INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA DE INTELIGÊNCIA ARTIFICIAL




Sistema Multiagente Inteligente Para Gestão Energética Baseado Em Dispositivos IoT Distribuídos

JOÃO FILIPE SOARES CAÇOTE
julho de 2022

Sistema Multiagente Inteligente Para Gestão Energética Baseado Em Dispositivos IoT Distribuídos

João Filipe Soares Caçote

Aluno nº: 1200132

Dissertação para obtenção do Grau de

Mestre em Engenharia de Inteligência Artificial

Orientador: Doutor Tiago Manuel Campelos Ferreira Pinto, Professor Adjunto Convidado do Instituto Superior de Engenharia do Instituto Politécnico do Porto e Professor Auxiliar da Universidade de Trás-os-Montes e Alto Douro

Co-Orientadora: Doutora Zita Maria Almeida do Vale, Professora Coordenadora Principal do Instituto Superior de Engenharia do Instituto Politécnico do Porto

Supervisor: Doutor Luis Filipe Oliveira Gomes, Investigador do Instituto Superior de Engenharia do Instituto Politécnico do Porto

Júri

Presidente**:**

Doutora Maria Goreti Carvalho Marreiros, Professora Coordenadora com Agregação do Instituto Superior de Engenharia do Instituto Politécnico do Porto

Vogais**:**

Doutor Paulo Jorge Pinto Leitão, Professor Coordenador Principal do Instituto Politécnico de Bragança

Porto, 28 de junho de 2022

Resumo

Devido à pandemia mundial que se iniciou no primeiro trimestre do ano de 2020 e que persiste até aos dias de hoje, inúmeros confinamentos populacionais foram ocorrendo, verificando-se com isso uma redução da procura de energia elétrica a nível mundial. No entanto, com o gradual regresso à normalidade, assiste-se ao aumento dessa procura de energia elétrica. Devido a este aumento atípico da procura, saber como gerir a energia elétrica que é gasta, é essencial. Neste trabalho é proposto um sistema inteligente de gestão de energia elétrica em edifícios, recorrendo para isso a dispositivos de Internet das Coisas, tais como sensores de temperatura, sensores de presença e sensores de luminosidade. A gestão da interação entre todos esses dispositivos foi realizada através de um sistema multiagente, onde cada agente do sistema é responsável por um dispositivo IoT. Esses dispositivos possuem capacidade de aprendizagem própria, conseguido através da implementação de redes neuronais artificiais nos microprocessadores dos dispositivos IoT. Desta forma, gera-se a capacidade do sistema em agir e reagir em tempo real e em função dos dados que são existentes e recolhidos. Este é um dos principais pontos do sistema, considerando que implementar redes neuronais artificiais em microprocessadores ainda é um desafio considerável, e ao mesmo tempo é uma das grandes vantagens do mesmo. Foram atingidos resultados muito satisfatórios no que respeita à experimentação realizada, dando especial ênfase ao caso de estudo em que foi utilizada a temperatura e um sistema HVAC, onde o dispositivo IoT responsável por esse parâmetro, efetua a recolha da temperatura atual sentida, para introduzir depois esse dado na rede neuronal artificial que o próprio possui com a finalidade de obter o resultado em previsão, ou seja, o resultado do que seria a ação que o HVAC deveria fazer em função dos valores previstos. Ter este tipo de inteligência num sistema como o desenvolvido é importante e relevante porque possibilita a ação correta do mesmo nos mais diversos casos de ação, uma vez que dados muito díspares não induzem o sistema em erro, porque os resultados previstos são obtidos através das redes neuronais artificiais aplicadas e utilizadas. Trata-se assim de um sistema mais preciso e mais assertivo, algo importante quando se fala em gestão energética inteligente.

Palavras-chave: Aprendizagem automática, gestão energética de edifícios, internet das coisas, redes neuronais artificiais, sistema multiagente

Abstract

Due to the global pandemic that began in the first quarter of 2020 and persists to this day, numerous population confinements have taken place, resulting in a reduction in demand for electricity worldwide. However, with the gradual return to normality, there is an increase in this demand for electricity. Due to this unusual increase in demand, knowing how to manage the electrical energy that is spent is essential. In this work, an intelligent system for managing electrical energy in buildings is proposed, using Internet of Things devices, such as temperature sensors, presence sensors and light sensors. The management of the interaction between all these devices was carried out through a multi-agent system, where each agent in the system is responsible for an IoT device. These devices have their own learning capacity, achieved through the implementation of artificial neural networks in the microprocessors of IoT devices. In this way, the system's ability to act and react in real time is generated and based on the data that is existing and collected. This is one of the main points of the system, considering that implementing artificial neural networks in microprocessors is still a considerable challenge, and at the same time it is one of its great advantages. Very satisfactory results were achieved with regard to the experimentation carried out, giving special emphasis to the case study in which the temperature and an HVAC system were used, where the IoT device responsible for this parameter, collects the current temperature felt, to then introduce this data in the artificial neural network that it has in order to obtain the result in prediction, that is, the result of what would be the action that the HVAC should do in terms of the predicted values. Having this type of intelligence in a system like the one developed is important and relevant because it allows the correct action of the same in the most diverse cases of action, since very disparate data do not mislead the system, because the predicted results are obtained through the networks. artificial neurons applied and used. It is therefore a more precise and assertive system, something important when it comes to intelligent energy management.

Keywords: Artificial neural network, building energy management, internet of things, machine learning, multiagent system

Agradecimentos

Em primeiro lugar, quero agradecer aos meus orientadores Doutor Tiago Pinto, Doutora Zita Vale e Doutor Luís Gomes por todo o apoio, suporte, dedicação e incentivo que me deram ao longo de todo o percurso. São profissionais de excelência e excelentes pessoas, que estiveram disponíveis do primeiro ao último minuto, e que me aceitaram como seu orientando da forma mais positiva possível, algo que irei sempre valorizar.

Deixo uma palavra de agradecimento a todos os meus colegas de mestrado, pelas conversas, pela troca de ideias, pelos momentos de entreajuda e por terem feito parte deste meu percurso, foi sem dúvida um gosto tê-los comigo nesta jornada.

Aos meus amigos mais próximos, quero agradecer por direta ou indiretamente me terem acompanhado neste percurso, mostrando sempre a sua confiança no meu trabalho, enviando mensagens de apoio e de incentivo, e sinais de verdadeira amizade.

Quero agradecer à minha namorada, Raquel, por ter acompanhado o meu percurso sempre de forma minuciosa, com paciência, sempre com palavras de força, de motivação, de afeto. Uma ajuda valiosa para superar os momentos mais adversos deste percurso, sempre lá para mim, mesmo quando éramos só nós e um gigantesco silêncio. Sem dúvida uma pessoa que me ajuda a ser melhor a cada dia.

Por último, mas não menos importante, tenho que agradecer aos meus pais e irmão, porque desde sempre me ensinaram que somos capazes de tudo, desde que lutemos pelos objetivos a que nos propomos. São um apoio constante em qualquer decisão que eu tome para a minha vida, uma companhia para qualquer caminho que eu percorra e figuras de extrema importância para mim, a quem eu dedico esta dissertação de mestrado, algo extremamente valioso e marcante para mim, como sinal do meu agradecimento para com eles, agradecimento esse que perdurará no tempo.

Fernando Pessoa disse: “Não é o trabalho, mas o saber trabalhar que é o segredo do êxito no trabalho. Saber trabalhar quer dizer: não fazer um esforço inútil, persistir no esforço até ao fim e saber reconstruir uma orientação quando se verificou que ela era, ou se tornou, errada”.

A todos os mencionados e todos que participaram neste meu percurso, direta ou indiretamente, o meu sincero obrigado.

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

ANN	*Artificial neural network*
BPTT	*Backpropagation through time*
CA	*Communication act*
CiTIUS	*Centro Singular de Investigación en Tecnoloxías Intelixentes*
DHT22	*Digital temperature and humidity sensor*
DSM	*Data server manager*
D2D	*Device to device*
D2S	*Device to server*
FIPA	*Foundation for intelligent physical agents*
GECAD	*Research Group on Intelligent Engineering and Computing for Advanced Innovation Development*
HAN	*Home area network*
HVAC	*Heating, Ventilating and Air Conditioning*
HyFIS	*Hybrid Neural Fuzzy Inference System*
IA	Inteligência Artificial
IaaS	*Infraestruture as a service*
IoT	*Internet of Things*
JADE	*Java Agent Development Framework*
JID	*Jabber identifier*
JSON	*JavaScript object notation*
LDR	*Light dependent resistors*
LED	*Light-emitting diode*
LSTM	*Long-short term memory*

MAS *Multiagent system*

MASCEM *Multi-Agent Simulator for Competitive Electricity Markets*

MBaas *Back end as a service*

MQTT *Message Queuing Telemetry Transport*

OGI *Oil and gas industry*

OMEMO *OMEMO Multi-End Message and Object Encryption*

PaaS *Platform as a service*

PADE *Python Agent Development framework*

PEAK *Python-based Ecosystem for Agent Communities*

PGP *Pretty good privacy*

PIR *Passive infrared sensor*

PLC *Programmable Logic Controllers*

SaaS *Software as a service*

SCADA Sistemas de supervisão e aquisição de dados

SMA Sistemas Multiagente

SVR *Support vector regression*

S2S *Server to server*

ReLU *Rectified linear unit*

RF *Random forest*

RNA Redes neuronais artificiais

RNC Redes neuronais convolucionais

RNR Redes neuronais recorrentes

RMSE *Root Mean Square Error*

WLAN *Wireless local area network*

WM *Wang and Mendel*

XML *Extensible mark-up language*

XMPP *Extensible Messaging and Presence Protocol*

Lista de Símbolos

$\boldsymbol{CO_2}$ Dióxido de carbono

1 Introdução

1.1 Definição do problema

A gestão energética inteligente de um edifício pode assumir um papel de grande relevância, numa fase onde o consumo energético é abordado de forma tão intensa devido aos impactos que esses consumos têm no meio ambiente, e onde a produção de energia é um dos debates mais atuais. Com o surgimento da pandemia mundial provocada pelo vírus SARS-COV2, no primeiro trimestre de 2020, assistiu-se a uma diminuição drástica do consumo de energia elétrica mundial (Portuguesa, 2021). É estimado que tenha ocorrido uma queda de cerca de 4,5% do consumo de energia elétrica, desde o início da pandemia, até final do último trimestre do ano de 2020. A registar, é a maior queda em 76 anos, desde o final da Segunda Guerra Mundial.

Com os diversos confinamentos decretados e o abrandamento da economia, local e mundial, assistiu-se a uma diminuição drástica do consumo de energia elétrica. Esse valor voltou a subir com o regresso gradual à normalidade, tem vindo a surgir um aumento da procura por fontes energéticas. Esse aumento de procura faz com que o mercado tenha menos recursos para oferecer, o que, consequentemente, encaminha a um aumento do preço dessa oferta. Isto e as alterações de paradigma energético contribuíram ainda mais para que o preço atinja valores anormais em relação ao que é a média de mercado (Neutel & Petiz, 2021). Percebe-se que uma gestão cuidada da energia elétrica é cada vez mais uma necessidade real.

Para além destes fatores, também a mudança de paradigma energético, originado pelas acentuados e destrutivas mudanças climáticas, contribui positivamente para o aumento do preço da energia (Browning & Sharafedin, 2021). Várias foram as centrais de produção de combustíveis fósseis que fecharam, chegando ao fim do seu ciclo de produção, aumentando a dependência por fontes de energias renováveis, que está dependente de fatores voláteis como o clima, gera incerteza nos mercados e o consequente aumento do preço nos mesmos.

Desta forma, a gestão cuidada da energia elétrica que é gasta torna-se um bem de primeira necessidade, para evitar que ela se esgote e, consequentemente, para evitar um aumento descontrolado do valor que por ela é cobrado. Neste sentido, neste trabalho é proposto e desenvolvido um sistema capaz de realizar essa gestão energética inteligente, da forma mais adequada possível, utilizando para isso dispositivos de IoT - Internet of Things (Oracle, 2021), algoritmos de aprendizagem automática (Microsoft, 2021) e interação entre agentes. Os elementos de IoT são os responsáveis pela parte de obtenção dos dados. Os agentes localizados em cada um dos dispositivos IoT existentes, encarregam-se de representar na rede o dispositivo no qual está inserido, criando assim um sistema multiagente e ligando todos os dispositivos à rede. A aprendizagem automática está presente pela necessidade de criar agentes inteligentes, com capacidade de ação e reação individual e personalizada, em função das circunstâncias. Por

isto, estão aprimorados de inteligência artificial através de algoritmos dessa categoria, nomeadamente redes neuronais artificiais.

1.2 Contextualização do Problema

Com o objetivo de controlar a gestão energética de um edifício, foi realizada a conceção, desenvolvimento e avaliação de um sistema inteligente com o objetivo de representar e gerir a comunicação entre dispositivos IoT, como sensores ou atuadores.

Os dispositivos IoT são responsáveis pela recolha de dados do ambiente. Por cada dispositivo IoT é existir um agente responsável. Os agentes poderão ter diversas tarefas. Podem reagir aos dados lidos pelos sensores e atuar em conformidade. Essa atuação pode ser sobre si mesmo ou sobre outros, existindo comunicação entre os agentes do sistema multiagente. Possuem uma rede neuronal artificial a si associada, que utilizam para realizar interpretações em relação à informação que recebem e usam essa rede neuronal para classificar a informação, como o tipo de consumo num dado período temporal, a necessidade de aumentar o grau da luminosidade sentido numa divisão, ou gerir a temperatura através de sistemas HVAC. A abordagem da aplicação é realizada de forma distribuída, pois uma abordagem centralizada não é completamente possível devido às características dos dispositivos IoT que são usados para a recolha dos dados e devido à necessidade de respostas rápidas, distribuídas.

Do ponto de vista teórico, realizar a recolha dos dados em todo o edifício, agrupar os mesmos e trabalhar sobre eles seria uma solução ótima. Do ponto de vista prático, trabalhar em áreas delimitadas leva a uma melhor solução final. Os dados de determinada área são recolhidos e agrupados e é com base nessa área que são tomadas decisões sobre como o controlo dos diversos dispositivos deve acontecer. Esta abordagem acaba por ser mais simplista, no entanto, mais fiável.

Para ligar todas essas áreas, o sistema multiagente assume uma relevância positiva e forte, uma vez que agentes de diferentes áreas podem comunicar entre si. Desta forma pretende-se que todo o sistema tenha conhecimento do que acontece nas áreas mais parciais do todo. No entanto, a questão da comunicação entre agentes assume também uma questão muito pertinente e relevante. Não é possível assumir que todos os edifícios possuam dispositivos com essa capacidade. Mesmo que existam edifícios com essas propriedades, não é possível garantir que o mesmo aconteça em todas as áreas do edifício, que os dispositivos utilizem os mesmos protocolos de comunicação ou que todos eles sejam baseados no mesmo funcionamento.

O desejável para ultrapassar a questão do *retrofitting* seriam as potenciais modernizações do edifício, com a possível instalação de dispositivos, tais como microprocessadores. Pretende-se assim que seja possível lidar com todas as questões de comunicação, monitorização e controlo de cada dispositivo, independentemente das características únicas e individuais de cada um. Um outro fator muito relevante e importante é a necessidade de participação ativa dos consumidores. Quanto mais ativa esta participação, mais dados recolhidos são recolhidos, o que permite uma maior noção da variação global do consumo, ou seja, de que forma o consumo

do edifício varia. Com essa informação disponível é possível entender que variabilidade de fontes de energia renováveis podem ser usadas no sistema, em função do potencial de flexibilidade do consumo existente, do tipo de fontes de energia que já são utilizadas no local.

1.3 Objetivos

O principal objetivo do trabalho desenvolvido é a gestão energética inteligente de um edifício. Para que essa gestão fosse, efetivamente, inteligente, exigia-se a utilização de redes neuronais artificias, responsáveis por munir os dispositivos IoT da chamada inteligência artificial, que no fundo é a capacidade de interpretar os dados que os próprios obtêm e agir em função de todo o conhecimento prévio que possuem. Essa utilização de redes neuronais artificias nos microprocessadores é apresentada na secção 3.6 (Rede neuronal artificial) onde é descrita a rede neuronal artificial que foi aplicada aos microprocessadores para colmatar esta necessidade.

Sabendo que um único dispositivo IoT (como um sensor que lê a temperatura existente) não é suficiente para tornar um edifício inteligente, existiu a necessidade de criar uma ligação entre os vários dispositivos do sistema, para que, por exemplo, sempre que o sensor de temperatura obtivesse uma previsão sobre como deve o dispositivo HVAC agir em função da temperatura que foi lida, essa informação chegasse ao HVAC. Para superar esta necessidade, foi criado um sistema multiagente onde cada agente desse sistema é responsável por um único dispositivo IoT, o que permite garantir uma comunicação global de todo o sistema, garantindo que todos os dados e conhecimentos unitários podem chegar a qualquer lugar do sistema, caso exista essa necessidade de o fazer. Toda a informação relativa ao sistema multiagente pode ser encontrada na secção 3.3 (Sistema multiagente) e exemplos da comunicação entre todo o sistema podem também ser vistos na secção 4 (Casos de estudo) onde são apresentados exemplos concretos de como a comunicação entre elementos acontece, e como a informação é transmitida por todo o sistema desenvolvido.

1.4 Organização do documento

O documento está organizado em cinco capítulos principais: o primeiro capítulo é introdutório do que é descrito no documento, onde é possível entender o problema que o sistema visa resolver, a contextualização no qual o sistema foi desenvolvido, bem como os objetivos traçados.

No capítulo dois é apresentado o estado da arte, onde são apresentados de forma individual cada um dos elementos que fazem parte do sistema desenvolvido, nomeadamente os componentes de *Internet of Things*, de sistema multiagente, de redes neuronais artificias e de aprendizagem automática. Para cada um dos elementos, são apresentadas as suas definições, a sua constituição de uma forma geral, mas também de uma forma pormenorizada. Ainda neste capítulo são revistos os trabalhos relacionados de maior impacto, isto é, trabalhos desenvolvidos no mesmo âmbito e que se uma forma ou de outra contribuíram para o

desenvolvimento do sistema, terminando com uma secção onde são apresentados os benefícios das tecnologias, o que possibilita a compreensão para a escolha das mesmas e a sua utilização.

No terceiro capítulo são apresentados os desenvolvimentos e experimentações que foram desenvolvidos e aplicados ao sistema, e que levaram a que o mesmo fosse moldado na sua forma final. São apresentadas as soluções escolhidas para suprir cada necessidade encontrada, tanto ao nível do sistema multiagente, passando pelas redes neuronais artificias, até à comunicação geral de todos os elementos que constituem o sistema. Trata-se de um capítulo realmente determinante para ser possível compreender cada elemento de forma unitária, e concluir o sistema como um todo.

No quarto capítulo do documento são mencionados todos os casos de estudo que foram selecionados e desenvolvidos para comprovar as funcionalidades que o sistema possui. São constituídos por provas de funcionamento, exemplificações e demonstrações.

No quinto e último capítulo, mas não menos importante, é apresentada a conclusão do documento, onde é feito um breve resumo do mesmo e do sistema desenvolvido, os contributos do trabalho, importante para perceber o possível impacto que o sistema desenvolvido tem enquanto solução para o problema descrito. São também apresentadas as questões éticas e de proteção de dados, importante do ponto de vista de segurança do sistema e de assegurar privacidade e proteção para os utilizadores do mesmo, terminando depois a secção com as limitações e trabalho futuro, onde são apresentadas de forma breve e sucinta as limitações que foram sendo sentidas com o decorrer do trabalho, de que forma as mesmas foram ultrapassadas e que possível impacto tiveram no produto final. São também apresentadas projeções futuras, isto é, possíveis ideias e novas funcionalidades que podem ser implementadas no futuro e que, certamente, podem contribuir para melhorar ainda mais o sistema.

2 Estado da Arte

2.1 O sistema desenvolvido

O sistema desenvolvido permite a gestão energética inteligente de um edifício baseado em dispositivos IoT distribuídos. Tratando-se de um sistema que possui dispositivos IoT diversos, como sensores de luminosidade, presença, temperatura, entre outros, existe a necessidade de perceber como é possível controlar cada um desses dispositivos e fazer com que, em conjunto, existam e funcionem para o propósito do sistema. É com esse intuito de integração e de controlo dos vários dispositivos IoT que foi criado um sistema multiagente. Nesse sistema, cada agente está encarregue por um determinado dispositivo IoT. Em conjunto, todos os agentes envolvidos no processo integram o sistema multiagente do sistema e permitem assim que todos os dispositivos IoT tenham uma interação mútua. Para tornar esse sistema um sistema inteligente, foram utilizados algoritmos e métodos de aprendizagem automática. Pretende-se assim que os dispositivos IoT tenham capacidade avaliativa e saibam agir e reagir de acordo com a informação que captam do meio onde se encontram, mas também com experiências que já tenham realizado anteriormente e com todo o conhecimento que determinada informação posterior lhes forneceu. Assim, o sistema possui três componentes principais: dispositivos IoT, um sistema multiagente e aprendizagem automática. O estado da arte pretende dar a conhecer um pouco de cada um destes elementos na sua forma singular, de que forma trabalham, e as suas características. Para além disso, de que forma podem estes elementos ser integrados para um funcionamento em grupo, ou seja, de que forma é possível existirem dispositivos IoT que sejam providos de inteligência artificial através da aprendizagem automática que realizam, e de que forma cada um desses dispositivos existe no sistema, representados por um agente virtual, e de que forma um sistema multiagente pode ser um fator benéfico e importante para um sistema deste tipo.

2.2 Internet of Things

2.2.1 Definição

A definição de *Internet of Things* – também designada de IoT – é ainda hoje uma das perguntas mais realizadas e à qual mais se tenta responder (Berte, 2018). Na verdade, não existe uma definição única de IoT, mas sim várias hipóteses que foram sendo criadas ao longo do tempo.

O conceito de *Internet of Things* surgiu em 1999 por Kevin Ashton, que mencionou IoT como sendo um ou mais objetos conectados, de identificação única, com tecnologia de identificação por radiofrequência.

O *Oxford Dictionary* define *Internet of Things* como um desenvolvimento da Internet onde objetos do cotidiano têm conectividade de rede, o que lhes permite enviar e receber dados (Anon., 2021). A revista *Forbes* define *Internet of Things* como a capacidade de conectar um dispositivo à Internet ou a outros dispositivos existentes na rede.

A *Internet of Things* é vista como a próxima geração da Internet, onde existirá a inclusão de elementos físicos em ambiente virtual, com a possibilidade de lhes aceder e de os identificar na rede. Isto implica que os elementos sejam capazes da troca de dados entre si.

2.2.2 Arquiteturas

Não existe um único modelo de arquitetura IoT, os modelos existentes variam conforme as necessidades de aplicação e a finalidade com que os dispositivos serão utilizados.

2.2.2.1 Arquitetura de três camadas

Uma das arquiteturas mais utilizadas e simples de implementar é a arquitetura de três camadas, que possui três camadas distintas (Al-Qaseemi, et al., 2017). A primeira das três camadas desta arquitetura é a camada de *perceptron* que é responsável pela recolha de todos os dados do meio. De seguida, surge a camada de rede, responsável pela transmissão e processamento dos dados que foram recolhidos na camada anterior. Por fim, a última camada é a denominada de camada de aplicação e é responsável por transmitir a informação ao utilizador, em função do tipo de serviço para o qual a aplicação se designa.

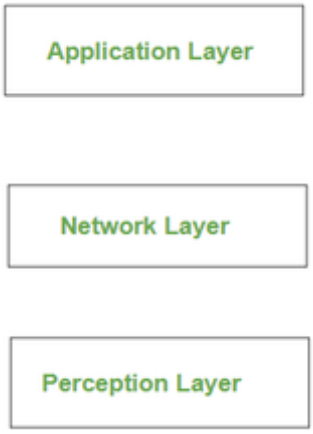

Figura 1 – Arquitetura de três camadas

2.2.2.2 Arquitetura de cinco camadas

Uma arquitetura que se inspira na arquitetura apresentada anteriormente, é a arquitetura de cinco camadas, que surgiu para situações com maior complexidade, onde as aplicações possuem um maior número de parâmetros (Kumar & Mallick, 2018). Possuem mais duas camadas do que a arquitetura anteriormente apresentada, no entanto, não necessitam de conter as três camadas que existem na arquitetura de três camadas. A escolha das camadas existentes é aquela que mais favorecer a situação para o qual a mesma será aplicada.

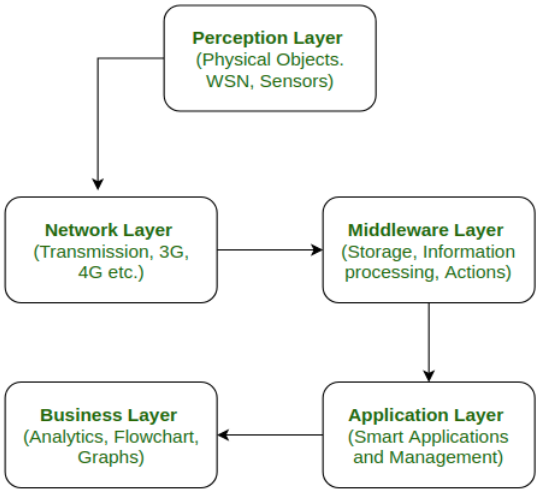

Figura 2 – Arquitetura de cinco camadas

2.2.2.3 Arquitetura orientada a serviços

Um outro modelo de arquitetura possível de encontrar em IoT é a arquitetura orientada a serviços (Gokhale, et al., 2007). Apresenta-se sobre a forma de quatro camadas, distintas entre si. A primeira camada é a sensitiva, onde estão presentes os dispositivos da rede, e onde é realizada a captura de dados do ambiente por parte de cada um deles. De seguida surge a camada de rede, onde os vários dispositivos são conectados entre si, sendo os dados de cada um são partilhados com os restantes elementos da rede e todos os dispositivos tomam conhecimentos dos outros dispositivos constituintes da rede. De seguida surge a camada de serviço, assim denominada pois é onde são realizadas as operações da rede. Essas operações podem incluir a procura de dados na rede, a procura de objetos específicos para executar uma determinada ação ou realizar a ligação à interface onde os dados serão apresentados ao utilizador. Por último, a camada de interface, a última camada desta arquitetura e onde são geridos os possíveis problemas de compatibilidade que possam existir, devido à presença de dispositivos com características distintas, ou devido a outros motivos. É nesta camada que problemas desse tipo são corrigidos para que sejam evitadas incompatibilidades que possam

prejudicar a troca de informações e a execução dos serviços existentes na aplicação (Mishra & Sarkar, 2021).

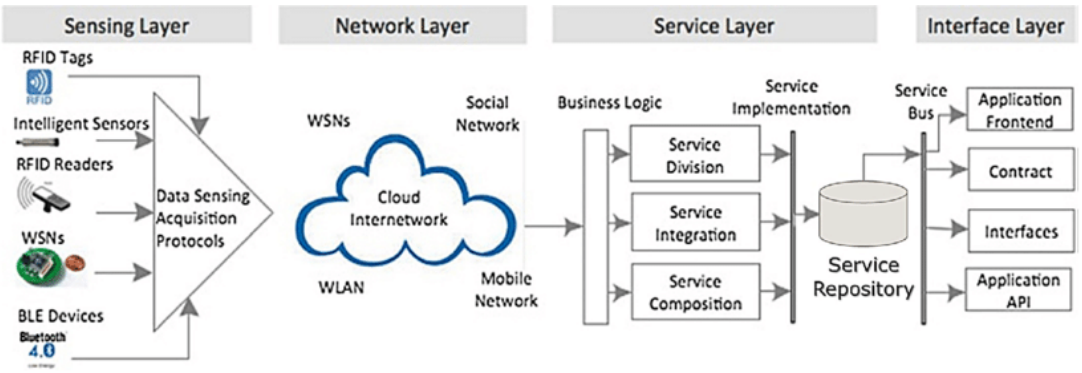

Figura 3 – Arquitetura orientada a serviços

2.2.3 Protocolos de Comunicação

A divisão protocolar de comunicação em IoT pode ser dividida em três componentes principais. Os principais intervenientes são apenas dois: dispositivos e servidor. O que diferencia cada tipo de protocolo de comunicação é a variação entre quais elementos a informação é transmitida.

O protocolo D2D é um tipo de protocolo de comunicação onde a mesma é realizada entre dispositivos que se encontram fisicamente próximos uns dos outros (Mahdavinejad, et al., 2018). Ainda sobre protocolos dos quais fazem parte dispositivos como um dos elementos integrantes, existe o protocolo D2S que é um tipo de protocolo de comunicação onde todos os dados que são recolhidos pelos diversos dispositivos são, posteriormente, enviados para o servidor (Mahdavinejad, et al., 2018). Em relação a protocolos onde a comunicação apenas é realizada com intervenção de servidores, deve-se mencionar o protocolo S2S, um tipo de protocolo de comunicação onde todos os dados são transmitidos entre servidores (Al Sibahee, et al., 2020).

2.2.4 Processamento de dados

O processamento de dados recolhidos por dispositivos IoT pode variar em função do tipo de computação. A principal diferença entre as computações e a forma de processar os dados está no local onde esse processamento é realizado. Enquanto algumas *frameworks* processam os dados assim que eles são recolhidos pelos dispositivos, outras não o fazem.

2.2.4.1 Computação fog

A computação fog é uma computação de processamento de dados limitada, tanto em relação à sua capacidade de rede, como em relação à sua capacidade de armazenamento (Shi, et al., 2015). A alocação do poder de processamento é mais perto do limite da rede, do que comparativamente com outras computações existentes. A análise que é realizada aos dados é a curto prazo, isto é, os dados são analisados ainda dentro do dispositivo que realizou a sua recolha. Neste tipo de computação, é fornecido aos dispositivos uma inteligência lógica, que tem como missão a filtragem de dados para o servidor. Essa ação é realizada com a preocupação de garantir uma baixa latência.

2.2.4.2 Computação em nuvem

Quando na na presença de uma computação em nuvem, os dispositivos que recolhem os dados, processam o seu envio para servidores, onde os mesmos serão analisados e só depois se tornam dados possíveis de serem acedidos (Mahdavinejad, et al., 2018). Este tipo de computação pode ser dividido em diferentes tipos de acordo com a aplicação em questão. A computação em nuvem IaaS - infraestrutura como um serviço – ocorre quando a empresa responsável pela computação possui todo o tipo de equipamento do sistema, como o *hardware*, caso dos dispositivos IoT, os servidores onde os dados são guardados e também as redes que são utilizadas para proceder ao envio de dados (Akkaya, et al., 2018). A computação em nuvem PaaS – a plataforma como um serviço – ocorre quando o equipamento utilizado para suportar o sistema não é da empresa que possui o sistema, ou seja, tanto os dispositivos, como os servidores ou a rede utilizada, são fornecidos por empresas provedoras. Na computação em nuvem Saas – software como um serviço – verifica-se a existência de um modelo de software distribuído, que é da responsabilidade do provedor de serviços. É o provedor de serviços que aloja esse software e o disponibiliza para uso por parte dos utilizadores da aplicação (Lovas, et al., 2018). Por fim, contamos ainda com a computação em nuvem MBaaS – o *backend* como um serviço – onde um caminho é desenvolvido para levar o utilizador à conexão com aos dados que estão na nuvem, já recolhidos e armazenados (Izzuddin, et al., 2018)

2.2.4.3 Computação distribuída

Contrariamente àquilo que se verifica na computação fog, que possui uma limitada capacidade de processamento de dados, a computação distribuída foi projetada com o intuito de ser capaz de processar um volume de dados superior (Di Pascale, et al., 2018). Neste tipo de computação todos os dados existentes são divididos por diferentes máquinas de processamento, o que contribui para o seu nome, uma vez que esse processamento não é realizado todo no mesmo local, ou na mesma máquina, mas sim de forma distribuída (Mahdavinejad, et al., 2018).

2.3 Sistemas Multiagente

Como referido na introdução do artigo, cada dispositivo IoT tem a si associado um agente. Agente esse que tem a tarefa de comunicar com outros agentes, responsáveis por outros dispositivos IoT e interligar toda a rede de dispositivos. Dada a existência de múltiplos agentes, estamos na presença de um sistema multiagente.

2.3.1 Definição

Um sistema multiagente é um grupo de entidades unitárias e autónomas que têm como propósito a resolução de tarefas de forma colaborativa (Ye, et al., 2017). Para essa colaboração ser eficiente e real os diversos agentes interagem com outros agentes e com o ambiente, com a finalidade de aprender algo através dessa interação e realizar novas ações. É através do seu conhecimento que decidem e realizam ações.

Apesar da descrição e definição de agente e das suas potenciais capacidades, uma definição única não é encontrada de forma consensual. Alguns autores definem o SMA – sigla para sistema multiagente – como um tipo de sistema distribuído onde os componentes desse sistema, no caso os agentes, são independentes e apenas se interessam pelo seu próprio benefício, procurando de forma exclusiva a obtenção dos seus objetivos (Dorri, et al., 2018).

Outros autores definem os agentes como uma entidade que está presente num determinado ambiente e que tem como principal função a deteção de informação, tendo por base certos parâmetros que são utilizados para que o agente seja capaz de decidir com a finalidade de atingir o seu objetivo (Dorri, et al., 2018).

2.3.2 Comunicação entre agentes

Em 1995 foi estabelecida a FIPA – Fundação para Agentes Físicos Inteligentes – uma organização sem fins lucrativos para produzir especificações para interfaces abertas sobre a área dos agentes (O'brien & Nicol, 1998). São eles os criadores da linguagem que tem a capacidade de realizar a comunicação entre os diversos agentes, bem como transmitir os seus desejos e intenções.

A linguagem criada por eles é denominada de FIPA ACL e individualiza a comunicação entre agentes através de uma semântica formal que foi desenvolvida para esse efeito e que possui cinco níveis distintos (O'brien & Nicol, 1998). O primeiro nível é o protocolo, onde são definidas as regras sociais pelas quais serão pautadas as comunicações entre agentes. De seguida, no segundo nível, temos o CA – ato da comunicação – que define o tipo de comunicação que está a ser realizada. Pode ser um pedido por parte de um agente, por exemplo. No terceiro nível, surge a mensagem propriamente dita, que é definida como meta informação. Inclui informação que permite a identificação do agente que a enviou, o agente que se espera receber a mensagem, o contexto e o conteúdo. No quarto nível existe a linguagem de conteúdo, que se

trata da definição de ambas gramática e semântica com as quais a mensagem será expressa. Por fim, no último nível, a ontologia, que nos apresenta a definição do vocabulário, o significado dos termos que foram utilizados na mensagem e os conceitos inseridos na expressão do conteúdo.

2.3.3 Framework

As *frameworks* aplicadas aos agentes possuem propósitos como ajudar a adicionar técnicas de inteligência artificial ou desenvolver protocolos de comunicação. Nesse sentido, a utilização de *frameworks* em aplicações onde os agentes se encontram presentes é algo lógico e natural.

Uma das mais conhecidas e utilizadas *frameworks* é JADE, um software implementado em linguagem Java, que ajuda a implementar um sistema multiagente através do seu middleware que considera as especificações FIPA dos agentes, adicionando ainda um conjunto de ferramentas gráficas que auxiliam no desenvolvimento das aplicações (Lez-Briones, et al., 2018). Um fator positivo é a sua capacidade de sistema distribuído que permite que os vários agentes se multipliquem por várias máquinas.

Uma *framework* desenvolvida utilizando a linguagem *Python* é o SPADE (Gregori, et al., 2006). Tal como a *framework* JADE, também utiliza as especificações FIPA nos seus agentes. Foi desenvolvida utilizando uma *framework* de comunicação denominada *Jabber*, que possibilita a notificação de presenças dos agentes existentes. Para além disso, um detalhe interessante da sua criação foi a inclusão do MUC – *Multi-user conference* – que permitia a criação de fóruns, permitindo aos agentes a sua ligação a eles quando estivessem a desempenhar atividades diversas.

Existem ainda outras *frameworks* como o PADE que é também uma *framework* que permite o desenvolvimento de um sistema de forma distribuída. Uma diferença relevante em relação à *framework* anterior é a possibilidade de criar os agentes utilizam a linguagem Python. Para além disso, utiliza todos os padrões FIPA, como a linguagem FIPA-ACL e os protocolos de comunicação FIPA (Lez-Briones, et al., 2018).

Uma nova solução, e a escolhida para a criação do sistema multiagente do sistema desenvolvido foi a *framework* PEAK (Ribeiro, et al., 2021). Trata-se de uma *framework* que se encontra ainda em desenvolvimento no GECAD e que é uma nova abordagem a este tipo de conceito, tendo como fator diferenciador o facto de permitir a existência de vários ecossistemas MAS, no entanto, existindo a garantia que nenhum deles tem uma intervenção de qualquer tipo nos restantes, fundamental para que se possa garantir que um deles não interfere de forma negativa num outro grupo de agentes. Tem por base o SPADE, já apresentado anteriormente nesta secção, o que indica que é também ele baseado no protocolo de comunicação XMPP, no entanto, difere do mesmo porque permite execução concorrente de agentes, algo que não se verifica no SPADE.

2.4 Aprendizagem automática

Os agentes do sistema multiagente têm a capacidade de aprender de forma autónoma e com experiências anteriores. Essa aprendizagem é realizada através da utilização de algoritmos de aprendizagem de inteligência artificial que permite aos agentes ter essa capacidade.

2.4.1 Definição

Aprendizagem automática é um tipo de inteligência artificial que possibilita a capacidade de uma máquina aprender sem necessitar de um qualquer tipo de programação adicional. Utiliza dados para realizar o ensino junto das máquinas que a possuem. Os algoritmos de machine learning obtêm a sua categoria em função do tipo de dados que utilizam.

2.4.2 Categorias da aprendizagem

As categorias de aprendizagem dependem do tipo de dados que são utilizados para treino. Existem três categorias que são as principais relativamente aos algoritmos deste tipo.

Quando existem dados com rótulo, estamos na presença de algoritmos de aprendizagem supervisionada, que são capazes de classificar os dados e obter resultados precisos (Jiang, et al., 2020). Neste tipo de aprendizagem, o dataset de treino já possui tanto os dados que serão utilizados como input, como os resultados – output – que são desejados.

Quando os dados existentes não são rotulados, estamos na presença de uma aprendizagem não supervisionada (Usama, et al., 2019). O conjunto de dados de treino apenas contém os dados de entrada, ao contrário da aprendizagem supervisionada, que contém também o resultado que é desejado de ser obtido (Jiang, et al., 2020). Também neste tipo de aprendizagem, são criadas estruturas nos dados, como *clusters* que são pequenos agrupamentos de pontos. Dentro desses pequenos grupos, procuram-se pontos em comum.

O terceiro e último tipo de aprendizagem é a aprendizagem por reforço. Neste tipo de aprendizagem o principal objetivo é a maximização daquilo que é chamado de sinal numérico de recompensa (Henderson, et al., 2018). Não é dito a quem aprende aquilo que deve fazer, mas sim entender de qual das ações realizadas é aquela em que é possível obter uma maior maximização desse sinal numérico de recompensa.

2.5 Redes Neuronais Artificiais

As redes neuronais artificiais – também conhecidas como RNA – são um dos tipos de machine learning existente. São um modelo artificial do cérebro humano, o que significa que o seu comportamento replica, de certa forma, o comportamento do cérebro humano (Villarrubia, et al., 2018). O paralelo com o cérebro humano é ainda mais profundo, dado que as camadas de uma rede artificial tendem a replicar, ainda que de forma artificial, o comportamento biológico do cérebro humano.

As RNA possuem três camadas características, sendo a primeira denominada de camada de entrada (Zhang, et al., 2019). Trata-se de uma representação artificial dos dendritos do neurónio biológico. A segunda camada é onde são realizadas as funções matemáticas com os dados de entrada, da primeira camada. Por fim, a terceira camada da RNA representa o sistema axónio e as sinapses do neurónio biológico.

2.5.1 Redes neuronais convolucionais

As redes neuronais convolucionais – também conhecidas como RNC – são um tipo de rede neuronal artificial cuja utilização é recorrente em sistemas de processamento e de reconhecimento de imagem.

As RNC possuem determinados elementos básicos tais como as camadas convolucionais, onde são realizadas as operações de convolução. Possuem também camadas de *pooling*, onde são realizadas as operações de *pooling*, que no fundo é a redução da dimensão da matriz imagem. Um outro elemento básico de uma RNC é a função de ativação, que por norma é do tipo ReLU - *rectified linear unit* (Wang, et al., 2019).

Para além dos elementos básicos apresentados, possíveis de encontrar em qualquer rede deste tipo, as RNC possuem a particularidade de ter todas as camadas completamente conectadas, isto é, todos os nós de uma camada estão conectados a todos os nós da camada seguinte, bem como a todos os nós da camada que os antecede (Gu, et al., 2018).

Uma das vantagens das redes neuronais convolucionais é a não necessidade de aprendizagem da totalidade do conjunto de dados que é utilizado para treino, sempre que o mesmo sofre uma alteração (Zhang, et al., 2019). Para além disto, são redes com uma boa capacidade de aprendizagem e que obtêm resultados robustos.

Relativamente às desvantagens apresentadas, a principal, e que deve ser mencionada, é a ineficiência temporal naquilo que é o processo de treino, ou seja, são redes que demoram bastante tempo até terem o seu treino concluído (Mijwel, 2018). Para além disso são redes neuronais que precisam de uma grande quantidade de dados para o seu correto funcionamento.

2.5.2 Redes neuronais recorrentes

As redes neuronais recorrentes – também conhecidas como RNR – são um tipo de rede neuronal artificial que possuem a capacidade de memorizar aquilo que já aconteceu (Wang, et al., 2019). Isto é particularmente útil quando se lida com sequências de dados e quando é necessário considerar mais entradas na rede do que aquelas que foram previamente vistas. Podem ser utilizadas para diversas finalidades, como língua natural, tradução automática, previsão de séries temporais e até previsão da trajetória de veículos (Wang, et al., 2019).

Este tipo de rede neuronal pode ser apresentado de diversas formas. Existem redes neuronais recorrentes básicas, que são denominadas de redes neuronais *vanilla*, quando a ligação entre apenas entre dois nós da rede (Chen, et al., 2019). Para além deste caso mais básico, as redes neuronais recorrentes podem ser ainda de várias nó para apenas um nó ou de vários nós para vários nós.

2.6 Trabalhos relacionados

2.6.1 Aprendizagem automática e internet das coisas

Existem bastantes soluções já desenvolvidas que utilizam diversos algoritmos de aprendizagem automática em simultâneo com dispositivos IoT para a criação de sistemas inteligentes capazes de realizar ações de controlo. Algumas dessas soluções têm como foco a mesma área da aplicação que se pretende desenvolver, enquanto que outras soluções possuem uma aplicabilidade nas mais diversas áreas.

Foi desenvolvido um estudo onde foram comparados um total de trinta e seis arquiteturas de machine learning para se perceber qual a arquitetura que melhor respondia às suas exigências (Alawadi, et al., 2020). A sua aplicação é utilizada em sistemas de *Heating, ventilation, and air conditioning* (HVAC) – ventilação, calor e ar condicionado. A finalidade seria a utilização dos algoritmos para a previsão da temperatura interna de um edifício inteligente. O dataset utilizado foi obtido do centro de pesquisa CiTIUS e também foram recolhidas medições de sensores de uma estação meteorológica local. Para comparar a precisão dos diversos algoritmos foi utilizado o coeficiente R e o erro quadrático médio da raiz, enquanto que para obter a performance dos algoritmos, foi utilizada a classificação de Friedman. Após a realização dos testes, conclui-se que a arquitetura com melhor resultado foi a ExtraTrees com uma precisão obtida através do coeficiente R de 0,97% e uma precisão obtida através do RMSE de 0,058%. Em relação à sua performance, teve o maior resultado na classificação de Friedman.

Milos Manic, em parceria com outros autores, desenvolveu um estudo para a comparação de algoritmos de *deep learning* para previsão da demanda energética em edifícios (Manic, et al., 2016). Utilizaram um total de três arquiteturas de machine learning distintas: LSTM – *long short term memory* – LSTM S2S e RNC. O dataset utilizado foi relativo ao consumo elétrico de um indivíduo residente no edifício e contém um histórico de informação de quatro anos. Os

primeiros três anos foram utilizados para treinar as três arquiteturas, com o último ano a ser utilizado como dados de teste. A primeira arquitetura testada foi o LSTM, com a sua saída da rede a representar o consumo de energia previsto, relativamente ao histórico existente. Para o treino do LSTM foi utilizado o padrão BPTT – *backpropagation through time* – com a adição do método do gradiente descendente. Seguidamente foi testada a arquitetura LSTM S2S, que é um método de mapeamento sequências que possuem comprimentos arbitrários. Nesta arquitetura existem duas redes LSTM básicas, onde uma delas funciona como um codificador e a outra como um descodificador. Ao codificador é exigida a conversão da transformação das sequências de entrada, realizando a sua transformação de uma sequência que possui um comprimento variável para uma sequência de comprimentos fixo que será usado como o input do descodificador. O descodificador é quem obtém o vetor de saída que é a previsão dos valores de energia que serão gastos futuramente. A última arquitetura a ser testada foi a rede neuronal convolucional onde os dados são vistos sobre a forma de uma matriz de uma única dimensão. A rede é sustentada com um número predefinido de dados de consumo, que são posteriormente utilizados na fase de convolução e de pooling, antes de serem passados a uma rede totalmente conectada. É só nesta fase que alguns dados do dataset, como a data e a hora dos dados são utilizados pelo sistema. No treino desta arquitetura, à semelhança daquilo que se verificou no LSTM, foi utilizado BPTT com um modelo de gradiente descendente para a fase de treino.

Existem também estudos acerca da segurança de um mecanismo de gestão de dados – DSM – utilizado numa *smart grid* que utiliza elementos provenientes de IoT (Babar, et al., 2020). Esse DSM está estrategicamente no centro da smart grid onde recebe informação de diversas HAN – redes de área doméstica, que por sua vez recebem a energia diretamente de postes de eletricidade ou de fontes de energia renováveis como o sol ou o vento. O objetivo é a proteção do DSM de ataques maliciosos. Para classificar a informação recebida pelo DSM como sendo informação segura ou informação maliciosa, foi usado o classificador Naive-Bayes e era pretendido treinar o analista de monitorização, um dos elementos do agente resiliente, que é o responsável por manter a integridade do DSM, e por verificar que as entidades que tentam entrar na rede são entidades honestas e seguras. Foram definidos os valores de segurança, sendo que um valor igual a 1 representa um segurança do DSM completa, valores menores que 1 e superiores a 0,51 consideram que o sistema ainda se encontra seguro e valores inferiores a 0,51 considera-se que o sistema se encontra inseguro, sendo que um valor de segurança igual menor que 0,26 já indica que não existe qualquer tipo de segurança no sistema. Numa primeira fase o Naive-Bayes foi aplicado a uma matriz com dimensão 340x900 sendo que 900 eram serviços e os restantes 340 eram consumidores. Para medir a eficiência dos resultados obtidos pelo classificador Naive-Bayes, foi utilizada uma matriz de confusão. Concluiu-se que o classificador tinha taxas de acertos superiores a 90% para valores verdadeiros, fossem eles negativos ou positivos e uma taxa de acerto inferior a 8% para os valores classificados como falsos, o que traduz uma classificação bastante boa por parte do classificador utilizado.

Yu Liu et al desenvolveram um estudo para deteção de anomalias através da utilização de elementos de IoT com vista ao controlo do clima interno dos edifícios (Liu, et al., 2020). O dataset utilizado para o treino e testes dos modelos de machine learning eram acerca dos níveis

de CO_2presente num lar de idosos na Suécia. Os níveis de CO_2 eram recolhidos a cada trinta segundos. Um segundo dataset foi utilizado, com o histórico das temperaturas que iam sendo registadas no local. Os modelos de redes neuronais aplicados ao estudo foram as redes neuronais artificiais – ANN – o LSTM e o bi-LSTM e LSTM-ED. Alguns hiperparâmetros foram devidamente usados para a performance ótima de cada modelo. Como métrica para a função de perda foi utilizado o erro quadrático médio e para um ajuste da taxa de aprendizagem, foi utilizado o otimizador de Adam. Os períodos de treino foram feitos em 10 épocas para um melhor equilíbrio entre desempenho e a eficiência de tempo. Garantiram ainda que em cada época o conjunto de treino era baralhado para que o modelo tivesse um alcance global e reduzisse o overfitting. Para além disto, 10% dos pontos do conjunto de treino eram selecionados como um conjunto de validação em cada época. Pretendiam assim que o modelo acabasse por se ajustar em demasia aos dados de treino. Após a realização do treino e dos testes, conclui-se que dos três, o modelo com os melhores resultados registados foi o LSTM-ED, um modelo adicional utilizado, que obteve os melhores resultados de previsão tanto no dataset relativos aos níveis de CO_2 registado no local, como em relação à temperatura.

Aria Jozi et al escreveram em 2019 sobre a implementação de um sistema SCADA no edifício do GECAD (Jozi, et al., 2019). O sistema tem como principal função a monitorização e gestão da energia no edifício. Para além do sistema SCADA - Sistemas de Supervisão e Aquisição de Dados – foi ainda proposto a junção com um sistema de *forescasting* em tempo real e automático do consumo de energia. A ideia da junção destes dois elementos era para aproveitar a influência da gestão energética de um edifício, através do valor de consumo previsto confiável. O sistema SCADA implementado possui diversos elementos tais como: sensores, medidores de energia e alguns PLCs - *Programmable Logic Controllers.* A principal função do sistema é ser capaz de obter em tempo real todos os dados relativos ao edifício. Esses dados são vários e dizem respeito à temperatura sentida, aos níveis de CO_2 existentes no local, à energia consumida, entre outros dados. De realçar que o sistema está incorporado com cinco métodos de *forecasting*, nomeadamente ANN, SVR, RF, WM e HyFIS. Esses métodos são utilizados em linguagem *Python* e em linguagem R. Após a utilização de todos os métodos, os seus resultados foram comparados e foi concluído que os métodos que utilizam linguagem R têm melhores resultados comparativamente aos métodos que utilizam linguagem *Python*, sendo que de entre os métodos utilizados, o que produziu melhores resultados foi o SVM.

2.6.2 Sistemas multiagente e internet das coisas

Roana Pal e outros autores desenvolveram uma revisão acerca da aplicação de sistemas multiagente a uma smart grid. Para além de possuir um sistema multiagente, estão também envolvidos nessa smart grid alguns dispositivos IoT, utilizados principalmente para uma gestão energética (Pal, et al., 2021). A smart grid é então gerida com um sistema multiagente, que possibilita a autonomia, reatividade, decisão e outro tipo de característica do sistema. A sua framework funciona através da coordenação de trabalho dos diversos agentes, que têm como objetivo a gestão da informação e a sua monitorização, recebendo informações do sistema e fazendo a sua transição para a camada de agente que é superior a si, na hierarquia do sistema.

Para a smart grid em estudo, que é para a gestão de estações de carregamento inteligente, os componentes presentes na mesma são o agente de ativação, que responsável pela monitorização e representa uma empresa prestadora de serviços, como um fornecedor energético. O agente de transporte público elétrico cuja utilização pretende prever e simular o comportamento dos transportes públicos elétricos de forma a maximizar lucros. O agente de estação de carregamento que fornece todos os serviços relacionados com carregamentos. O agente *power grid* que recebe as respostas de necessidade de carregamento e ajusta toda a distribuição do sistema. Todos estes agentes possuem um papel bem definido e fundamento no sistema da smart grid de estação de carregamento, e todos agem de acordo com a informação que é recolhida pelos diversos dispositivos IoT integrantes do sistema.

Takuo Suganuma et al apresentaram uma proposta de arquitetura FLEC que tem como objetivo a resolução de problemas que ocorrem em sistemas tradicionais de IoT e de *edge computing*, essencialmente devido à rigidez desses sistemas (Suganuma, et al., 2018). A arquitetura FLEC é um modelo mais flexível e capaz de ser adaptar tanto ao ambiente como ao utilizador. A configuração desta arquitetura, apresentada pelos autores do artigo, tem como base a utilização de um sistema multiagente, que controlam cada componente que integra a arquitetura. Os agentes podem ou não estar inseridos nos dispositivos IoT. Caso os agentes estejam inseridos nesses dispositivos, existe uma camada de software denominada de plataforma de agente, que é instalada em cada dispositivo para coordenar as necessidades de cada agente. Os agentes e os seus protocolos são quem fornece à arquitetura a sua capacidade de adaptação ao ambiente. Cada agente possui um protocolo de constituição da organização com base na privacidade do contrato, o que implementa a configuração e reconfiguração dinâmica de uma organização virtual com base nas possíveis mudanças existentes no ambiente onde a arquitetura atua, o que garante a capacidade de adaptação ao ambiente. Com a utilização de um sistema multiagente, a plataforma escolhida para implementar o sistema foi a COSAP, que quando utilizada para uma arquitetura FLEC inclui cinco agentes principais. O agente do utilizador que transforma um requisito de um utilizador em algo que pode ser compreendido pelos outros agentes. O agente MGR que é responsável pela gestão de sensores e aplicações, geridas de forma dinâmica e em função dos pedidos do utilizador. O agente de sensor que age como um sensor do utilizador. O agente da aplicação, que funciona como uma aplicação do utilizador. E por fim um agente de recurso, cuja tarefa é recolher e gerir as informações relativas a cada recurso que é utilizado pela arquitetura.

Michal Janošek apresentou uma revisão acerca de casas inteligentes e sistemas multiagente (Janošek, 2019). A utilização dos sistemas multiagentes deve-se à sua capacidade de modelar sistemas complexos e garantir soluções viáveis. Essas soluções podem aparecer sob a forma de simulações, sendo que o objetivo da mesma é simular o comportamento humano dentro da casa, utilizando os agentes inteligentes. Essa simulação possui três partes fundamentais que são o editor de ambiente visual, a criação de agente autónomos e o núcleo de simulação. No que diz respeito à arquitetura apresentada pelos agentes, o autor da revisão refere que cada agente representa um dispositivo como uma televisão ou uma fonte luminosa. No fundo, os agentes possuem na sua constituição sensores, atuadores, módulos de tarefas, de comunicação, entre outros. Para a conexão dos agentes com a internet, foi utilizado o IoT Gateway. Quanto à

arquitetura apresenta pelo sistema multiagente, composto por todos os agentes envolvidos, são descritos quatro agentes. Um deles é o agente de sensor, responsável pela gestão dos sensores. Um outro agente é o agente da base de dados, responsável por toda a gestão dos dados que são recolhidos do ambiente pelos diversos sensores. Depois, o agente de decisão, que determina a necessidade de ação de cada agente, em função do objetivo final e dos dados existentes no momento da decisão, e por fim, o agente de ação, responsável pela execução das ações que são determinadas pelo agente de decisão.

Em 2019 foi também publicada uma revisão tendo como tema principal as formas de reduzir os custos operacionais, aumentar a eficiência e os resultados obtidos nas organizações OGI, que são as organizações da indústria de petróleo e gás (Hanga & Kovalchuk, 2019). A utilização de sistemas multiagente entram como uma opção de solução devido à sua capacidade de distribuição de tarefas, o que é ideal para aplicação em organizações onde potencialmente existem objetivos que são conflituantes com os demais. Para a maximização da quantidade de petróleo produzido, com a limitação do volume de areia e água utilizada, foi proposto um sistema multiagente para controlo e otimização da produção de petróleo. Alguns artigos referem como ferramenta de design para sistemas multiagente a ferramenta *Prometheus*, que possibilita uma construção de sistema multiagente com múltiplos agentes sendo que a sua produção tem como base as tarefas que lhe serão atribuídas. Outros autores apresentam como proposta o ambiente de agentes inteligentes denominado JACK que possibilita a construção, execução e integração de sistemas do tipo. No que diz respeito a questão de conflito de controlo em áreas de atuação como organizações OGI, foi desenvolvido um modelo de controlo multiobjetivo extensível, que se trata de um modelo MAS estratificado onde os objetivos de controlo são atribuídos a vários agentes através de um meio de contexto de negociação de três camadas, que são a camada de decisão, a camada tática e a camada operacional. Para além da correta gestão das organizações OGI para aumento de lucro e redução de custos operacionais, questões de segurança, nomeadamente no que diz respeito ao evitar de fugas de componentes, foram apresentadas formas de o realizar. Uma das soluções apresentada foi um sistema MAS para prevenção e controlo de vandalismo em oleodutos, onde a engenharia dos sistemas multiagente era constituída tanto por agentes reativos e agentes proativos que executavam juntos para atingir os seus objetivos de segurança. Existiam alguns agentes predefinidos tais como o agente inteligente que tinha como tarefa a deteção de uma pressão anormal e que alertava outro agente, denominado de agente móvel. O agente de sensor era o responsável por detetar irregularidades no sistema através dos seus sensores e avisava o agente mestre, responsável pela central de controlo, sempre que alguma irregularidade fosse detetada. Por fim, o agente de proteção móvel era o responsável pela primeira linha de proteção. Os sistemas para segurança e manutenção eram diversos e diversificados. Podiam ou não ter os mesmos agentes em comum, no entanto partilhavam o mesmo objetivo, o de garantir a seguranças da indústria OGI.

Brígida Teixeira et al apresentaram em 2018 uma *framework* denominada TOOCC – *Tool Control Center* – que tem como objetivo a possibilidade de operação entre sistema heterogéneos de energia e potência, com recurso a ontologias definidas (Teixeira, et al., 2018). A *framework* é um sistema de suporte à decisão multiagente que é capaz de criar, analisar e simular cenários

do mercado de eletricidade, através da operação de sistema heterogéneos de simulação. Possui a capacidade de realizar projeções de uma determinada rede elétrica ou programar a rede de uma casa para um conjunto de horas futuras, com base em algumas restrições. Essas projeções baseiam-se em diferentes horizontes temporais e são a representação de uma perspetiva que é considerada proactiva, ou seja, que age antecipadamente, em função dos dados existentes. O sistema multiagente escolhido para funcionar no TOOCC foi o MASCEM - *Multi-Agent Simulator for Competitive Eletricity Markets* – que tem como principal objetivo a simulação de cenários relativos aos comportamentos dos mercados elétricos e das suas entidades, através da utilização de algumas regras de mercados internacionais como o mercado ibérico e o mercado europeu central. O sistema funciona com dados reais, como informações retiradas de HVAC, que são informações de consumo, mas também dados acerca da produção de energia por meio solar e do vento, e ainda as condições meteorológicas que se fazem sentir.

2.6.3 Sistemas multiagente com IoT e aprendizagem automática

Chao Liang et al desenvolveram uma solução para um sistema de deteção de intrusões, utilizando dispositivos IoT e com uma abordagem de machine learning (Liang, et al., 2019). Foram duas as técnicas de machine learning escolhidas e aplicadas à solução desenvolvida. Deep learning foi uma das técnicas escolhidas devido à sua transformação de dados complexos em expressões mais simples e abstratas. A outra técnica foi a aprendizagem por reforço multiagente, que é uma subárea importante da aprendizagem por reforço, onde os agentes podem aumentar a sua eficiência através de um processo de treino mais contínuo. Relativamente aos agentes e respetivo sistema multiagente desenvolvido, os autores alocaram os agentes em quatro módulos diferentes, sendo que cada um desses módulos comunica com os restantes através dos agentes inseridos em cada um deles. Os módulos existentes são de recolha de dados, o seu processamento, seguido de um módulo de deteção e análise e por fim um módulo de resposta. A linguagem utilizada por cada agente para a comunicação foi o FIPA-ACL. Existem um total de sete agentes, cada um responsável por uma atividade singular. O agente de comunicação é responsável pela comunicação entre todos os agentes presentes no sistema. O agente de recolha é quem recolhe dos dados dos dispositivos IoT e de enviar esses dados aos agentes de comunicação. O agente de gestão de dados tem como tarefa gerir os dados que são provenientes de diferentes redes, e realizar o tratamento que é necessário, como integração e padronização dos dados. O agente da base de dados é o único que pode alterar algum tipo de dado que está presente na base de dados, mas apenas com a supervisão do agente de comunicação. O agente de treino pode ler os dados da base de dados, mas nunca os alterar. É responsável pelo treino dos modelos de deteção e por enviar esses modelos aos agentes de comunicação, depois de concluídos os mesmos. O agente de deteção, em função do protocolo existente e do tipo de dados, escolhe o modelo de deteção que mais se enquadra para lidar com os possíveis ataques existentes. Por fim, o agente de resposta é quem envia avisos e medidas de defesa e medidas de defesa contra potenciais ataques, ao agente de comunicação.

No presente ano de 2022 foi apresentada uma proposta para a resposta às necessidades de realocação de recursos com utilização de sistemas multiagente e dispositivos industriais do IoT (Rosenberger, et al., 2022). A solução tem como objetivo colmatar os problemas que surgem devido às limitações computacionais existentes e à necessidade de utilizar de forma quase perfeita os recursos disponíveis para os dispositivos IoT. Assim, esta solução apresenta um sistema multiagente descentralizado, com um agente existente para lidar com os problemas de alocação de recursos e um outro agente que fica responsável pela alocação dos recursos de rede. Devido à descentralização de funções, torna-se possível que tarefas que não sejam necessárias de realizar em determinado momento, estejam a ser realizadas no mesmo instante temporal de tarefas cuja necessidade de realização é imperativa. Desta forma é garantida uma gestão muito mais eficiente do sistema, sem que existam perdas de performance ou problemas significativos de recursos, sejam eles de que tipo específico.

Tiago Pinto et al desenvolveram, no presente ano de 2022, uma solução para uma simulação inteligente para a gestão energética em edifícios e *microgrids*, denominada de MARTINE - *Multi-Agent based Real-Time INfrastruture for Energy* (Pinto, et al., 2022). Possui cinco camadas distintas sendo que uma delas é referente a um sistema multiagente e denominada camada do sistema multiagente. Esta camada permite a representação e modelação dos elementos que não podem ser representados de forma física pelo sistema na plataforma em questão. Possui ainda uma camada onde estão contidos os mecanismos de aprendizagem automática e de otimização, denominada camada de conhecimento. Incluem modelos e algoritmos que suportam a tomada de decisão. De realçar ainda a denominada camada ciberfísica onde se encontram os elementos IoT que fazem parte do sistema, responsável pela integração dos recursos físicos. O caso de estudo descrito envolve a previsão dos consumos que serão realizados tendo por base o histórico de consumo dos oito meses anteriores. Os dados dos oito meses anteriores foram considerados, tendo como objetivo a previsão de consumo existente no edifício no período dos seis dias seguintes em relação à data de início de previsão. Para a previsão, foram utilizados dois algoritmos de aprendizagem automática: redes neuronais e o algoritmo *k-nearest neighbor*. Para obter os erros de *forecasting* foram utilizadas três métricas distintas: WAPE, SMAPE e RMSPE. Conclui-se que o algoritmo com menor erro de *forecasting* foi a ANN, comparativamente com o *k-nearest neighbor*, comparativamente com uma percentagem de acerto na previsão entre 87% a 95%.

2.6.4 Lacunas identificadas

Uma das principais lacunas identificadas na pesquisa realizada por soluções existentes na área foi a falta de combinação dos três elementos que serão utilizados no sistema proposto, isto é, existem soluções válidas e interessantes que utilizam a componente IoT, algumas outras que utilizam a componente de aprendizagem automática ou a componente de sistema multiagente. No entanto, quando se realiza a pesquisa pelo conjunto das tecnologias, verifica-se que não existem muitas soluções já existentes nesse âmbito. Essa verificação é ainda proeminente quando se pesquisam por soluções que utilizem as três componentes e que sejam aplicadas à

área em questão, que neste caso é a área energética e mais concretamente o sistema multiagente que tenham como objetivo a gestão energética de um edifício.

2.7 Benefício das Tecnologias

Os dispositivos de Internet of Things, como sensores, por si só, não são elementos inteligentes ou que possuam capacidades de aprendizagem ou de adaptação. São dispositivos que cumprem a tarefa para a qual foram concebidos, como um sensor de temperatura que mede a temperatura do local onde está inserido, que por serem do tipo IoT, podem estar ligados à internet ou a uma rede, e que possuem a capacidade de armazenar os dados que vão sendo recolhidos (Hanga & Kovalchuk, 2019). Com a utilização da aprendizagem automática existe um aumento das potencialidades dos sistemas IoT, com a introdução de inteligência artificial nos mesmos. Para uma gestão inteligente de um edifício, como é o caso do sistema que se pretende desenvolver, é necessário que os dispositivos IoT utilizados tenham capacidade de imitar o comportamento do ser humano, o que implementa nestes dispositivos a capacidade de decisão, de forma justificada e planeada, sem a necessidade de intervenção humana. No fundo, tornam estes dispositivos IoT inteligentes, capazes de agir e reagir em função daquilo que já viram e daquilo que estão a ver no momento em que têm que tomar uma decisão sobre uma ação a realizar.

Apesar de os dispositivos IoT terem a capacidade de serem ligados à internet ou a uma rede particular utilizada num sistema concreto, de terem também a capacidade de manter os dados que recolhem, a sua integração como um sistema torna-se apetecível utilizando sistemas multiagente. Este tipo de sistema permite a integração dos dispositivos IoT integrantes do sistema, a troca de informação entre eles, mesmo que os dispositivos se encontrem em diferentes camadas hierárquicas (Kumar & Mallick, 2018). Os agentes, que podem ou não ser incluídos dentro de cada dispositivo IoT, são utilizados para a modelação de sistemas complexos, onde é necessária a troca de informação entre dispositivos, para a realização de determinada ação. Se imaginarmos um sistema que se pretende inteligente, responsável por ligar o aquecimento central de um apartamento em função de dois dados distintos: a temperatura sentida dentro do apartamento e a percentagem de humidade existente. São necessários dois sensores distintos, um que realize a medição constante da temperatura e outro que faça o mesmo para a humidade. O atuador responsável pela ativação do aquecimento central do apartamento necessita de receber os dados dos dois sensores, para agir em conformidade. Caso cada um dos dispositivos tenha a si associado um agente, num sistema multiagente, então o agente do sensor de temperatura, bem como o agente do sensor de humidade, envia os dados recolhidos pelos seus sensores, ao agente do atuador. Desta forma toda a informação existente e necessária chega com o devido sucesso ao agente do atuador. Com os dados em seu poder, e considerando que o sistema utiliza machine learning para conceber inteligência artificial aos seus vários componentes, podemos considerar que o sistema irá agir conforme aquilo que é esperado dele, ou seja, que caso a temperatura esteja inserida num determinado intervalo e

que se verifica o mesmo na percentagem de humidade, então o aquecimento central deve ser ativado por parte do atuador.

Pelo motivos apresentados e considerando as lacunas identificadas em soluções do mesmo género e que atuem na mesma área de atuação, a solução desenvolvida pode ter um papel de destaque e representar uma solução inovadora e única, uma vez que se trata de um sistema onde os dispositivos IoT serão utilizados, tendo por base um sistema multiagente onde cada agente irá representar um desses dispositivos, e onde a inteligência artificial automática tem um papel fundamental de incutir a capacidade de aprendizagem automática a todo o sistema, munindo assim os dispositivos IoT de uma capacidade de ação em função do seu conhecimento. Integram-se assim os três principais pilares do sistema, e cria-se uma solução com esse potencial, na área da eficiência energética.

3 Desenvolvimento e experimentação

Neste capítulo serão apresentadas as experimentações realizadas e que constituem a solução final do sistema desenvolvido. Tem como principal objetivo a apresentação e detalhe dos métodos e ferramentas selecionadas, cobrindo os casos de estudo selecionados e devidamente testados, cobrindo as três principais áreas do sistema.

3.1 Arquitetura do sistema

Tratando-se de um sistema multiagente que controla dispositivos IoT que realizarão ações em função das condições que encontram no meio, é necessário fornecer a esses dispositivos as capacidades de ação/reação necessárias. Definiu-se que cada agente virtual PEAK é responsável por um dispositivo IoT, sendo essa a sua representação no sistema multiagente.

Uma vez que os dispositivos IoT utilizam o protocolo de comunicação MQTT para realizar a sua comunicação, mas os agentes PEAK utilizam o protocolo de comunicação XMPP, foi necessário criar *gateways* de comunicação, para que os agentes PEAK possam enviar mensagens para os dispositivos IoT e também para que os dispositivos IoT consigam enviar respostas para os agentes PEAK do sistema. Para isso foram desenvolvidos duas *gateways* de comunicação: uma delas recebe todas as mensagens que os agentes PEAK pretendem enviar para os dispositivos IoT e efetuam o trabalho de publicar essas mensagens para os tópicos corretos para que os dispositivos recebem as mensagens que lhes são endereçadas. Uma outra gateway faz o sentido contrário da comunicação, isto é, recebe todas as mensagens que os dispositivos IoT enviam, através da publicação para um tópico predefinido, e após receberem essas mensagens, tratam de as enviar para os agentes PEAK de destino. A arquitetura do sistema pode ser observada na imagem seguinte.

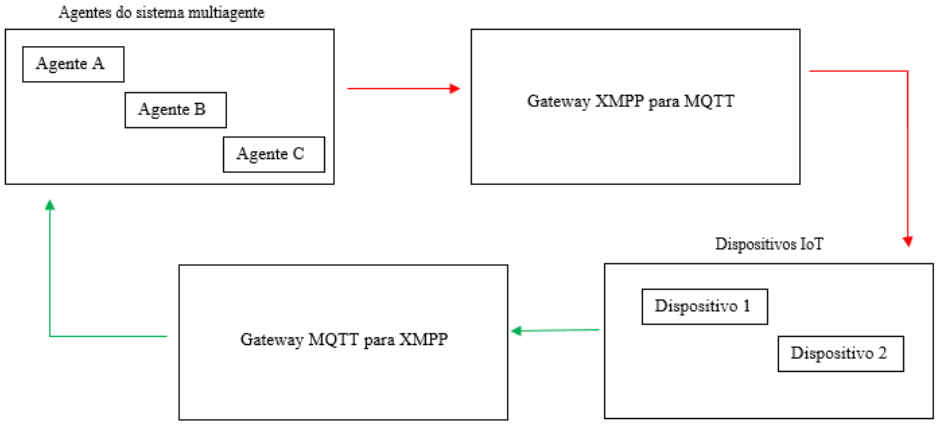

Figura 4 – Arquitetura do sistema

Na figura 4 é possível observar os principais elementos que fazem parte do sistema. O sistema é essencialmente agrupado por elementos que fazem parte do grupo de IoT e elementos que fazem parte do grupo de SMA. As setas a vermelho representam a comunicação que é efetuada através do protocolo de comunicação XMPP, enquanto que as setas verdes representam a comunicação que é efetuada através do protocolo de comunicação MQTT. O primeiro tipo de comunicação acontece entre os elementos do SMA, que neste caso estão representados como o sistema multiagente com diversos agentes no seu interior, e a gateway de comunicação que realiza a troca de protocolo de comunicação. No segundo tipo de comunicação, as setas verdes, representam a comunicação entre os elementos do IoT, que neste caso são os dispositivos IoT que estão representados, e o gateway de comunicação.

3.2 Sistema multiagente

A criação de um sistema multiagente exige alguns detalhes que variam em conformidade com as várias possibilidades existentes tais como a plataforma que será selecionada para a construção do sistema multiagente, o tipo de comunicação e protocolo de comunicação que pode ser utilizado nesse sistema ou o cliente que pode ser selecionado para alojar os agentes que serão criados e que constituirão o sistema multiagente. Por existirem várias possibilidades em cada uma das entidades mencionadas, a seleção das mesmas torna-se imperial.

Para a construção do sistema multiagente, a escolha recai sobre a plataforma de sistemas multiagente PEAK, escrito com recurso à linguagem de programação Python e que tem como base a plataforma multiagente SPADE, que se trata de um sistema instantâneo de mensagens denominado XMPP (Palanca, 2020). Esta plataforma PEAK possui as funcionalidades da plataforma na qual se inspira, mas para além disso potencia outras funcionalidades que são muito interessantes e úteis para o sistema que foi desenvolvimento, como é o caso da possibilidade de criação de grupos de agentes.

Para a criação de agentes é apenas necessária uma ligação XMPP a um servidor que pode ser criado de raíz e utilizado para o efeito. No caso concreto do sistema desenvolvimento, esse servidor é fornecido pelo GECAD. Nesse servidor os agentes são identificados pelo seu JID, do qual faz parte o seu nome próprio. Um exemplo de um JID de um agente da plataforma seria apresentado na seguinte forma: nome_agente@mas.gecad.isep.ipp.pt.

3.2.1 Protocolo de comunicação - XMPP

XMPP é o acrónimo de *Extensible Messaging and Presence Protocol* e trata-se de um protocolo de comunicação para serviços de mensagens instantâneas, multi-chats, chamadas de voz e de todo o tipo de conteúdo que possua data do tipo XML (Org, s.d.). É o protocolo de comunicação utilizado pela plataforma PEAK para gerir internamente as comunicações dos agentes do sistema, que depois de realizarem a sua conexão ao servidor do qual fazem parte, abrem um canal de comunicação XMPP, que é aberto e persistente, e que se encontra ligado à plataforma

3.2.2 *Cliente Pidgin*

Uma vez que os agentes do sistema multiagente são virtuais, para ser possível uma melhor perceção daquilo que acontece entre os agentes, pelo menos numa fase mais básica do sistema, foi selecionado o cliente Pidgin (Developers, s.d.). O que este cliente possibilita é a criação de uma conta nesse cliente, que funciona como um agente depois de criada. Trata-se de um cliente que já possui alguns fatores importantes como a segurança das comunicações, existindo uma encriptação end-to-end, onde é possível utilizar plugin de encriptação tais como OMEMO ou PGP. Possui ainda uma filosofia de mensagens descentralizadas. Na figura 5 é possível observar um exemplo de troca mensagens entre os agentes do sistema multiagente.

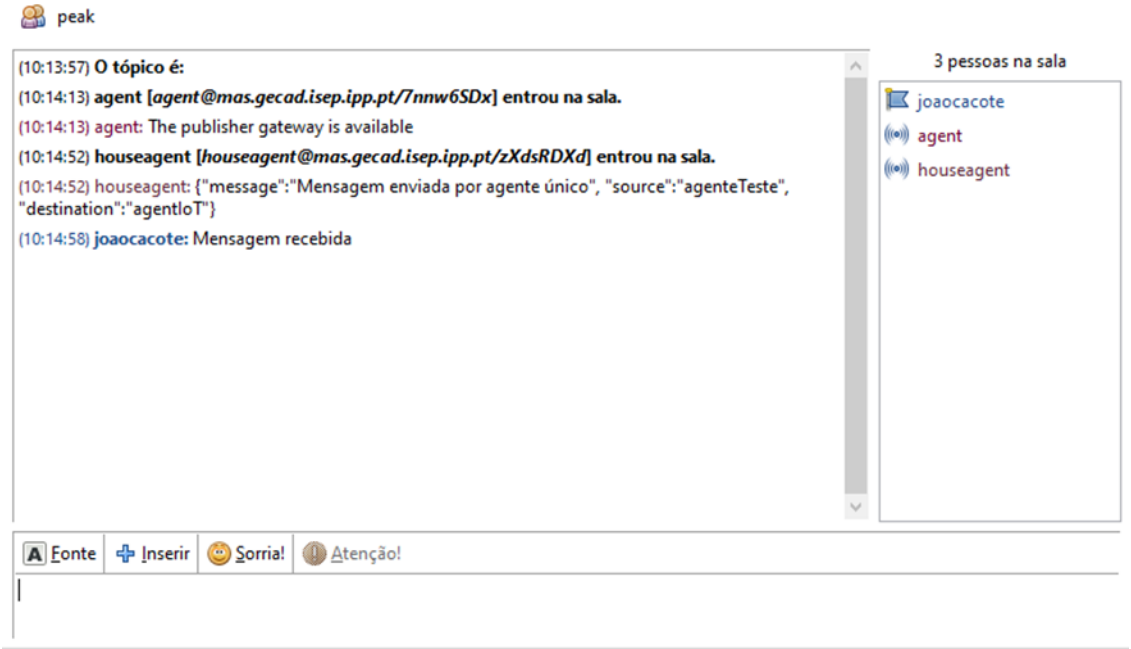

Figura 5 - Cliente Pidgin

O que a figura 5 representa no seu todo é a gateway de comunicação de XMPP para MQTT. É possível observar que quanto um agente envia uma mensagem para um dispositivo IoT, essa mensagem chega ao gateway no formato JSON, sendo depois processada pelo gateway para que possa ser enviada para o dispositivo IoT de destino.

3.2.3 *Criação de agentes*

A criação de um agente do sistema multiagente pode ser realizada através de um único script Python, escrito para essa função. Quando um agente é criado, é a escolha do utilizador decidir que tinha de ações esse agente executa, sendo que pode ter diversos tipos de comportamento como comportamentos únicos (enviar uma mensagem) ou comportamentos cíclicos, como estar atento a um determinado canal de comunicação e indicar que uma mensagem foi recebida nesse canal. Estes dois comportamentos foram os utilizados nos agentes criados para preencher as necessidades do sistema, isto é, ou os agentes eram portadores de um comportamento único

ou então eram portadores de um comportamento cíclico. Essa definição foi aplicada em função das necessidades do sistema.

3.2.4 Comunicação entre agentes

Como referido na arquitetura do sistema, qualquer mensagem que um agente PEAK pretenda enviar para um dispositivo IoT passa pela *gateway* de comunicação que é responsável por enviar a mensagem para o dispositivo de destino. O mesmo não acontece quando as mensagens são trocadas entre agentes PEAK, onde nesse tipo de situação as mensagens são trocadas diretamente entre os dois agentes PEAK sem a necessidade de intervenção da *gateway*. Os agentes são sempre identificados pelo seu JID, sendo essa a forma de se perceber para qual agente PEAK determinada mensagem deve ser enviada. Na imagem que se segue é possível ver a criação e envio de mensagem por parte de um agente.

```
class createAgent(Agent):

    class SendMessage(OneShotBehaviour):

        async def run(self):
            msg = Message(to='peak@conference.'+self.agent.jid.domain)
            msg.body = '{"message":"Mensagem enviada por agente único", "source":"agenteTeste", "destination":"agentIoT"}'
            await self.send_to_group(msg)
```

Figura 6 – Criação de agente e envio de mensagem

Quando um agente PEAK envia uma mensagem para um dispositivo IoT, essa mensagem tem que ser enviada no formato JSON para que possa ser corretamente interpretada no lado do MQTT e também porque é mais percetível retirar da mensagem os elementos individuais. Uma mensagem enviada por um agente PEAK tem sempre três elementos distintos: o corpo da mensagem, o destino que é a representação de quem a mensagem necessita de ser entregue e o identificador de quem enviou a mensagem, que neste caso é sempre o nome do agente que procedeu ao envio da mensagem. Este elemento é útil e fundamental para que o dispositivo IoT saiba a quem deve responder quando for necessário enviar de volta a sua resposta.

3.3 Gateway de comunicação de XMPP para MQTT

3.3.1 Envio de mensagem por parte de agente PEAK para dispositivo IoT

Sempre que uma mensagem é enviada e tem como destino um dispositivo IoT, essa mensagem passa pelo *gateway* responsável por enviar essa mensagem para o dispositivo IoT de destino. Esse gateway é perfeitamente coordenado, isto é, está sempre à escuta de alguma mensagem que seja necessária retransmitir para os dispositivos IoT, mas apenas atua quando uma recebe uma mensagem de algum agente PEAK. Enquanto essa mensagem não é recebida, a única tarefa que possui é aguardar e escutar o canal de comunicação.

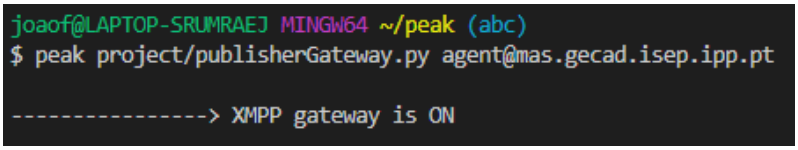

Figura 7 – Gateway XMPP para MQTT a aguardar mensagens

Assim que algum agente envia uma mensagem com destino a um dispositivo IoT, a gateway recebe essa mensagem, e retira da mesma os elementos necessários para saber qual o dispositivo IoT ao qual a mensagem se destina. Essa informação é obtida através do campo *destination* da mensagem JSON. Na figura 8 é possível ver que o agente responsável pela temperatura da casa 190 está a enviar uma mensagem ao seu dispositivo IoT.

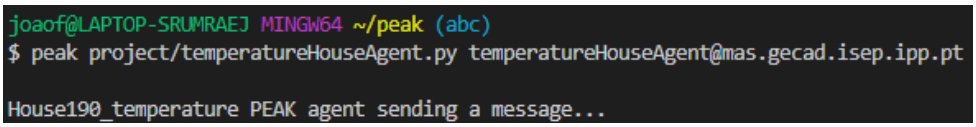

Figura 8 – Mensagem enviada pelo agente PEAK

Do lado da gateway a mensagem é recebida e processada e enviada para o tópico MQTT que é explicitado pelo destino que a mensagem possui.

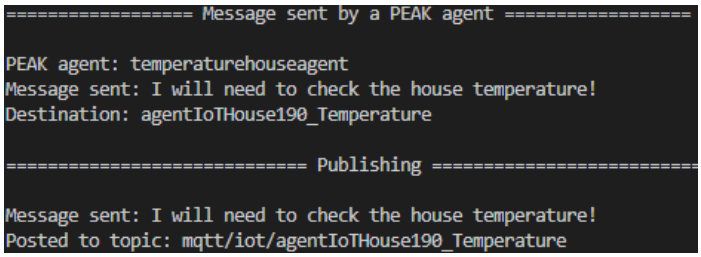

Figura 9 – Mensagem recebida e reenviada pela gateway

3.3.2 Receção de mensagem por parte do dispositivo IoT

O dispositivo IoT está em constante escuta no seu tópico específico, aguardando por alguma mensagem que lhe seja endereçada. Sempre que uma mensagem é publicada para o seu tópico específico, ele processa essa mensagem e, posteriormente, age em conformidade.

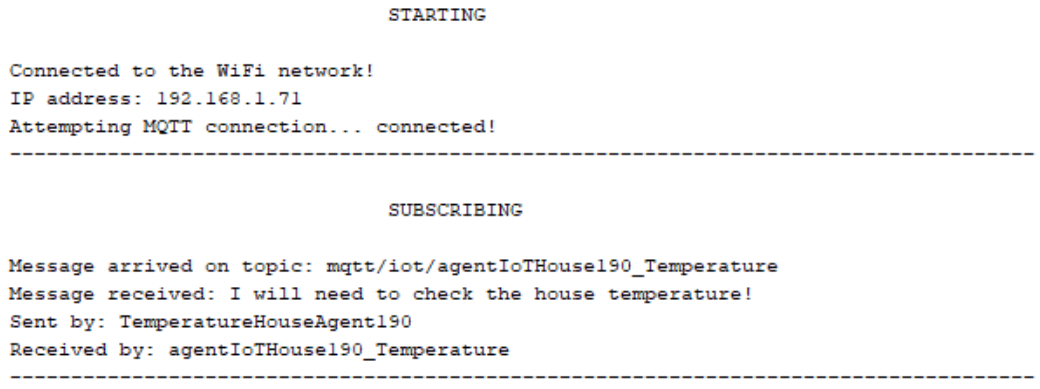

Figura 10 – Mensagem recebida pelo dispositivo IoT

Na figura 10 é possível observar o dispositivo IoT de temperatura a receber uma mensagem através do protocolo de comunicação MQTT. Cada dispositivo IoT subscreve um tópico MQTT, aquele no qual as mensagens lhe são enviadas. É possível observar na figura 10 qual o tópico que cada dispositivo IoT subscreve, bem como quem foi o agente PEAK que enviou a mensagem, o nome do próprio dispositivo IoT que recebe e o conteúdo da mensagem propriamente dita que o agente PEAK enviou ao dispositivo IoT.

3.4 Gateway de comunicação de MQTT para XMPP

Quando um dispositivo IoT pretende enviar uma mensagem para um agente PEAK, e tendo em conta que é utilizado o protocolo de comunicação MQTT, o dispositivo IoT publica uma mensagem para um tópico específico. Esse tópico é subscrito pela gateway de MQTT para XMPP e sempre que uma mensagem é publicada para esse tópico, essa gateway tem como tarefa receber essa mensagem, obter os seus elementos e enviar a mensagem para o agente PEAK de destino. Na figura 11 é possível observar o dispositivo IoT a enviar uma mensagem para um agente PEAK.

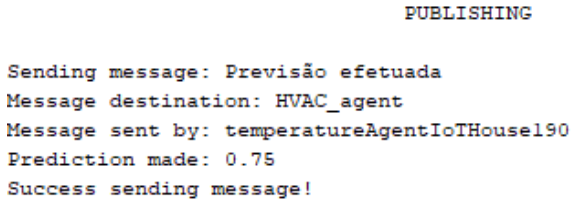

Figura 11 – Mensagem publicada pelo dispositivo IoT

É possível observar que nessa mensagem é explicitado quem é o dispositivo IoT que a está a enviar, para que o agente PEAK consiga saber quem enviou esta mensagem. É também sabido o agente PEAK de destino, informação fundamental para que a gateway saiba para quem deve endereçar a mensagem. Para além disso, é ainda enviado o corpo da mensagem e neste caso concreto, a previsão da rede neuronal artificial, que será objeto de maior detalhe mais à frente.

Quando a mensagem é recebida no tópico da gateway de comunicação de MQTT para XMPP, os elementos são recebidos e retirados da mensagem e é feito o envio para o agente PEAK correspondente, que neste caso específico tratava-se do HVAC_agent.

```
joaof@LAPTOP-SRUMRAEJ MINGW64 ~/peak (abc)
$ peak project/subscriberGateway.py sub@mas.gecad.isep.ipp.pt

-----> MQTT gateway is ON

================= Message sent by a IoT device =================

IoT device: temperatureAgentIoTHouse190
Message: Previsão efetuada
Destination: HVAC_agent
Prediction:0.75
```

Figura 12 – Mensagem recebida e enviada pela gateway

Quando a gateway envia a mensagem para o agente PEAK, o mesmo irá receber a mensagem, rever os seus componentes, mas mais importante: verificar a ação que é necessário tomar.

```
joaof@LAPTOP-SRUMRAEJ MINGW64 ~/peak (abc)
$ peak project/HVAC.py HVAC_agent@mas.gecad.isep.ipp.pt

================= HVAC PEAK AGENT =================

================= Received a message from an IoT device =================

IoT device sender: temperatureAgentIoTHouse190
Message: Previsão efetuada
Agent receiver: HVAC_agent
Action to make: warm up the division
```

Figura 13 – Mensagem recebida pelo agente PEAK

3.5 Rede neuronal artificial

Um dos grandes objetivos deste sistema era a implementação de redes neuronais artificiais nos microprocessadores para que os dispositivos IoT tivessem capacidade de decisão e inteligência para interpretar os valores que obtinham, e não apenas obterem esses valores. Desta forma foi desenvolvida uma rede neuronal artificial, com a sua configuração padrão descrita na figura 14. A rede neuronal artificial observada recebe input, que são depois sujeitos aos pesos da rede neuronal artificial, tanto na camada oculta intermédia, como na camada de saída. Os inputs da rede diferem do tipo de sensor que a está a utilizar.

Numa primeira fase, e de forma a garantir que o microprocessador tinha uma capacidade de processamento necessária, foi realizada uma pequena experiência com dois botões e um LED, onde o objetivo era treinar a rede fora do microprocessador, obter os valores dos pesos das duas camadas, tanto a camada oculta como a camada de saída, e colocar esses valores no interior do microprocessador. Esses valores são posteriormente utilizados para avaliar novos inputs que são recebidos do utilizador. Significa isto que a rede neuronal artificial foi treinada fora do microprocessador, e dentro do mesmo possui a capacidade de avaliação de novos dados de entrada.

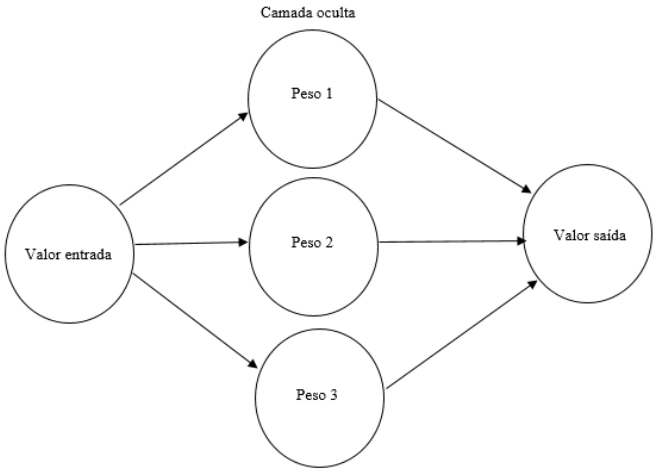

Figura 14 - Configuração base da rede neuronal artificial

3.5.1 Montagem e componentes do ESP32

Para a experimentação da aplicação de uma rede neuronal artificial a um microprocessador como o ESP32, foi desenvolvido um pequeno sistema experimental que é constituído por dois botões e por um LED.

A rede neuronal recebe como inputs os possíveis estados dos botões que são utilizados na montagem, bem como o possível estado do LED. Tratam-se de valores matriciais, o que indica que os valores de saída serão também eles no formato de uma matriz de valores.

Tabela 1 – Combinação de estados

Botão 1	Botão 2	LED
1	1	0
1	0	1
0	1	1
0	0	0

É esperado que o LED seja ativado quando apenas um dos botões está a ser premido, e que permaneça apagado caso ambos estejam a ser premidos, ou quando nenhum dos botões esteja a ser premido. De referir que a rede neuronal artificial é treinada para funcionar como uma porta lógica XOR, o que representa a tabela 1.

Para pequena demonstração da experiência descrita anteriormente é possível aceder ao seguinte link: https://youtu.be/HRNkdN9LKuM.

3.5.2 Rede neuronal experimental para alimentação do ESP32

A rede neuronal mencionada e utilizada para a experimentação foi desenvolvida em Python e trata-se de uma pequena rede neuronal de *feedforward* com *backpropagation*. A rede recebe como inputs os possíveis estados dos dispositivos físicos da montagem, retornando depois os pesos da rede neuronal como os valores de saída. De seguida apresenta-se um excerto da rede neuronal artificial experimental onde é possível observar os valores de entrada dos elementos de IoT, que neste caso são os dois botões e o LED, e a parte onde são realizados os cálculos para obter o output proveniente da rede neuronal artificial.

```
epochs = 10000                                  # Number of iterations
inputLayerSize, hiddenLayerSize, outputLayerSize = 2, 3, 1
L = 0.1                                         # learning rate

X = np.array([[0,0], [0,1], [1,0], [1,1]])      # Buttons states array
Y = np.array([ [0],   [1],   [1],   [0]])       # LED states array

def sigmoid (x): return 1/(1 + np.exp(-x))      # activation function
                                                # weights on layer inputs
Wh = np.random.uniform(size=(inputLayerSize, hiddenLayerSize))
Wz = np.random.uniform(size=(hiddenLayerSize,outputLayerSize))

for i in range(epochs):

    H = sigmoid(np.dot(X, Wh))                  # calculate forward part
    Z = np.dot(H,Wz)                            #
    E = Y - Z                                   # calculate error
    dZ = E * L                                  # delta Z
    Wz +=  H.T.dot(dZ)                          # calculate backpropagation part

    def sigmoid_derivate(H):
        return sigmoid(H) * (1 - sigmoid(H))

    dH = dZ.dot(Wz.T) * sigmoid_derivate(H)     #
    Wh +=  X.T.dot(dH)                          # update hidden layer weights
```

Figura 15 – Excerto da rede neuronal artificial

```
************** weights **************
input to hidden layer weights:
[[-0.19748447 -0.14539118 -0.45547055]
 [-0.18017022 -0.07207487 -0.45603191]]
hidden to output layer weights:
[[ 2.58310809]
 [-0.63574825]
 [-0.94996344]]
```

Figura 16 – Output da rede neuronal artificial

Estes dois vetores de saída são depois utilizados no código do ESP32, onde através desses pesos são realizados os cálculos para o caminho dentro da rede.

Dentro do ESP32, esses valores são recebidos para realizar os cálculos, considerando os pesos que são enviados pela rede neuronal artificial.

```
int X[1][2]    =   {{1,0}};

float W1[2][3] =   {{0.74000854,  4.47769531, -0.98692059},
                    {0.83034991,  4.48772758, -0.55733578}};

float W2[3][1] =   {{-6.17234487},
                    {4.8835918},
                    {1.28875386}};
```

Figura 17 – Valores de input no microprocessador

Depois de receber esses valores, eles serão utilizados para os cálculos necessários com a finalidade de determinar como os elementos IoT devem reagir em função dos estímulos provocados pelo ser humano, no caso desta experiência realizada, e que se trata da utilização dos botões para gerar ação luminosa no LED. É o valor de Y, calculado pela camada de saída, que irá definir a ativação ou não do LED.

```
/* Cálculo da direção forward com base nos pesos */
for(int i=0; i<1; i++)
{
    for(int j=0;j <3; j++)
    {
        for(int k=0; k<2; k++)
        {
            sum += X[i][k]*W1[k][j];
        }
        Wo1[i][j] = sigmoid(sum);
        sum = 0;
    }
}
/* Camada de saída */
for(int i=0; i<1; i++)
{
    for(int j=0;j <1; j++)
    {
        for(int k=0; k<3; k++)
        {
            Y += Wo1[i][k]*W2[k][j];
        }
    }
}
```

Figura 18 – Cálculos realizados pelo microprocessaodor

Após a realização desta implementação, foi possível perceber que o microprocessador possuía a capacidade necessária para incorporar uma rede neuronal artificial no seu interior. Um avanço muito benéfico uma vez que era assim possível assegurar que os microprocessadores iriam ser capazes de interpretar os dados das leituras que realizavam e inferir em função desses dados.

Essa dedução permite que o sistema seja autónomo na tomada de decisão, isto é, permite que à medida que um sensor efetue as suas medições, interprete os valores, decida e comunique a ação necessária de realizar ao agente PEAK responsável por si. Essa ação pode ser de aumentar a temperatura, de diminuir a intensidade luminosa das luzes de um determinado local, entre outros.

4 Casos de estudo

Neste capítulo pretende-se apresentar e descrever os casos de estudo que foram formulados e ensaiados para comprovar as funcionalidades do sistema desenvolvido. São cinco casos de estudo preparados e comprovados, onde são utilizados os dispositivos IoT e os agentes PEAK, bem como a demonstração da rede neuronal artificial cuja implementação foi feita no interior do microprocessador, concretizando assim o uso de inteligência artificial no sistema. O primeiro caso de estudo está relacionado com o sensor de temperatura e o dispositivo HVAC, e pretende-se mostrar de que forma o dispositivo IoT de temperatura lê a temperatura sentida, utiliza a ANN para obter uma classificação da ação que deve realizar, enviando essa informação para o HVAC. No segundo caso de estudo, é descrito o sensor de luminosidade, onde se pretende demonstrar de que forma as luzes podem ser controladas com base na utilização de uma ANN que está inserida no dispositivo IoT de luminosidade. No terceiro caso de estudo, é utilizado um sensor de presença para controlo das luzes. Demonstra como o dispositivo IoT prevê a presença num dado local e com base nessa informação, decide se as luzes devem estar ligadas ou desligadas. O quarto caso de estudo apresenta o agente PEAK responsável por uma divisão completa, neste caso a sala, recebendo informação dos três dispositivos IoT anteriores (temperatura, luminosidade e presença) para decidir como deve ser controlada essa divisão. Por último, o quinto caso de estudo apresenta o agente PEAK do edifício, que recebe informação de todos os agentes PEAK responsáveis por uma divisão, e pretende demonstrar como é feita a gestão energética do edifício, ou seja, com base nas informações recebidas pelos agentes PEAK das salas, decidir se a energia do edifício é suficiente, se por outro lado não o é e será necessário comprar energia, ou se existe excesso de energia, que pode ser vendida.

Para além da descrição detalhada dos casos de estudo, serão ainda descritos os processos de obtenção dos dados que cada caso de estudo utiliza, nomeadamente na rede neuronal artificial que foi utilizada em cada um. Tratando-se de um sistema que utiliza essas redes neuronais artificias para previsão de valores futuros e para incorporar inteligência na aplicação, a utilização de dados suficientes para o processo de treino é de extrema importância. Os dados utilizados para o teste e previsão foram, essencialmente, dados que provém dos diversos sensores existentes no edifício onde o sistema atua. Esses dados variam também em função daquilo que se pretende determinar, isto é, para diferentes aferições foram utilizados diferentes sensores e os valores obtidos através da leitura desses sensores não são sempre do mesmo tipo. Relativamente ao treino das redes neuronais artificias, os dados utilizados provêm de *datasets* que foram obtidos já construídos, ou que foram construídos propositadamente para os efeitos de desenvolvimento de sistema.

4.1 Caso de estudo 1: temperatura e dispositivo HVAC

Neste caso de estudo é utilizado o valor da temperatura, como um dos parâmetros da rede neuronal artificial, para obter uma previsão em função desses valores de entrada e perceber que tipo de ação é necessário realizar recorrendo a um sistema HVAC para controlar o ambiente do local no que diz respeito à temperatura que se faz sentir. Os elementos que constituem este caso de estudo e que intervêm no mesmo podem ser observados na imagem que se segue.

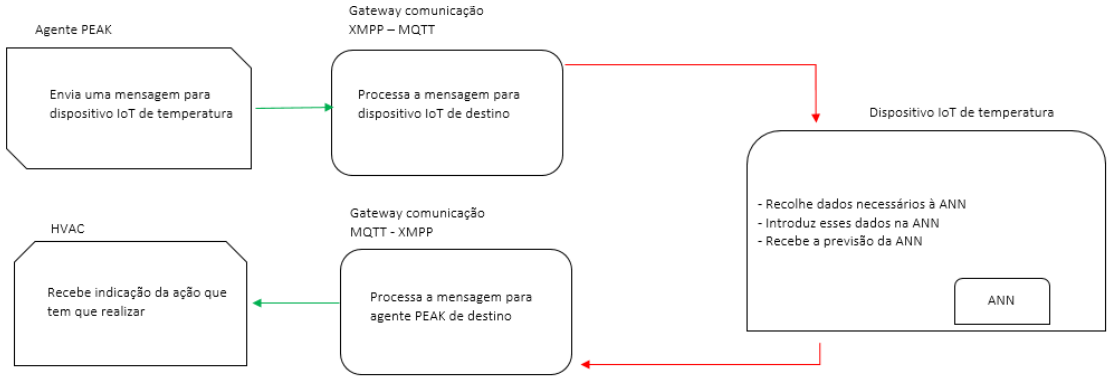

Figura 19 – Componentes e interação no caso de estudo 1

Na figura 19 estão representados os elementos desta interação, fazendo parte dela o agente PEAK atribuído ao dispositivo IoT de temperatura, ambas as *gateways* de comunicação, uma para realizar a comunicação de XMPP para MQTT e a outra para realizar a comunicação de MQTT para XMPP. Está também representado o dispositivo IoT de temperatura, com indicação para a presença da ANN dentro de si, e por fim o HVAC. Para cada elemento existe uma breve descrição daquilo que cada um faz, apenas para ser percetível de que forma cada elemento interage com o sistema.

4.1.1 Rede neuronal artificial

A rede neuronal artificial apresentada na figura 5 mantêm a sua configuração com exceção do número de inputs, dado que recebe neste caso particular um total de três inputs: tempo em que a leitura da temperatura foi efetuada (*timestamp*), a temperatura que foi obtida e a zona que indica onde o sensor se encontra. Esses dados são utilizados para treinar a rede neuronal artificial fora do microprocessador, obter os pesos da camada intermédia oculta e da camada de saída, para posteriormente serem utilizados para realizar de previsões dentro do ESP32.

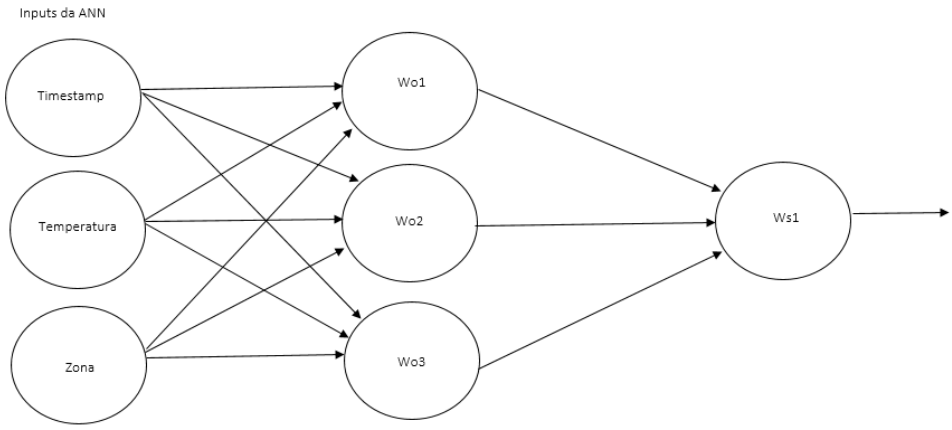

Figura 20 – Configuração da ANN do caso de estudo 1

Da fase de treino da rede neuronal artificial, realizado fora do microprocessador, resultam duas matrizes que serão utilizadas para os cálculos da fase de teste. Essas matrizes representam os valores dos pesos da camada oculta (camada intermédia da rede neuronal artificial) bem como os valores dos pesos da camada de saída.

Estes valores são manualmente introduzidos no microprocessador responsável pela previsão da temperatura. Estes valores são os utilizados para o cálculo matricial, sendo que os valores de input no microprocessador são os três que estão mencionados na figura 20. Importa ainda referir que é importante que estes pesos sejam atualizados com alguma frequência através de um novo treino da rede neuronal artificial, para manter a rede atualizada e ser possível obter corretas classificações.

```
================ Valor dos pesos de input para a camada hidden ================

[[1.84028562e+07 3.95913785e+07 2.22787666e+07]
 [9.24578638e-01 7.02094095e-01 1.23132978e+00]
 [6.16371025e+03 3.34723093e+03 8.01666204e+03]]

================ Valor dos pesos de output ================

[[0.29453703]
 [0.06683862]
 [0.38862435]]
```

Figura 21 – Resultado do treino da ANN do caso de estudo 1

4.1.2 Dataset utilizado no treino da rede neuronal artificial – caso estudo 1

O *dataset* utilizado para o treino da rede neuronal artificial (Duarte Roa, et al., 2020) é muito completo e essencial e que vai de encontro ao pretendido, dado que contém informação do tipo de aceitação que existe em relação a uma determinada temperatura. É uma informação realmente útil e necessária, considerando que é essa previsão que a rede neuronal artificial

deve ser capaz de prever na fase de teste, quando recebe os dados provenientes do sensor de temperatura. No caso concreto da aplicação do *dataset*, o mesmo não foi aplicado na totalidade porque algumas das colunas não eram relevantes para o caso de estudo, ficando assim reduzido às colunas do *timestamp* de leitura, a temperatura propriamente dita, a zona onde o sensor se encontra localizado e a coluna onde é determinada se a temperatura deve ser aumentada, se deve manter como está ou se deve ser diminuída. As colunas que foram removidas tinham informação que não era necessária de ser utilizada, tal como o identificador único do sensor que havia sido utilizado, o tempo de início da leitura realizada, o tempo final da leitura realizada, e algumas perguntas que foram respondidas pelos utilizadores e que não justificaram serem relevantes de utilizar.

4.1.3 Valores de input para o teste da rede neuronal

Para obter o valor da temperatura no local é utilizado um sensor de temperatura DHT22, que se liga ao microprocessador ESP32 para que possa ser alimentado. Esse sensor está no local onde se pretende realizar a medição e obtêm o valor da temperatura naquele local. O valor obtido da temperatura sentida é utilizado como input para a rede neuronal artificial deste dispositivo IoT, mas não é o único. A esse valor juntam-se ainda outros dois valores: o timestamp, valor obtido de forma automática pelo microprocessador e que representa o instante em que a leitura da temperatura ocorre e a zona do sensor, um valor numérico e que representa o local onde o sensor de temperatura está colocado. Este valor difere de local para local e server para distinguir os diversos locais onde sensores deste tipo se encontram posicionados. Trata-se de um valor fixo para cada sensor, isto é, um sensor colocado num quarto terá um valor identificativo de 8903, por exemplo, enquanto que um sensor colocado na sala de jantar poderá ter um valor identificativo de 9304.

```
Sensor zone = 8903
Timestamp = 1654426846
Temperature: 24.70
```

Figura 22 – Leitura efetuada pelo DHT22

4.1.4 Fluxo de ação

O agente PEAK responsável pelo dispositivo IoT que lida com as questões da temperatura, envia uma mensagem a esse dispositivo IoT para que ele execute a sua tarefa, isto é, obtenha o valor da temperatura no local, bem como o *timestamp* da leitura, para que esse valor seja utilizado na previsão a enviar para o HVAC. Essa mensagem pode ser vista na figura 23.

```
joaof@LAPTOP-SRUMRAEJ MINGW64 ~/peak (abc)
$ peak project/temperatureHouseAgent.py temperatureHouseAgent@mas.gecad.isep.ipp.pt

House190 temperature PEAK agent sending a message...
```

Figura 23 – Mensagem enviada pelo agente PEAK para dispositivo IoT de temperatura

Essa mensagem enviada pelo agente PEAK é recebida pela gateway de comunicação XMPP para MQTT uma vez que se trata de uma mensagem enviada por um agente PEAK e que tem como destino um dispositivo IoT. Essa mensagem é processada pela gateway e publicada para o tópico esperado.

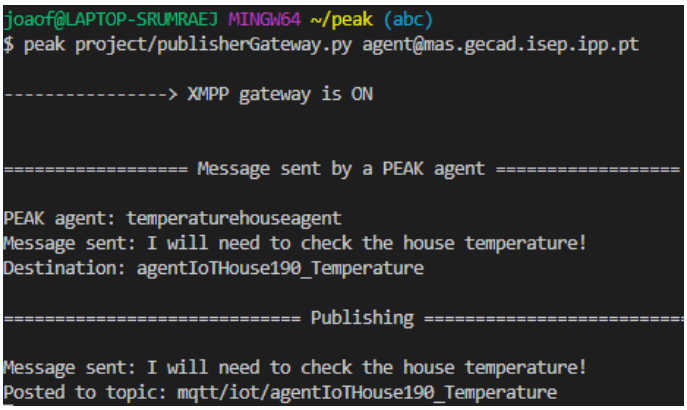

Figura 24 – Publicação da mensagem para o tópico do dispositivo IoT

O dispositivo IoT recebe a mensagem publicada tópico que o próprio escuta, efetua a recolha de dados (temperatura no local e timestamp) e com esses valores recolhidos e em adição ao valor que corresponde ao local onde o sensor se encontra, a rede neuronal artificial é alimentada. Desse processo resulta um valor de previsão, que é posteriormente enviado pelo dispositivo IoT para o agente responsável pelo HVAC. Esse valor tem como finalidade perceber que tipo de ação o HVAC terá que realizar: aumentar a temperatura no local, diminuir a temperatura ou manter a mesma temperatura, ou seja, não atuar.

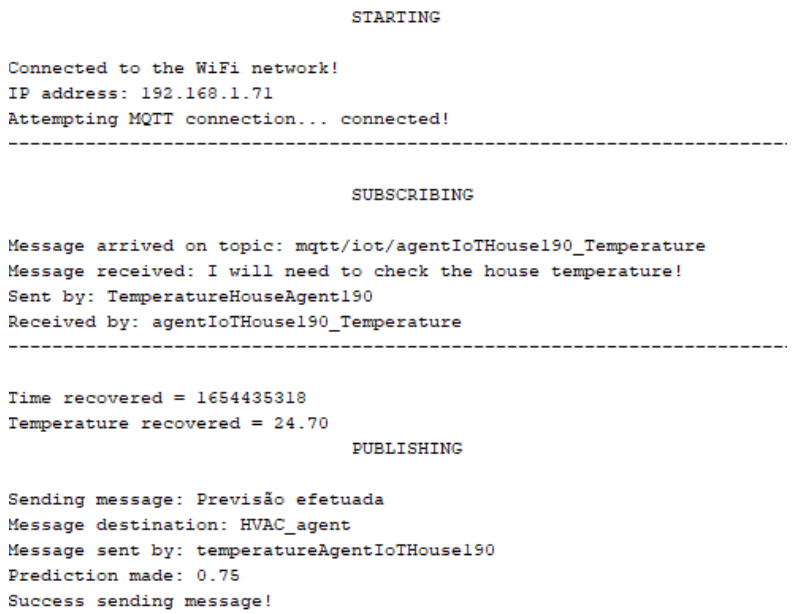

Figura 25 - Funcionamento total do dispositivo IoT para temperatura

O resultado da ação do dispositivo IoT é enviado para a gateway de comunicação de MQTT para XMPP para que seja enviada para o agente PEAK de destino, que neste caso é o responsável pelo HVAC. Quando a mensagem é recebida, o HVAC saberá o tipo de ação que tem que realizar, sendo que no caso da demonstração aqui expressa, se conclui que é necessário que o local seja aquecido.

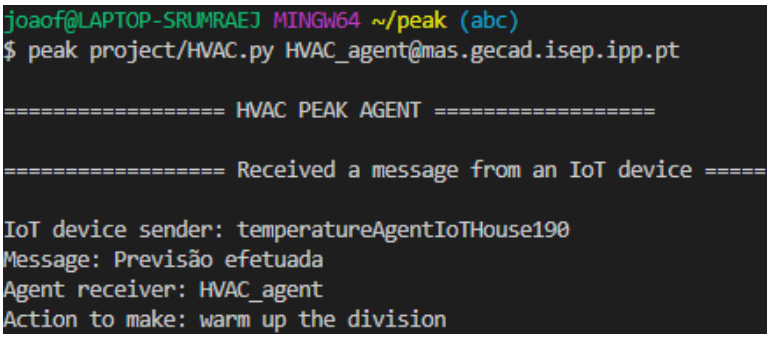

Figura 26 – Ação a realizar pelo HVAC

4.1.5 Análise dos erros de classificação da ANN

De forma a obter a classificação mais precisa possível da rede neuronal artificial que se encontra no dispositivo IoT de temperatura, foram realizados testes com dados de input estáticos, dados conhecidos e que se esperava que tivessem um determinado output pela rede neuronal artificial. Este procedimento foi necessário pelo facto de a rede neuronal artificial fornecer o valor da classificação em formato numérico, ou seja, através de um valor numérico que iria necessitar de ser interpretado. Os testes realizados utilizaram valores do *dataset* usado para treinar a rede neuronal artificial. Para estes acertos, invés de obter o valor da temperatura sentida em determinada divisão através da utilização do sensor de temperatura, esse valor foi introduzido manualmente como valor de teste para a rede. Com isto, e considerando a classificação obtida por esse valor no *dataset* utilizado, era esperada uma determinada classificação. Supondo que esse valor deveria indicar que a temperatura da divisão deveria ser reduzida, ou seja, que o HVAC deveria ser ativado para arrefecer a divisão, caso a classificação da rede fosse de que a temperatura era ideal para a divisão ou que se deveria aquecer a divisão, então era possível concluir que a interpretação do valor final de output da rede neuronal ainda não se encontrava correta. Os testes prosseguiram nessa lógica de interpretação e acerto até se obter classificações consideradas satisfatórios, ou seja, introduzir valores presentes no *dataset* e que se sabe qual a classificação que a rede deve ter, e obter essa mesma classificação assim que o *output* é interpretado. Depois da classificação afinada com dados estáticos, ou seja, com dados idênticos aos presentes no *dataset,* deu-se início à experimentação com dados reais, lidos através da utilização do sensor de temperatura, onde se verificou o acerto da rede, em função dos valores de temperatura que eram obtidos pelo sensor e introduzidos na rede neuronal, e a classificação final que era obtida com recurso a esses valores.

A precisão da ANN foi obtida em duas fases distintas, de forma a perceber se os resultados que estavam a ser obtidos eram realmente fruto de uma boa classificação, ou se existiam valores

que poderiam não estar em concordância com o esperado. Observando a tabela 2, podemos consultar os valores de precisão obtidos na fase de treino da ANN e na fase de teste da ANN.

Tabela 2 – Precisão ANN de temperatura

	Precisão
Fase de treino da ANN	0.92
Fase de teste da ANN	0.90

4.2 Caso de estudo 2: luminosidade

No segundo caso de estudo apresentado pretende-se perceber se a luminosidade de um determinado local se encontra nos níveis esperados, em função da atividade que se está a realizar. Segundo uma escala internacional, existem intensidades luminosas apropriadas para as diferentes atividades que estejam em curso, concluindo que a intensidade luminosa para se estar a cozinhar na cozinha é diferente da esperada quando se está no quarto a estudar ou a ler um livro. O que se pretende com esta solução é perceber se a intensidade luminosa de um determinado local está em concordância com os valores pretendidos, em função da atividade que está a ser realizada. Os elementos que constituem este caso de estudo e que intervêm no mesmo podem ser observados na imagem que se segue.

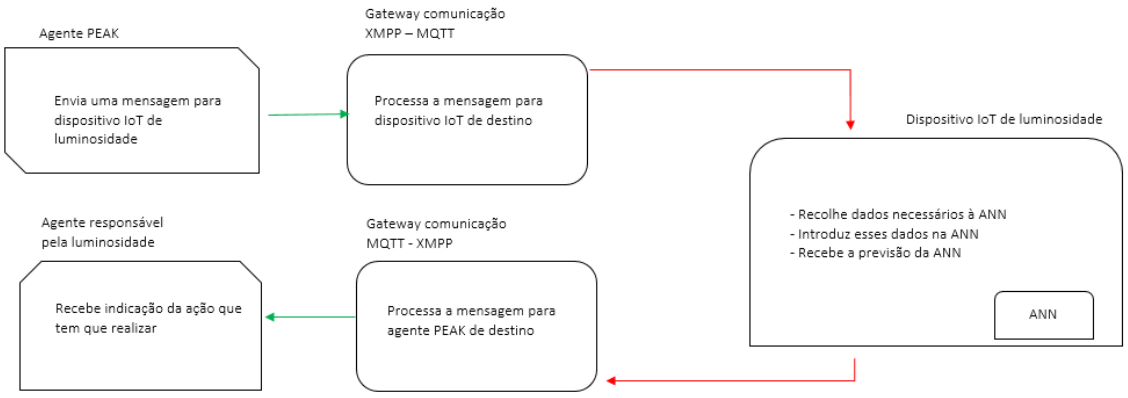

Figura 27 – Componentes e interação no caso de estudo 2

Na figura 27 estão representados os elementos desta interação, fazendo parte dela o agente PEAK atribuído ao dispositivo IoT de temperatura, ambas as *gateways* de comunicação, uma para realizar a comunicação de XMPP para MQTT e a outra para realizar a comunicação de MQTT para XMPP. Está também representado o dispositivo IoT de luminosidade, com indicação para a presença da ANN dentro de si, e por fim o agente responsável pelas luzes. Para cada elemento existe uma breve descrição daquilo que cada um faz, apenas para ser percetível de que forma cada elemento interage com o sistema.

4.2.1 Rede neuronal artificial

Novamente partimos da configuração padrão de rede neuronal artificial expressa na figura 14 e apenas é necessário alterar os inputs que a rede neuronal artificial irá receber, que neste caso particular são quatro: *timestamp* (momento em que a luminosidade é obtida), a zona onde o sensor que recolhe os dados se encontra, o tipo de atividade que está a ser realizada e o valor propriamente da luminosidade naquele local. São estes quatro dados de entrada que são fornecidos à rede neuronal artificial para que a mesma realize o seu treino, com a finalidade de obter os pesos da camada oculta e os pesos da camada de saída, para serem utilizados no teste da rede dentro do microprocessador.

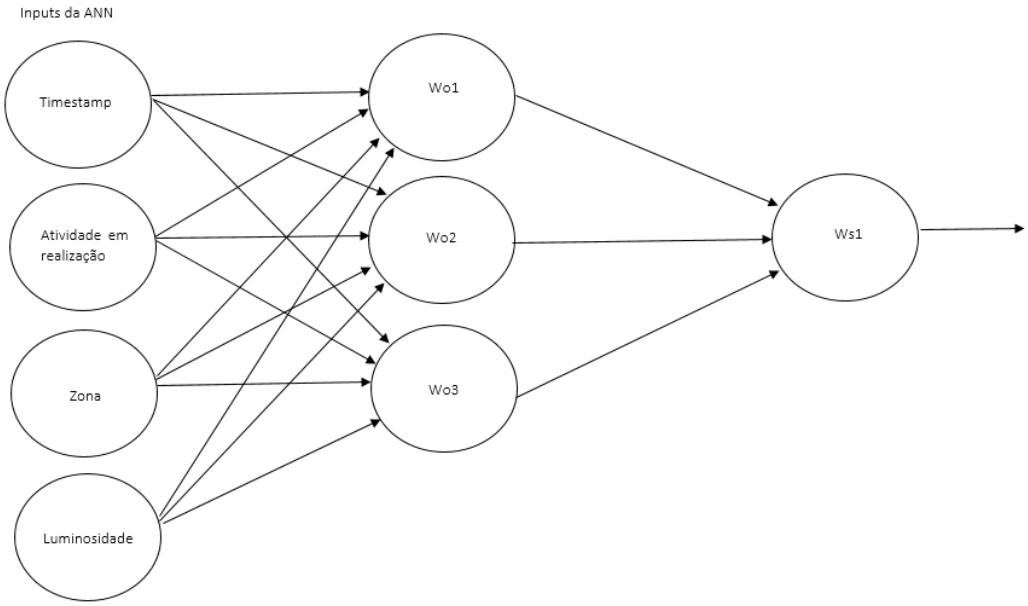

Figura 28 – Configuração da ANN do caso de estudo 2

Da fase de treino da rede neuronal artificial, realizado fora do microprocessador, resultam duas matrizes que serão utilizadas para os cálculos da fase de teste. Essas matrizes representam os valores dos pesos da camada oculta (camada intermédia da rede neuronal artificial) bem como os valores dos pesos da camada de saída.

```
================ Valor dos pesos de input para a camada hidden ================

[[-6.82574467e+06  4.07908652e+07  4.77181319e+07]
 [-1.71683459e+00 -1.76754923e-01  9.18466021e-01]
 [ 2.55840152e+03  3.15066715e+03  2.76338630e+03]
 [-2.15148317e+00  9.97254836e+00  1.17385980e+01]]

================ Valor dos pesos de output ================

[[0.4585766 ]
 [0.20020868]
```

Figura 29 – Resultado do treino da ANN do caso de estudo 2

Estes valores são manualmente introduzidos no microprocessador responsável pela previsão da temperatura. Estes valores são os utilizados para o cálculo matricial, sendo que os valores de input no microprocessador são os quatro que estão mencionados na figura 28.

4.2.2 Dataset utilizado no treino da rede neuronal artificial – caso estudo 2

Devido à falta de um *dataset* que fosse capaz de preencher os requisitos necessários, existiu a necessidade de desenvolver um dataset personalizado que fosse capaz de colmatar essa falta. Contrariamente a dados como a temperatura, onde a opinião do utilizador sobre como o HVAC deve agir, pode variar de indivíduo para indivíduo, a luminosidade pode ser ajustada com base em valores internacionais que são padronizados e que indicam o nível de luminosidade em função do tipo de atividade que está a ser realizada. Com esta informação, foi desenvolvido um *dataset* onde são considerados os seguintes dados: *timestamp*, luminosidade, zona do sensor, e o tipo de atividade que está a decorrer.

4.2.3 Recolha dos valores de input

Para a recolha do valor de luminosidade no local foi utilizado um sensor LDR que se encontra ligado ao ESP32 para que possa ser alimentado. Novamente e tal como no caso de estudo 1, o valor do instante de tempo em que a recolha do valor da luminosidade é obtido é recolhido de forma automática pelo microprocessador, no momento da recolha do restante valor. A zona onde o sensor se encontra mantêm-se como um valor fixo e que apenas é utilizado para que seja possível saber de que local se trata. O tipo de atividade que se está a realizar não é algo que seja possível de obter por parte do sensor, mas sim um dado que é enviado pelo agente PEAK quando envia a mensagem para o dispositivo IoT. O agente indica o tipo de atividade que está a ser realizada naquele momento, sendo esse dado recuperado pelo dispositivo IoT quando recebe a mensagem, para que possa ser utilizado como input. Antes dessa utilização acontecer, essa informação é convertida em valor numérico, isto é, diferentes atividades possuem valores distintos, como por exemplo: cozinhar teria um valor de 011, mas ler um livro teria um valor de 015. Assim é possível utilizar essa informação como input da rede neuronal.

4.2.4 Fluxo de ação

O agente PEAK responsável pelo dispositivo IoT de luminosidade envia uma mensagem para esse dispositivo IoT, onde indica a atividade que está em curso. Essa mensagem é recebida pela *gateway* de comunicação de XMPP para MQTT e publicada para o tópico correspondente ao dispositivo IoT de destino.

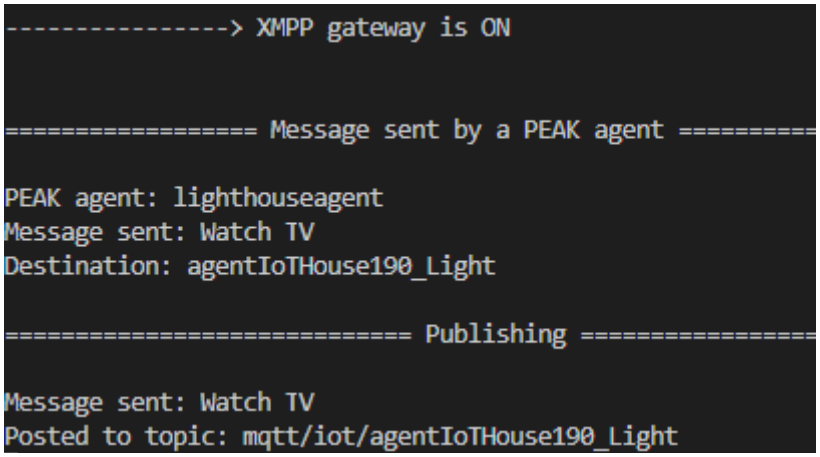

Figura 30 - Publicação da mensagem para dispositivo IoT de luminosidade

Assim que o dispositivo IoT recebe essa mensagem, retira dela o elemento que é necessário para o input na rede neuronal artificial: a atividade. Essa atividade é utilizada para a obtenção de um código, que é o valor a utilizar como input. Para além disto, utilizando o sensor LDR, irá obter a intensidade luminosa presente no local, e obtém ainda o *timestamp* relativo ao momento em que obteve essa leitura. A zona onde o sensor se encontra é um valor fixo, pelo que está previamente atribuído. Na presença destes quatro valores, os dados são enviados para a rede neuronal artificial para obter uma previsão. Assim que esse resultado é obtido, o dispositivo IoT realiza a publicação dos resultados para o tópico da *gateway* de comunicação de MQTT para XMPP, para que a mensagem seja enviada ao agente responsável pelas luzes do local em questão.

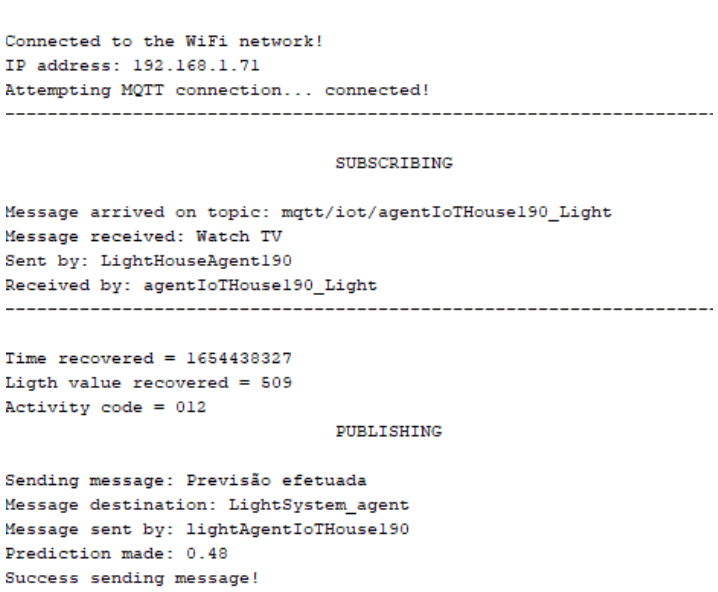

Figura 31 - Funcionamento total do dispositivo IoT para luminosidade

Por fim a mensagem chega ao agente PEAK de destino, que processa o tipo de ação que é necessário realizar em função das informações que recebeu.

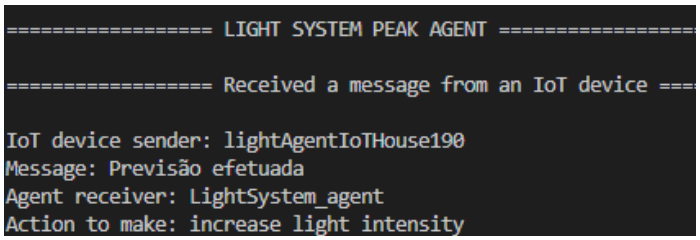

Figura 32 - Ação a realizar sobre a luminosidade

4.2.5 Análise dos erros de classificação da ANN

Da mesma forma que a rede neuronal artificial do dispositivo IoT de temperatura teve de ser devidamente ensaiada, também a rede neuronal artificial do dispositivo IoT de luminosidade demanda que o processo de lapidação da classificação obtida seja realizado. Os testes para obter a melhor classificação possível da rede neuronal artificial foram realizados através da utilização do *dataset* com o qual a rede neuronal artificial realiza o seu treino. Considerando os valores presentes no *dataset* e a classificação esperada de cada um deles, era necessário que na fase de teste a rede neuronal fosse capaz de obter a mesma classificação para o mesmo valor de input. Para estes acertos, invés de do valor de teste ser diretamente obtido do sensor de luminosidade, ele foi introduzido manualmente na rede, sabendo-se *à priori* qual a classificação esperada para esse valor de entrada. Os testes prosseguiram nessa linha de raciocínio, ou seja, sempre a tentar calibrar a rede para que os valores de teste que provinham da rede neuronal artificial fossem classificados na prática, como o eram na rede neuronal. Dessa forma estava-se mais próximo de garantir que a rede neuronal é capaz de classificar conforme o esperado. Depois de realizados os testes com os valores do *dataset*, da classificação ter sido afinada com esses dados, deu-se início à experimentação com dados reais, lidos através do sensor de luminosidade, onde se verificou o acerto da rede, em relação aos valores de leitura que haviam sido obtidos e a classificação que a rede apresenta.

Da mesma forma que a ANN da temperatura teve a sua precisão calculada, também com a ANN da luminosidade essa precisão foi obtida, com o mesmo objetivo específico, de perceber de que forma a classificação era realizada. A tabela 3 apresenta os resultados da previsão da ANN de luminosidade para a fase de treino e para a fase de teste.

Tabela 3 – Precisão ANN de luminosidade

	Precisão
Fase de treino	0.92
Fase de teste	0.93

4.3 Caso de estudo 3: deteção de presença

No terceiro caso de estudo apresentado pretende-se detetar a presença humana ou não no local onde o sensor de presença está localizado. Para que este caso de estudo seja ainda mais preciso, é necessário conhecer um dado concreto: se as luzes estão ligadas ou não. Essa informação provém do agente PEAK que envia a mensagem para o dispositivo IoT de presença, onde na mensagem é referido qual o estado das luzes no momento. O dispositivo IoT recebe esse dado e armazena-o, uma vez que lhe será útil e necessário. Caso seja detetada presença, não é de imediato que o dispositivo IoT comunica ao agente PEAK e necessidade de ligar as luzes, considerando que as luzes podem já estar ligadas. Quer isto dizer que a resposta que este dispositivo IoT envia para o seu agente PEAK é sempre em função da informação que recebeu do mesmo. Caso exista confirmação de presença e o agente PEAK tenha informado que as luzes se encontravam desligadas, então o dispositivo IoT enviará a informação de que é necessário ligar as luzes. Caso seja detetada presença, mas as luzes já estejam ligadas, então o dispositivo IoT apenas indicará que não é necessário realizar qualquer tipo de ação. Por outro lado, caso o dispositivo IoT não detete nenhuma presença, mas exista a informação de que as luzes se encontram ligadas, então o dispositivo IoT indica ao seu agente PEAK que as luzes podem ser desligadas, uma vez que não existe presença no local e não existe necessidade de as luzes permanecerem acesas.

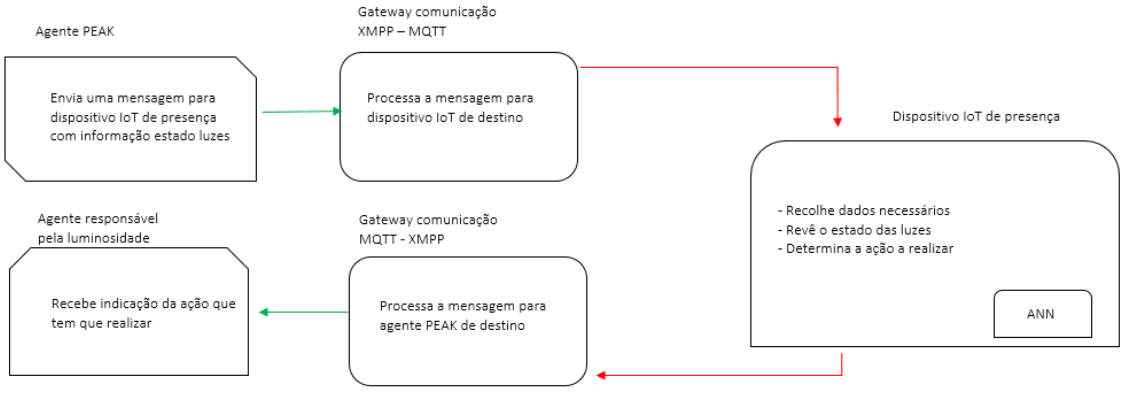

Figura 33 - Componentes e interação no caso de estudo 3

Na figura 33 estão representados os elementos desta interação, fazendo parte dela o agente PEAK atribuído ao dispositivo IoT de temperatura, ambas as *gateways* de comunicação, uma para realizar a comunicação de XMPP para MQTT e a outra para realizar a comunicação de MQTT para XMPP. Está também representado o dispositivo IoT de presença, com indicação para a presença da ANN dentro de si, e por fim o agente responsável pelo sistema de luzes. Para cada elemento existe uma breve descrição daquilo que cada um faz, apenas para ser percetível de que forma cada elemento interage com o sistema.

4.3.1 Rede neuronal artificial

A base da configuração padrão da rede neuronal artificial está expressa na figura 14. Para este caso de estudo, é necessário alterar os inputs que a rede neuronal irá receber, que neste caso particular são quatro: o *timestamp* (momento em que a leitura é realizada), e um intervalo de valores que são referentes aos três instantes anteriores de verificação, ou seja, para esta rede neuronal artificial, que será de previsão, receberá o *timestamp* atual e os teus valores das últimas três verificações e com base nesses dados, é realizada a previsão para o estado da presença no momento seguinte. São estes quatro dados de entrada que são fornecidos à rede neuronal artificial para que a mesma realize o seu treino, com a finalidade de obter os pesos da camada oculta e os pesos da camada de saída, para serem utilizados no teste da rede dentro do microprocessador.

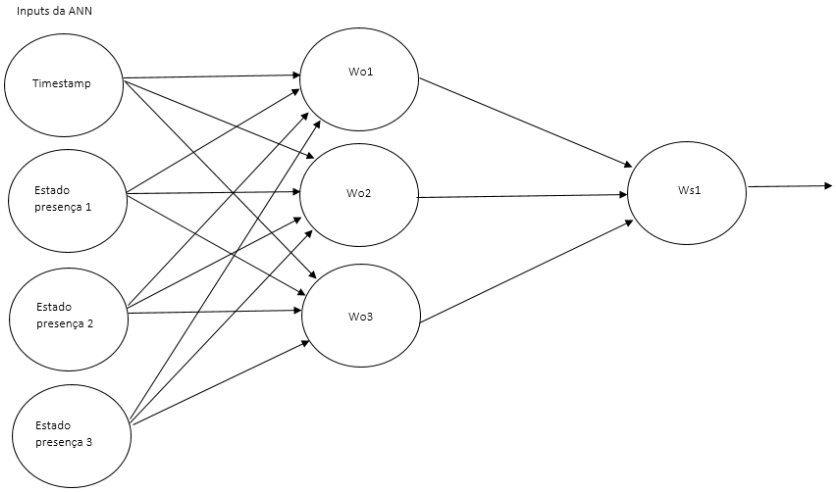

Figura 34 - Configuração da ANN do caso de estudo 3

Da fase de treino da rede neuronal artificial, realizado fora do microprocessador, resultam duas matrizes que serão utilizadas para os cálculos da fase de teste. Essas matrizes representam os valores dos pesos da camada oculta (camada intermédia da rede neuronal artificial) bem como os valores dos pesos da camada de saída.

```
================ Valor dos pesos de input para a camada hidden ==============

[[ 2.62322343e+07  2.68017602e+07 -2.26482650e+07]
 [ 2.58081142e-01  2.51953293e-01  4.90960501e-01]
 [ 5.84836253e-01  8.02786607e-01  1.15939405e+00]
 [ 1.82488612e+00  1.48379583e+00  1.26853151e+00]]

================ Valor dos pesos de output ================

[[0.25162257]
 [0.24837743]
 [0.15021581]]
```

Figura 35 - Resultado do treino da ANN do caso de estudo 3

4.3.2 *Dataset utilizado no treino da rede neuronal artificial – caso estudo 3*

Para colmatar a falta de um *dataset* que fosse constituído com os dados necessários a serem utilizados no treino da rede neuronal artificial, foi construído um, personalizado, e que suprime as necessidades existentes. Esta tarefa foi facilitada pelo facto de não estarem presentes dados intelectuais, isto é, opiniões ou escolhas do utilizador, considerando que um sensor de presença indica a existência ou não de presença, pelo que falámos de valores binários. A construção do *dataset* existe com a inclusão de cinco valores distintos: o primeiro é referente ao tempo atual da verificação, ou seja, o instante em que é verificada a existência de presença no local. Os três valores seguintes são as três últimas verificações, que ocorreram antes da verificação atual, com a informação de presença ou ausência em cada um desses instantes. E por último, o valor da verificação atual. Este último valor será o valor previsto pela rede neuronal artificial que se encontra dentro do microprocessador.

4.3.3 *Recolha dos valores de input*

Para a verificação de presença foi utilizado um sensor denominado PIR, ligado ao ESP32. Como referido na secção 4.3.1, são utilizados quatro valores de input. O primeiro que é referente ao *timestamp* é recolhido de forma automática pelo microprocessador. Os seguintes valores são valores de medições anteriores, no caso das últimas três medições que foram efetuadas. Com os quatro dados reunidos, os inputs a fornecer à rede neuronal artificial encontram-se prontos.

4.3.4 *Fluxo de ação*

O agente PEAK responsável pelo dispositivo IoT de presença envia uma mensagem para esse dispositivo IoT, onde indica o estado da luz, isto é, se a luz se encontra desligada ou ligada. Essa mensagem é recebida pela gateway de comunicação de XMPP para MQTT e publicada para o tópico correspondente ao dispositivo IoT de destino.

```
----------------> XMPP gateway is ON


================= Message sent by a PEAK agent =================

PEAK agent: presencehouseagent
Message sent: Light on
Destination: agentIoTHouse190_Presence

============================ Publishing ============================

Message sent: Light on
Posted to topic: mqtt/iot/agentIoTHouse190_Presence
```

Figura 36 - Publicação da mensagem para dispositivo IoT de presença

Assim que o dispositivo IoT recebe essa mensagem, existe um dado que é necessário recolher e que chega na mensagem que é enviada: a informação relativa ao estado das luzes, uma vez que o estado das luzes é um dos elementos fundamentais para o desenvolvimento da atividade do dispositivo IoT. Com essa informação, o dispositivo IoT vai verificar qual o estado de presença no local, ou seja, se essa presença é confirmada ou, se pelo contrário, não é confirmada. Assim que os dados são analisados, o dispositivo IoT determina qual a ação que é necessário que seja realizada publica essa informação para o tópico da *gateway* de comunicação de MQTT para XMPP, para que a mensagem seja enviada ao agente responsável pelas luzes do local em questão.

```
                                SUBSCRIBING

Message arrived on topic: mqtt/iot/agentIoTHouse190_Presence
Message received: Light on
Sent by: PresenceHouseAgent190
Received by: agentIoTHouse190_Presence
-------------------------------------------------------------------------

                                PUBLISHING

Sending message: Unnoticed presence
Message destination: PresenceLightSystem_agent
Message sent by: presenceAgentIoTHouse190
Prediction made: Turn off the light
Success sending message!
```

Figura 37 – Ação realizada pelo dispositivo IoT de presença

Por fim a mensagem chega ao agente PEAK de destino que recebe o tipo de ação que deve realizar, em função da informação que o dispositivo IoT lhe enviou. Neste caso particular, o agente PEAK havia informado o dispositivo IoT de que a luz se encontrava ligada, e o sensor de presença não detetou a presença humana no local, o que concluí que as luzes devem ser desligadas.

```
================== LIGHT PRESENCE SYSTEM PEAK AGENT ==================

================== Received a message from an IoT device ==============

IoT device sender: presenceAgentIoTHouse190
Agent receiver: PresenceLightSystem_agent
Message: Unnoticed presence
Action to make: Turn off the light
```

Figura 38 – Ação a realizar sobre as luzes

4.3.5 Análise dos erros de classificação da ANN

Contrariamente às redes neuronais dos dispositivos IoT de temperatura e de luminosidade, a rede neuronal do dispositivo IoT de presença, é de previsão e apenas prevê um de dois valores, nomeadamente se existe ou não presença na divisão onde o sensor se encontra. No entanto, foi ainda assim necessário realizar o ajuste desta previsão para que a mesma fosse o mais precisa possível. Os testes para obter a melhor previsão foram realizados com recurso ao *dataset* utilizado para o treino da rede neuronal para ser possível compreender os desvios das previsões reais, em relação às previsões esperadas, presentes no *dataset*. Após estes testes realizados, e os ajustes realizados, deu-se início à fase de teste com dados reais, ou seja, com os dados lidos do sensor de presença. Verificou-se também nesse capítulo a previsão da rede neuronal aplicada ao dispositivo IoT de presença.A tabela 4 apresenta os resultados da precisão da ANN de presença para a fase de treino e para a fase de teste.

Tabela 4 – Precisão ANN de presença

	Precisão
Fase de treino	0.89
Fase de teste	0.89

4.4 Caso de estudo 4: agente responsável pela sala

No quarto caso de estudo apresentado é introduzido um novo agente do sistema. Trata-se do agente responsável por uma sala, ou seja, um agente que controla um local onde todos os sensores estão incluídos. Considerando que numa sala existem os três sensores introduzidos nos três casos de estudo anteriores (sensores de temperatura, luminosidade e presença). Os agentes virtuais PEAK, responsáveis por cada dispositivo IoT, enviam uma mensagem para esses dispositivos para que os mesmos realizam as suas ações, tal como até aqui acontecia. Cada um desses dispositivos IoT vai realizar as suas ações, obter as suas previsões ou classificações, e enviar as mesmas para o agente PEAK responsável pela sala. Esse agente PEAK recebe e verifica os dados enviados pelos dispositivos IoT, e tem particular atenção à previsão enviada pelo dispositivo IoT de presença. Caso seja enviada a previsão de que não existirá presença na sala no período seguinte, então o agente PEAK informará os dispositivos IoT do HVAC e das luzes para que estes elementos saiam de funcionamento, uma vez que não existe presença, logo não existe razão para os mesmos estarem em funcionamento. Se pelo contrário existir presença, então o agente PEAK responsável pela sala comunica aos dispositivos IoT do HVAC e das luzes o que os mesmos devem fazer, e essa decisão é baseada na informação que recebem dos dispositivos IoT dos sensores de temperatura e de luminosidade.

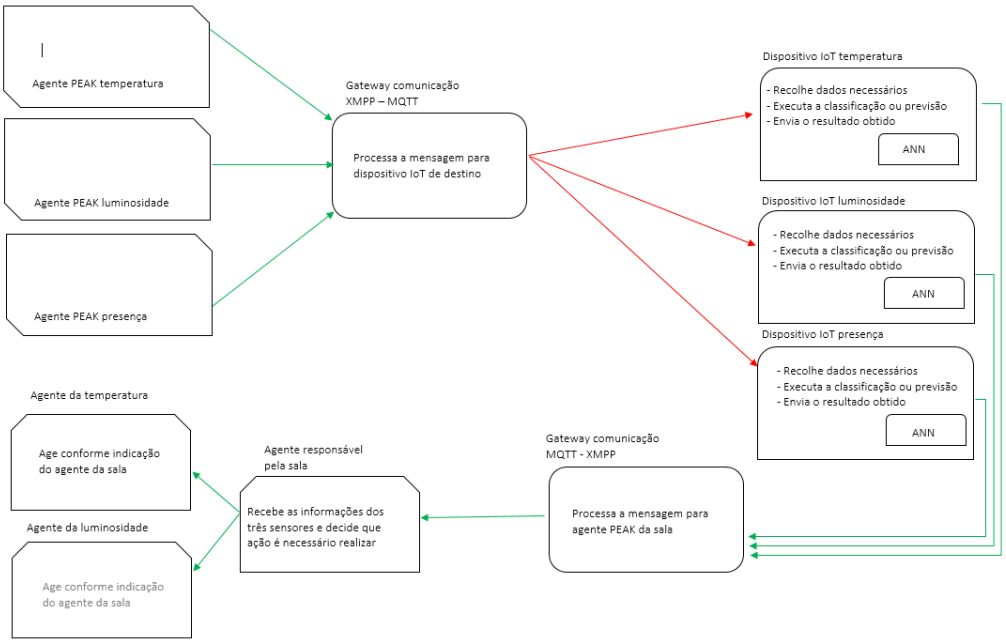

Figura 39 - Componentes e interação no caso de estudo 4

Considerando os elementos do caso de estudo em questão, é possível perceber que as redes neuronais artificias presentes são as já descritas e pertencentes a cada um dos dispositivos IoT em questão. Observando a figura 39 podemos compreender melhor o funcionamento do caso de estudo 4. Os agentes PEAK responsáveis pelas três áreas (temperatura, luminosidade e presença) enviam cada um uma mensagem ao seu dispositivo IoT. Essa mensagem passa pela gateway, chega aos dispositivos, que utilizando as suas ANN enviam o feedback ao agente da sala que indica que ação dos agentes de temperatura e luminosidade devem fazer.

Cada um dos agentes PEAK responsáveis por cada um dos três dispositivos IoT de obtenção de classificação (temperatura e luminosidade) ou previsão (presença) envia uma mensagem para ativar esses dispositivos, isto é, para que os mesmos realizem as suas tarefas e obtenham a informação que é esperada. Essas mensagens são recebidas pela gateway de comunicação de XMPP para MQTT e publicadas para os tópicos correspondentes a cada dispositivo IoT destino. As mensagens são recebidas pelos dispositivos IoT, os dados necessários para a utilização das redes neuronais artificias são recolhidos por cada um desses dispositivos. Assim que o resultado da rede neuronal artificial é obtido em cada dispositivo IoT, cada um deles envia a informação através da publicação para o tópico da gateway de comunicação de MQTT para XMPP, para que a mensagem seja enviada para o agente PEAK responsável pela sala.

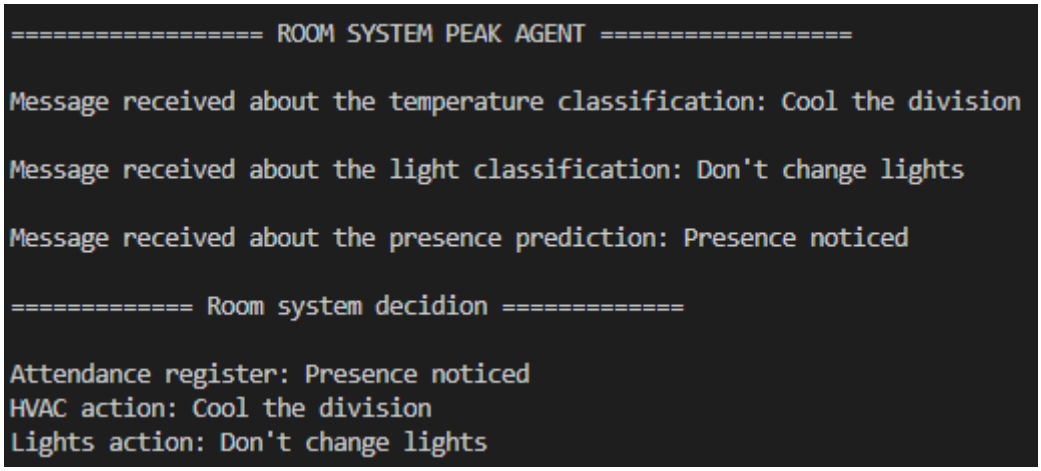

Figura 40 – Dados recebidos pelo agente PEAK responsável pela sala

Conforme é possível observar na figura 40, o agente PEAK responsável pela sala vai recebendo os dados relativos a cada um dos sensores. Neste caso, como a previsão da rede neuronal do sensor de presença indica a presença de alguém na sala, então o agente PEAK considera as classificações obtidas pelas redes neuronais dos sensores de temperatura e luminosidade, e usa essas informações para indicar a ação que tanto o HVAC como o sistema de luzes devem realizar. De realçar que, independentemente da classificação das redes neuronais de temperatura e de luminosidade, se a previsão de presença indiciasse a não existência de presença, a ordem seria para que ambos os sistemas (HVAC e luzes) fossem desligados, uma vez que ninguém estaria no local, e os mesmos não estavam a ser corretamente utilizados.

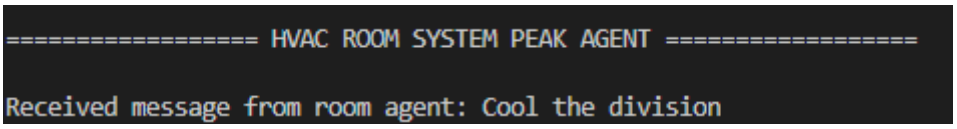

Figura 41 – Ação a realizar recebida pelo agente PEAK da temperatura da sala

Na figura 41 é possível ver a mensagem que o agente PEAK do HVAC da sala recebe vindo do agente PEAK da sala, e que corresponde à ação que o agente PEAK da sala concluí que deve ser feita pelo sistema. Já na figura 42 é possível ver a mensagem que o agente PEAK do sistema de luzes da sala recebe vindo do agente PEAK da sala, que mais uma vez corresponde à ação que o agente PEAK da sala concluí que deve ser feita pelo sistema.

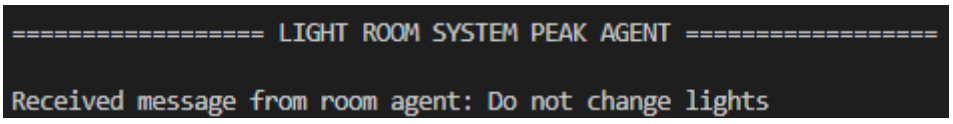

Figura 42 – Ação a realizar recebida pelo agente PEAK do sistema de luzes da sala

Assim, o agente da sala concluí o processo, enviando aos agentes PEAK da temperatura e da luminosidade as ações que se espera que ambos realizam em função da informação que o agente PEAK da sala recebeu, e que em primeira instância é influenciada pela existência, ou não, de presença. Neste caso a presença existe, portanto, a temperatura e luminosidade serão ajustadas em função dos resultados obtidos pelas ANN.

4.5 Caso de estudo 5 – agente responsável pelo edifício

No quinto caso apresentado é introduzido uma vez mais um novo agente do sistema: o agente do edifício. Trata-se do agente responsável pelo edifício e que controla os níveis de energia que o edifício possui, percebendo qual a gestão que deve ser feita. Para que esse controlo seja o mais acertado possível, este agente recebe informação de todos os agentes das salas, ou seja, ele tem conhecimento das informações acerca de cada uma das salas existente. No caso de estudo apresentado na secção 4.4, foi introduzido o agente PEAK responsável pela sala, que após receber informação dos sensores, decide o tipo de ações a realizar. Após essa decisão, esse agente comunica para o agente do edifício com a informação da variação da energia naquela sala, ou seja, se o gasto de energia vai aumentar, se por outro lado será um gasto menor ou se o gasto não será alterado em relação ao normal esperado. O agente edifício recebe esses dados de todas as salas, analisa a informação que recebe e indica as conclusões que obtêm.

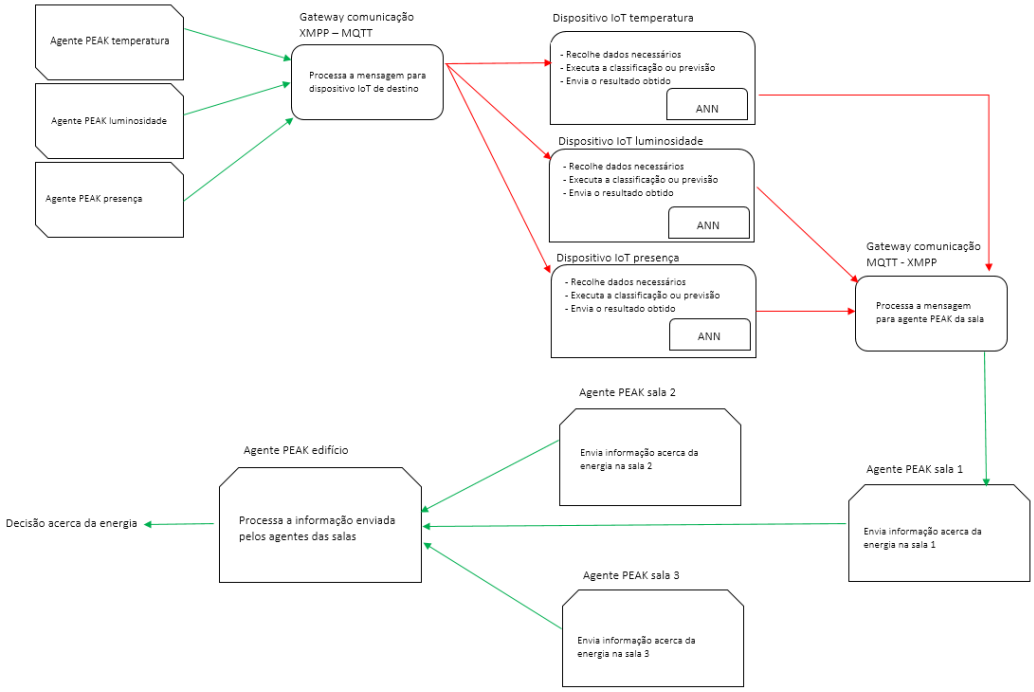

Figura 43 – Componentes e interações no caso de estudo 5

O fluxo de ação do caso de estudo 5 começa, novamente, com a comunicação dos agentes PEAK para os dispositivos IoT relacionados a si, neste caso dispositivo IoT de temperatura, dispositivo IoT de luminosidade e dispositivos IoT de presença. Cada um destes dispositivos, depois de receber indicação do seu agente PEAK, realiza as suas funções, utiliza as ANN que estão em si construídos e obtém as previsões de que o sistema necessita.

```
                                SUBSCRIBING

Message arrived on topic: mqtt/iot/agentIoTHouse190_Presence
Message received: Light on
Sent by: PresenceHouseAgent190
Received by: agentIoTHouse190_Presence
--------------------------------------------------------------------------------


                                PUBLISHING

Sending message: Presença revista
Message destination: RoomSystem_agent
Message sent by: presenceAgentIoTHouse190
Prediction made: Presence noticed
Success sending message!
```

Figura 44 – Comunicações do dispositivo IoT de presença

Como podemos ver pela figura 43, todos os três dispositivos IoT processam as mensagens que os agentes PEAK enviam, utilizam a sua ANN e envia as previsões ou classificações após os testes realizados pela ANN.

Sempre que um dos dispositivos IoT que estão presentes na sala, respondem ao pedido pelo dispositivo PEAK responsável pelo pedido que foi efetuado, essa resposta chega ao agente PEAK responsável pela sala, que fica nesse momento com total conhecimento das previsões e das classificações que foram realizadas pelos dispositivos IoT.

```
================= ROOM SYSTEM PEAK AGENT ==================

Message received about the temperature classification: Cool the division

Message received about the light classification: Don't change lights

Message received about the presence prediction: Presence noticed

============= Room system decidion =============

Attendance register: Presence noticed
HVAC action: Cool the division
Lights action: Don't change lights
```

Figura 45 - informação recebida pelo agente PEAK da sala

Na figura 45 podemos observar a receção dessa informação por parte do agente PEAK da sala. O agente PEAK da sala possui conhecimento, isto é, interpreta os dados recebidos e age em função do que recebe. Neste caso concreto, como existia presença notada na divisão, o que o agente PEAK da sala faz é concluir que uma vez que existe presença, o HVAC e o sistema de

luzes devem agir conforme as classificações que os seus dispositivos IoT enviaram. Como se pode observar, neste caso as luzes não precisavam de ser alteradas, mas o HVAC deveria arrefecer a divisão. Com esta informação, o agente PEAK da sala enviará ao agente PEAK do edifício de que forma a energia do edifício será afetada por estas ações. Considerando que nesta situação o HVAC iria ter que ser ativado para que a divisão pudesse ter a sua temperatura regulada, a energia gasta iria aumentar, sendo essa informação que o agente PEAK da sala comunica ao agente PEAK do edifício.

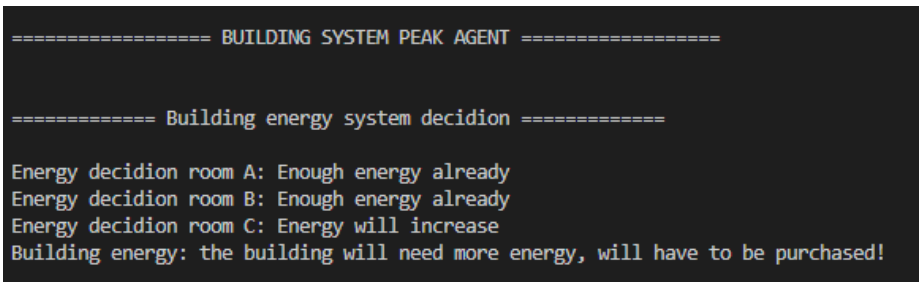

Figura 46 – Conclusões do agente PEAK do edifício

O agente PEAK do edifício recebe informação de todas as salas presentes no edifício, como podemos observar pela figura 46. A sala C é a sala da figura X, que indica ao agente PEAK do edifício que, como o HVAC vai trabalhar para que a temperatura da sala seja ajustada, existe um aumento da energia despendida. O agente PEAK do edifício analisa as informações enviadas por todas as salas, sendo que neste caso é indicado que as outras duas salas já possuem energia para as atividades que estão a realizar, e a sala C necessita de mais energia. Nesse sentido, o agente PEAK do edifício concluí que existe a necessidade de adquirir mais energia.

Importa salientar que as decisões são sempre em função da análise de todas as salas. Caso a sala A e sala B comunicassem ao agente PEAK do edifício que a energia para essas divisões é a necessária para realizar as ações, e o mesmo acontecesse com a sala C, então o agente PEAK do edifício iria concluir que a energia do edifício era a suficiente, sendo que nenhuma ação era necessária realizar no mercado de energia. Se pelo contrário, as salas A e B comunicassem o mesmo, mas a sala C comunicasse que a energia utilizada iria diminuir, numa situação onde não existisse presença na divisão, então o agente PEAK do edifício iria concluir que a energia presente no edifício estava em excesso, que o excedente poderia ser vendido no mercado de energia. Procura-se assim que o sistema tenha a melhor gestão da eficiência energética possível.

5 Conclusão

5.1 Resumo

O trabalho desenvolvido propõe a aplicação de um sistema multiagente, onde agentes individuais são responsáveis por dispositivos IoT, para o controlo inteligente da gestão energética de um edifício. Para que esse controlo seja, efetivamente, considerado inteligente, o sistema foi munido com redes neuronais artificiais aplicadas aos dispositivos IoT, isto é, foram aplicadas redes neuronais artificiais aos microprocessadores que se encontram ligados aos sensores presentes no edifício em questão.

Desta forma os sensores obtêm os valores de leitura correspondentes a cada um, interpretam esse valor e usam-no como input na sua rede neuronal artificial e uma previsão é encontrada. Essa previsão serve para perceber que ação deve ser considerada em função da área do sensor em questão, ou seja, se for um sensor de temperatura, então a previsão obtida será aplicada num dispositivo de HVAC para o controlo de temperatura num dado local. Caso estejamos a falar de um sensor de luminosidade, então a previsão já diz respeito ao ajuste de luminosidade.

5.2 Contributos do trabalho

De um ponto de vista científico, a dissertação tem uma contribuição importante no que respeita à aplicação de redes neuronais artificias em microprocessadores. Realmente é uma prova lúcida de que é possível que este tipo de dispositivo seja munido com inteligência, com capacidade de interpretação e com autonomia. Um contributo que é também um ponto de partida para um futuro trabalho e para autores que pretendam a implementação de redes neuronais artificias mais complexas e mais detalhadas em microprocessadores do tipo.

Importa também ressalvar a exploração de agentes virtuais, relativos a um sistema multiagente e que acabam por ser responsáveis por dispositivos IoT. Um contributo que se estende a uma importante troca de protocolo de comunicação, uma vez que os dois elementos utilizam dois protocolos distintos, sendo utilizado o protocolo de comunicação XMPP para o sistema multiagente e o protocolo de comunicação MQTT para os dispositivos IoT.

A nível de segurança e proteção de dados, importa salientar o máximo cuidado pelo tratamento dos dados utilizados, por forma a garantir que não existem dados confidenciais a serem usados ou divulgados, tentar com que os dados não tenham uma origem previsível para quem não conhece o sistema e para quem possa pretender utilizar os dados para algo que não o valor máximo do sistema desenvolvido, que é a gestão energética inteligente de edifícios.

De realçar que do ponto de vista de preenchimento de uma lacuna identificada, este sistema é muito válido e vem preencher um lugar importante nesse domínio, considerando que abrange

a integração de três áreas: sistemas multiagente, inteligência artificial e dispositivos IoT. Assim mostra-se que é possível que estas três áreas possam coexistir e que a sua utilização conjunta é uma mais valia para a criação de sistema deste tipo, como aquele que foi apresentado ao longo de todo o documento. Seguramente que o sistema desenvolvido pode ser considerado como um importante ponto de partida para que mais sistemas da mesma área de atuação possam ser desenvolvidos, e que sistema do mesmo tipo, com ainda mais funcionalidades, mais poder computacional possam surgir, entretanto. A identificação das lacunas e a procura pelo seu preenchimento está presente, tornando este como um dos maiores contributos que este trabalho oferece do ponto de vista científico.

5.3 Questões éticas

A ética, no sentido lato do termo, pode ser definida como sendo um determinado grupo de regras de conduta que devem ser seguidas de forma individual, para que possam ser cumpridas em grupo, considerando que aqui grupo se trata da representação da sociedade onde cada indivíduo se encontra inserido (Priberam, 2022). A ética em sistemas que lidam com dados e análises de comportamento, não é um assunto amplamente discutido e difundido, no entanto, a ética em sistemas como aquele que se pretende desenvolver são de uma importância fundamental, para cumprir os requisitos mais importantes nos dias modernos: a segurança do indivíduo e a proteção dos seus dados.

Um conceito existente em relação à ética em sistemas deste tipo é o de Design Ético (Baldini et al., 2018). Este conceito define que quem estaria sob o controlo das decisões acerca da proteção e segurança das aplicações seriam os utilizadores da mesma, ao contrário daquilo que é por norma feito, onde essa segurança é implementada no sistema propriamente dito, criando mecanismos de segurança que partem de dentro e onde o utilizador não possui influência direta. No fundo, neste referido design ético, o sistema é construído com a capacidade de capacitar os utilizadores a tomarem determinadas escolhas em função daquilo que são as suas escolhas éticas.

Acredita-se que os aspetos éticos de cada sistema ou aplicação desenvolvida acabem por diferir de quem está encarregue da sua construção. Poderá existir quem considera que um determinado tipo de dado que é recolhido pelo sistema não é assim tão impactante em relação à informação que fornece sobre o utilizador do sistema, mas poderão existir outros desenvolvedores que tenham uma opinião contrária a essa, e que acabem por considerar esse dado como extremamente sensível no que diz respeito à informação que transmite.

Mas independentemente de qual o paradigma seguido para cumprir os requisitos de um sistema do ponto de vista ético, é necessário perceber que fundamentalmente a ética se traduz como o seguimento das regras necessárias de serem tomadas para que possa ser garantida a segurança do sistema e dos seus utilizadores. É de uma importância extrema que os utilizadores se sintam seguros em relação a disponibilizar os seus dados pessoais para um sistema que

realmente necessita de os utilizar, mas com a convicção de que esses dados estão protegidos, a qualquer instante temporal e de qualquer possível ameaça externa ao sistema.

Essa segurança esperada por parte de um sistema, de IA ou IoT, por exemplo, acaba por classificar a forma como o sistema é catalogado. Supondo que existe um sistema de IA para condução autónoma, o que pressupõe que com maior ou menor intervenção humana, o carro será capaz de conduzir sozinho. Caso o carro acabe por ter um acidente, então a tendência será de classificar esse sistema como inseguro, enquanto que se esse sistema for infalível, ou se pelo menos nunca for registado um incidente de maior notoriedade, então o sistema será classificado como um sistema seguro (Green, 2018).

Na área da energia e dos sistemas de energia como aqueles que se pretende desenvolver, a ética estará seguramente relacionada com garantir a segurança de quem fornece esses dados, isto é, evitar que os dados sejam passíveis de revelar localizações, padrões comportamentais e outro tipo de informação que possa comprometer a segurança de quem fornece os seus dados para uma determinada aplicação os utilize.

5.4 Proteção de dados

Depois de realizada uma breve revisão acerca do que é a ética, o que ela representa na sua ampla generalidade, mas também em sistemas como aquele que se pretende desenvolver, é importante perceber de que forma as questões de segurança e proteção de dados podem ser implementadas para evitar a descredibilização do sistema e para garantir a proteção do mesmo.

Atualmente aspetos de proteção e segurança dos dados assumem-se como fundamentais em qualquer área de aplicação, não sendo isto uma exceção na área da energia e dos dispositivos inteligentes. Como se sabe, os dados que alimentam o sistema são provenientes de leituras realizadas pelos diversos dispositivos IoT instalados no edifício, como sensores de temperatura, luminosidade ou presença. Isto significa que através dos dados, é possível extrair um conjunto de informações que podem comprometer a privacidade de quem utiliza esse sistema. Ainda que um sensor de presença apenas tenha como valores de leitura zeros ou uns, a verdade é que uma análise a esse tipo de dado pode indicar rotinas que aconteçam dentro desse edifício, como saber-se que existem alturas do dia onde é muito pouco comum haverem leituras de presença, o que pode indicar a ausência dos proprietários do local nesses intervalos de tempo. O estudo dos valores lidos pelos sensores de luminosidade pode também indicar o tipo de atividade do proprietário, bem como as leituras dos sensores de temperatura, utilização dos aparelhos de aquecimento como ar-condicionado, entre outros.

Uma das vantagens do sistema em relação a este tipo de situação é a não transmissão dos dados sem tratamento, isto é, os dados lidos de sensores nunca são enviados ou utilizados para nenhum tipo de efeito a não ser para o teste das redes neuronais artificiais. Os dados que são utilizados para input no microprocessador, são já dados de saída da rede neuronal, ou seja, não se tratam efetivamente dos dados brutos, mas sim de dados já tratados e processados. Significa

isso que esses dados nunca são enviados para lado algum, mas apenas ficam no interior do microprocessador e nunca são armazenados. São dados obtidos em tempo real, utilizados para serem inputs da rede neuronal artificial e que desaparecem assim que o microprocessador deixa de ser utilizado, ou que novos valores são obtidos num intervalo de tempo futuro.

Uma outra forma de evitar fugas de segurança em relação aos dados medidos pelos sensores, é não lhes serem atribuídos uma geolocalização. É certo que com os dados brutos podem ser concluídas algumas informações que podem ser consideradas de alto risco de segurança, no entanto, esse tipo de informação acaba por ser muito pouco útil se não estiver ligado a nenhum local físico, ou seja, percebe-se que existe uma tendência comportamental de um determinado edifício perante a análise feita aos dados, mas não é possível saber onde fica localizado esse edifício, e portanto os dados passam a ser dados irrelevantes em questões de segurança. Do ponto de vista prático, os sensores possuem a si atribuídos um valor que corresponde ao local onde esse sensor está aplicado ou está em utilização, no entanto, esse valor é meramente um valor numérico que não possui qualquer tipo de associação, pelo que por si só, não nada revela.

5.5 Limitações e trabalho futuro

Em relação às limitações sentidas durante o desenvolvimento do sistema, especial atenção nas que envolvem as redes neuronais artificias nos microprocessadores. Ainda se verifica difícil a utilização de redes neuronais artificiais mais complexas dentro dos microprocessadores, com particular atenção para as redes neuronais artificias que utilizam *TensorFlow*. A fase de teste que deve ser realizada dentro do microprocessador não é intuitivamente implementável, o que faz com que ocorram muitos erros e problemas na sua utilização.

Uma outra limitação que é possível identificar no sistema desenvolvido é a necessidade de enviar manualmente as matrizes de treino para a rede neuronal artificial realizar a fase de teste dentro do dispositivo IoT. Considerando que existe a necessidade de ir atualizando o processo de treino fora dos dispositivos, para manter a rede neuronal atualizada, é necessário que esses dados matriciais sejam alterados manualmente dentro do dispositivo IoT.

O que é uma limitação acaba por ser também um trabalho de desenvolvimento futuro, ou seja, a inclusão de redes neuronais artificias mais complexas e exigentes do ponto de vista computacional seria algo muito interessante de ser explorado. Essa possibilidade de redes neuronais mais complexas possivelmente tornariam o sistema mais preciso, com uma maior capacidade de previsão e de classificação, tornando as suas conclusões mais enriquecidas e ao mesmo tempo mais completas e seguras do ponto de vista de decisão.

Numa tentativa de também mitigar uma outra limitação identificada, um trabalho de melhoria futura seria o envio dos dados matriciais provenientes da fase de treino da rede neuronal artificial para o dispositivo IoT de forma automática, ou seja, sempre que a rede neuronal artificial realizasse um novo treino e obtivesse novos valores matriciais para utilizar no processo de teste, seria muito interessante que esses dados pudessem ser atualizados sem a necessidade de o utilizador o fazer de forma manual.

Um outro desenvolvimento futuro muito interessante de implementar, seria a integração de dispositivos físicos reais como é o caso dos HVAC e de sistemas de luzes. Capacitando assim o sistema com uma capacidade real de controlar todos os detalhes do edifício, não só ao nível de previsão e de classificação, mas também com ações reais possíveis de serem realizadas, através da utilização dos mencionados dispositivos físicos.

Considerando os elevados progressos na eletrificação automóvel, uma possível melhoria futura seria a integração de sistemas de carregamento para veículos elétricos, permitindo assim que os veículos deste tipo pudessem realizar o seu carregamento no edifício.

Com as sugestões para trabalho futuro, surge também a perspetiva de integração de agentes do sistema multiagente que seriam responsáveis por ações de maior complexidade no que diz respeito à utilização de inteligência artificial e que seriam responsáveis pela aplicação de modelos avançados de apoio à decisão para a gestão do edifício, tendo como contribuição as decisões tomadas pelos dispositivos IoT que integram o sistema. Tratar-se-iam de agentes que não tinham uma ligação a um dispositivo IoT, como outros que constituem o sistema, mas cujo objetivo seria a tomada de decisão. Com isto, novas implementações, tais como o sistema de carregamento para veículos elétricos, poderia ser gerido de forma mais eficiente, um pouco como todo o edifício, contribuindo para melhores tomadas de decisão e uma mais eficiente gestão da energia elétrica.

6 Referências

Akkaya, K., Cherkaoui, S., Andersson, K. & Al-Turjman, F., 2018. *Proceedings of the 43rd Annual IEEE Conference on Local Computer Networks.* Chicago: s.n.

Al Sibahee, M. A. et al., 2020. Lightweight Secure Message Delivery for E2E S2S Communication in the IoT-Cloud System. *IEEE Access,* Volume 8, pp. 218331-218347.

Alawadi, S. et al., 2020. A comparison of machine learning algorithms for forecasting indoor temperature in smart buildings. *Energy Systems.*

Al-Qaseemi, S. A., Almulhim, H. A., Almulhim, M. F. & Chaudhry, S. R., 2017. IoT architecture challenges and issues: Lack of standardization. *FTC 2016 - Proceedings of Future Technologies Conference.*

Anon., 2021. *Jornal Expresso.* [Online]
Available at: https://expresso.pt/
[Acedido em 09 12 2021].

Anon., 2021. *Oxford Learner's Dictionaries.* [Online]
Available at: https://www.oxfordlearnersdictionaries.com/
[Acedido em 10 12 2021].

Babar, M., Tariq, M. U. & Jan, M., 2020. Secure and resilient demand side management engine using machine learning for IoT-enabled smart grid. *Sustainable Cities and Society,* Volume 62.

Berte, D.-R., 2018. Defining the IoT. *Proceedings of the International Conference on Business Excellence.*

Browning, N. & Sharafedin, B., 2021. *Reuters.* [Online]
Available at: https://www.reuters.com/business/energy/fossil-fuel-demand-shakes-off-pandemic-blow-climate-fight-2021-10-04/
[Acedido em 27 Dezembro 2021].

Chen, Z., Zhang, J., Arjovsky, M. & Bottou, L., 2019. Symplectic Recurrent Neural Networks.

Developers, G., s.d. *Gajim.* [Online]
Available at: https://gajim.org/
[Acedido em 20 Janeiro 2022].

Developers, P., s.d. *Pidgin.* [Online]
Available at: https://www.pidgin.im/
[Acedido em 20 Janeiro 2022].

Di Pascale, E. et al., 2018. The Network As a Computer: A Framework for Distributed Computing over IoT Mesh. *IEEE Internet of Things Journal,* pp. 2107-2119.

Dorri, A., Kanhere, S. S. & Jurdak, R., 2018. Multi-Agent Systems: A Survey. *IEEE Access,* Volume 6, pp. 28573-28593.

Duarte Roa, C., Schiavon, S. & Parkinson, T., 2020. *Targeted occupant surveys: A novel method to effectively relate occupant feedback with environmental conditions.* [Online] Available at: https://zenodo.org/record/4268447#.Ymv7i-iZOUl [Acedido em 19 Fevereiro 2022].

Gokhale, P., Gokhale, P. & Gokhale, P., 2007. Introduction to IOT. *International Advanced Research Journal in Science, Engineering and Technology ISO.*

Gregori, M., Cámara, J. & Bada, G., 2006. A Jabber-based Multi-Agent System Platform.

Gu, J. et al., 2018. Recent advances in convolutional neural networks. *Pattern Recognition,* Volume 77, pp. 354-377.

Hanga, K. M. & Kovalchuk, Y., 2019. Machine learning and multi-agent systems in oil and gas industry applications: A survey. *Computer Science Review,* Volume 34.

Henderson, P. et al., 2018. *Deep Reinforcement Learning that Matters,* s.l.: s.n.

Izzuddin, M. et al., 2018. *Smart Traffic Light based on IoT and mBaaS using High Priority Vehicles Method.* Malang, s.n.

Janošek, M., 2019. *Smart houses and multi-agent systems: A short review.* s.l., s.n.

Jiang, T., Gradus, J. L. & Rosellini, A. J., 2020. *Supervised Machine Learning: A Brief Primer,* s.l.: s.n.

Jozi, A. et al., 2019. Demonstration of an energy consumption forecasting system for energy management in buildings. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics),* Volume 11804 LNAI, pp. 462-468.

Kamilaris, A. & Prenafeta-Boldú, F. X., 2018. A review of the use of convolutional neural networks in agriculture. *Journal of Agricultural Science,* Volume 156, pp. 312-322.

Kumar, N. M. & Mallick, P. K., 2018. *The Internet of Things: Insights into the building blocks, component interactions, and architecture layers.* s.l., s.n.

Lez-Briones, A. G. et al., 2018. Multi-agent systems applications in energy optimization problems: A state-of-the-art review. *Energies.*

Liang, C. et al., 2019. *Intrusion Detection System for Internet of Things.* s.l., s.n.

Liu, Y., Pang, Z., Karlsson, M. & Gong, S., 2020. Anomaly detection based on machine learning in IoT-based vertical plant wall for indoor climate control. *Building and Environment,* Volume 183.

Lovas, R. et al., 2018. *PaaS-oriented IoT platform with connected cars use cases.* s.l., s.n.

Mahdavinejad, M. S. et al., 2018. Machine learning for internet of things data analysis: a survey. *Digital Communications and Networks.*

Manic, M., Amarasinghe, K., Rodriguez-Andina, J. J. & Rieger, C., 2016. Intelligent Buildings of the Future: Cyberaware, Deep Learning Powered, and Human Interacting. *IEEE Industrial Electronics Magazine,* Volume 10, pp. 32-49.

Mijwel, M. M., 2018. *Maad M. Mijwel Artificial Neural Networks Advantages and Disadvantages,* s.l.: s.n.

Mishra, S. K. & Sarkar, A., 2021. Service-oriented architecture for Internet of Things: A semantic approach. *Journal of King Saud University - Computer and Information Sciences.*

O'brien, P. & Nicol, R., 1998. *FIPA-towards a standard for software agents,* s.l.: s.n.

Org, X., s.d. *XMPP.* [Online]
Available at: https://xmpp.org/
[Acedido em 21 Janeiro 2022].

Palanca, J., 2020. *SPADE.* [Online]
Available at: https://spade-mas.readthedocs.io/en/latest/readme.html#
[Acedido em 20 Janeiro 2022].

Pal, R. et al., 2021. A comprehensive review on IoT-based infrastructure for smart grid applications. *IET Renewable Power Generation,* Volume 15, pp. 3761-3778.

Pinto, T. et al., 2022. Intelligent Simulation and Emulation Platform for Energy Management in Buildings and Microgrids. *Intelligent Systems Reference Library,* Volume 121, pp. 167-181.

Portuguesa, A. d. R. d. E. d. P. d. L. O., 2021. *Relatório sobre os Impactos do Covid-19,* s.l.: s.n.

Priberam, 2022. *Priberam.* [Online]
Available at: https://dicionario.priberam.org/%C3%A9tica
[Acedido em 28 Janeiro 2022].

Ribeiro, B., Pereira, H., Gomes, L. & Vale, Z., 2021. Python-based Ecosystem for Agent Communities simulation.

Rosenberger, J. et al., 2022. Deep Reinforcement Learning Multi-Agent System for. *MDPI.*

RTP, 2021. *RTP Notícias.* [Online]
Available at: https://www.rtp.pt/noticias/
[Acedido em 9 12 2021].

Shi, Y. et al., 2015. *The Fog Computing Service for Healthc,* China: IEEE.

Shi, Y. et al., s.d. *The Fog Computing Service for Healthcare,* s.l.: s.n.

Suganuma, T. et al., 2018. Multiagent-Based Flexible Edge Computing Architecture for IoT. *IEEE Network,* Volume 32, pp. 16-23.

Teixeira, B. et al., 2018. Multi-agent decision support tool to enable interoperability among heterogeneous energy systems. *Applied Sciences (Switzerland),* Volume 8.

Usama, M. et al., 2019. Unsupervised Machine Learning for Networking: Techniques, Applications and Research Challenges. *IEEE Access,* Volume 7, pp. 65579-65615.

Villarrubia, G., De Paz, J. F., Chamoso, P. & la Prieta, F. D., 2018. Artificial neural networks used in optimization problems. *Neurocomputing,* Volume 272, pp. 10-16.

Wang, R. et al., 2019. *Convolutional Recurrent Neural Networks for Text Classification; Convolutional Recurrent Neural Networks for Text Classification.* s.l.:s.n.

Ye, D., Zhang, M. & Vasilakos, A. V., 2017. A survey of self-organization mechanisms in multiagent systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems.*

Zhang, Q. et al., 2019. Artificial neural networks enabled by nanophotonics. *Light: Science and Applications,* Volume 8.

Zhang, Q. et al., 2019. Recent advances in convolutional neural network acceleration. *Neurocomputing,* Volume 323, pp. 37-51.