados para solucionar determinados problemas em domínios específicos. A área médica, desde o início das pesquisas, tem sido uma das áreas mais beneficiadas pelos sistemas especialistas, por ser considerada detentora de problemas clássicos possuidores de todas as peculiaridades necessárias, para serem instrumentalizados por tais sistemas [\[2\]](#).

### 2.2.1 Machine Learning

Há um constante crescimento no interesse pelo aprendizado de máquina ou *machine learning*. Isso se deve ao crescente volume e variedade de dados disponíveis, aumento e barateamento do processamento computacional e ao armazenamento de dados de forma mais eficiente que tornou possível produzir modelos preditivos mais rapidamente e com maior precisão.

De forma geral, aprendizado de máquina é um campo da inteligência artificial que visa estudar e construir algoritmos que possibilitam o reconhecimento e extração de padrões a partir de um grande volume de dados, construindo dessa maneira um modelo de aprendizado. Esse aprendizado é obtido da observação dos dados e uma vez que tenha construído o modelo, é capaz de executar tarefas complexas e dinâmicas, fazer previsões com alta precisão, reagir em situações diversas e comportar-se de forma inteligente.

Existem atualmente três métodos de aprendizados muito utilizados [\[31\]](#). São eles:

- **Aprendizado supervisionado:** o algoritmo de aprendizagem recebe um conjunto de entradas junto com as saídas corretas correspondentes, e o algoritmo aprende comparando a saída real com as saídas do modelo. Em seguida, ele modifica o modelo para aumentar sua precisão.

- Aprendizado não supervisionado: basicamente é usado em dados que não possuem rótulos, ou seja, o sistema não sabe a “resposta certa” a priori. O algoritmo deve achar um padrão nos dados e associá-los, por esse motivo é mais difícil utilizar este método de aprendizado.
- Aprendizado por reforço: a máquina tenta aprender qual é a melhor ação a ser tomada, dependendo das circunstâncias na qual a ação será executada, por meio de tentativa e erro. Neste método é dado uma recompensa caso a máquina acerte e uma punição caso escolha uma ação ruim, sendo que seu objetivo neste cenário é maximizar sua recompensa ou diminuir sua punição.

## 2.2.2 Random Forest

*Random Forest* é um algoritmo de aprendizagem supervisionada que combina várias *decision trees* para fazer uma predição. Seu funcionamento consiste em inicialmente escolher aleatoriamente N amostras do seu conjunto de treinamento, recolocando as amostras no conjunto de treinamento após utilizá-las, criando uma *decision tree* a partir destas amostras. Isto é repetido até que se tenha uma floresta de tamanho P, quando for necessário fazer uma previsão é selecionado aleatoriamente M *decision trees* com  $M < P$  e combina-se às decisões destas M árvores para se fazer a predição [\[32\]](#).

### 2.2.2.1 Decision Trees

*Decision Tree* é um algoritmo de aprendizagem supervisionada, muito utilizado em problemas de classificação. Seu objetivo é particionar o espaço recursivamente em sub-regiões homogêneas, ou seja, separar e agrupar os dados que tem maior semelhança entre si.

Para se criar uma *decision tree* deve-se escolher a variável que melhor particiona o conjunto de treinamento como nó raiz da árvore, em seguida, devemos escolher as variáveis que melhor particiona as sub-regiões formadas pelo nó raiz até que se obtenham regiões completamente homogêneas ou até acabarem as variáveis [\[33\]](#).

## 2.3 CEP (Complex Event Processing)

O CEP existe para ajudar a entender e controlar os sistemas de informação baseados em eventos. Essas técnicas são baseadas no conceito de eventos que só poderiam acontecer porque outros eventos aconteceram antes. A ideia é que os eventos são relacionados pela razão no qual eles aconteceram, quando aconteceram e a associação entre eles e outros eventos. De certa forma, o CEP distingue-se das abordagens “estáticas” relacionadas ao banco de dados, concentrando-se na natureza dinâmica dos dados a serem processados: os eventos são criados de forma ordenada, propagados em fluxos e devem ser processados, agregados e filtrados em tempo real [\[14\]](#).

## 2.4 Metodologias de HAR

O aprendizado de atividades humanas visa aprender e compreender as atividades e situações observadas em um ambiente, sendo útil para uma ampla gama de aplicações e serviços, como detecção de emergências e monitoramento de saúde, detecção precoce de doenças, automação residencial, segurança e intervenção comportamental. Isso envolve recursos como reconhecimento (classificação), detecção, segmentação e previsão de atividades.

Considerando técnicas de segmentação e classificação de atividades como parte importante da tecnologia envolvida no reconhecimento de padrões comportamentais em nosso projeto, as principais abordagens propostas segundo Aminikhanghahi e Cook (2019) [\[5\]](#) são a AR-W, AR-WT, AR-SM e AR-SS, cujas siglas não são explicadas, porém o funcionamento de cada abordagem está descrito em detalhes na referência.









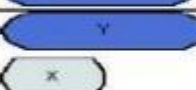

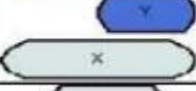


## 2.5 Séries temporais

As atividades em uma casa inteligente incluem as atividades dos residentes como também interações com o meio ambiente. Isso pode incluir caminhar, sentar no sofá, acender uma lâmpada, usar a cafeteira e assim por diante. Vemos que essas atividades não são instantâneas, mas têm horários de início e fim distintos.

Também vemos que há relacionamentos bem definidos entre os intervalos de tempo para diferentes atividades. Essas relações temporais podem ser representadas usando as relações temporais de Allen [29] e podem ser usadas para a descoberta de padrões nas atividades do dia-a-dia. Essas descobertas podem ser usadas para o desenvolvimento de sistemas que detectam anomalias e ajudam os cuidadores a tomar medidas preventivas [30].

Allen [29] listou treze relações (visualizadas na tabela 2) compreendendo uma lógica temporal: *before*, *after*, *meets*, *meet-by*, *overlaps*, *overlapped-by*, *starts*, *started-by*, *finishes*, *finished-by*, *during*, *contains* e *equals*. Essas relações temporais desempenham um papel importante na identificação de atividades sensíveis ao tempo que ocorrem em uma casa inteligente. Considere, por exemplo, um caso em que o residente ligue a televisão antes de sentar no sofá. Neste exemplo essas duas atividades, ligando a TV e sentando no sofá, são frequentemente relacionadas no tempo, de acordo com a relação temporal "antes". Restrições temporais podem ser úteis ao pensar sobre atividades, caso uma restrição temporal não seja satisfeita, uma possível situação "anômala" ou "crítica" pode ter ocorrido [30].

Tabela 2 - Relações temporais de Allen

Temporal Relations	Pictorial Representation	Interval constraints
X Before Y		$StartTime(X) < StartTime(Y);$ $EndTime(X) < StartTime(Y)$
X After Y		$StartTime(X) > StartTime(Y);$ $EndTime(Y) < StartTime(X)$
X During Y		$StartTime(X) > StartTime(Y);$ $EndTime(X) < EndTime(Y)$
X Contains Y		$StartTime(X) < StartTime(Y);$ $EndTime(X) > EndTime(Y)$
X Overlaps Y		$StartTime(X) < StartTime(Y);$ $StartTime(Y) < EndTime(X);$ $EndTime(X) < EndTime(Y)$
X Overlapped-By Y		$StartTime(Y) < StartTime(X);$ $StartTime(X) < EndTime(Y);$ $EndTime(Y) < EndTime(X)$
X Meets Y		$StartTime(Y) = EndTime(X)$
X Met-by Y		$StartTime(X) = EndTime(Y)$
X Starts Y		$StartTime(X) = StartTime(Y);$ $EndTime(X) \neq EndTime(Y)$
X started-by Y		$StartTime(Y) = StartTime(X);$ $EndTime(X) \neq EndTime(Y)$
X Finishes Y		$StartTime(X) \neq StartTime(Y);$ $EndTime(X) = EndTime(Y)$
X Finished-by Y		$StartTime(X) \neq StartTime(Y);$ $EndTime(X) = EndTime(Y)$
X Equals Y		$StartTime(X) = StartTime(Y);$ $EndTime(X) = EndTime(Y)$

Fonte: [\[30\]](#)

Contudo, existem muitos sensores dentro de uma casa inteligente e fazer uma correlação temporal entre todos estes sensores não só seria muito custoso computacionalmente como também diminui a acurácia preditiva desta abordagem, pois caso existam sensores pouco acionados na análise isso aumentaria a variância do sistema, o que dificultaria a avaliação. Por isso, antes de se utilizar um algoritmo de série temporal deve se fazer um *data mining* na base a fim de se encontrar os sensores que são mais frequentemente acionados.

## 2.6 Data mining

O termo *data mining* (ou mineração de dados) representa “o processo de explorar grandes quantidades de dados à procura de padrões consistentes, como regras de associação ou sequências temporais, para detectar relacionamentos sistemáticos entre variáveis, detectando assim novos subconjuntos de dados”. [\[28\]](#)

### 2.6.1 AprioriAll

Trata-se de um algoritmo de *data mining* para se encontrar padrões sequenciais em uma base de dados [\[1\]](#).

Primeiramente a base deve estar dividida em períodos (dias, horas, etc), com isso feito deve-se seguir os seguintes passos:

1. Criar um conjunto sequencial de tamanho 1 com seu respectivo mínimo suporte, sendo que o mínimo suporte é o número de ocorrências de uma determinada sequência na base, excluindo as ocorrências duplicadas em um mesmo período.
2. Retirar todas as sequências que tenham um mínimo suporte menor que um limite X arbitrário.
3. Guardar as sequências encontradas.
4. Gerar um novo conjunto sequencial de tamanho N+1 a partir do conjunto sequencial de tamanho N.
5. Repetir os passos 2, 3 e 4 até o novo conjunto gerado ser vazio.
6. Remover todas as sequências que sejam uma subsequência de uma sequência maior.

A seguir encontra-se um exemplo da execução do algoritmo para um mínimo suporte igual a 2:

Figura 1 - Base analisada

$$\begin{aligned}
 &\langle \{1\} \{5\} \{2\} \{3\} \{4\} \rangle \\
 &\langle \{1\} \{3\} \{4\} \{3\} \{5\} \rangle \\
 &\langle \{1\} \{2\} \{3\} \{4\} \rangle \\
 &\langle \{1\} \{3\} \{5\} \rangle \\
 &\langle \{4\} \{5\} \rangle
 \end{aligned}$$

Fonte: [1] adaptado

Figura 2 - Conjunto sequencial de tamanho 1

Sequence	Support
$\langle 1 \rangle$	4
$\langle 2 \rangle$	2
$\langle 3 \rangle$	4
$\langle 4 \rangle$	4
$\langle 5 \rangle$	4

Fonte: [1] adaptado

Figura 3 - Conjunto sequencial de tamanho 2

Sequence	Support
$\langle 1 \ 2 \rangle$	2
$\langle 1 \ 3 \rangle$	4
$\langle 1 \ 4 \rangle$	3
$\langle 1 \ 5 \rangle$	3
$\langle 2 \ 3 \rangle$	2
$\langle 2 \ 4 \rangle$	2
$\langle 3 \ 4 \rangle$	3
$\langle 3 \ 5 \rangle$	2
$\langle 4 \ 5 \rangle$	2

Fonte: [1] adaptado

Figura 4 - Conjunto sequencial de tamanho 3

Sequence	Support
$\langle 1\ 2\ 3 \rangle$	2
$\langle 1\ 2\ 4 \rangle$	2
$\langle 1\ 3\ 4 \rangle$	3
$\langle 1\ 3\ 5 \rangle$	2
$\langle 2\ 3\ 4 \rangle$	2

Fonte: [1] adaptado

Figura 5 - Conjunto sequencial de tamanho 4

Sequence	Support
$\langle 1\ 2\ 3\ 4 \rangle$	2

Fonte: [1] adaptado

Figura 6 - Padrões sequenciais encontrados

Sequence	Support
$\langle 1\ 2\ 3\ 4 \rangle$	2
$\langle 1\ 3\ 5 \rangle$	2
$\langle 4\ 5 \rangle$	2

Fonte: [1] adaptado

A figura 1 ilustra a base que será utilizada para o exemplo, enquanto que a figura 2 mostra as sequências de tamanho 1 que existem na base utilizando um mínimo suporte de 40%, ou seja, apenas sequências que tenham mínimo suporte igual ou maior que 2. O mínimo suporte para  $\{3\}$  é igual a 4, contudo na base há 5 aparições de  $\{3\}$ , isso ocorre porque  $\{3\}$  está duplicado em um dos períodos.

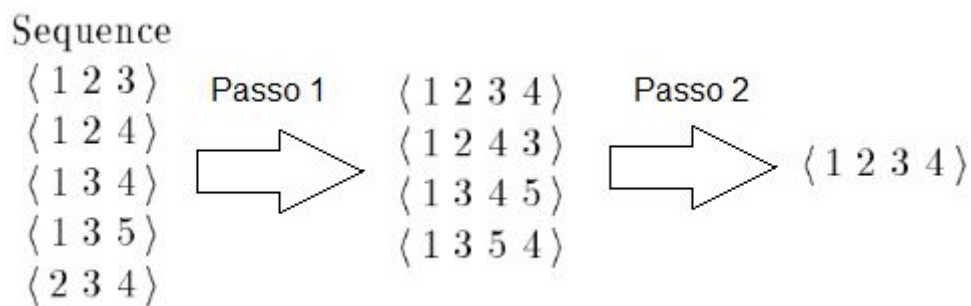
Para as figuras seguintes foram seguidos os passos 2,3,4 descritos anteriormente nesta seção.



### 2.6.1.1 Geração de conjuntos

Dada uma sequência  $X$  no conjunto, caso exista outra sequência  $Y$  que seja idêntica a sequência  $X$ , exceto pelo seu último elemento, é acrescentado o último elemento de  $Y$  à  $X$ , sendo este processo repetido para todas as sequências de um conjunto de tamanho  $N$ . Por exemplo, considere  $X$  igual a  $\langle 1, 2, 3 \rangle$  e  $Y$  igual a  $\langle 1, 2, 4 \rangle$ , a sequência de tamanho  $N+1$  resultante seria  $\langle 1, 2, 3, 4 \rangle$ , o passo seguinte é eliminar todas as sequências do conjunto  $N+1$  que tenham uma subsequência que não esteja presente no conjunto  $N$  [1].

Figura 7 - Exemplo da geração de conjuntos



Fonte: adaptado de [1]

Aqui podemos ver que as sequências  $\langle 1, 2, 4, 3 \rangle$ ;  $\langle 1, 3, 4, 5 \rangle$ ;  $\langle 1, 3, 5, 4 \rangle$  foram retiradas porque não existiam as subsequências  $\langle 2, 4, 3 \rangle$ ;  $\langle 3, 4, 5 \rangle$ ;  $\langle 3, 5, 4 \rangle$  respectivamente no conjunto  $N$ .

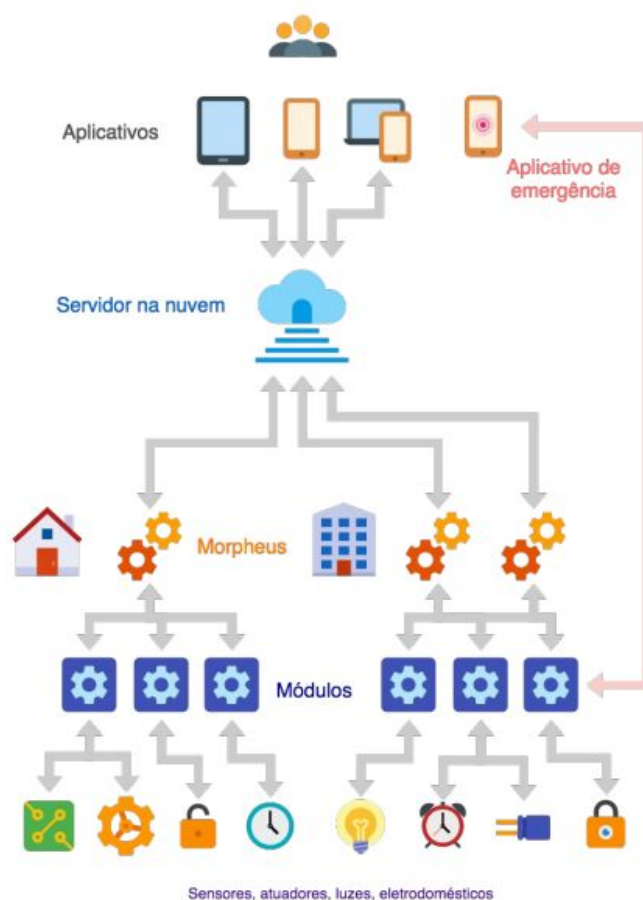
### 3. Tecnologias Utilizadas

Aqui serão descritas tecnologias utilizadas direta ou indiretamente no projeto proposto.

#### 3.1 Arquitetura do sistema atual

Este projeto está baseado em um sistema atualmente implementado em uma residência: o Projeto Hedwig [16], que visa a implementação de diversos dispositivos IoT com o objetivo de fornecer melhor qualidade de vida para os habitantes da residência, seja em aspectos de segurança (*safety* e *security*), automação de atividades cotidianas ou eficiência em gastos de luz e água. Sua arquitetura é representada pelo diagrama a seguir:

Figura 8 - Diagrama arquitetural da casa inteligente



Fonte: YASSUDA, D. et al., 2019

Segundo YASSUDA et al. (2017), alguns dos módulos que podem ser incluídos no sistema são:

- Quarto (despertador, iluminação, monitoramento de temperatura e umidade);
- Cozinha (timer, iluminação, monitoramento de presença e gás);
- Acesso (controle de abertura, monitoramento de estado);
- Externo (monitoramento de temperatura, umidade, energia elétrica e consumo de água);
- Corredor (monitoramento de presença, iluminação);
- Chuveiro (controle de temperatura/potência a partir do perfil de usuário e temperatura externa)
- Ar condicionado (controle da potência a partir do monitoramento das temperaturas internas e externas da casa).

Sua arquitetura fornece 3 níveis de funcionamento: Online, Local e Offline, para garantir a disponibilidade mesmo com problemas (queda do servidor, internet indisponível, falha no roteador), com medidas como reconexão, monitoramento e manutenções preventivas e corretivas do sistema. Além disso, o sistema é construído de forma modular, garantindo independência de funcionamento dos módulos, o que contribui para sua robustez, custo e manutenção.

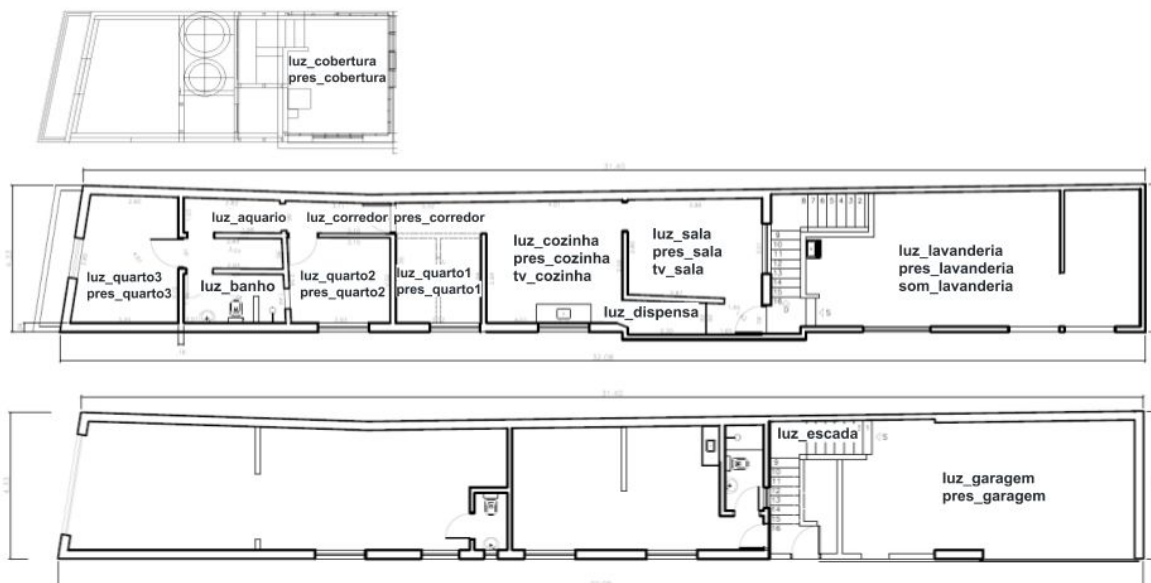
Os módulos deste sistema que estão diretamente relacionados com nosso projeto são os sensores, uma vez que esses fornecem a massa de dados bruta para análise e aplicação das metodologias de reconhecimento de atividades.

### **3.2 Tipos de sensores**

Em relação aos sensores que fornecem os dados comportamentais da casa é possível separá-los em duas principais categorias: sensores de presença e sensores de liga/desliga. Para os sensores de presença, estes possuem um contador que é incrementado enquanto uma nova pessoa não é detectada. Quando ocorre a detecção, este contador é zerado. Já no caso dos sensores de liga/desliga, estes podem estar relacionados tanto com interruptores de luzes nos cômodos

(indicando luz acesa ou apagada), como também a aparelhos eletrônicos (TV, rádio, etc.). Estes sensores estão dispostos na casa conforme a planta a seguir:

Figura 9 - Disposição dos sensores na casa



Fonte: [\[17\]](#)

### 3.3 Reconhecimento de atividades (HAR)

As saídas dos sensores são exportadas através de dados em formato de *log* CSV, contendo em cada coluna os valores registrados pelos sensores em um determinado instante de tempo (*timestamp*). Estes dados servirão a um código Python tanto no treinamento da base para segmentação e classificação de atividades, como também como amostra de teste para reconhecer se um padrão comportamental é considerado aceitável ou anômalo.

## 4. Metodologia do Trabalho

### 4.1 Definição do escopo

A motivação inicial de nosso trabalho seria detecção e análise de comportamentos anômalos para fins de segurança (*safety*) em condomínios. Porém após algumas reuniões com o professor, a orientação foi mudar o escopo para a classificação de atividades em casas inteligentes desempenhadas pelos residentes da casa, uma vez que existia um projeto anterior sobre uma casa conectada já em operação [17], servindo de acelerador para a elaboração deste projeto.

### 4.2 Projeto

Com a definição do escopo, o grupo buscou por trabalhos e pesquisas recentes voltados para casas inteligentes, sobretudo na parte de reconhecimento de atividades, para então escolher as referências [5] e [6] como base para nossos estudos iniciais.

Em relação à fase de implementação, o projeto é constituído por duas vertentes principais de aprendizado de máquina: aprendizado supervisionado, que conta com algoritmo de *Random Forest* para avaliação de atividades, e aprendizado não-supervisionado, através de um algoritmo de série temporal [30].

A vertente de aprendizado supervisionado conta com segmentação de atividades baseado em critério de presenças e sensores de luz, classificação de atividades de acordo com um conjunto de características atribuídas a elas e verificação de padrões de normalidade ou anormalidade de acordo com treinamento de base através de *Random Forest*.

Já a vertente de aprendizado não-supervisionado conta com um algoritmo de *data mining* [1] que servirá para descobrir quais são os padrões sequenciais mais frequentes na nossa base de dados para posteriormente obtermos relações temporais entre estas sequências, por fim, com isso podemos dizer se um dado acionamento de sensor é anormal ou não.

## 4.3 Cronograma

Este projeto está sendo concebido desde Março/19, data da primeira reunião entre os alunos e uma equipe conselheira, que vem realizando um trabalho conjunto de acompanhamento e definição de próximos passos. Desde então, reuniões vêm ocorrendo em média a cada 15 dias, onde os alunos trazem atualizações de sugestões de implementações, e juntamente com a equipe conselheira decisões de projeto são propostas, bem como os próximos passos para a evolução deste.

A fim de seguir um planejamento de implantação e testes, sugerimos abaixo os prazos de finalização relacionados a cada módulo:

- Segmentação: Julho/2019
- Classificação: Julho a Agosto/2019
- Análise de padrões: Setembro a Dezembro/2019
- Algoritmo de data mining: Setembro/2019
- Algoritmo de série temporal: Outubro a Novembro/2019
- Cenários de teste: Outubro a Dezembro/2019

### 4.3.1 Divisão de tarefas

Essencialmente, cada um dos membros do grupo trabalhou em uma vertente distinta de reconhecimento de atividades, de forma a realizar um trabalho especializado ao longo do desenvolvimento do projeto. Hugo realizou a frente de série temporal, sendo este o método de aprendizado não-supervisionado. Já o Victor realizou a arquitetura que utiliza o *Random Forest* como forma final de avaliação das atividades, sendo esta uma metodologia de aprendizado supervisionado.

## 4.4 Evolução do projeto

Como descrito anteriormente na seção 4.2 deste documento: “Em relação à fase de implementação, o projeto será constituído por duas vertentes principais de aprendizado de máquina: aprendizado supervisionado (...) e aprendizado não-supervisionado”.

O método supervisionado é composto por segmentação, classificação, e verificação de padrões. Inicialmente, a etapa de verificação de padrões estava

prevista para ser implementada através de rede neural, porém ao decorrer do projeto optamos por trocar essa abordagem pelo algoritmo de *Random Forest*. Essa decisão foi tomada pois este algoritmo é mais adequado para aprendizado baseado em dados estruturados, em que certa categoria de classificação depende dos valores de um conjunto de variáveis, o que caracteriza exatamente o ambiente de funcionamento da etapa de verificação de padrões. Por sua vez, a rede neural é mais adequada para dados não-estruturados, como imagens, vídeos e texto, com tempo de processamento superior e processo de aprendizado mais complexo em relação ao Random Forest. [\[25\]](#) [\[26\]](#). Os códigos fonte de cada um dos módulos podem ser verificados no projeto GitHub referente a este trabalho [\[27\]](#).

Como decisão de projeto decidimos adicionar o método não supervisionado a nossa solução a fim de comparar com o método supervisionado além de fazer análises sobre o uso de cada método.

## **4.5 Testes e resultados previstos**

Durante a implementação dos 3 módulos principais, serão realizados testes separadamente a fim de verificar o funcionamento apropriado de cada um.

Para a verificação completa do sistema será utilizado um dataset de treinamento de 2 meses , enquanto que para a validação do sistema usamos um dataset de um 1 dia, sendo que a validação ocorrerá da seguinte forma:

### **4.5.1 Comportamentos normais**

Aqui serão utilizados dados reais, gerados pelos sensores da casa a fim de se testar o modelo treinado para o cenário de falsos positivos. Espera-se que o sistema emita no máximo alertas de nível baixo.

### **4.5.2 Comportamentos anormais**

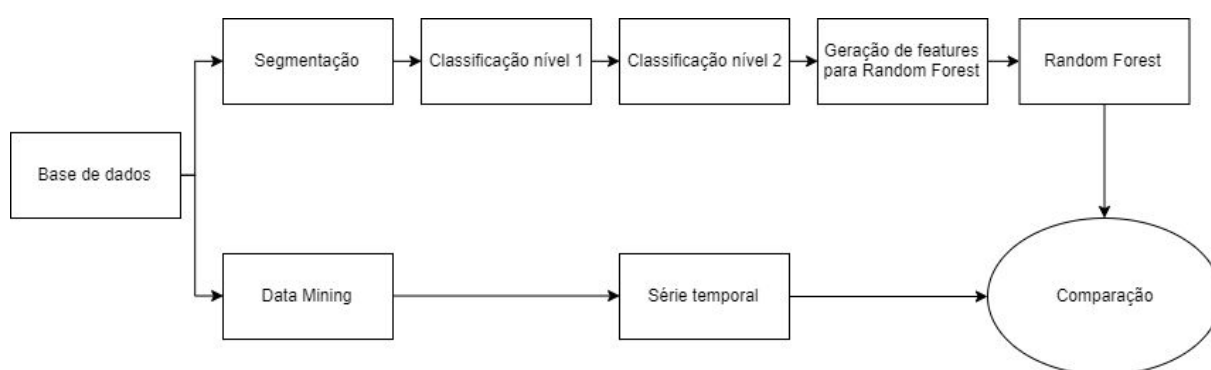
Dados originais dos sensores serão manipulados de forma que o conjunto das features simboliza uma atividade anômala a fim de se testar o modelo para o cenário de falsos negativos. Espera-se que o sistema emita alarmes de nível médio ou alto, em sua maioria.

## 5. Especificação de Requisitos do Sistema

Utilizamos a base de dados fornecida pelo sistema Hedwig [\[16\]](#), contudo nosso sistema não oferece nenhum tipo de integração com o sistema Hedwig, apenas utilizamos os dados gerados por esse sistema para estudo do comportamento da casa.

### 5.1 Metodologia

Figura 10 - Fluxo de módulos do sistema



Fonte: Autores

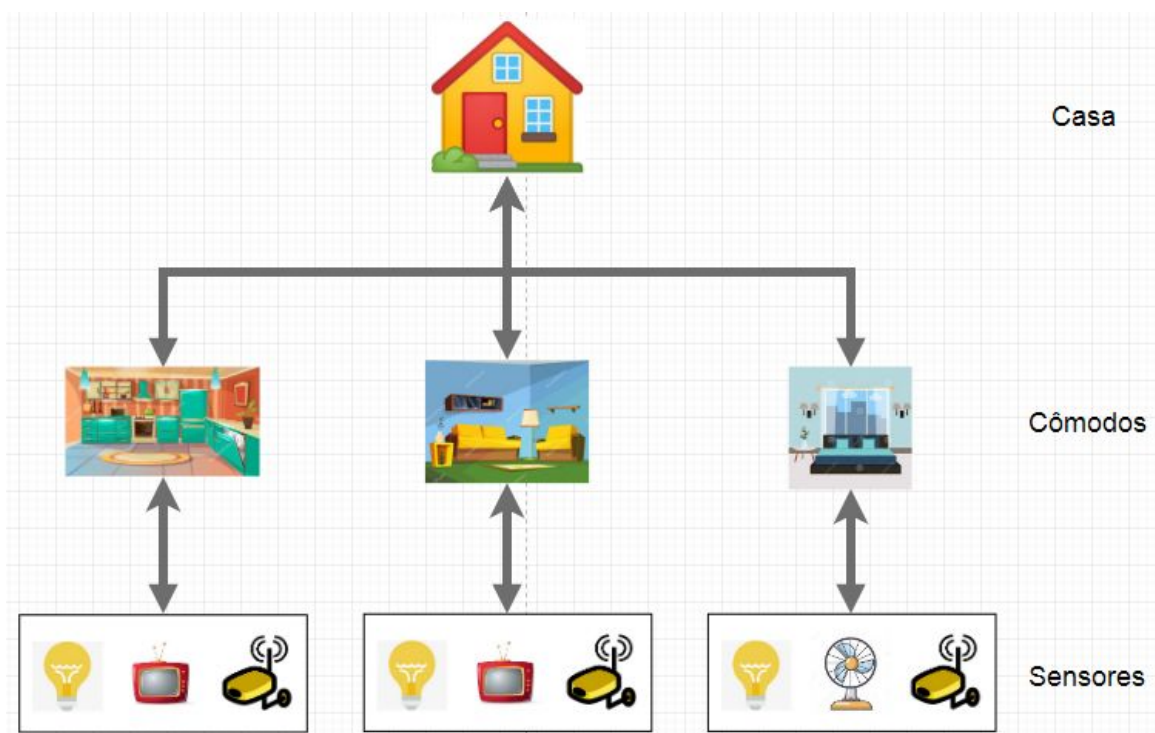
Os dados de sensores extraídos do sistema atual da casa inteligente alimentam o sistema proposto neste documento para a execução das vertentes de série temporal e *Random Forest*, visto que ambas possuem a mesma origem de dados, como observado no diagrama acima.

Com isso, o funcionamento de cada um dos módulos de ambas as vertentes será exposto em detalhes no capítulo 6, bem como os resultados dessas diferentes abordagens serão expostos e comparados no capítulo 7 deste documento.



## 5.2 Arquitetura do sistema

Figura 11 - Diagrama arquitetural do sistema



Fonte: Autores

Como mencionado na seção 3.1, o sistema foi idealizado de forma que fosse compatível com a infraestrutura já existente no sistema Hedwig [16], além de levar em consideração também aspectos não funcionais do projeto como disponibilidade e latência. Por isso, optou-se por fazer um sistema modularizado em software, onde cada módulo funciona independentemente, uma vez que os arquivos de saídas gerados por alguns módulos servem de alimentação (entrada) para outros módulos.

Por exemplo, no caso de uma arquitetura de hardware que possua os módulos para cada cômodo da casa, caso um módulo de um cômodo (ou o cômodo como um todo) fique indisponível, isso não comprometerá o fluxo de funcionamento do sistema como um todo.

Os sensores de cada cômodo são a base do sistema, pois estes farão a coleta dos dados dos residentes. No caso, a base de dados com os registros dos sensores é fornecida através de uma extração gerada pelo sistema Hedwig.

## 5.3 Requisitos do sistema

### 5.3.1 Requisitos funcionais

Tabela 3 - Requisitos funcionais

#RF	Nome	Descrição
RF01	Segmentação de atividades	Determinar quando há mudança de estado (transição de atividade)
RF02	Classificação de atividades	Determinar a qual classe uma atividade pertence baseado em suas features
RF03	Análise de padrões de atividades	Verificar se uma atividade é considerada comum ou anômala, de acordo com histórico de comportamento dos usuários
RF04	Graus de anormalidade	O sistema deverá classificar atividades anômalas segundo os seguintes graus de periculosidade: baixo, médio e alto

Fonte: Autores

Estes requisitos estão relacionados com a metodologia de projeto explicada na seção 4.2 deste documento.

### 5.3.2 Requisitos não funcionais

Tabela 4 - Requisitos não-funcionais

#RNF	Nome	Descrição
RNF01	Privacidade	Garantir o sigilo das informações dos moradores
RNF02	Disponibilidade	Garantir o funcionamento contínuo do sistema
RNF03	Latência	Sistema deve avaliar atividades com duração máxima próxima a 15 minutos

Fonte: Autores

Como citado anteriormente na seção 5.2, todas as escolhas de projeto foram tomadas de forma a atender todos os requisitos mencionados nesta seção.

## 6. Projeto e implementação

As seções 6.1 e 6.2 versam sobre estudo da base de dados fornecida pelo sistema Hedwig, tanto em sua estrutura quanto a respeito das informações contidas. Os capítulos 6.3, 6.4, 6.5 dizem respeito às 3 etapas principais da vertente de aprendizado supervisionado: segmentação, classificação e verificação de padrões, respectivamente. Já o capítulo 6.6 está relacionado à vertente de aprendizado não-supervisionado através de algoritmo de séries temporais.

### 6.1 Base de dados

Para estudo do comportamento da casa, foram fornecidos 2 arquivos CSV que dizem respeito a registros de diferentes cômodos da casa. O arquivo “access.csv” possui registros a respeito do acionamento do acesso no portão principal da casa, contando com informações de data e hora do acesso, e quem foi o usuário que acessou a casa. Já o arquivo “home.csv” fornece informações a respeito das atividades realizadas nos cômodos da casa, através dos valores de data e hora (timestamp) e dos 3 tipos de sensores encontrados na casa: luz, presença e aparelho.

A tabela 5 descreve a relação entre os cômodos da casa e seus respectivos sensores instalados, separados por tipo:

Tabela 5 - Correlação entre cômodos e sensores na base “home.csv”

		Luz	Aparelho	Presença
1	SALA	luz_sala	tv_sala	pres_sala
2	COZINHA	luz_cozinha	tv_cozinha	pres_cozinha
3	LAVANDERIA	luz_lavanderia	som_lavanderia	pres_lavanderia
4	QUARTO1	luz_quarto1	vent_quarto1	pres_quarto1
5	QUARTO2	luz_quarto2	vent_quarto2	pres_quarto2
6	QUARTO3	luz_quarto3	vent_quarto3	pres_quarto3
7	CORREDOR	luz_corredor	--	pres_corredor
8	GARAGEM	luz_garagem	--	pres_garagem
9	COBERTURA	luz_cobertura	--	pres_cobertura
10	ESCADA	luz_escada	--	--
11	DISPENSA	luz_dispensa	--	--
12	AQUARIO	luz_aquario	--	--
13	BANHO	luz_banho	--	--

Fonte: Autores

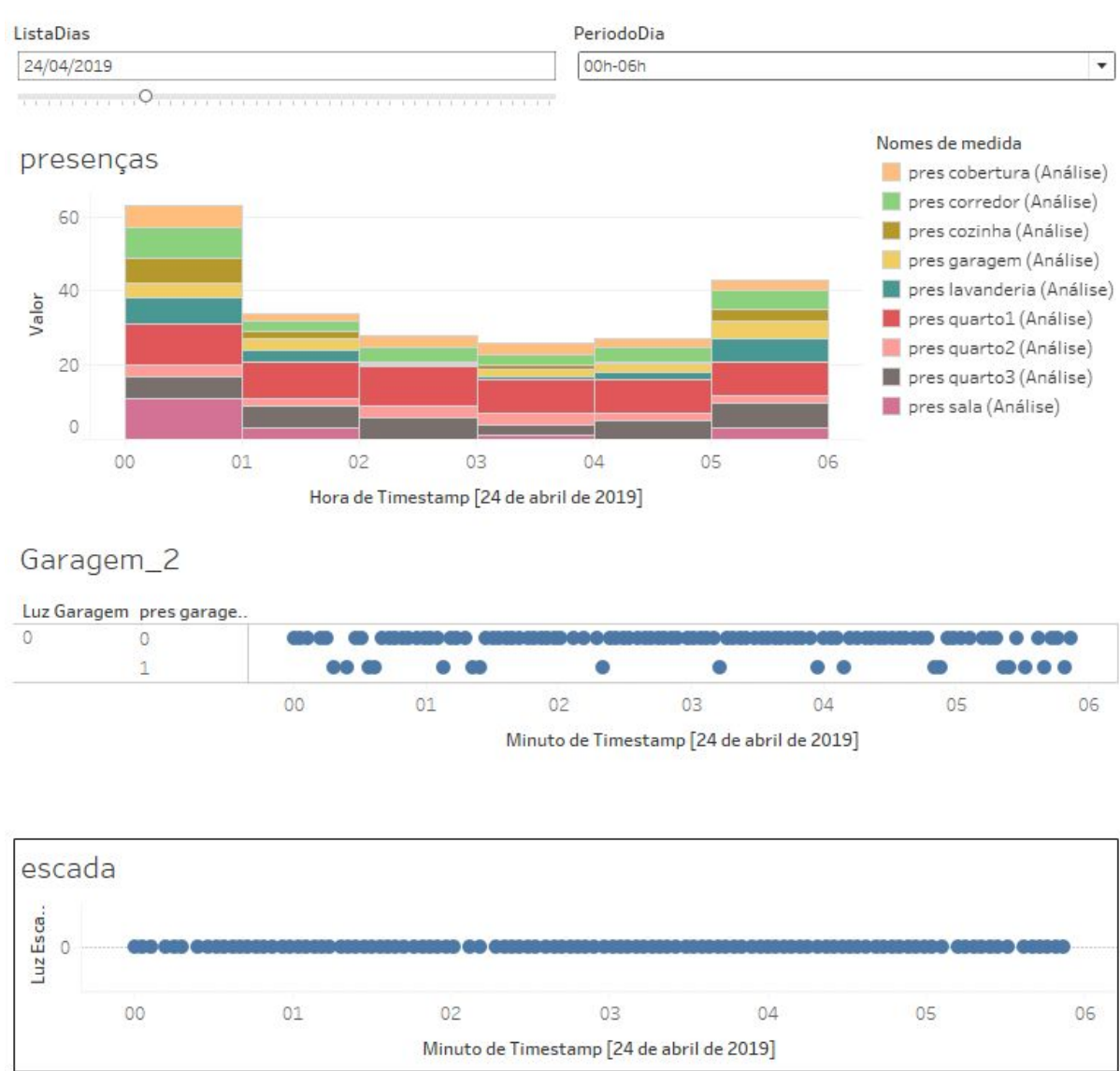
Estas bases fornecem registros obtidos entre 14/04/2019 e 27/05/2019.

## 6.2 Estudo de comportamento da casa

Com o conhecimento da estrutura da base fornecida e da planta da casa estudada, o próximo passo foi fundamental para endossar as tomadas de decisões para as etapas de classificação e *Random Forest*: o estudo de padrões de comportamentos da casa, considerando variáveis como dia útil/fim de semana, período do dia, ordem de registros nos cômodos, além dos próprios valores registrados na base pelos sensores.

Para facilitar a visualização das combinações de valores e correlações entre cômodos, foi utilizada a ferramenta de visualização de dados Tableau, e a dashboard produzida está exemplificada na imagem abaixo:

Figura 12 - Dashboard Tableau



Fonte: Autores

A dashboard Tableau consiste na visualização dos estados de cada cômodo de acordo com a combinação de saídas produzidas por seus sensores relacionados, de acordo com uma faixa de horários determinada para um dia específico. Este recurso foi essencial para produzir como resultado um estudo aprofundado do comportamento da casa, que serve como consulta para a justificativa dos rótulos criados para etapa de classificação, bem como a conformidade da etapa de verificação de padrões.

## 6.3 Segmentação

Como citado por Aminikhanghahi e Cook [5]: “Consideramos o problema de detectar transições de atividade como um problema de detecção de *change point* em dados de séries temporais. Essa suposição é baseada na hipótese de que as transições de atividade podem ser detectadas sem conhecer as categorias de atividade a priori, avaliando apenas as mudanças nas distribuições de dados do espaço de recurso.” Valendo-se dessa afirmação, a primeira etapa será a definição de um mecanismo que consiga diferenciar e definir estados de acordo com uma sequência de eventos e suas características (*features*) relacionadas.

Nossa metodologia de segmentação é baseada de acordo com uma função densidade de probabilidade (FDP), onde dada uma série temporal TS assume-se que o registro de tempo  $t$  é um ponto de transição se a função densidade de probabilidade  $f$  criada a partir de dados observados antes de  $t$  é suficientemente diferente dos dados observados imediatamente após  $t$ . Contudo, o autor faz esse tipo de análise em um cenário monousuário, enquanto que estamos trabalhando com um cenário multiusuário, e dispunha de uma infraestrutura maior do que a nossa.

Pensando nisso decidimos alterar o algoritmo AR-SS para se adequar ao nosso projeto, onde em vez de utilizar uma FDP para fazer a segmentação, utilizamos a seguinte regra para fazer a segmentação: a partir de um momento  $T$ , caso o estado dos sensores em  $T+1$ ,  $T+2$  e  $T+3$  for diferente de  $T$ . O ponto  $T$  é considerado ponto de transição.

Feitas essas considerações, a primeira etapa implementada em código pode ser verificada pelo código “segmentacao.py”, presente no repositório GitHub deste projeto. Este módulo desempenha a função de ler a base de dados “home.csv” e identificar momentos de transições de cada cômodo da casa, baseando-se em mudanças nos valores dos sensores relacionados, que delimitam os segmentos caracterizadores das atividades desses cômodos. A fim de evitar que segmentos muito longos possam comprometer a avaliação periódica do sistema, este módulo quebra segmentos de longa duração em segmentos menores de aproximadamente 15 minutos de duração.

## 6.4 Classificação

Com a segmentação finalizada, o próximo passo é categorizar as atividades segmentadas, de forma a fornecer uma classificação mais apropriada destas, uma vez que a etapa anterior define a separação dos segmentos baseada em mudança de valores de sensores, porém não nomeia o que um determinado conjunto de valores representa. Para efeito de comparação, uma criança em processo de aprendizado sabe diferenciar a cor preta da cor branca baseado em seus aspectos, porém em seu processo de alfabetização lhe é ensinado que a cor mais escura chama-se “preto”, enquanto a cor mais clara é chamada de “branco”, sendo a nomenclatura dessas cores um conhecimento previamente adquirido e repassado adiante por outras pessoas.

Para isso, as atividades segmentadas serão diferenciadas através de categorias (rótulos), definidas de acordo com um conjunto de valores dos sensores da casa que cada segmento carrega consigo.

### 6.4.1 Arquivos de configuração

Levando em consideração o exemplo dado acima, a abordagem de classificação dos segmentos em nosso projeto também é baseada em conhecimento prévio, determinada pelos autores através de 3 arquivos de configuração principais, que servem como leitura para o processamento neste módulo.

#### 6.4.1.1 Arquivo de atividades

Caracterizado pelo nome “atividades.csv”, contém os rótulos de cada atividade e algumas features importantes relacionadas, como período do dia, horário de início e fim da atividade, categoria do dia (dia útil ou fim de semana), se possui um cômodo específico relacionado, entre outros. Na tabela abaixo, segue uma relação entre as colunas do arquivo e suas respectivas funcionalidades:

Tabela 6 - Campos do arquivo de atividades

<b>Nome</b>	<b>Função</b>
id_atividade*	Identificador da atividade registrada
desc_atividade*	Rótulo da atividade para classificação
periodo_dia*	1 - Madrugada (00h-06h); 2 - Manhã (06h-12h); 3 - Tarde (12h-18h); 4 - Noite (18h-00h)
hora_1	Se preenchido, descreve um horário mínimo em que a atividade deverá ser reconhecida
hora_2	Se preenchido, descreve um horário máximo em que a atividade deverá ser reconhecida
categ_dia*	0 - Indiferente; 1 - Dia útil (Segunda a Sexta-feira); 2 - Fim de semana (Sábado e Domingo)
id_comodo*	Se preenchido com zero, não atribui cômodo específico à atividade; caso contrário, relaciona com o id_comodo registrado no arquivo de cômodos e sensores
valor_luz	Se preenchido, descreve o valor lido do sensor de luz relacionado ao cômodo para o reconhecimento da atividade. Exemplo: se valor_luz = 1, a atividade somente será reconhecida se a luz estiver obrigatória acesa
valor_aparelho	Se preenchido, descreve o valor lido do sensor de aparelho relacionado ao cômodo para o reconhecimento da atividade

(\*) Campo obrigatório

Fonte: autores

### 6.4.1.2 Arquivo de cômodos e sensores

Caracterizado como “id\_comodo\_sensores.csv”, determina a relação entre os cômodos e seus respectivos sensores, separados por sensor de luz, presença e aparelho.

Os campos id\_luz, id\_presenca e id\_comodo correspondem às posições das colunas dos sensores relacionados no arquivo “home.csv”, considerando zero como índice inicial. Exemplo: para o cômodo sala, são registrados os valores id\_luz = 2, id\_presenca = 15 e id\_aparelho = 27, sendo estes índices correspondentes às 3º, 16º e 28º coluna do arquivo “home.csv”, respectivamente.



Tabela 7 - Campos do arquivo de cômodos e sensores

<b>Nome</b>	<b>Função</b>
id_comodo*	Identificador do cômodo registrado
desc_comodo*	Rótulo do cômodo
tipo*	Caracteriza se o cômodo possui 1 sensor (luz), 2 sensores (luz e presença) ou 3 sensores (luz, presença e aparelho)
id_luz	Identificação do sensor de luz do cômodo na base “home.csv”
id_presenca	Identificação do sensor de presença do cômodo na base “home.csv”
id_aparelho	Identificação do sensor de aparelho do cômodo na base “home.csv”

(\*) Campo obrigatório

Fonte: autores

### 6.4.1.3 Arquivo de correlação entre sensores

Caracterizado como “correlacao\_sensores.csv”, contém uma relação entre pares de sensores correlacionados a uma mesma atividade. Exemplo: a atividade de acordar apenas será reconhecida caso haja atividades recentes nos sensores de presença da sala e cozinha.

Caso a atividade não contenha correlação entre sensores e possua um único sensor de cômodo relacionado, os índices do sensor e da atividade devem ser preenchidos nos campos id\_sensor\_1 e id\_atividade, respectivamente, deixando o campo id\_sensor\_2 em branco.

Tabela 8 - Campos do arquivo de correlação de sensores

<b>Nome</b>	<b>Função</b>
id_sensor_1*	1º sensor do par de correlação
id_sensor_2	2º sensor do par de correlação
id_atividade*	Id da atividade relacionada ao par de correlação

(\*) Campo obrigatório

Fonte : autores

## **6.4.2 Classificação**

Para fins de desempenho e eleição apropriada de atividades elegíveis para classificação, esta etapa é realizada em 2 níveis, explicados a seguir.

### **6.4.2.1 Classificação nível 1**

Nesta etapa são verificados os registros dos arquivos de configuração de atividades e de cômodos e sensores, citados no item 6.4.1, aplicando as regras citadas nas descrições de campos referentes aos arquivos de configurações.

Como resultado, é gerado um arquivo de saídas (“`classif_results_lv1.csv`”) que representa a classificação realizada em um primeiro nível. Este arquivo representa atividades candidatas a uma classificação final, pois algumas dessas atividades necessitam de uma correlação de cômodos e/ou sensores entre elas para serem caracterizadas, e este primeiro nível de classificação não representa uma condição suficiente para esse cenário. As atividades que necessitam de definição de sensores correlatos são aquelas que não pertencem a um cômodo específico.

### **6.4.2.2 Classificação nível 2**

Para solucionar o problema exposto na classificação nível 1, partimos para uma abordagem de correlação de sensores, através dos critérios para configuração do arquivo citado em 6.4.1.3. Como resultado, é gerado um arquivo de saídas (“`classif_results_lv2.csv`”).

## **6.5 Verificação de padrões (Random Forest)**

Esta etapa será realizada por um Random Forest treinada para aprender o hábito da residência e a partir disso avaliar se uma dada atividade reconhecida pelo sistema está dentro ou fora dos padrões. Caso o sistema avalie a atividade como sendo anormal, deverá emitir alertas de acordo com o grau de periculosidade da atividade.

Antes de executar o treinamento, os dados precisam receber uma preparação para que possam ser lidos de forma mais eficiente pelo algoritmo. A primeira parte dessa preparação está no módulo “Geração de features para Random Forest”, que conta com funções que atribuem features relacionadas aos registros processados pela execução da classificação nível 2. Estas features não estão presentes explicitamente na forma de dados brutos, porém se calculadas podem ser muito úteis para a etapa de verificação de padrões com Random Forest.

Tabela 9 - Features adicionais geradas para treinamento de Random Forest

<b>Nome</b>	<b>Função</b>
lig_desl_luz	Quantidade de mudanças no estado de luz dentro do segmento analisado
lig_desl_aparelho	Quantidade de mudanças no estado de aparelho dentro do segmento analisado
tempo_lig_luz	Porcentagem do tempo em que a luz permanece acesa dentro do período do segmento analisado
tempo_lig_aparelho	Porcentagem do tempo em que o aparelho permanece ligado dentro do período do segmento analisado

Fonte: autores

Com as features adicionais geradas, é hora de realizar o treinamento e testes do Random Forest. A segunda parte da preparação de dados para treinamento consiste em selecionar a partir da saída do módulo “Geração de Features” apenas colunas que são relevantes para treinamento, além de conversão de dados não-numéricos (categóricos) através de codificação One-Hot. Como resultado, é produzido o modelo de dados de acordo com a figura 13:

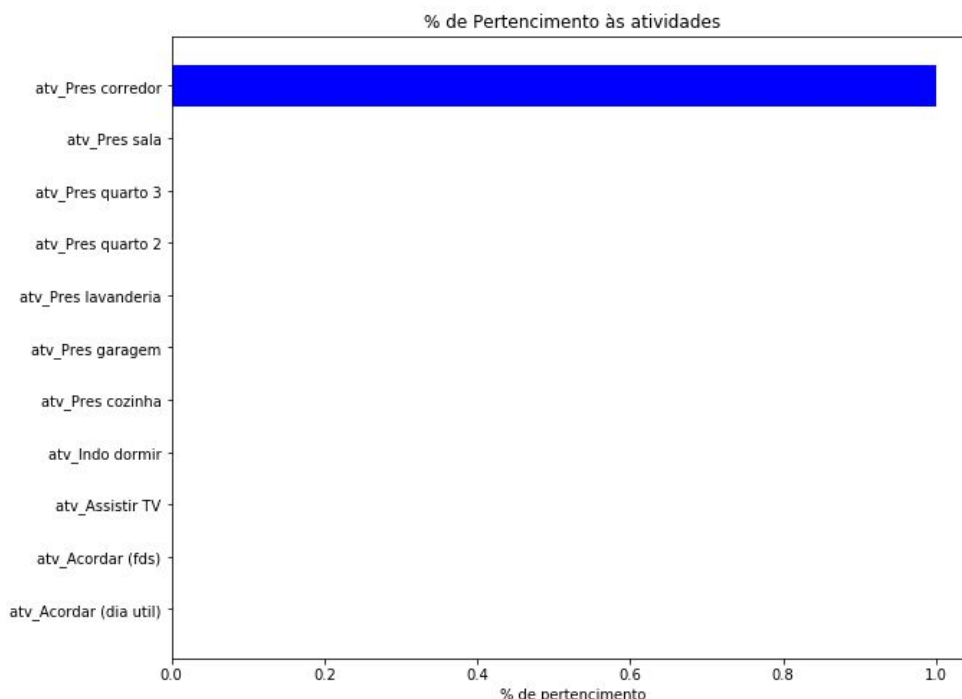
Figura 13 - Features selecionadas para treinamento de Random Forest

	Hora	Minuto	Fim de Semana	Comodo	lig_desl_luz	lig_desl_aparelho	tempo_lig_luz	tempo_lig_aparelho	Atividade
0	13	28	1	Cozinha	0	0	1.0	0.0	Pres cozinha
1	13	28	1	Sala	0	0	1.0	0.0	Pres sala
2	13	28	1	Quarto 3	0	0	0.0	0.0	Pres quarto 3
3	13	28	1	Garagem	0	0	0.0	0.0	Pres garagem
4	13	34	1	Cozinha	0	0	1.0	0.0	Pres cozinha
...	...	...	...	...	...	...	...	...	...
9709	22	54	0	Sala	0	0	0.0	0.0	Pres sala
9710	23	4	0	Cozinha	0	0	1.0	0.0	Pres cozinha
9711	23	4	0	Sala	0	0	0.0	0.0	Pres sala
9712	23	4	0	Lavanderia	0	0	1.0	0.0	Pres lavanderia
9713	23	4	0	Corredor	0	0	0.0	0.0	Pres corredor

Fonte: Autores

Por fim, é realizado o treinamento, utilizando os registros preparados de todos os dias de registro exceto do último, pois este será utilizado para os diferentes cenários de testes a serem relatados. Com o treinamento do algoritmo e comparação da amostra de testes, o algoritmo devolve a probabilidade de uma amostra de features pertencer a cada atividade, como relatado na figura 14, que relata um cenário de absoluta certeza quanto à atividade pertencente a tal amostra:

Figura 14 - Distribuição de probabilidades entre possíveis atividades



Fonte: Autores

Podem existir também situações em que o algoritmo não possui tanta certeza quanto ao grau de pertencimento às categorias de atividades candidatas. Isso pode ocorrer com a distribuição de probabilidades entre categorias, bem como uma distribuição próxima de 50/50 em relação a duas categorias candidatas. Aproveitamos esse critério de avaliação pela Random Forest para determinar graus de anormalidade de uma atividade, a fim de gerar alarmes a partir disso, adotando thresholds baseados na máximo valor de probabilidade observado dentro da distribuição probabilística de cada amostra analisada:

Tabela 10 - Thresholds para níveis de anormalidade

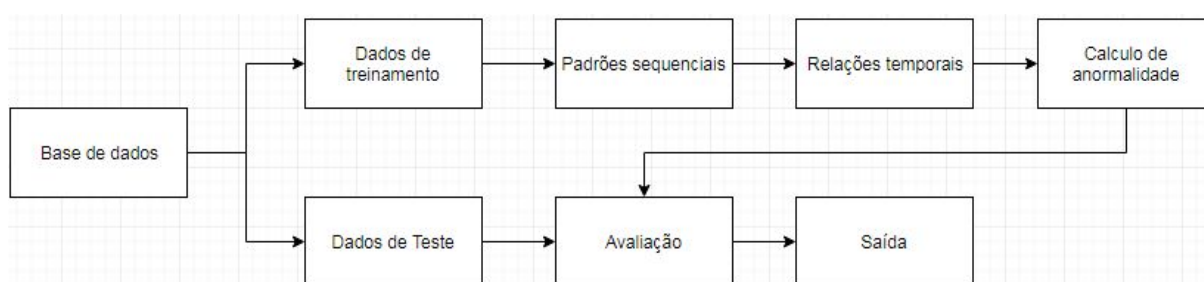
Máxima probabilidade na distribuição (X)	Nível de anormalidade
$X \geq 90\%$	Normal (sem anormalidade)
$75\% \leq X < 90\%$	Baixo
$60\% \leq X < 75\%$	Médio
$X < 60\%$	Alto

Fonte: Autores

## 6.6 Algoritmo de série temporal

Nesta etapa, foi criada uma linha de trabalho paralela à principal para o desenvolvimento desta solução que utilizaremos para validar nossa proposta principal. Esta solução se consiste basicamente na implementação da proposta de um artigo [\[30\]](#), sendo sua arquitetura representada pela figura 15.

Figura 15 - Arquitetura do algoritmo de série temporal



Fonte: autores

Primeiramente a base de dados é separada em 2 conjuntos: um conjunto de treino e outro de teste que segue a mesma lógica descrita na seção 4.5, onde é utilizado dois meses para treino e um dia para teste. A seguir usa-se o algoritmo AprioriAll [\[1\]](#) com mínimo suporte de 95% para se encontrar os padrões sequenciais mais frequentes presentes no conjunto de treino, após isso os elementos que compõem cada vetor são separados um a um, eliminando elementos repetidos.

Para a etapa seguinte cada um desses elementos são pareados 2 a 2, sendo que um par (A, B) é diferente de um par (B, A), a fim de se obter as relações temporais de cada par, nesta etapa são utilizadas as relações de Allen [\[29\]](#). Entretanto as únicas relações temporais que serão úteis para este projeto são: *before*, *contains*, *overlaps*, *meets*, *starts*, *started-by*, *finishes*, *finished-by* e *equals*, pois queremos detectar uma anomalia no momento que ela ocorre, não depois. Estas relações são então tabeladas para poderem ser utilizadas no cálculo de anormalidade de evento de um sensor.

O cálculo de anormalidade é feito para todos os pares, sendo primeiramente calculado a probabilidade que um sensor C será acionado. O suporte para esta

probabilidade baseia-se na ocorrência de outro evento que tenha uma relação temporal com C. Considere a probabilidade de ocorrência de C, uma vez que um sensor B foi acionado, a fórmula para se calcular  $P(C|B)$  é dada pela equação na figura 14.

Figura 16 - Fórmula para o cálculo de probabilidade

$$P(C|B) = (|Before(B, C)| + |Contains(B, C)| + |Overlaps(B, C)| + |Meets(B, C)| + |Starts(B, C)| + |StartedBy(B, C)| + |Finishes(B, C)| + |FinishedBy(B, C)| + |Equals(B, C)|) / |B|$$

Fonte: [\[30\]](#)

Podemos também calcular a probabilidade de um dado X ocorrer baseado na ocorrência de outros 2 sensores pela seguinte fórmula:  $P(C|A \cup B) = P(C \cap (A \cup B)) / P(A \cup B)$ . Finalmente podemos calcular o valor de anormalidade de um sensor usando a seguinte fórmula:  $A(C|B) = 1 - P(C|B)$

Por fim, para a avaliação do acionamento de um dado sensor X verifica-se se o valor de anormalidade deste sensor não ultrapassa um dado *threshold*, *threshold* este que é calculado pela média dos valores de anormalidade de todos os pares mais 2 vezes o desvio padrão. Dois desvios padrões da média representam aproximadamente 95% da população de dados, portanto, qualquer valor que não esteja nesta população será reportado como anormal.

## **7. Testes e Avaliação**

Como dito na seção 4.5., os testes se dividirão em duas etapas, sendo uma delas para a detecção de falsos positivos e a outra para a detecção de falsos negativos.

### **7.1 Comportamentos anormais**

Para este cenário os testes foram separados em 2 tipos: randômicos e específicos. Os testes randômicos servem para demonstrar que a nossa abordagem consegue reconhecer comportamentos anômalos de natureza aleatória, enquanto que os específicos visam simular situações de cenários anômalos específicos na casa.

Nos testes específicos consideramos 2 tipos de cenários, um onde a casa é invadida e outro onde é recebido um convidado pelos moradores. Para criarmos estes cenários alteramos os registros na nossa base de testes de forma a simular os cenários propostos. No cenário de invasão alteramos os registros na base entre 03:00h - 04:00h, 15:02h - 16:17h e 20:05 - 21:14. Enquanto que para o cenário de visita alteramos das 15:12h - 19:14h. O cenário de invasão é representado pelo acessos de cômodos na sequência garagem - sala - cozinha - corredor - quartos, com movimentação constante no corredor, sala, cozinha e quartos. Já o cenário de visita é representado pela sequência garagem - sala - lavanderia.

#### **7.1.1 Teste Randômico**

Este teste servirá para avaliar se nossa solução consegue detectar comportamentos anormais de uma forma geral.

##### **7.1.1.1 Série temporal**

Ao realizarmos o teste randômico percebemos que o sistema nunca acusava qualquer sequência de acionamento de sensores como anormal devido a alta dispersão nos dados de treino. Entretanto, a avaliação deste método é binária, ou seja, existem apenas 2 valores possíveis para avaliação, anômalo ou normal, o que

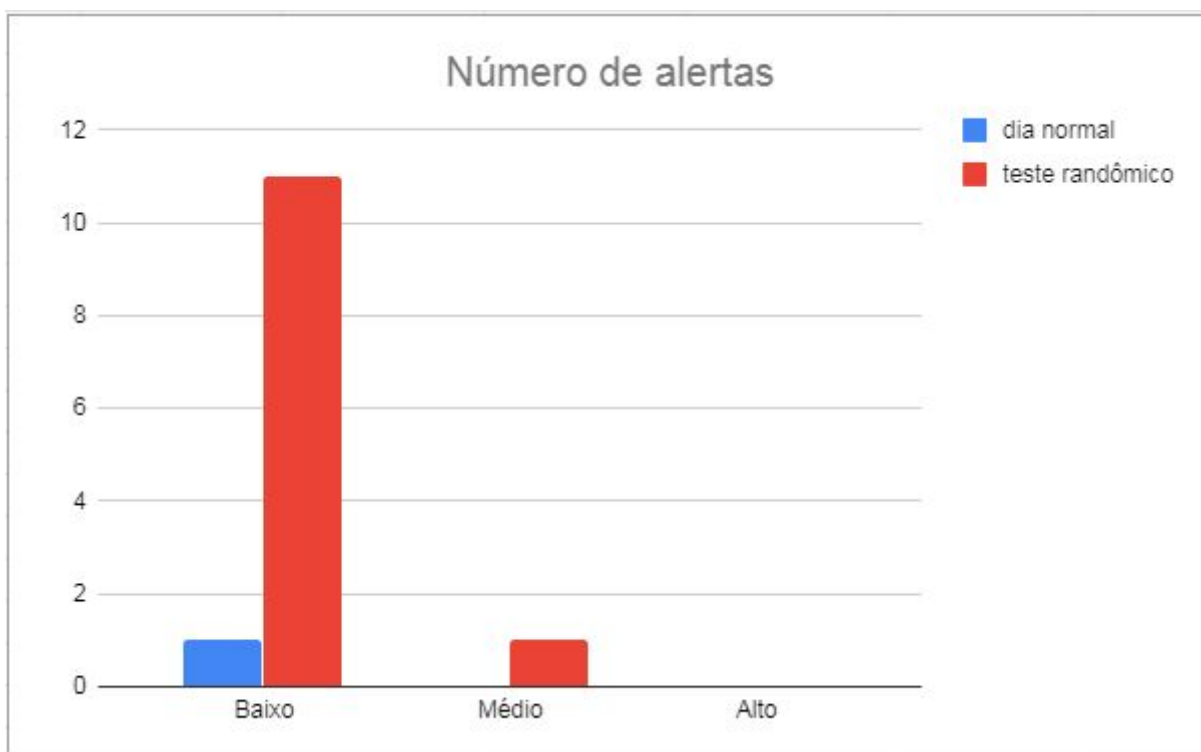


difere um pouco das especificações do sistema descrito na seção 5.3 que contêm graus de anormalidade para as atividades avaliadas como anômala. Além disso é mencionado no artigo [30] que apenas os acionamentos anômalos que são considerados “surpreendentes” o suficiente são reportados, ou seja, apenas os comportamentos que representem risco aos residentes são reportados, o que é equivalente ao nosso grau de anormalidade alto. Com isso podemos afirmar que não foi encontrado nenhum comportamento anormal de nível alto.

Após alguns testes verificamos que os valores de *threshold* para vários dias era de aproximadamente 75%, portanto, decidimos adotar um novo valor para o *threshold* baseado em níveis de periculosidade, sendo mostrados a seguir:

- Baixo -  $80\% < X \leq 89\%$
- Médio -  $89\% < X \leq 94\%$
- Alto -  $94\% < X$

Figura 17 - Número de alertas em cenário randômico de série temporal



Fonte: autores

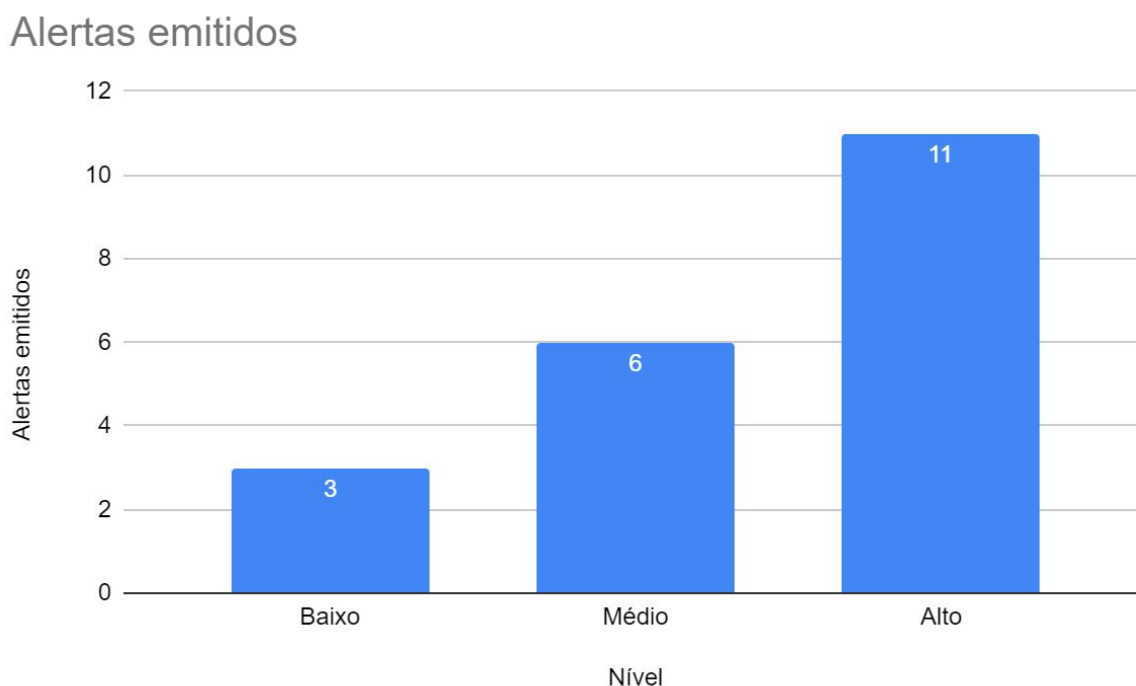
Com a implementação dos graus de anormalidade, percebemos que o sistema apresenta uma grande quantidade de alertas para o teste randômico em comparação com o número de alertas gerados por um dia normal, o que nos leva a crer que o sistema consegue detectar comportamentos anômalos na casa.

### 7.1.1.2 Random Forest

Para a geração de amostra randômica de anormalidade, foram alterados 40 registros aleatórios da amostra de teste, escolhendo-se 2 features aleatoriamente a cada registro analisado. As atividades candidatas para avaliação em todos os cenários são: “Acordar (dia útil)”, “Acordar (fds)”, “Assistir TV”, “Indo dormir”, “Pres corredor”, “Pres cozinha”, “Pres garagem”, “Pres lavanderia”, “Pres quarto 2”, “Pres quarto 3” e “Pres sala”.

Com a execução do treinamento e teste a partir destes cenários, os seguintes números de alertas de acordo com o critério demonstrado conforme a tabela 10:

Figura 18 - Número de alertas no cenário randômico Random Forest



Fonte: Autores

Estes números mostram que o Random Forest é capaz de reconhecer certos graus de anormalidade de atividade de acordo com sua base de treino, com grau de criticidade alto, apresentando um comportamento dentro do esperado.

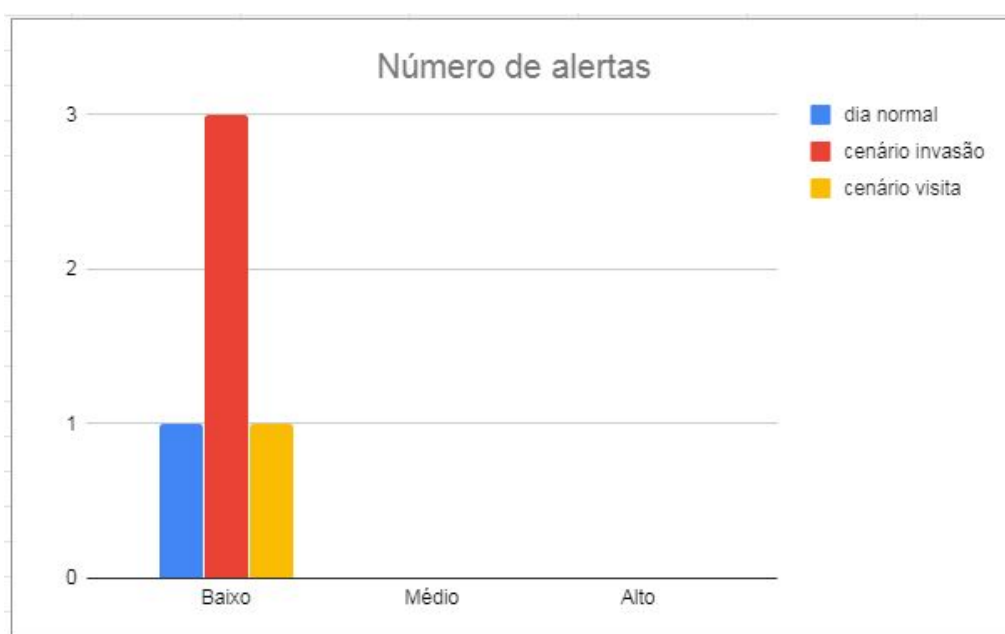
## 7.1.2 Teste específico

Neste teste é avaliado se a solução proposta consegue detectar um comportamento anormal em um cenário específico elaborado previamente.

### 7.1.2.1 Série temporal

Ao analisar as figuras 19 e 20 percebemos que o algoritmo conseguiu identificar 3 casos de atividade anômala para o cenário de invasão, onde cada índice dos vetores no registro de log representa respectivamente o sensor acionado, a hora que foi acionado, a probabilidade dele ser acionado, seu valor de anormalidade e o nível de alerta. Dentre os casos identificados como anômalos apenas 2 foram oriundos das alterações feitas na base de testes, sendo eles o primeiro e o último registro de log. Enquanto que para o cenário de visita o algoritmo não conseguiu distinguir qualquer atividade anômala além da já detectada na base de teste.

Figura 19 - Número de alertas em diferentes cenários de série temporal



Fonte: autores

Figura 20 - Registro de log para cenário de invasão

```
> 0: ['luz_corredor', '2019-05-27 15:24', 0.19633772727996998, 0.80366227272003, 'baixo']
> 1: ['luz_corredor', '2019-05-27 19:59', 0.16702012714301057, 0.8329798728569895, 'baixo']
> 2: ['luz_corredor', '2019-05-27 20:18', 0.13967081360834321, 0.8603291863916568, 'baixo']
```

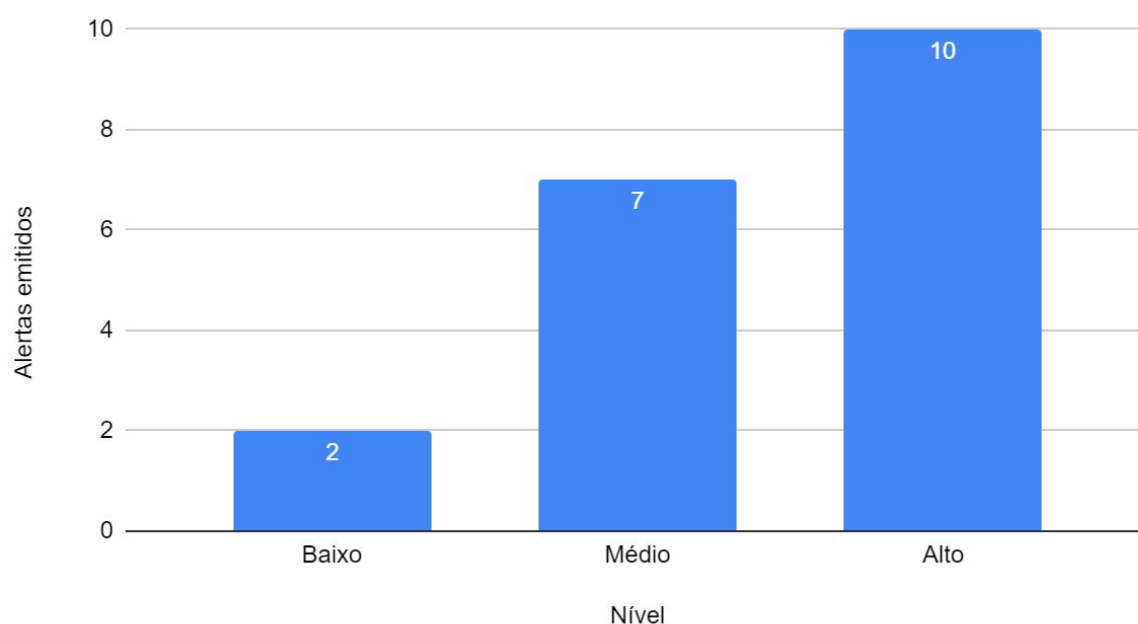
Fonte: autores

### 7.1.2.2 Random Forest

A seguir estão os resultados referentes ao cenário de invasão descrito no capítulo 7.1:

Figura 21 - Número de alertas em cenário de invasão Random Forest

#### Alertas emitidos



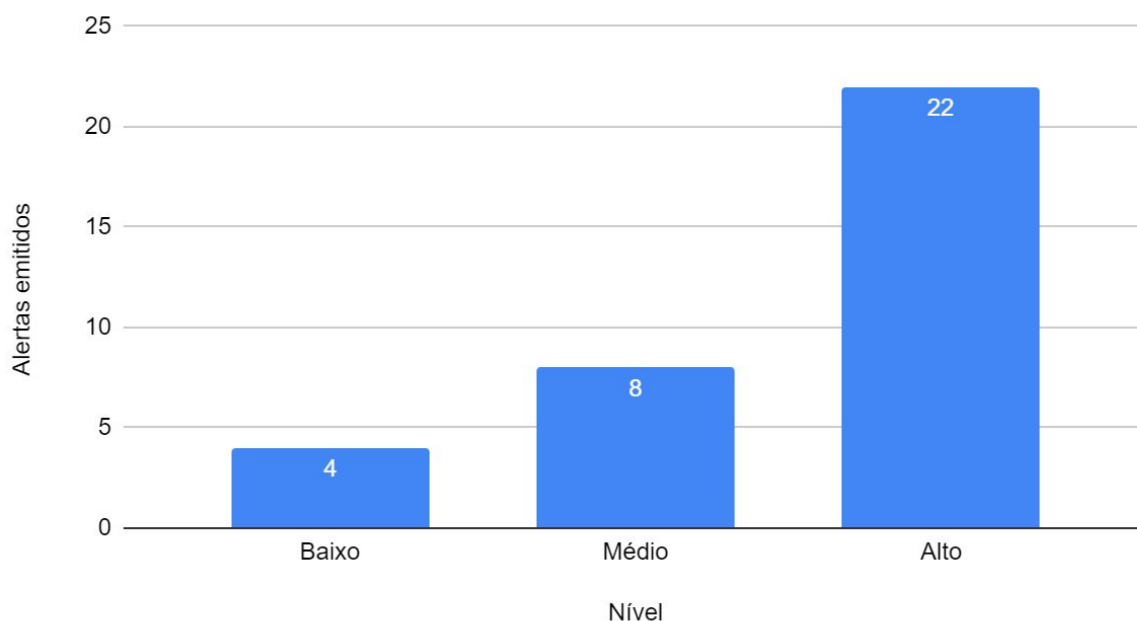
Fonte: Autores

Novamente, são notados alertas de nível alto em sua maioria, o que a princípio condiz com a situação esperada. Porém, nenhum desses alertas está relacionado com movimentação da casa no período entre 3:00 e 4:00 da manhã, representando uma notável brecha de segurança. Isso ocorreu devido a ruídos nos registros da base de treinamento, que conta registros esporádicos no período

supracitado. A fim de representar uma situação mais condizente com o trabalho da casa, realizamos a limpeza de quaisquer registros da casa no período entre 2:00 e 5:00, período este em que todos seus residentes estão inativos. Com a nova base tratada, repetimos o procedimento, observando mudanças nos números obtidos:

Figura 22 - Número de alertas em cenário de invasão Random Forest tratado

### Alertas emitidos



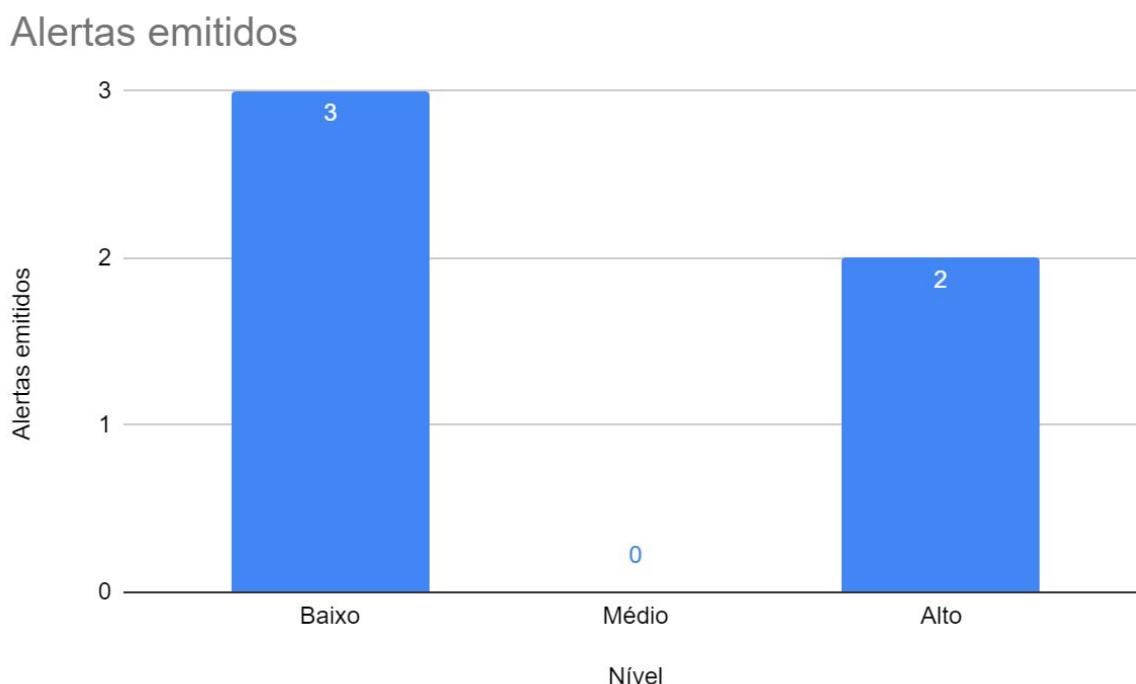
Fonte: Autores

O crescimento no número de alertas de nível alto agora se justifica, uma vez que são reconhecidas atividades de movimentação da casa no período de 3:00 a 4:00 da manhã, que desta vez não estão previstos dentro da nova base de treinamento.

Em relação aos demais alertas de nível alto, estes podem ser justificados por diversos motivos. O principal argumento para isso baseia-se ambiguidade entre uma atividade de invasão e movimentação comum na casa durante o dia de acordo com o modelo de dados fornecido, o que eventualmente pode caracterizar a insuficiência de variáveis envolvidas para a diferenciação dessas atividades ambíguas.

Em relação ao cenário de visita, foram observados os seguintes números:

Figura 23 - Número de alertas em cenário de visita Random Forest



Fonte: Autores

Os alertas de nível alto correspondem a um cenário de dúvida entre atividade de dormir e presença em um cômodo, o que representa uma brecha de ambiguidade em nosso algoritmo. Por outro lado, a visita realizada durante o período definido em 7.1 passou despercebida pelo algoritmo, o que tem o lado de positivo de não representar um falso cenário de ameaça, ao mesmo tempo que não detecta uma leve alteração no comportamento da casa para o horário.

## 7.2 Comportamentos normais

### 7.2.1 Série temporal

Neste cenário o algoritmo de série temporal foi treinado com uma base de dados dos meses de Abril e Maio/19, sendo que o último dia do mês de maio foi retirado da base de treino para ser utilizado na validação do algoritmo, obtendo os seguintes resultados:

Figura 24 - Médias e desvios padrões no comportamento normal de série temporal

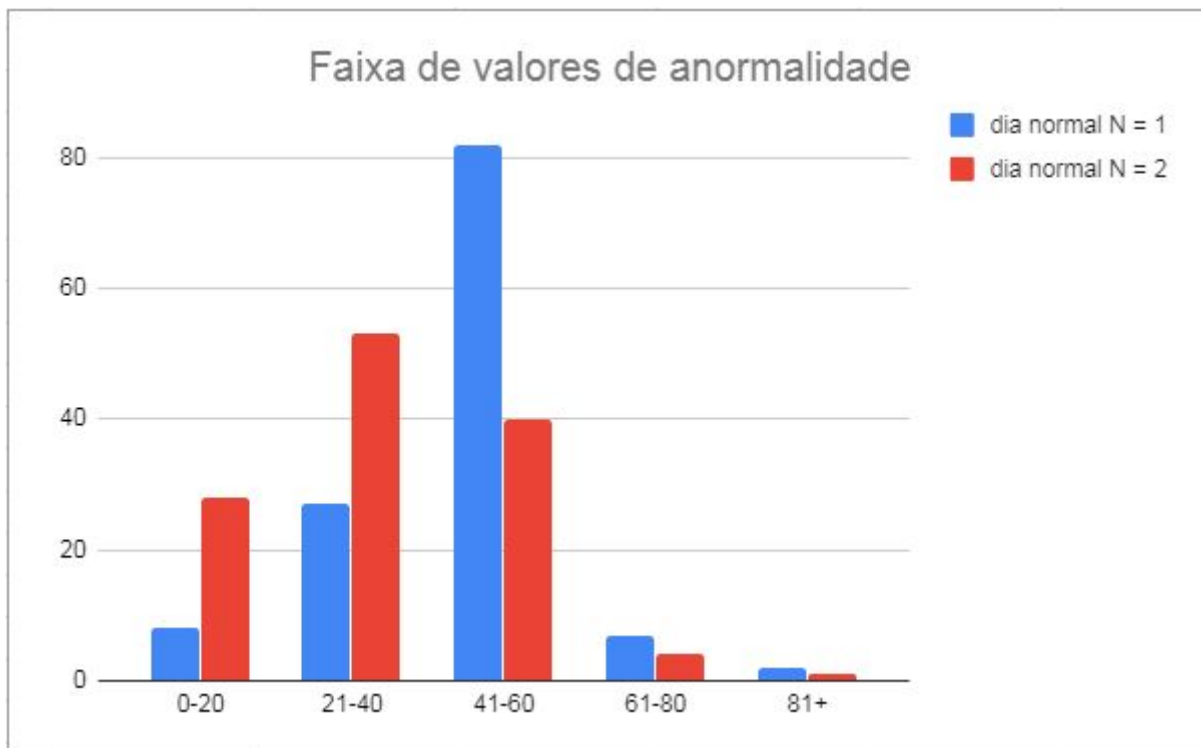
```
desv_pad_hist_N_1: 0.19236356098990606
media_hist_N_1: 0.562346725699602
desv_pad_hist_N_2: 0.23047729960017424
media_hist_N_2: 0.47922377475650846
desv_pad_N_1: 0.13784734098170726
media_N_1: 0.4574389735695953
desv_pad_N_2: 0.17669276422654195
media_N_2: 0.3323153150091101
```

Fonte: autores

Como dito na seção 6.6, o algoritmo de série temporal pode avaliar o acionamento de um sensor com base no último sensor acionado ou com base nos últimos N sensores acionados. A figura 24 mostra os desvios padrões e médias tanto para N = 1 e N = 2 para um dia, quanto para a série histórica.

- “desv\_pad\_hist\_N\_1” - desvio padrão para N = 1 dos dados de treino
- “media\_hist\_N\_1” - média dos valores de anormalidade para N = 1 dos dados de treino
- “desv\_pad\_hist\_N\_2” - desvio padrão para N = 2 dos dados de treino
- “media\_hist\_N\_2” - média dos valores de anormalidade para N = 2 dos dados de treino
- “desv\_pad\_N\_1” - desvio padrão para N = 1 dos dados de teste
- “media\_N\_1” - média dos valores de anormalidade para N = 1 dos dados de teste
- “desv\_pad\_N\_2” - desvio padrão para N = 2 dos dados de teste
- “media\_N\_2” - média dos valores de anormalidade para N = 2 dos dados de teste

Figura 25 - Avaliação dos sensores acionados em um dia por faixa



Fonte: autores

Figura 26 - Número de alertas para o dia de teste



Fonte: autores



Podemos ver que com a adição de mais um sensor ( $N = 2$ ) na análise comportamental da casa, diminuiu significativamente o valor de anormalidade na avaliação dos sensores, entretanto isso aumenta levemente o desvio padrão, uma vez que o aumento do número de sensores na análise diminuiu o valor de anormalidade das atividades frequentemente realizadas (hábitos), enquanto mantém aproximadamente constante ou incrementa levemente os valores de anormalidade para atividades pouco realizadas, o que acentua a dispersão da amostra, isso pode ser verificado pela figura 25, onde temos um grande decréscimo na faixa de 41% a 60% de anormalidade, enquanto que as faixas superiores permanecem razoavelmente constantes. Além disso, percebemos que o *threshold* permanece constante (~94%) tanto para  $N = 1$  como para  $N = 2$ , o que nos leva a crer que a medida que aumentamos o valor de  $N$ , aumentamos a discriminação entre atividades normais e anômalas.

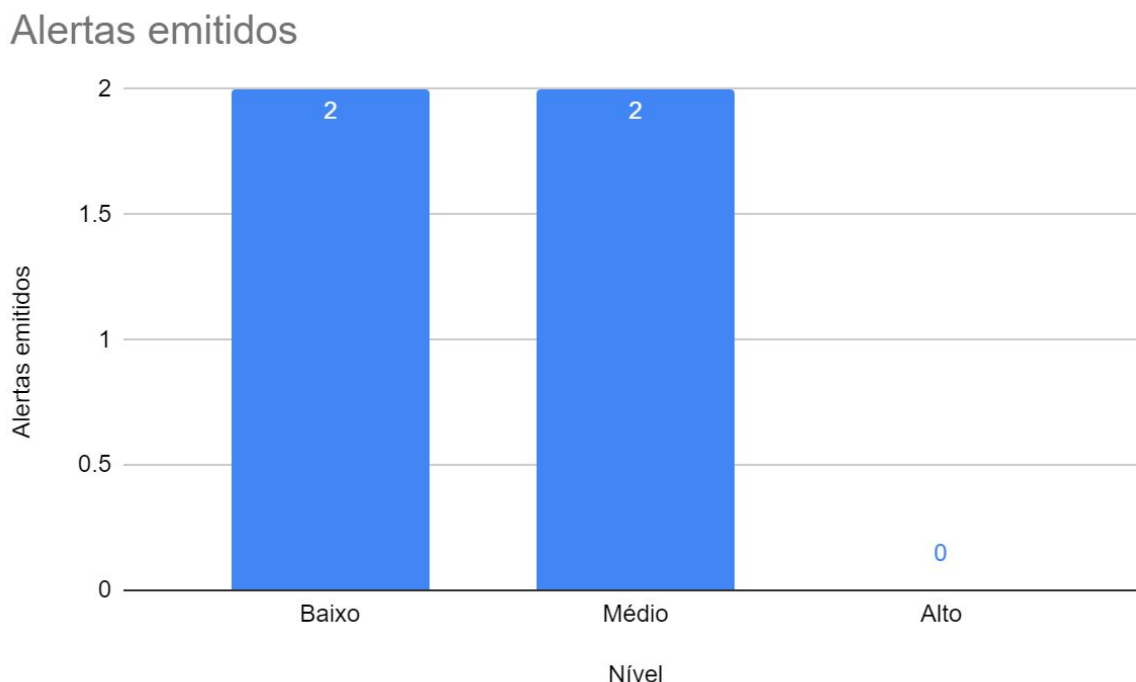
Outra coisa que podemos notar é a diferença entre as médias de anormalidade dos dados de teste e dos de treino, isso ocorre porque os residentes historicamente devem ter realizado atividades que estão fora do seu hábito diário o que é perfeitamente compreensível e normal, como acontece quando recebemos algum convidado, isto pode ser verificado pelo alto valor do desvio padrão.

Por fim, vemos que o algoritmo apenas acusou apenas um alerta de nível baixo para  $N = 2$  como mostrado na figura 26. Este seria um resultado promissor caso o desvio padrão da base de treino não fosse tão alto, o que dificulta a avaliação de atividades normais e anômalas como dito na seção anterior.

## 7.2.2 Random Forest

Para treinamento e teste de Random Forest, foi utilizado a mesma amostra de testes descrita no item 7.2.1. Após a execução da avaliação, foram obtidos os seguintes resultados:

Figura 27 - Número de alertas no cenário normal Random Forest



Fonte: Autores

Em relação aos cenários de anormalidade observados anteriormente, nota-se que o sistema emite alertas de menor criticidade, o que condiz muito com o cenário proposto.

Em relação aos falsos positivos relacionados (alertas baixos e médios), estes estão relacionados a tarefas que possuem um pequeno grau de ambiguidade entre elas, porém nada que prejudique a análise como um todo.

### 7.3 Avaliação

Analisando ambas as abordagens verificou-se que existem prós e contras em cada uma. Primeiramente a série temporal obteve a pior acurácia entre as duas abordagens, isso é devido ao grande valor de dispersão na base de treino, o que dificulta a detecção de atividades anômalas pelo algoritmo. Assim sendo, vemos que o método precisa de um valor de dispersão baixo para funcionar adequadamente e a única forma de se fazer isso é reduzir o número de sensores avaliados, ou seja, para se conseguir uma dispersão menor deverá ser aumentado o valor de mínimo

suporte, o que implicará na redução do número de sensores avaliado, e além disso a série avalia apenas uma sequência de acontecimentos, não sendo possível avaliar o instante de tempo em que ocorrem. O algoritmo de série temporal possui uma vantagem essencial para qualquer sistema de reconhecimento de atividades humanas, que é a avaliação de uma sequência de eventos, pois as atividades humanas são uma sequência de eventos complexa que é dificilmente avaliável. Sendo que esta característica não está presente na abordagem

Já o algoritmo de aprendizagem supervisionada por Random Forest se mostrou um pouco mais eficaz em certas circunstâncias, pois este leva em consideração uma quantidade maior de variáveis em relação ao algoritmo de série temporal, exceto nos casos de anormalidade onde pode haver ambiguidade entre atividades candidatas devido à má configuração das features ou insuficiência de features para distinguir atividades específicas. Por outro lado, o Random Forest não leva em consideração a sequência de acionamento de sensores, porém apenas a correlação entre eles, representando assim um ponto positivo para a série temporal.

Por fim, podemos ver que a série temporal se enquadra melhor a cenários onde temos uma rotina bem definida ou que tenham algumas atividades que sejam muito mais frequentes que o restante das atividades, enquanto que a aprendizagem supervisionada por Random Forest, por sua vez, fornece cobertura a quaisquer cenários, independente de sua frequência de acontecimento, desde que estes cenários estejam previstos nos arquivos de configuração citados no capítulo 6.4.1.

Visando aproveitar os benefícios oferecidos por cada abordagem estudada no projeto, pode-se utilizar alguma forma de integração entre estes 2 fluxos de reconhecimentos de atividades, como alguma forma de *ensemble* entre os algoritmos apresentados, por exemplo.

## 8. Considerações Finais

### 8.1 Conclusões do Projeto de Formatura

Em relação ao módulo de série temporal, não atingimos os resultados esperados, apesar de conseguirmos detectar alguns casos de atividades anômalas, isso é devido a dispersão na base de treino, como comentado anteriormente. Dentre as hipóteses levantadas para justificar a alta dispersão estão:

- A base de treino ser um cenário multiusuário (4 residentes na casa);
- A casa não ter um hábito bem definido;
- A base ser pequena;
- A fatores externos, como visitas ou obras durante a coleta de dados.

Dentre as hipóteses levantadas a mais provável para justificar o valor de dispersão é que este algoritmo não se adapta bem a um cenário multiusuário, pois cada residente pode ter uma rotina diferente e a intersecção dessas rotinas poluiu as relações temporais entre os sensores avaliados. Uma possível solução seria aumentar a base de treino a fim de diminuir o ruído que a intersecção das atividades causa nas relações temporais dos sensores.

Em relação ao aprendizado supervisionado por Random Forest, este atingiu parcialmente as expectativas criadas pelos autores do trabalho, uma vez que atividades anômalas em cenários atípicos demonstraram serem reconhecidas em certo grau, apesar da existência de um número considerável de falsos positivos, o que compromete de certa forma a acurácia do sistema desenvolvido até o presente momento. Por outro lado, se olharmos os módulos anteriores ao treinamento propriamente dito da Random Forest (segmentação, classificação e geração de features), estes demonstram ser uma forma mais assertiva e segura de definir categorias de atividades em uma residência, uma vez que a configuração de características particulares de atividades, cômodos e correlação entre atividades podem ser configuradas de forma prática pelo usuário, representando regras bem definidas a serem obedecidas por esses módulos.

## 8.2 Contribuições

O artigo do algoritmo de série temporal [\[30\]](#) apresenta algumas inconsistências que foram percebidas durante o desenvolvimento deste método, entre elas estão o fato de os autores usarem dados sintéticos para validar sua abordagem em vez de dados reais, durante o teste com os dados reais não foi encontrado nenhuma anomalia, por isso foram gerados dados sintéticos a fim de provar o ponto de vista dos autores. Outra inconsistência é que os autores dizem ser possível avaliar o acionamento de um dado sensor X a partir dos N últimos sensores acionados, entretanto, eles não fornecem uma forma de generalizar os valores de N, apenas deixam claro como fazer para  $N = 2$ .

Em relação à abordagem supervisionada por Random Forest e também ao projeto como um todo, foram propostas possibilidades de processos e variáveis a serem consideradas no que diz respeito a reconhecimento de atividades humanas em um ambiente conectado por dispositivos IoT. Apesar da abordagem apresentada não estar perto da ideal, o estudo realizado pelos autores ao decorrer do ano se mostra muito válido, servindo de provocação e inspiração para futuros voluntários que decidam se aprofundar em estudar e desenvolver metodologias mais eficazes de HAR.

## 8.3 Trabalhos futuros

O projeto proposto possui um escopo que pode ser explorado além do que foi proposto neste projeto, com a possibilidade de se tornar um sistema com muitas outras funcionalidades. Pensando nisso, a seguir são propostas algumas possíveis formas de continuar este projeto.

### 8.3.1 Redes neurais e Deep Learning

Uma possibilidade de tecnologia que pode ser utilizada em outra vertente de análise comportamental em nosso projeto está relacionada tanto com bibliotecas e frameworks da linguagem Python voltados para Deep Learning, como também serviços específicos de processamento de redes neurais na nuvem. Dentre as bibliotecas Python, podemos listar Keras [\[18\]](#), TensorFlow [\[19\]](#), PyTorch [\[20\]](#), entre

outras. Em relação aos serviços de processamento na nuvem, podemos listar serviços da Amazon [\[21\]\[22\]](#), Google [\[23\]\[24\]](#), entre outros. Neste caso, a rede neural parte de um método de aprendizado não-supervisionado que teria como entrada diretamente as entradas brutas fornecidas pelo arquivo de atividades “home.csv”, assim como já ocorre de maneira semelhante no flow de processos da série temporal.

### **8.3.2 Sistema de notificação ao usuário e ação**

Com a geração de alertas separados em níveis pelo nosso sistema, uma outra possibilidade de continuidade está relacionada a tomada de ações relacionadas a uma atividade anormal detectada. Isso pode ser implementado desde simples notificações nos smartphones dos residentes, até a tomada de alguma ação, como notificar a polícia.

### **8.3.3 Sistema de Healthcare**

Futuramente sistema que ajudam a cuidar de idosos serão muito importantes, pois a população mundial está envelhecendo. Pensando nisso nossa abordagem atual que foca apenas na detecção de anomalias pode ser incrementada para passar a detectar também ausência de atividades, como por exemplo um idoso que esqueceu de tomar o seu remédio matinal. Outro fator que ajuda na implementação desse sistema é que o algoritmo de série temporal pode detectar ausência de atividades, contudo como não era o foco do projeto não foi implementado.

## Referências

- [1] Agrawal, R. and Srikant, R. (1995). **Mining sequential patterns**. In Proceedings of the International Conference on Data Engineering, pp. 3-14
- [2] Mendes, R. D.; **Inteligência artificial: sistemas especialistas no gerenciamento da informação**. 1997.
- [3] P. Vepakomma, D. De, S.K. Das, S. Bhansali; **A-wristocracy: deep learning on wrist-worn sensing for recognition of user complex activities**. 2015 IEEE 12th International Conference on Wearable and Implantable Body Sensor Networks (BSN), IEEE (2015). pp. 1-6
- [4] Patel, K. K.; Patel, S. M.; **Internet of Things-IOT: Definition, Characteristics, Architecture, Enabling Technologies, Application & Future Challenges**. Faculty of Technology and Engineering-MSU, Vadodara, Gujarat, India, 2016. 10p.
- [5] Aminikhanghahi, S.; Cook D. J.; **Enhancing Activity Recognition using CPD-based Activity Segmentation**. Washington State University, Pullman, WA: School of Electrical Engineering and Computer Science, 2019. 35p.
- [6] Dahmen, J. et al. **Smart Secure Homes: A Survey of Smart Home Technologies that Sense, Assess, and Respond to Security Threats**. Washington State University, Pullman, WA: School of Electrical Engineering and Computer Science, 2017. 23p.
- [7] J. Qin, L. Liu, Z. Zhang, Y. Wang, L. Shao. **Compressive sequential learning for action similarity labeling**. IEEE Trans. Image Process., 25 (2) (2016), pp. 756-769
- [8] Y. Kim, B. Toomajian. **Hand gesture recognition using micro-doppler signatures with convolutional neural network**. IEEE Access, vol. 4. 2016. pp. 7125-7130.

- [9] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu; **Deep learning for sensor-based activity recognition: A Survey**. Pattern Recognit. Lett., Feb. 2018.
- [10] Y. Chen, Y. Xue. **A deep learning approach to human activity recognition based on single accelerometer**. Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on, IEEE (2015), pp. 1488-1492
- [11] W. Jiang, Z. Yin. **Human activity recognition using wearable sensors by deep convolutional neural networks**. MM, ACM (2015), pp. 1307-1310
- [12] T. Plötz, N.Y. Hammerla, P. Olivier. **Feature learning for activity recognition in ubiquitous computing**. IJCAI, 22 (2011), p. 1729
- [13] A. Wang, G. Chen, C. Shang, M. Zhang, L. Liu. **Human activity recognition in a smart home environment with stacked denoising autoencoders**. International Conference on Web-Age Information Management, Springer (2016), pp. 29-40
- [14] Hallé, S. Gaboury, S. Bouchard, B. **Activity Recognition Through Complex Event Processing: First Findings**. AAAI-16 Workshop on artificial intelligence applied to assistive technologies and smart environments (ATSE 2016), At Phoenix, AZ, USA. 6 p.
- [15] A. Alberdi et al. **Smart Home-Based Prediction of Multidomain Symptoms Related to Alzheimer's Disease**. IEEE Journal of Biomedical and Health Informatics, vol. 22, no. 6. 2018. pp. 1720-1731
- [16] YASSUDA et al. **Hedwig - Casa Conectada**. Trabalho de conclusão de curso (graduação em engenharia de computação) - Escola Politécnica da Universidade de São Paulo, São Paulo. 2017.



[17] HAYASHI, V. T. São Paulo. **Domus Smart Home Testbed**. Disponível em: <<https://github.com/vthayashi/domus>>. Acesso em: 23 jun. 2019.

[18] **Keras**. Disponível em: <<https://keras.io/>>. Acesso em: 23 jun. 2019.

[19] **TensorFlow**. Disponível em: <<https://www.tensorflow.org/>>. Acesso em: 23 jun. 2019.

[20] **PyTorch**. Disponível em: <<https://pytorch.org/>>. Acesso em: 23 jun. 2019.

[21] **Aprendizado profundo na AWS**. Disponível em: <<https://aws.amazon.com/pt/deep-learning/>>. Acesso em: 23 jun. 2019.

[22] **Crie, treine e implante modelos de Machine Learning com o Amazon SageMaker**. Disponível em: <<https://aws.amazon.com/pt/sagemaker/>>. Acesso em: 23 jun. 2019.

[23] **Google Cloud AI**. Disponível em: <<https://cloud.google.com/products/ai/>>. Acesso em: 23 jun. 2019.

[24] **Deep Learning VM Image | Deep Learning VM | Google Cloud**. Disponível em: <<https://cloud.google.com/deep-learning-vm/>>. Acesso em: 23 jun. 2019.

[25] **Which classifier is better, random forests or deep neural networks?**

Disponível em :

<<https://www.quora.com/Which-classifier-is-better-random-forests-or-deep-neural-net-works>>. Acesso em: 19 out. 2019.

[26] **Random Forests vs Neural Networks: Which is Better, and When?**

Disponível em:

<<https://www.kdnuggets.com/2019/06/random-forest-vs-neural-network.html>>.

Acesso em: 19 out. 2019.

[27] **GitHub - Smarthome TCC.** Disponível em: <[https://github.com/vrbarreira/smarthome\\_tcc](https://github.com/vrbarreira/smarthome_tcc)>. Acesso em: 19 out. 2019.

[28] **Data Mining (Mineração de Dados): O que é, conceito e definição | Blog Cetax.** Disponível em: <<https://www.cetax.com.br/blog/data-mining/>>. Acesso em: 19 out. 2019.

[29] Allen, J. F. and Ferguson, G. (1994). **Actions and events in interval temporal logic.** Journal of Logic and Computation, 4(5):531-579.

[30] Jakkula, R. V.; Crandall, S. A.; Cook D. J.; **Enhancing anomaly detection using temporal pattern discovery**

[31] **Os Três Tipos de Aprendizado de Máquina.** Disponível em: <<https://lamfo-unb.github.io/2017/07/27/tres-tipos-am/>>. Acesso em: 19 out. 2019.

[32] Gislason, P.O.; Benediktsson, J.A.; Sveinsson, J.A.; **Random Forests for land cover classification.** Department of Electrical and Computer Engineering, University of Iceland. 2006.

[33] Monard, M. C.; Baranauskas, J. A.; **Indução de Regras e Árvores de Decisão.** Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto da Universidade de São Paulo. p. 57-74.