UNIVERSIDADE
LUSÓFONA

CENTRO UNIVERSITÁRIO DE LISBOA

ESCOLA DE COMUNICAÇÃO, ARQUITECTURA, ARTES E TECNOLOGIAS DA INFORMAÇÃO

MESTRADO EM ENGENHARIA INFORMÁTICA E SISTEMAS DE INFORMAÇÃO

ALGORITMOS DE INTELIGÊNCIA ARTIFICIAL PARA DETEÇÃO DE QUEDAS EM DISPOSITIVOS

Dissertação de Mestrado apresentada a provas públicas para a obtenção do grau de Mestre, orientada pelo Prof. Doutor João Pedro Leal Abalada de Matos Carvalho (ULusófona-CUL) e Prof. Doutor Sérgio Duarte Correia (ESTG-IPPortalegre).

Aluno:
Pedro Maria de Oliveira Martins dos Santos Bessa
Nº: a21102885

2024

www.ulusofona.pt

UNIVERSIDADE
LUSÓFONA

CENTRO UNIVERSITÁRIO DE LISBOA

ESCOLA DE COMUNICAÇÃO, ARQUITECTURA, ARTES E TECNOLOGIAS DA INFORMAÇÃO

MESTRADO EM ENGENHARIA INFORMÁTICA E SISTEMAS DE INFORMAÇÃO

ALGORITMOS DE INTELIGÊNCIA ARTIFICIAL PARA DETEÇÃO DE QUEDAS EM DISPOSITIVOS

Dissertação defendida em provas públicas na Universidade Lusófona - Centro Universitário de Lisboa no dia 28 de fevereiro de 2024, nomeado pelo Despacho de Nomeação nº581/2024, de 09 de janeiro, para a obtenção do grau de Mestre, perante o júri com a seguinte composição:

Presidente: Prof. Doutor Paulo Jorge Tavares Guedes (ULusófona-CUL)
Arguente: Prof. Doutor Filipe de Carvalho Moutinho (FCT/UNL)
Orientador: Prof. Doutor João Pedro Leal Abalada de Matos Carvalho (ULusófona-CUL)

Este trabalho também foi orientado por: Prof. Doutor Sérgio Duarte Correia (ESTG-IPPortalegre)

Aluno:
Pedro Maria de Oliveira Martins dos Santos Bessa
Nº: a21102885

2024

Algoritmos de Inteligência Artificial para Deteção de Quedas em Dispositivos

Agradecimentos

Expresso os meus agradecimentos à Universidade Lusófona por me acompanhar durante o meu percurso académico, por me proporcionar todas as condições intelectuais, de aprendizagem e pela escolha programática da licenciatura e mestrado em que estive envolvido como aluno. Ainda na mesma linha, expresso os meus agradecimentos à coordenação do Curso e aos Professores que se foram cruzando comigo, e que de alguma forma me acrescentaram conhecimento, método e sabedora.

Um agradecimento muito especial aos orientadores da minha tese, Professores João Carvalho e Sérgio Duarte Correia, que me acompanharam sempre, com muita paciência e sabedoria, e sem os quais este trabalho não seria uma realidade.

Agradeço profundamente à minha Família, em particular à Cristina, Mafalda e Francisca, por me permitirem a ausência em momentos em que deveria estar presente – e por isso e muito mais, o meu muito obrigado por toda a compreensão, amor e suporte familiar que recebi de vós.

À minha Mãe, que foi assistindo às dores de crescimento deste processo, e que me deu sempre um grande incentivo para continuar. Obrigado Mãe.

Resumo

Neste documento serão executadas atividades de análise e implementação de algoritmos de inteligência artificial para a deteção de quedas em idosos com base na medição de aceleração, considerando a sua implementação futura em equipamentos de baixo poder de computação.

Considerando o caráter temporal dos sinais, serão analisadas e implementadas redes recursivas com memória de curto prazo ou *Long Short-term Memory*, as LSTM. Os trabalhos terão em conta: a análise do atual estado da arte e a implementação e análise de performance das topologias de rede consideradas mais promissoras.

É estudada uma rede neuronal baseada em LSTM, sobre a qual são aplicadas técnicas no sentido de melhorar o desempenho dos sistemas visados. São testadas múltiplas camadas LSTM considerando janelas temporais mais alargadas, que na sua dimensão conseguem manter uma performance acima dos sistemas de referência.

Tendo em conta o modelo de classificação com três dimensões, os resultados apurados nesta abordagem demonstram-se satisfatórios, com métricas aperfeiçoadas relativamente ao modelo clássico de dados *Sisfall* e com níveis de *probabilidade de acerto, sensibilidade e especificidade* em linha com os melhores resultados dos sistemas em análise.

O modelo atingiu um máximo de *probabilidade de acerto* no treino de 94,93% e 0,14% de *valor de perda*, sendo os valores finais de validação de 93,11% e 0,19% respetivamente, tendo-se verificado que o modelo tem evolução em todas as dimensões.

Palavras-chave: *Artificial Intelligence, Wearables, Fall Detection, Embedded Programming*

Abstract

Analysis and implementation of artificial intelligence algorithms for detecting falls in the elderly based on the measurement of acceleration and/or angular velocity and taking into account their future implementation in equipment with low computing power.

Considering temporal nature of the signals, recursive networks with short-term memory or LSTM will be considered. This dissertation will focus on: analysing the current state of the art, implementing and analysing the performance of the most promising network topologies.

This dissertation studies a neural network based on LSTM, on which techniques are applied to improve the performance of the proposed systems. Multiple LSTM layers are tested considering wider time frames, which in their processing dimension manage to maintain a higher performance than the analysed systems.

Considering the non-binary classification model with multiple dimensions, the results of this approach are promising, with improved metrics compared to the classic *Sisfall* data model and with levels of *accuracy*, *sensitivity*, and *specificity* in line with the best results of the proposed systems.

This model achieved a maximum training *accuracy* of 94.93% and a 0.14% *loss value*, with final validation values of 93.11% and 0.19% respectively. It can be verified that the model has evolved in all its dimensions.

Keywords: *Artificial Intelligence, Wearables, Fall Detection, Embedded Programming*

Lista de Abreviaturas

AC	Accuracy
ADL	Activities of daily living
AVD	Atividades da Vida Diária
CNN	Convolutional Neural Network
CUSUM	Cumulative Sum
FC	Fully Connected
FD-CNN	Fall Dection CNN
FDS	Fall Detection System
FN	Falsos Negativos
FP	Falsos Positivos
FPGA	Field Programmable gate array
GPU	Graphics Processor Unit
GRF	Ground Reaction Force
IA	Inteligencia Artificial
IOT	Internet of things
KNN	K-Nearest Neighbors
LPWAN	Low Power WAN
LSTM	Long Short-term memory
MCU	Micro controller Unit
R&D	Research and development
RNN	Recurrent Neural Network
SHIMMER	Sensing Health with Inteligence, Modularity and Experimental Reusability
SMV	Signal Magnitude Vector
SPC	Controlo Estatístico de Processos
SVM	Support Vector Machine
TPU	Tensor Processing Unit
URFD	UR Fall Detection Dataset
VN	Verdadeiros Negativos
VP	Verdadeiros Positivos
WAN	Wide Area Network

Índice

Índice de Ilustrações

Índice de Tabelas

1 - Introdução

Em cenários domésticos, nos quais os idosos vivam de forma solitária, as quedas tornam-se comuns, provocando constrangimentos físicos que resultam em desconforto e perda da qualidade de vida. O mesmo acontece em lares ou centros de acolhimento, onde cerca de 30 a 50% dos residentes apresentam eventos de quedas recorrentes, segundo dados da Organização Mundial de Saúde. Importa referir que estas quedas se qualificam em ferimentos ligeiros a graves – 20% a 30%, e 40% delas acabam por conduzir à morte (Torti et al., 2019).

1.1 – Motivação e objetivos de trabalho

As quedas são uma das principais causas de lesões. Mais de um terço dos idosos sofre pelo menos uma queda grave a cada ano (Rajagopalan et al., 2017). A tendência para o crescimento do número de idosos é grande, e segundo a Pordata – no contexto nacional, a taxa de idosos por cada 100 jovens é de 182,1 %, sendo que em 1960 era de 27,3% (PORDATA, 2021).

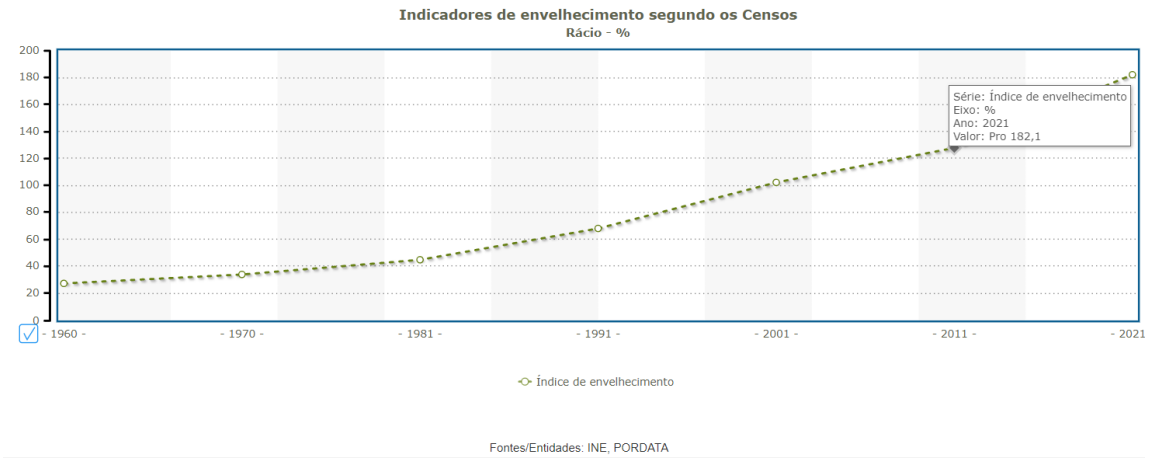


Figura 1 - Índice de envelhecimento - Dados Pordata – 2021.

Não obstante, as quedas em si não são uma ameaça - na maior parte dos casos. O problema maior acontece em contextos onde as pessoas levam uma vida mais independente até uma idade mais avançada. Fruto dessa liberdade, em eventos de queda em que as pessoas não são socorridas pelos sistemas de saúde de forma atempada, esses eventos deixam sequelas que poderão evoluir para problemas futuros. Por outro lado, se uma queda for antecipada com

precisão ou reconhecida a tempo, os riscos de queda e os seus efeitos podem ser significativamente minimizados.

Não sendo possível implementar mecanismos de prevenção ou monitorização humana em contextos de liberdade de mobilidade, torna-se pertinente uma abordagem de "deteção de quedas", que atue como sendo um sistema de reconhecimento de quedas e identificação correta das atividades diárias (Islam et al., 2020).

Pode-se encontrar na literatura científica diversos métodos para identificar quedas de pessoas idosas. Esses são baseados na combinação de técnicas *de machine learning*, da Internet das Coisas (IoT), e no reconhecimento de imagens, entre outras tecnologias.

A previsão de quedas é o processo de monitorização contínua de uma pessoa idosa, que se combina com a utilização de objetos incorporados ou em vestuário, para determinar a probabilidade de uma queda, e em que circunstancias essa queda pode acontecer (Rajagopalan et al., 2017).

A previsão de quedas será o principal objetivo deste trabalho, pelo que serão estudadas as redes neuronais consideradas mais adequadas à sua previsão. Será feita a análise do atual estado da arte, bem como a implementação e análise de performance das topologias de rede consideradas mais promissoras, com o objetivo futuro de as implementar em plataformas computacionais de baixo poder de computação, reduzindo desta forma as necessidades energéticas e computacionais, tal como efetuado em (Torti et al., 2018).

1.2 – Estrutura do documento

Neste documento a formulação da proposta de trabalho está estruturada da seguinte forma:

Em primeiro lugar, existe uma secção com o levantamento e estudo sobre os trabalhos relacionados, de seguida passar-se-á para uma descrição pormenorizada do método proposto, metodologias aplicadas, dificuldades encontradas e soluções aplicadas. Segue-se uma secção onde se apresentam os resultados experimentais obtidos sobre um conjunto de dados selecionados, tendo como base os trabalhos considerados como referência (Torti et al., 2019) e (Musci et al., 2018). Por fim, a última secção conclui o documento, discorrendo-se nesse momento sobre os resultados obtidos e se discutem propostas de trabalho futuro, concluindo-se deste modo o estudo proposto.

2 - Revisão da literatura

Neste capítulo será feito o levantamento e estudo sobre os trabalhos relacionados, com enquadramento inicial tendo em conta os sistemas de prevenção e deteção de quedas. Serão apresentados os tipos de sistemas utilizados – Vestíveis e não Vestíveis, bem como apresentadas as técnicas de *deep learning* adequadas ao tema e as arquiteturas de redes neuronais comummente utilizadas para a deteção de quedas.

2.1 - Sistemas de prevenção e deteção de quedas

O desenvolvimento de sistemas de prevenção e deteção de quedas tem-se tornado num tópico de investigação nos últimos anos. Várias abordagens têm sido consideradas no seu processo evolutivo. Esses sistemas são classificados em duas categorias maiores: Sistemas Vestíveis e não Vestíveis (Usmani et al., 2021).

2.1.1 - Sistemas não Vestíveis

Segundo (Usmani et al., 2021), os Sistemas não Vestíveis são compostos por sensores colocados nas proximidades dos humanos, para a recolha de dados e monitorização dos movimentos, nomeadamente o andar humano. Estes são, por sua vez, subdivididos em sensores de visão e outros instalados no chão (Muro-de-la-Herran et al., 2014). Sensores de visão, como câmaras e sensores de infravermelhos, fazem medições óticas e usam processamento de imagem para análise. A vigilância por vídeo é um tipo comum desses sistemas, que capturam imagens e usam diferentes algoritmos para determinar a ocorrência de quedas. Por outro lado, os sensores instalados no chão – baseados no piso, como sensores de Força de Reação do Solo (GRF) e sensores de pressão, registam a força exercida pelo pé humano para observar a queda (Serra et al., 2016).

O número de sensores varia para cada cenário, sendo esta a principal desvantagem dos sistemas não vestíveis, onde o raio de ação é limitado pela sua natureza. Tais sistemas podem ser implementados em escritórios, residências e laboratórios, tornando-os menos adaptáveis e caros na sua adoção, não sendo ideais para a maioria das aplicações da vida real.

2.1.2 - Sistemas Vestíveis

No que se refere aos Sistemas Vestíveis, estes consistem em dispositivos ou sensores que se podem acoplar ao corpo humano para a recolha de dados. Utilizam-se acelerómetros, giroscópios, magnetómetros, entre outros (Lou et al., 2020). A vantagem destes sistemas é a sua capacidade de recolher dados fora de ambientes controlados, como sejam os laboratoriais. Deste modo, estes sistemas consideram-se como uma solução viável para a análise e deteção de quedas e prevenção das mesmas durante a realização de atividades diárias. Os sensores são frequentemente incorporados em equipamentos - como sejam telemóveis, sem que haja a necessidade de investimento adicional, permitindo desta forma a recolha de dados. Não obstante, esses dados precisam de processamento adicional utilizando algoritmos para a tomada de decisão. Abaixo, apresenta-se uma lista com o nome dos sensores relevantes e respetivas funcionalidades:

- Acelerómetro - Mede a taxa de variação da velocidade (aceleração) de um objeto ao longo de seu eixo.
- Giroscópio – Mede as mudanças de rotação em relação à orientação. Com esses dados, calcula a velocidade angular ao longo de três eixos: X, Y e Z.
- Magnetómetro – Mede a mudança relativa de um campo magnético, a sua direção e força.

Outros aspetos apresentam-se como relevantes e são objeto de estudo. Os dispositivos vestíveis são por natureza instâncias de componentes eletrónicos miniaturizados, por norma incluindo sensores e uma unidade de processamento, e como já referido, acoplados a humanos ou acomodados por cima de roupa.

Várias soluções de dispositivos vestíveis foram referidas pela literatura. O de (Nyan et al., 2008), que propõe um sistema que usa acelerómetros e giroscópios. Esta abordagem pressupõe que os sensores se liguem através de redes sem fios – e uma placa baseada num processador Intel PXA255 – onde decorre o processamento. Noutro sistema, é um circuito integrado que carrega um acelerómetro ADXL345 - usado para medir a aceleração nos três eixos: x, y e z, e uma FPGA para realizar o processamento dos sinais (Abdelhedi et al., 2016). Ainda na mesma linha (Cola et al., 2017) desenvolveram um dispositivo para uso na cabeça, contendo um acelerómetro e um barómetro, integrados num microcontrolador da *Texas Instruments* MSP430.

Segundo os autores (Torti et al., 2019), do ponto de vista de software, os métodos de deteção de queda aqui referidos são executados nos próprios dispositivos, usando um limite pré-definido aplicado aos sinais recebidos ou aos indicadores estatísticos. Ainda sobre o tema, estes autores referem que existem métodos mais sofisticados noutras abordagens, mas com o ónus de o processamento ocorrer remotamente numa estação de trabalho ligada ao sistema. Uma dessas abordagens é baseada na plataforma de sensores integrados SHIMMER (*Sensing Health with Intelligence, Modularity, Mobility and Experimental Reusability*) (Hossain et al., 2016). Neste caso, o processamento remoto dos sinais do acelerómetro é realizado através de máquinas de suporte vetorial - *Support Vector Machine* (SVM); os autores também comparam o último método com *k-Nearest Neighbors* (kNN) e árvores complexas. Estes também referem outra abordagem semelhante que é descrita em (Fafoutis et al., 2018), onde se reúnem e processam os sinais de forma offline, recolhidos por acelerómetros e giroscópios. Neste caso os dados são analisados recorrendo-se a algoritmos de *machine learning*. Nesse artigo alude-se igualmente à questão dos consumos energéticos, uma vez que o desenho e a arquitetura devem ser personalizados com o objetivo específico de minimizar o consumo de energia num sistema de deteção de quedas (Nguyen Gia et al., 2018).

2.2 - Técnicas de *Deep Learning*

Recentemente, os sistemas de *deep learning* vêm sendo utilizados em uma miríade de áreas do conhecimento (Lecun et al., 2015). Segundo estes, o *deep learning* permite que modelos computacionais compostos por várias camadas de processamento, aprendam representações de dados com vários níveis de abstração. Estes métodos melhoraram drasticamente, como se pode observar através do estado da arte, no que se refere ao reconhecimento da fala, reconhecimento visual de objetos, sua deteção, e em muitos outros domínios, como sejam a descoberta de novas drogas e avanço na ciência da genómica. O *deep learning* descobre estruturas complexas em grandes conjuntos de dados usando o algoritmo de retro propagação, para indicar como uma máquina deve alterar os seus parâmetros internos que são usados para calcular a representação em cada camada a partir da representação na camada anterior.

Existem várias técnicas de *deep learning* para reconhecimento de eventos de quedas. Alguns sistemas são desenvolvidos utilizando apenas um método de *deep learning*, pese embora outros utilizem diferentes combinações para atingir uma maior taxa de sucesso na deteção de eventos de quedas.

Os autores (Islam et al., 2020) fizeram uma revisão de literatura que abrange 37 artigos que abordam o tema deste trabalho. Nessa revisão, os 37 artigos identificam 21 sistemas, dos quais 56.8% usou uma arquitetura *Convolutional Neural Network* (CNN) para a deteção de quedas. Um total de 12 sistemas usou a arquitetura LSTM e um total de 4 sistemas usou uma arquitetura de *auto-encoder*. Relativamente a esta arquitetura de *auto-encoder*, não será o foco do trabalho. Não obstante, é de referir que se trata de uma rede neuronal artificial não supervisionada (Dong, 2018), (Alqahtani et al., 2018). Existem 4 dimensões consideradas para o desenvolvimento de um modelo de *auto-encoder*, que se compõe por; *encoder, bottleneck, decoder e reconstruction* (Rangamani et al., 2018), (Charte et al., 2019).

2.2.1 - Redes Neuronais Convolucionais

Os sistemas de deteção de quedas baseados em redes neuronais CNN, representam as quedas e as atividades diárias em formato de imagens (Yao et al., 2019), (Z. Zhang et al., 2019). Este processo pode ser executado diretamente, usando imagens ou vídeos capturados por sensores, ou representando outros dados de sensores em forma de imagem. A arquitetura das

CNN é muito eficiente quando se pretende encontrar padrões e formas dentro de determinadas imagens (Rawat & Wang, 2017), (Voulodimos et al., 2018). Desta forma, os sistemas de deteção de quedas que se socorrem de várias arquiteturas baseadas em CNN, aproveitam o seu poder de categorização ou deteção de imagens.

2.2.2 - Redes Neuronais de Memória de Longo e Curto Prazo

As redes neuronais de memória de longo e curto prazo são um aperfeiçoamento das *recurrent neural network* (RNN). Tal como as CNN, as LSTM podem manipular dados de imagem. Não obstante, a sua principal vocação é a capacidade em lidar com dados sequenciais, como sejam dados de séries temporais – *time series data* (Miikkulainen et al., 2018), (Veeriah et al., 2015).

Uma vez que os dados dos sensores que registam eventos de queda são sequenciais – *time specific*, as LSTM podem ser usadas com sucesso e eficiência para distinguir entre eventos de queda e as atividades diárias (ADL em inglês – *activities of daily living*).

Segundo os autores (Miikkulainen et al., 2018), (Veeriah et al., 2015), em alguns dos sistemas revistos foram utilizadas combinações de LSTM e CNN para ajudar à melhor deteção de quedas e superar problemas relacionados com a visão, ruído das imagens, segmentação incorreta e generalizações com as perspetivas. Continuam a análise referindo que, enquanto as arquiteturas LSTM e CNN são fundamentalmente usadas em ambientes supervisionados, os *auto-encoders* são preferencialmente usados para compreender de forma eficiente as representações de dados em ambientes não supervisionados (Ge et al., 2018), (Dong, 2018). Geralmente, algumas técnicas são aplicadas nos dados recolhidos pelos sensores e transformados em dados específicos para diferentes tipos de quedas e em atividades diárias - ADL. Os *auto-encoders* são depois usados nos dados transformados – os *datasets*, para distinguir entre os eventos registados.

Já os sistemas supervisionados são categorizados com base no método utilizado para inferir ou deduzir os eventos registados. A categorização foca-se em como os métodos relevantes (CNN, LSTM e *auto-encoders*) lidam com eventos de queda capturados pelos sensores.

Nas últimas décadas, vários sistemas e métodos de *deep learning* foram desenvolvidos para o reconhecimento das quedas e ADL – conforme já identificado acima. Muitas organizações e comunidades de *research* têm trabalhado em investigação e desenvolvimento, no sentido de desenvolver métodos mais eficientes no reconhecimento de quedas, diminuindo a valores residuais os falsos alarmes nas situações do mundo real.

2.3 - Pesquisa sobre o tema

De seguida serão apresentadas as mais recentes pesquisas sobre o tema e de como as técnicas desenvolvidas estão a criar sistemas de deteção de quedas mais apurados, considerando: (1) - o trabalho proposto, (2) - os princípios aplicados, (3) - os *datasets* utilizados para o treino das redes e por fim, (4) - a análise crítica e de desempenho do esquema.

A maioria dos sistemas analisados usa as CNN para o desenvolvimento de sistemas automáticos de deteção de quedas. A CNN captura uma imagem como dado de entrada e de seguida processa-a e classifica a imagem processada em determinadas categorias (Q. Zhang et al., 2019), (Voulodimos et al., 2018).

A técnica das CNN consiste na passagem de cada imagem de entrada através de uma série de camadas convolucionais recorrendo a filtros, camadas de *pooling*, camadas ligadas – as *fully connected* (FC) (Voulodimos et al., 2018), e uma função denominada de *softmax* com o objetivo de treino e testes à rede. Certos sistemas de deteção de quedas são propostos por (S. Wang et al., 2016), (Solbach & Tsotsos, 2017). Os sistemas de deteção de quedas baseados em CNN são apresentados um pouco mais abaixo.

2.3.1 - Enquadramento sobre os resultados

Como nota prévia, para que se entendam os resultados, estes apresentam-se considerando os elementos de uma matriz de confusão, com a seguinte estrutura (Mariano, 2021):

Tabela 1 - Elementos de uma matriz de confusão

	Classificado com *Fall*	Classificado como ADL
Queda	**VP** (Verdadeiro Positivo)	**FN** (Falso Negativo)
ADL	**FP** (Falso Positivo)	**VN** (Verdadeiro Negativo)

Verdadeiro positivo (VP): quando o método utilizado diz que a classificação é positiva e, ao verificar a resposta, confirma-se que a classificação era positiva;

Verdadeiro negativo (VN): quando o método utilizado diz que a classificação é negativa e, ao verificar a resposta, confirma-se que a classificação era negativa;

Falso positivo (FP): quando o método utilizado diz que a classificação é positiva, mas ao verificar a resposta, afinal a classificação era negativa;

Falso negativo (FN): quando o método diz que a classificação é negativa, mas ao verificar a resposta, afinal a classificação era positiva;

A **probabilidade de acerto** (*accuracy*) / rácio de deteção, é considerada uma das métricas mais importantes, uma vez que avalia a percentagem de valores corretos – os acertos, sobre o somatório de todos os itens, conforme a equação 1:

$$\frac{VP+VN}{VP+FN+VN+FP} \quad (1)$$

Outra métrica utilizada é a **sensibilidade**, também conhecida como *recall*. Neste caso, avalia-se a capacidade de detetar com sucesso os positivos e o seu cálculo obtém-pela equação 2:

$$\frac{VP}{VP+FN} \quad (2)$$

Por outro lado, a **especificidade** avalia a capacidade de detetar resultados negativos. O seu cálculo obtém-pela equação 3:

$$\frac{VN}{VN+FP} \quad (3)$$

Finalmente, a **precisão** é uma métrica que avalia a quantidade de verdadeiros positivos sobre a soma de todos os valores positivos, sendo esta a sua representação dada pela equação 4:

$$\frac{VP}{VP+FP} \quad (4)$$

No contexto de um sistema de deteção de quedas, onde a classe *FALL* é de maior interesse, a métrica mais importante geralmente é a **sensibilidade**, uma vez que se mede a proporção de instâncias de queda que são corretamente identificadas pelo modelo em relação ao total de instâncias de queda. A **sensibilidade** vem revelar a capacidade do modelo em identificar todas as quedas verdadeiras, minimizando os falsos negativos.

Ainda no contexto das quedas, a **especificidade** é menos relevante por comparação com a sensibilidade. A **especificidade** mede a capacidade de o modelo identificar corretamente as instâncias negativas, ou seja, as atividades diárias normais (BKG) ou os alertas (ALERT). Embora seja importante ter a métrica **especificidade** equilibrada, para evitar a identificação incorreta das outras atividades diárias como sendo quedas, o foco principal está em garantir que o modelo identifique a maioria das quedas verdadeiras (VP), minimizando os falsos negativos (FN), pelo que a **sensibilidade** é a métrica adequada para avaliar se as quedas são identificadas com rigor, porquanto a especificidade poderá ser menos relevante neste contexto.

A **probabilidade de acerto** geral é também uma medida importante pois avalia a precisão geral do modelo em todas as classes, incluindo BKG, ALERT e FALL. Representa a proporção da previsão acertada pelo modelo em relação ao total de previsões realizadas. Embora a **probabilidade de acerto** geral permita ter uma visão ampla do desempenho do modelo, é importante considerar também as métricas específicas para cada uma das classes. Atingir valores altos na **probabilidade de acerto** geral é desejável, mas é fundamental avaliar também o desempenho do modelo em cada classe individualmente para garantir que o modelo identifica corretamente as quedas, minimizando tanto os falsos positivos bem como os falsos negativos.

2.3.2 - Sistemas de deteção de quedas baseados em CNN

No primeiro estudo, (Yhdego et al., 2019), os autores propuseram uma abordagem recorrendo a um modelo pré-treinado em *Kinematics* – cinemática, com recurso a algoritmos de *machine learning*, alimentando-se de um conjunto de dados guardados (*datasets*) com os registos dos acelerómetros. A Figura 2 ilustra o modelo proposto, descrevendo os diversos processos de transformação das imagens recorrendo à função convolucional bem como várias camadas totalmente ligadas – *fully connected*, para ajudar à identificação de representações complexas e não lineares dos padrões nos dados, permitindo a melhor classificação do modelo à saída.

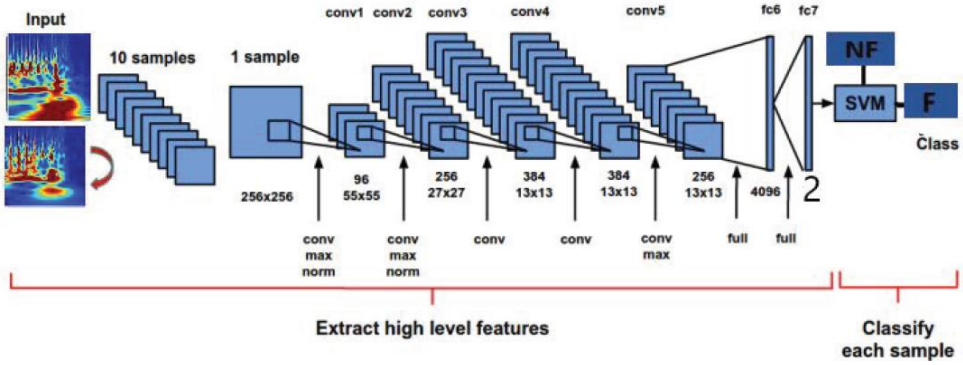


Figura 2 - Arquitetura CNN proposta. Fonte: (Yhdego et al., 2019)

Neste modelo, os dados são convertidos em imagens usando uma "*continuous wavelet transform*", conforme se pode observar no escalograma da Figura 3.

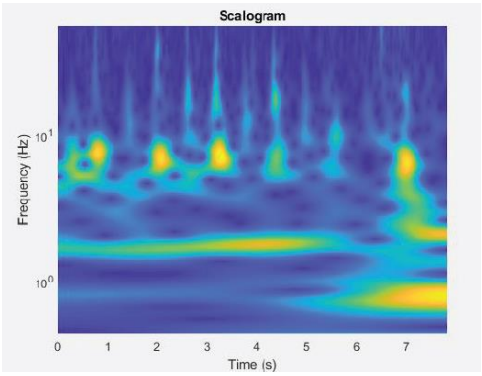


Figura 3 - Amostragem de um escalograma. Fonte: (Yhdego et al., 2019)

O resultado dessa aprendizagem é depois utilizado para treinar uma rede CNN baseada nas imagens transformadas. Foi utilizado um conjunto de dados disponibilizado por um *dataset UR Fall Detection Dataset* (URFD) – contendo 30 sequencias de quedas e 40 sequências de atividades correntes. Nesse *dataset* foi executada uma separação dos dados para teste e validação da rede à razão de 80% e 20% respetivamente O sistema proposto obteve uma probabilidade de acerto na ordem dos 96,43%, precisão e sensibilidade de 95,83 %, e 96,875 % de especificidade, conforme se pode avaliar pela Figura 4.

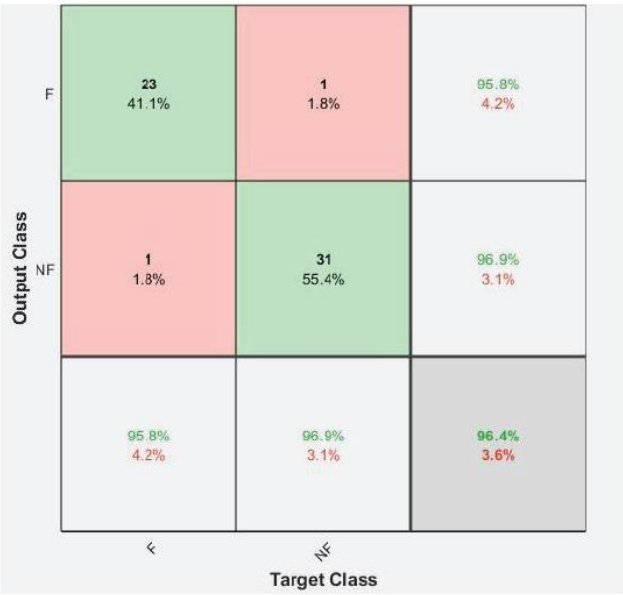


Figura 4 - Matriz de Confusão com os resultados obtidos. Fonte: (Yhdego et al., 2019)

Já em (He et al., 2019), foi desenvolvido um método de deteção de quedas baseada em FD-CNN (*fall detection*), com recurso a um sistema de baixa potência. Este sistema foi desenvolvido tendo como base um conjunto de sensores acionados por interruptores, num circuito impresso MPU6050 e um sistema sem fios baseado no protocolo ZigBee - de baixo consumo, para amostragem e *caching* da velocidade angular bem como de 3 eixos de aceleração. A Figura 5 refere o esquema com os sensores, suas dependências e diagrama de processamento dos dados bem como compara o seu tamanho com uma moeda de 1 cêntimo, para que se possa ter uma perspetiva das dimensões dos circuitos.

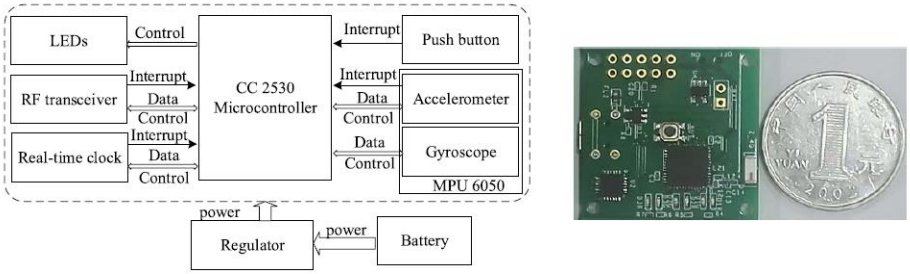


Figura 5 - A placa com os sensores, sua arquitetura e tamanho do circuito. Fonte: (He et al., 2019)

De seguida, os dados recolhidos são mapeados em imagens RGB de 3 canais e combinados com dados transformados de *datasets Sisfall*[1] e *MobiFall*[2], sendo o seu resultado utilizado para o treino da rede em FD-CNN, conforme se pode ver pelo fluxo de dados apresentados na Figura 6.

Os datasets *Sisfall* e *Mobifall* contêm dados sobre quedas. O *Sisfall* agrega dados sobre quedas e atividades da vida diária (AVD) adquiridos com um dispositivo desenvolvido pelos próprios, composto por dois tipos de acelerómetros e um giroscópio. Consiste em 19 ADL e 15 tipos de quedas realizadas por 23 jovens adultos, 15 tipos de ADL realizadas por 14 participantes saudáveis e independentes com mais de 62 anos e dados de um participante de 60 anos que realizou todas as ADL e quedas (Sucerquia et al., 2017).

No que se refere ao *MobiFall*, baseia-se em dados de sensores inerciais de smartphones. Incorpora sinais registados a partir dos sensores do acelerómetro e do giroscópio

[1] Fonte de dados em: https://www.kaggle.com/datasets/gangulahemanthreddy/sis-fall-dataset

[2] Fonte de dados em: https://www.kaggle.com/datasets/kmknation/mobifall-dataset-v20/code

para quatro quedas diferentes e nove atividades da vida diária (ADL) diferentes (Vavoulas et al., 2013).

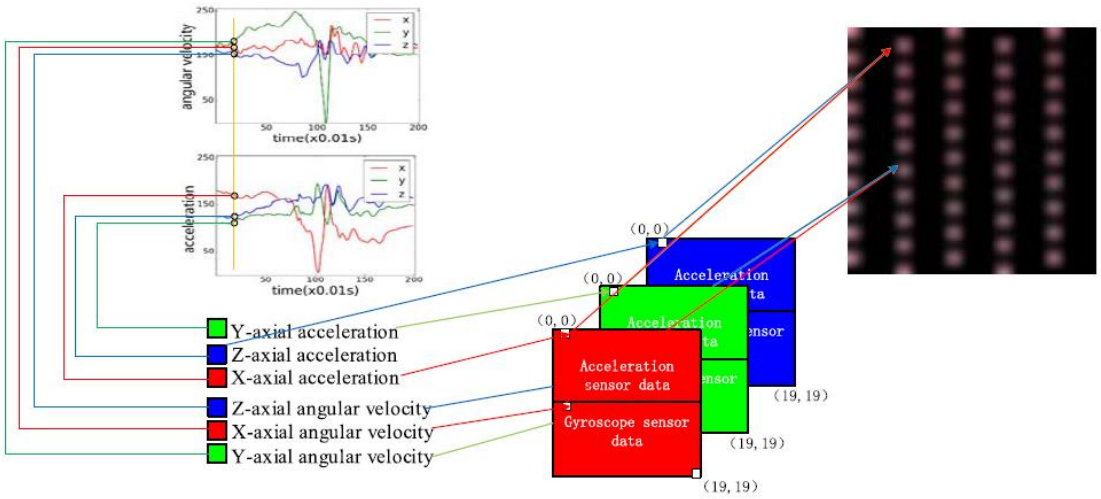


Figura 6 - Esquema dos mapeamentos dos dados para o mapa de bits RGB. Fonte: (He et al., 2019)

Sobre estes dados combinados é executada uma separação de rácio 90% para treino e 10% para validação, que serve de base para a aprendizagem da rede. Como resultados desta proposta, os valores de performance apurados foram os seguintes:

O desempenho da tecnologia proposta alcançou uma *probabilidade de acerto* média de 98,61%, enquanto as *especificidade* e *sensibilidade* médias de 99,80% e 98,62%, respetivamente, usando um procedimento de validação cruzada (*fold cross-validation*) de 10 vezes. A validação cruzada é uma técnica comum para avaliar o desempenho de um modelo estatístico, especialmente quando se tem um conjunto de dados limitado. Esta técnica ajuda a perceber o quão bem um modelo pode generalizar para dados que não foram utilizados no treino. O processo de aprendizagem segue da seguinte forma:

Divisão do Conjunto de Dados: O conjunto de dados é dividido em k partições.

Treino e Teste: O modelo é treinado k vezes. Em cada iteração, k-1 partições são utilizadas para treinar o modelo, e a partição restante é utilizada para testar o modelo.

Avaliação do Desempenho: A métrica de desempenho - por exemplo a *precisão*, e a *sensibilidade,* é calculada para cada iteração.

Média das Métricas: A métrica de desempenho final é a média das métricas obtidas em cada iteração.

Os autores concluem que a performance pode ser melhorada se forem utilizadas tecnologias de ponta - *edge computing*, recorrendo a métodos de computação em rede de baixo consumo.

Um método mais generalizado combinando CNN com um algoritmo de deteção de mudança – "*efficient change detection algorithm*", é proposto pelos autores (Tasoulis et al., 2019), para responder às dificuldades causadas pela generalização e o problema da otimização dos recursos. Este método de "*wide generalization*", trabalhou sobre os dados em bruto de acelerómetros com baixa exigência de recursos. O sistema desenvolvido utilizou um *dataset* da *SmartFall*[3], onde os dados recolhidos das experiências efetuadas a 7 indivíduos diferentes, utilizaram um *wearable* da Microsoft no seu pulso – o Microsoft Band 2.

O objetivo do projeto *Smartfall* é combinar a dinâmica de movimento do mundo real de um indivíduo com os dados recolhidos pelo acelerómetro através de um relógio de pulso comum e de um telemóvel na anca oposta para melhorar a deteção de quedas fortes e leves (Ngu et al., n.d.). Foi utilizado um algoritmo CUSUM (*cumulative sum*), para melhorar a performance do sistema. A soma cumulativa é uma técnica estatística utilizada para monitorizar a consistência ou estabilidade de um processo ao longo do tempo. É frequentemente aplicada no controlo estatístico de processos (SPC) para detetar alterações ou desvios significativos. O CUSUM calcula a soma cumulativa dos desvios observados em relação a um valor de referência ao longo do tempo. É particularmente útil para detetar alterações pequenas e graduais. O objetivo é identificar quando a soma acumulada ultrapassa um limite predefinido, indicando que pode ter ocorrido uma mudança significativa na atividade em causa. A interpretação do CUSUM envolve observar os pontos em que a soma acumulada ultrapassa os limites de controlo especificados. Esses pontos indicam períodos em que algo pode ter desviado significativamente da condição normal. O *dataset* foi sendo alimentado pelos registos das ações de 7 pessoas diferentes, totalizando 51192 amostras. Neste caso houve uma separação sobre os dados à razão de 66% para treino da rede e 44% para validação tendo os resultados sido os seguintes: A *precisão* e a *precisão balanceada* do sistema proposto atingem os 96,79% e 95,35%

[3] Fonte de dados em: https://userweb.cs.txstate.edu/~hn12/data/SmartFallDataSet.zip

respetivamente. Por fim, é referido que o modelo pode ser treinado e testado com mais eficiência se forem utilizados *datasets* contendo dados reais de atividades diárias.

Na mesma linha, (L. Wang et al., 2019) propuseram um sistema para a deteção de quedas bem como para o reconhecimento das atividades diárias.

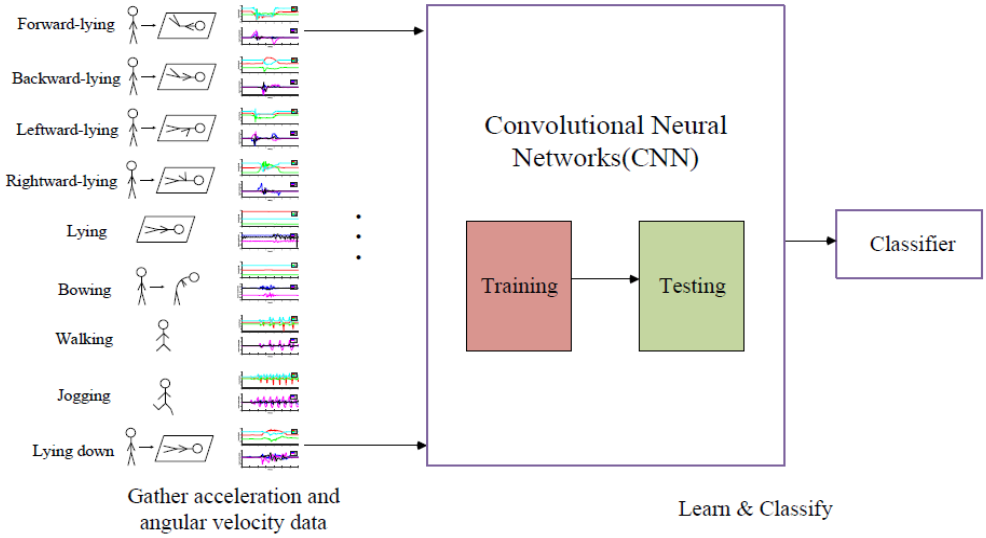


Figura 7 - Ilustração do sistema proposto de deteção de quedas. Fonte: (L. Wang et al., 2019)

O sistema combina um acelerómetro com três eixos e um sensor de giroscópio instalado numa palmilha inteligente. A Figura 7 ilustra o sistema, onde se registam as atividades diárias através dos dados de aceleração e com esses dados se executam os treinos e testes de uma rede CNN, terminando com um modelo de classificação para a deteção de quedas.

É usado um algoritmo de CNN para melhorar o nível de precisão sem que haja grande complexidade computacional. O *dataset* é alimentado por voluntários que usam a palmilha inteligente. A principal virtude deste sistema é conseguir aplicar a CNN diretamente nos dados não tratados do sensor, recolhidos pelo acelerómetro de três eixos e do giroscópio. O sistema foi testado num conjunto de dados personalizados, contendo 800 quedas – em séries de 200 de cada, onde a primeira se refere ao movimento de deitar na cama, a segunda ao movimento de se curvar, a terceira a deitar-se, e por último, a caminhar e caminhada larga – jogging. A palmilha inteligente foi concebida para a recolha de dados dos voluntários.

Figura 8 - A palmilha inteligente é ligada ao telemóvel para o registo de dados. Fonte: (L. Wang et al., 2019)

A Figura 8 apresenta um conjunto de três imagens, onde no seu lado esquerdo pode ser observada a palmilha inteligente numa vista superior e lateral. Do lado direito da figura apresenta-se uma representação do que significa a capacidade de a palmilha enviar os dados para análise - todos os dados relativos à aceleração e à velocidade angular são enviados para o smartphone para armazenamento de dados via *Bluetooth*.

A *probabilidade de acerto* média do sistema proposto é de aproximadamente 98,61%. Para além destas métricas, são alcançados 97,58% de *sensibilidade* e 99,58% de *especificidade*. Os autores referrem também, como nota final, que em termos de *sensibilidade* e *especificidade*, o método proposto supera outros algoritmos tradicionais.

Numa abordagem ligeiramente diferente, (Casilari et al., 2020) propuseram um sistema de deteção de quedas também baseado numa CNN. O sistema deteta eventos de queda reconhecendo os padrões recebidos do acelerómetro portátil baseado em 3 eixos. Para o estudo desta abordagem, o sistema recorreu a múltiplos conjuntos de *datasets* como origem para os dados de treino, incluindo vários disponíveis de forma aberta como sejam: *MobiAct*[4], *SisFall*, *MobiFall*, *UniMiB SHAR*[5], etc.

O *MobiAct* é um conjunto de dados disponível ao público que inclui dados de um smartphone quando os participantes estão a realizar diferentes tipos de atividades e uma série de quedas (Vavoulas et al., 2016). O UniMiB SHAR é um novo conjunto de dados de amostras de aceleração adquiridas com um smartphone Android concebido para o reconhecimento da atividade humana e a deteção de quedas. O conjunto de dados inclui 11 771 amostras de atividades humanas e quedas realizadas por 30 indivíduos com idades compreendidas entre os 18 e os 60 anos (Laboratory, n.d.).

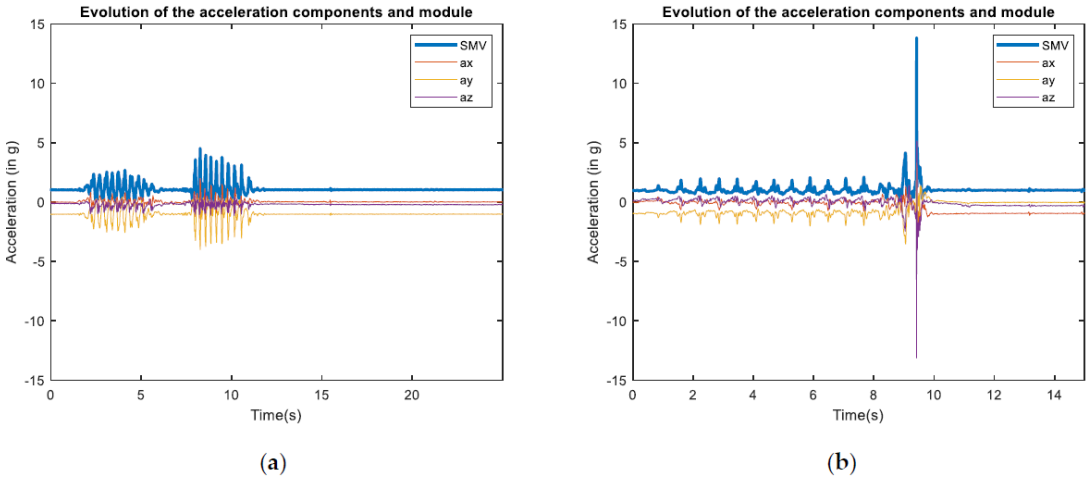


Figura 9 - Progresso dos componentes de aceleração (Ax, Ay, Az). Fonte: (Casilari et al., 2020)

Na Figura 9 os sensores registam o progresso dos componentes de aceleração (Ax, Ay, Az) e magnitude (SMV) para duas amostras no *dataset SisFall* – (a) – Subidas e descidas rápidas em ADL, (b) – queda para a frente durante uma caminhada, causada por um tropeção.

[4] Fonte de dados em: https://www.kaggle.com/code/cclearning/fall-mobiact/notebook

[5] Fonte de dados em: https://www.dropbox.com/s/x2fpfqj0bpf8ep6/UniMiB-SHAR.zip?dl=0

Todas as atividades estão descritas e apresentadas na Tabela 6 - Tipos de atividades diárias *dataset Sisfall*.

A técnica de *Signal Magnitude Vector* (SMV) – descrita em (Casilari et al., 2020), tem como princípio a análise de dados de sensores, com foco nos acelerómetros e giroscópios e aplica-se à monitorização do movimento humano, como seja a deteção de atividade física e à captura de dados do movimento. A característica base do SMV é conseguir concentrar as informações dos diferentes eixos de um sensor num único valor.

O sistema analisou fundamentalmente as características desses conjuntos de dados, seus recursos disponíveis, para a deteção de quedas e métricas de performance, aplicando a arquitetura proposta nos 14 conjuntos de dados. A Figura 10 ilustra uma fase do processo de treino e validação da probabilidade de acerto do modelo proposto.

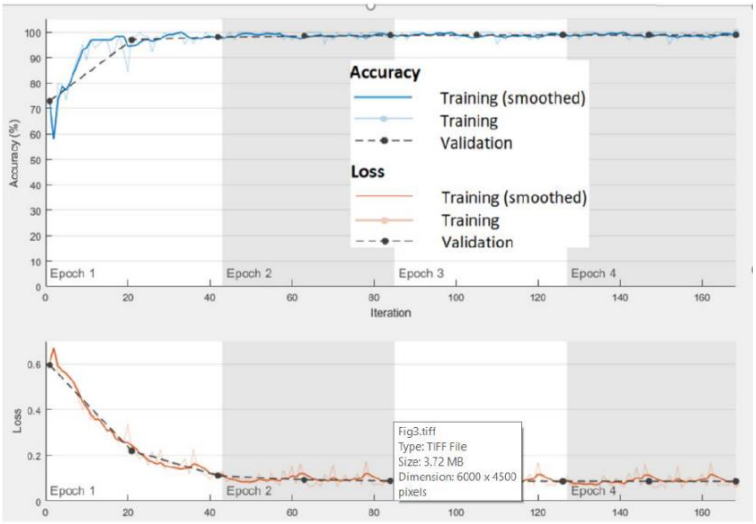


Figura 10 - Processo de treino e validação da probabilidade de acerto. Fonte: (Casilari et al., 2020)

Este método alcançou uma probabilidade de acerto de 99,22%, sensibilidade e especificidade no *dataset SisFall*, de 98,64% e 99,63%, respetivamente.

Para melhorar o desempenho em situações de vida real e diminuir a limitação da capacidade de extrapolação, os autores sugerem que se implemente uma rede CNN mais eficiente ou outra arquitetura, como por exemplo uma LSTM.

Por último, (Q. Zhang & Zhu, 2018) utilizaram telefones móveis para a deteção em tempo real das atividades diárias em humanos.

Os telemóveis fazem parte do nosso quotidiano, pelo que neste caso os autores propuseram um sistema baseado em *deep learning* – CNN, assente num acelerómetro de três eixos que recebe dados em bruto. Foi utilizado um *dataset* da *UniMib* (University of Milano Bicocca Smartphone-based Human Activity Recognition) para o processo de treino da rede, que é feito através da geração de três imagens de 1 dimensão em múltiplos ângulos de visualização, correspondentes às atividades diárias de um humano.

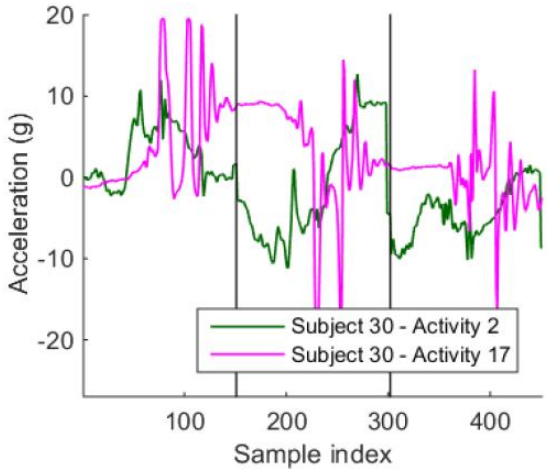


Figura 11 - Exemplo da dificuldade em diferenciar atividades humanas.

Fonte: (Q. Zhang & Zhu, 2018)

O gráfico da Figura 11 é exemplificativo da dificuldade em diferenciar diferentes atividades humanas, mesmo para um indivíduo (**atividade 2**) - Levantar de uma posição deitado, (**atividade 17**) - cair para a esquerda. Três segmentos de cada curva correspondem à aceleração de três dimensões medidas simultaneamente.

As imagens são aplicadas no modelo com base numa CNN, na busca por padrões ocultos. O conjunto de dados utilizado era composto por 11,770 registos de 17 tipos de atividades, recolhidos de pacientes com faixas etárias de largo espectro. O sistema atingiu uma taxa de deteção de 91,5 %, utilizando uma validação cruzada em 5 iterações – *5- fold cross-validation*.

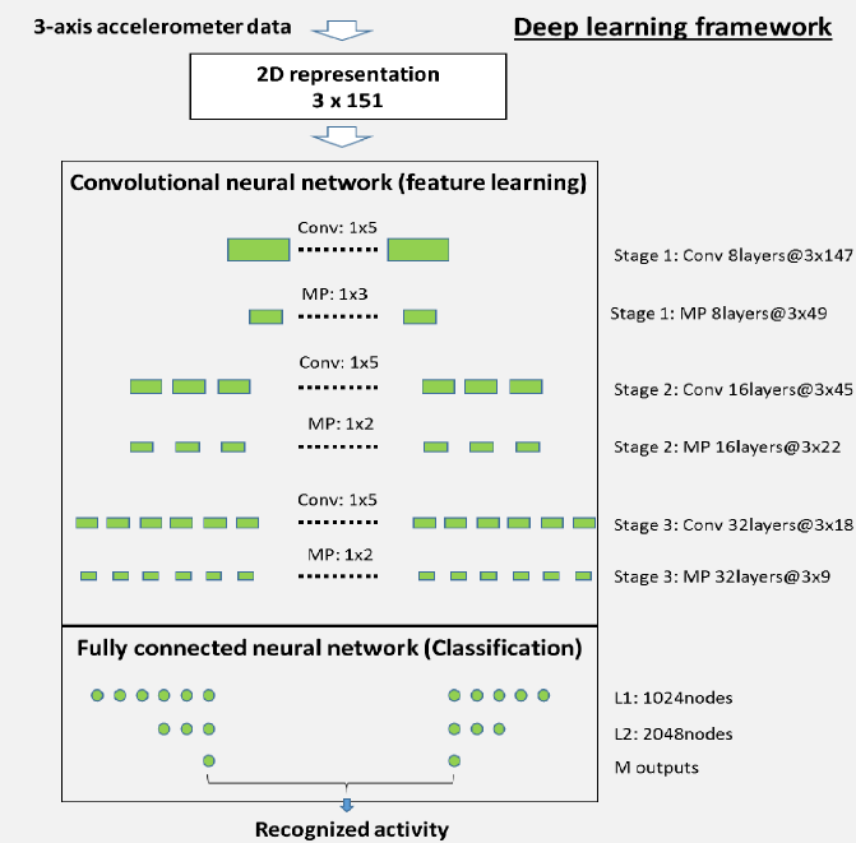


Figura 12 - Vista global do modelo convolucional. Fonte: (Q. Zhang & Zhu, 2018)

A Figura 12 permite ter uma visão global do modelo convolucional – CNN, para aprendizagem e reconhecimento das atividades humanas.

2.3.2.1 - Desempenho dos modelos revistos em CNN

A Tabela 2 resume o desempenho dos modelos revistos, nas dimensões já referidas: *probabilidade de acerto, precisão, sensibilidade e especificidade*:

Tabela 2 - Resumo de desempenho dos modelos revistos em CNN

CNN				
Autores	Probabilidade de acerto	Precisão	Sensibilidade	Especificidade
(Yhdego et al., 2019)	96,43%	95,93%	95,83%	96,875%
(He et al., 2019)	98,61%	N/D	98,62%	99,80%
(Tasoulis et al., 2019)	96,79%	N/D	N/D	N/D
(L. Wang et al., 2019)	98,61%	N/D	97,92%	99,58%
(Casilari et al., 2020)	99,20%	N/D	98,64%	99,63%
(Q. Zhang & Zhu, 2018)	91,50%	N/D	N/D	N/D

Os modelos CNN revistos apresentam uma probabilidade de acerto geralmente alta, variando entre os 91,50% e 99,20%. Isso sugere que os modelos têm uma boa capacidade de prever a deteção de quedas, com base nos *datasets* utilizados em cada uma das propostas. A precisão não foi considerada nos estudos analisados, tendo sido apenas medida numa ocasião, pelo que não é possível fazer uma comparação completa entre os modelos com base nessa métrica. Relativamente à sensibilidade e especificidade, os resultados apurados estão em linha com as métricas apresentadas para a probabilidade de acerto, verificando-se que os modelos têm a capacidade de identificar com precisão tanto as quedas quanto as atividades normais, minimizando assim tanto os falsos positivos quanto os falsos negativos.

2.3.3 - Os sistemas de deteção de quedas baseados em LSTM

Conforme referido no início da exposição, a maioria dos sistemas analisados usa as CNN para o desenvolvimento de sistemas automáticos de deteção de quedas, mas, por outro lado, as redes LSTM podem manter memórias de longo prazo de forma mais eficiente.

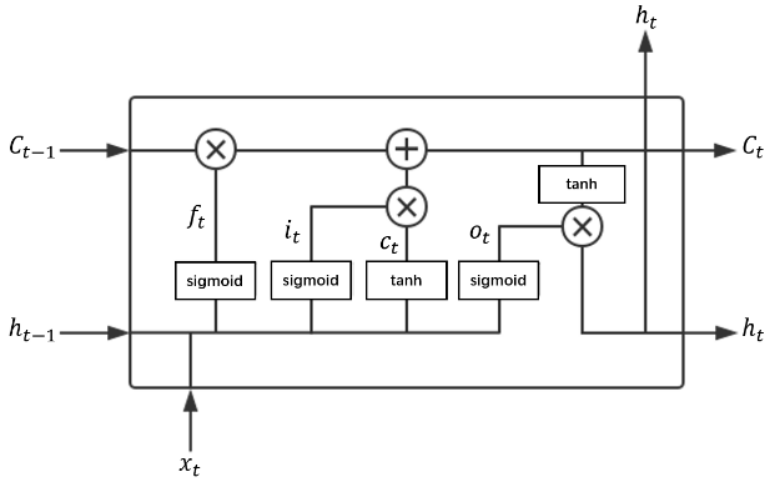


Figura 13 - Estrutura de uma LSTM. Fonte: (Xu et al., 2019)

As RNN desempenham um papel importante no processamento de sinais de voz e séries temporais – já referido acima. No entanto, as RNN têm dificuldade em lidar com os problemas de dependência de longo prazo, o que pode levar ao problema do gradiente – uma vez que este pode explodir nas camadas iniciais de uma rede profunda, com múltiplas *hidden layers* - camadas ocultas. As LSTM resolvem os problemas de dependência de longo prazo, adicionando portões de memória à RNN, melhorando assim os resultados (Xu et al., 2019). A Figura 13 representa um bloco de uma LSTM num momento definido de {t}. A resposta de saída da célula nervosa no momento t é determinada pela entrada nesse momento e pela resposta de saída no momento anterior: A memória pode ou não ser guardada pela rede, dependendo dos dados e do contexto. Os mecanismos de *gating* – portões, das LSTM preservam as dependências de longo prazo. As LSTM podem armazenar ou libertar memória utilizando o mecanismo de *gating* (Veeriah et al., 2015), (Hammerla et al., 2016), conforme referido acima.

Apresentam-se alguns modelos que tiveram como base as LSTM e que foram objeto de revisão.

Um sistema de Internet das coisas – IOT foi proposto por (Xu et al., 2019), tendo sido usada uma *feature* – característica, de extração e aproveitamento das características combinadas da CNN e LSTM para o processamento rápido de séries temporais – *time series*. No caso em específico, o algoritmo desenvolvido recebe dados de aceleração como *inputs* de um acelerómetro de três eixos e de baixo custo. A Figura 14 ilustra o modelo proposto, onde se pode apreciar em particular a secção da rede LSTM, com duas camadas de 32 neurónios, seguida de duas *fully connected* que terminam numa *softmax* para a conversão das classes na camada de saída do modelo de classificação.

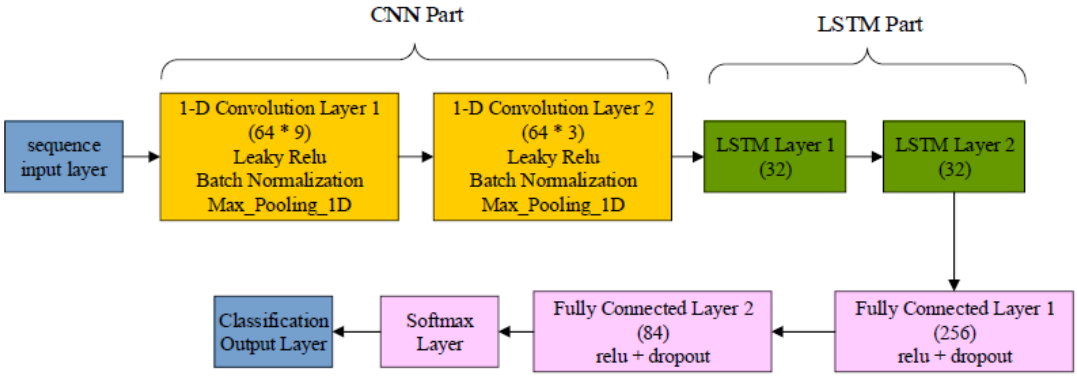


Figura 14 - Vista global da arquitetura de rede. Fonte: (Xu et al., 2019)

O sistema obteve um desempenho melhor do que o método combinado baseado em *Support Vector Machine* (SVM) e CNN. O treino e testes foram realizados sobre um conjunto de dados *MobiAct*, contendo eventos de quedas combinados com outros 6 eventos de atividades diárias. No *dataset* foi efetuada uma separação dos dados com o rácio de 80% - /20% - para treino da rede e validação, respetivamente.

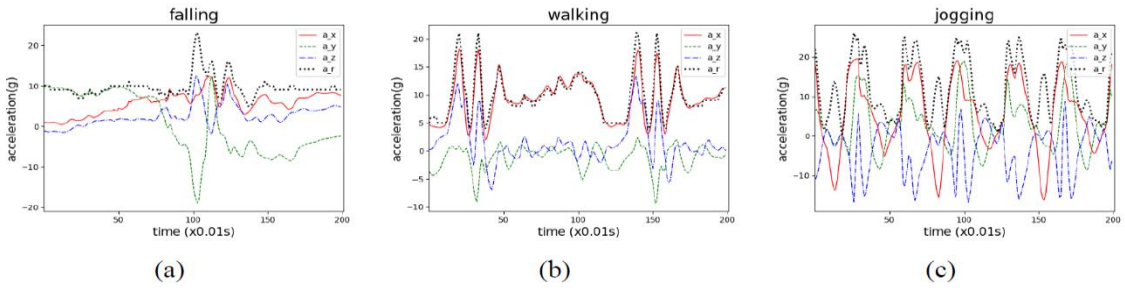


Figura 15 - Representação das curvas de aceleração das ADL's. Fonte: (Xu et al., 2019)

A Figura 15 apresenta a representação das curvas de aceleração das ADL's. (**a**): curva de aceleração de queda, (**b**): curva de aceleração de caminhada, (**c**): curva de aceleração de

corrida. Na figura 16 podemos observar os resultados da *probabilidade de acerto* obtidos no processo de treino para as dimensões – LSTM+CNN, LSTM e CNN.

A *probabilidade de acerto* média da classificação atingiu os 98,98% e a *especificidade* e *precisão* da deteção alcançadas foram de 99,76% e 98,61% respetivamente.

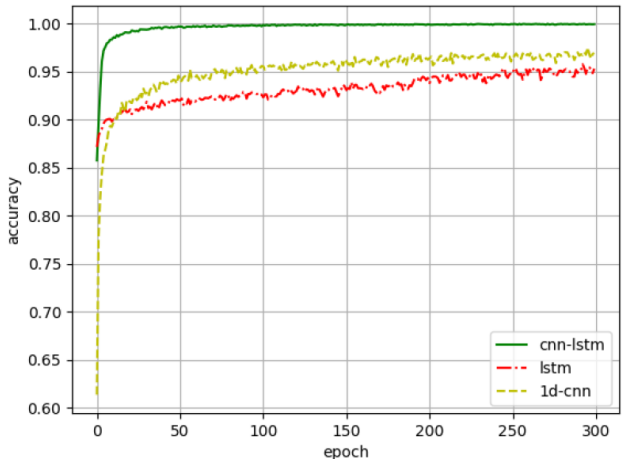


Figura 16 - Representação da curva de probabilidade de acerto no processo de treino. Fonte: (Xu et al., 2019)

Um outro sistema, proposto por (Ajerla et al., 2019) apresenta uma *framework* para a deteção de quedas também baseada em LSTM. A proposta passou por utilizar equipamentos laptop locais para a computação, em vez de enviar os dados em bruto para um sistema baseado em nuvem, onde os algoritmos de predição de quedas seriam executados em tempo real. Na Figura 17 apresenta-se uma visão geral da *framework* proposta, onde se pode verificar o fluxo de recolha de dados, através de sensores da MetaMotionR[6]. Este equipamento, produzido pela MbientLab, lida com os dados em bruto provenientes de um acelerómetro de 3 eixos e um serviço "apache flink", através do qual é executada a análise dos dados transmitidos.

[6] Fonte para o sensor: https://mbientlab.com/store/metamotionr/

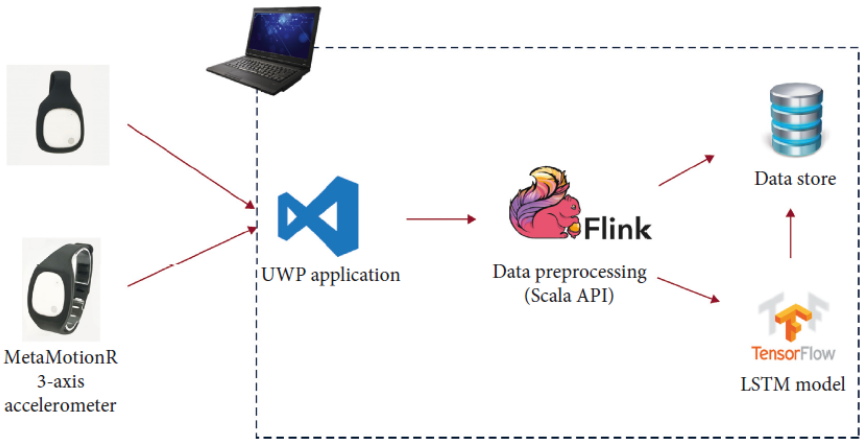


Figura 17 - Framework de *edge computing* para a deteção de falhas. Fonte: (Ajerla et al., 2019)

Importa referir que o MetaMotionR, representado na Figura 18, é um módulo compacto que incorpora uma série de sensores para medir diferentes aspetos do movimento e do ambiente. A MbientLab é conhecida por produzir dispositivos IoT e *wearables*, e o MetaMotionR foi concebido para permitir a monitorização de atividades físicas.

Figura 18 - Sensores MetaMotionR da MBientLab. Fonte: (Ajerla et al., 2019)

O treino e testes da arquitetura foram executados, tendo como base um subconjunto do *dataset* público da *MobiAct*. O sistema concluiu que a cintura seria o melhor lugar para a colocação dos sensores, depois de serem testados em várias posições, conforme ilustrado na Figura 19.

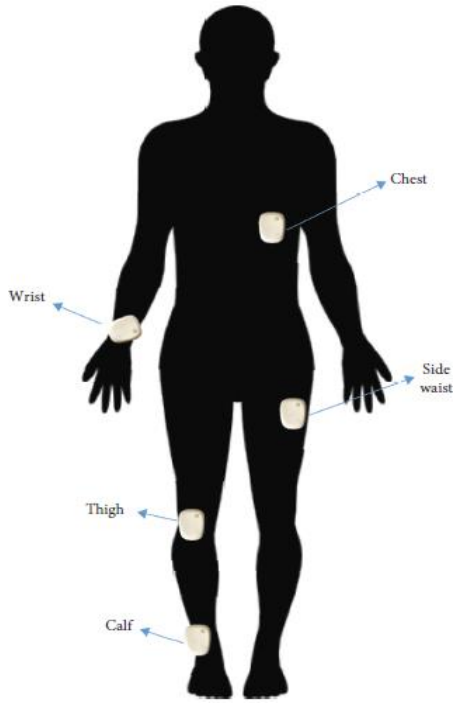


Figura 19 - Posicionamento dos sensores em diversas partes do corpo. Fonte: (Ajerla et al., 2019)

Esta framework consegue prever eventos de quedas recorrendo a dados em tempo real com uma *probabilidade de acerto* de 95,8 %. Concluem os autores deste estudo que o uso de vários sensores e um maior fluxo de dados pode resultar num melhor desempenho geral do sistema de deteção de quedas.

Uma outra arquitetura de deteção de quedas, que utilizou um sistema existente de monitorização de saúde, foi proposta por (Pena Queralta et al., 2019). Estes combinaram um sistema de *edge computing* com *fog computing*, com os dados a serem transmitidos numa rede WAN de baixo consumo (LPWAN), recorrendo a um algoritmo de compressão, conseguindo desta forma reduzir a latência do sistema.

Tendo como base os dados recebidos, duas redes neuronais foram implementadas em conjunto no sistema de *edge computing* - LSTM e RNN, para a deteção de quedas. A arquitetura do sistema é composta pelos sensores que recolhem os dados e se ligam à estrutura de rede local. Conforme se pode acompanhar pela Figura 20, são enviados alertas e notificações a partir destes *edge gateways*, bem como são transmitidos para uma nuvem todos os dados em bruto para análise em tempo real. Este sistema está preparado para funcionar em locais com fraca conectividade e faz uma gestão adequada dos recursos que lhe permite uma maior autonomia das baterias. Foram alcançados níveis médios de probabilidade de acerto, na ordem dos 95,3% e precisão de 90,1% na previsão de eventos de queda, utilizando um dataset da *MobiAct*.

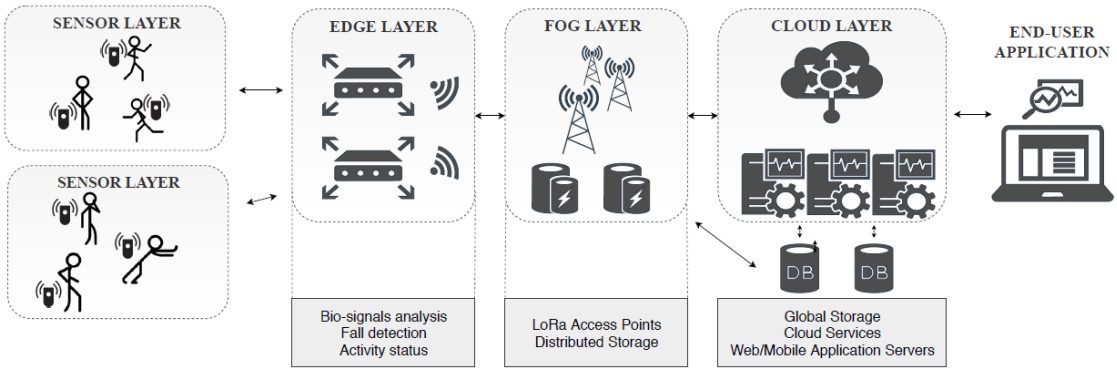


Figura 20 - Arquitetura do sistema proposto. Fonte: (Pena Queralta et al., 2019)

Segundo os autores, a performance global pode ser melhorada, fazendo parte do trabalho futuro o aperfeiçoamento do modelo de previsão e uma análise de desempenho mais exaustiva. Indicam também que irão expandir o modelo para classificar diferentes atividades diárias para além da deteção de quedas.

Por último, apresenta-se um método que foi proposto por (Torti et al., 2018). Este consiste numa implementação de uma arquitetura baseada em RNN em simultâneo com uma LSTM - ambas concorrem para a deteção de quedas e podem ser integradas numa unidade microcontrolador (MCU) com acelerómetros de 3 eixos. A implementação foi executada na plataforma Tensorflow.

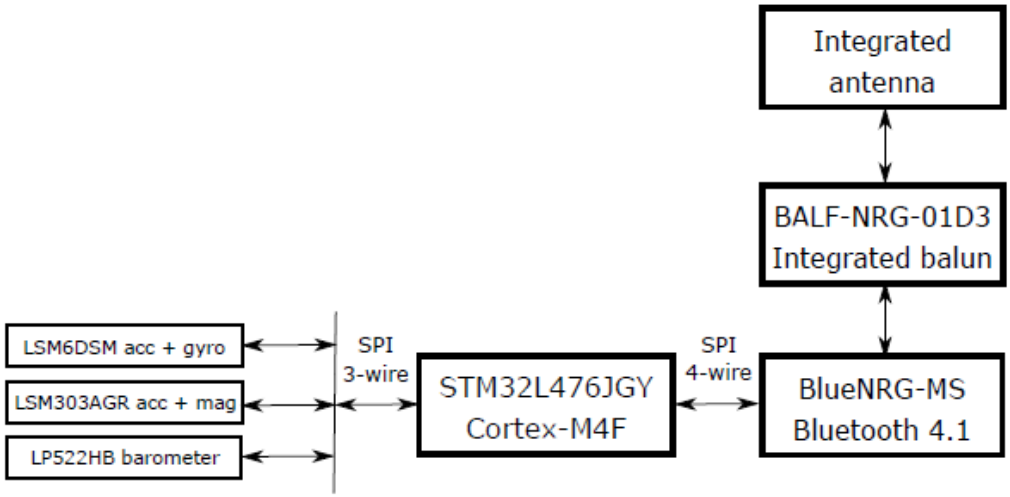


Figura 21 - Arquitetura *SensorTile*. Fonte: (Torti et al., 2018)

O *TensorFlow* [7]é uma ferramenta utilizada para construir, treinar e implementar modelos de *machine learning*. Pode ser usado na classificação bem como no processamento de linguagem natural e visão computacional e permite o aproveitamento do poder de processamento de unidades de processamento gráfico - *Graphics Processor Unit* (GPUs) bem como unidades de processamento tensorial - *Tensor Processing Unit* (TPUs), acelerando significativamente o treino dos modelos.

O modelo treinado, foi testado com um conjunto de dados *Sisfall* devidamente preparado (*annotated* - ficheiro de configurações específicas) e aos dados foi aplicada uma separação para treino e validação com um rácio de 80% - 20% respetivamente.

[7] Fonte: https://www.tensorflow.org/guide/basics?hl=pt-br

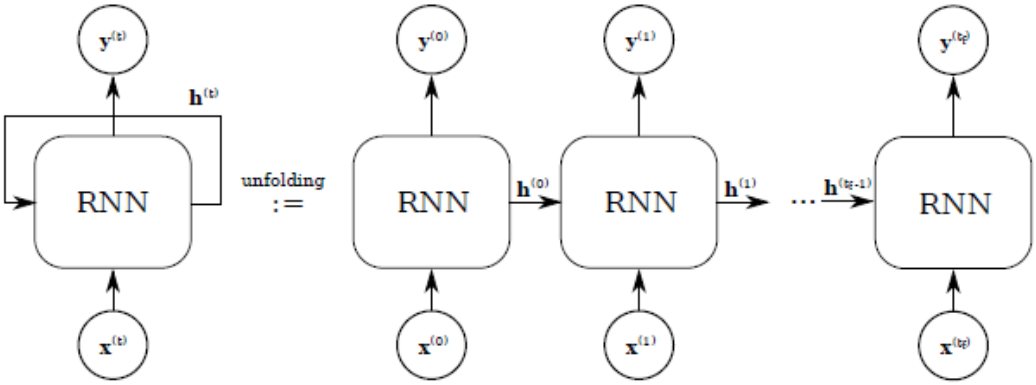


Figura 22 - Uma única célula (à esquerda) é desdobrada temporariamente para treino. Fonte: (Torti et al., 2018)

Na Figura 22 podemos avaliar uma única célula RNN (à esquerda), que é desdobrada temporariamente para treino. A profundidade desse desdobramento é também a profundidade da rede. O modelo proposto foi capaz de atingir uma *probabilidade de acerto* de 98,33% na previsão de eventos de queda bem como valores para a *sensibilidade* e *especificidade* na ordem dos 98,73% e 97,93%, respetivamente. Foi também implementado um módulo de deteção em tempo real num dispositivo *SensorTile*[8], e validado com os resultados obtidos com o *TensorFlow* instalado numa estação de trabalho. A *SensorTile*, criada pela empresa *STMicroelectronics*, é uma plataforma de desenvolvimento compacta que integra vários sensores num único dispositivo, como sejam giroscópio e magnetómetro, facilitando o desenvolvimento de aplicações que requerem a recolha de dados sensoriais em tempo real. Possui conectividade sem fio, como Bluetooth, e está otimizada para atingir baixos consumos de energia.

Ambos desempenharam papéis distintos, com a *SensorTile* focada na aquisição de dados do mundo real e o *TensorFlow* na construção e implementação do modelo de *machine learning*. Uma vez que o dispositivo *SensorTile* opera em regime de baixo consumo energético, consegue alcançar níveis de duração que chegam a cerca de 20 horas até nova carga.

[8] Fonte: https://www.st.com/en/evaluation-tools/steval-stlkt01v1.html#overview

2.3.3.1 - Desempenho dos modelos revistos em LSTM

A tabela abaixo, resume o desempenho dos modelos revistos, nas dimensões já referidas: *probabilidade de acerto, precisão, sensibilidade e especificidade*:

Tabela 3 - Resumo de desempenho dos modelos revistos em LSTM

LSTM				
Autores	Probabilidade de acerto	Precisão	Sensibilidade	Especificidade
(Xu et al., 2019)	98,98%	98,61%	N/D	99,76%
(Ajerla et al., 2019)	95,80%	N/D	N/D	N/D
(Pena Queralta et al., 2019)	95,30%	90,10%	N/D	N/D
(Torti et al., 2018)	98,33%	N/D	98,73%	97,93%

Os modelos LSTM revistos apresentam complexidade na sua implementação, integrando múltiplas arquiteturas para a construção dos modelos. Utilizam computação em nuvem para permitir uma reação célere, *fog computing* para atenuar os desafios da fraca conectividade e sistemas centralizados para a constante melhoria das redes LSTM.

À semelhança dos modelos baseados nas redes neuronais convolucionais, os resultados apurados relativamente à probabilidade de acerto são globalmente altos, sempre acima dos 95%, indicando que os modelos têm uma boa capacidade de prever a deteção de quedas. No que à sensibilidade e especificidade dos modelos revistos diz respeito, somente alguns dos trabalhos apresentam essas métricas. No entanto, pode afirmar-se que esses modelos classificam acima de 97%, indicando que identificam corretamente os eventos de quedas bem como as atividades normais, minimizando tanto os falsos positivos quanto os falsos negativos

Por fim, a precisão das duas propostas que apresentam resultados sugere que os modelos têm uma baixa prevalência de falsos positivos em relação ao total de positivos previstos, se bem que um deles apresenta uma taxa de precisão na ordem dos 98,61%, havendo espaço para melhorar a taxa de falsos positivos na proposta de (Pena Queralta et al., 2019)

2.3.4 - Sumário dos sistemas desenvolvidos para deteção de quedas

Como resumo de todos os sistemas desenvolvidos para deteção de quedas, apresenta-se abaixo a tabela contendo uma sumarização dos modelos apreciados.

Tabela 4 - Resumo dos modelos revistos em CNN e LSTM

Autores	Ano	Categoria	Dispositivo de captura	Método	Tipo deteção	Dataset utilizado
(Yhdego et al., 2019)	2019	Sensor	Acelerómetro	CNN	ADL e Queda	URFD
(He et al., 2019)	2019	Sensor	Acelerómetro 3 eixos, Giroscópio	FD-CNN	Queda	SisFall, MobiFall
(Tasoulis et al., 2019)	2019	Sensor	Acelerómetro	CNN, CUSUM	Queda	SmartFall
(L. Wang et al., 2019)	2019	Sensor	Acelerómetro 3 eixos, Giroscópio	CNN	ADL e Queda	Voluntários que usam palmilha, pelo que dados personalizados
(Casilari et al., 2020)	2020	Sensor	Acelerómetro de 3 eixos portátil	CNN	Queda	MobiAct, Sisfall, MobiFall, UniMiB, SHAR, UP-Fall, entre outros
(Q. Zhang & Zhu, 2018)	2019	Sensor	Acelerómetro 3 eixos, com dados brutos	CNN	ADL e Queda	UniMiB SHAR
(Xu et al., 2019)	2019	Sensor	Acelerómetro 3 eixos	LSTM	ADL e Queda	MobiAct
(Ajerla et al., 2019)	2019	Sensor	Acelerómetro 3 eixos	LSTM	Queda	MobiAct
(Pena Queralta et al., 2019)	2019	Sensor	MPU9250 com Acelerómetro de 3 eixos, Giroscópio de 3 eixos, Magnetómetro 3 eixos	RNN, LSTM	ADL e Queda	MobiAct
(Torti et al., 2018)	2018	Sensor	Acelerómetro 3 eixos	RNN, LSTM	Queda	SisFall

3 - Configuração experimental

Nesta secção do documento pretende-se fazer a implementação de uma rede baseada numa função de perda ou função de custo (Rangamani et al., 2018). Foi tomada como ponto de partida, a arquitetura da rede neuronal presente no artigo (Torti et al., 2019). Nesse artigo, é tida uma abordagem que adota dispositivos *wearable* para deteção de quedas em tempo real.

O foco desta tese de mestrado centra-se no artigo (Torti et al., 2019) e num segundo (Musci et al., 2018), onde se detalha todo o processo do primeiro. Pretende-se executar um estudo da rede LSTM descrita, introduzindo alterações no modelo de dados utilizado em (Torti et al., 2019). Implementa-se num ambiente Matlab, considerando a utilização de hiper parâmetros distintos para a afinação e desempenho global da rede LSTM. Daqui sairão resultados que se compararão com os referenciais mencionados.

Detalhando, (Torti et al., 2019) propõem a incorporação de uma arquitetura de RNN numa unidade microcontrolador (MCU) alimentada por dados de acelerómetros de três eixos, que por sua vez são alimentados pelos sensores acoplados. Para avaliar a viabilidade dessa abordagem de *deep learning* - com recursos limitados, o artigo de referência apresenta algumas fórmulas gerais para determinar a ocupação de memória, a carga computacional e o consumo de energia. A alimentação do modelo é feita através da utilização de um *dataset Sisfall* anotado[9] (Sisfall Temporally Annotated, n.d.).

Segundo os autores (Torti et al., 2019), entre os vários *dataset* disponíveis para o treino e validação do sistema de deteção de quedas (FDS - *Fall detection System*), foi escolhido o *Sisfall*, uma vez que se provou ser o mais adequado para o pretendido, introduzindo a capacidade de classificação em três dimensões – atividades diárias, alertas e quedas.

O treino e validação do modelo foi alvo de uma seleção precisa, não incluindo todos os registos pré-existentes no dataset original - contrariando diametralmente a abordagem desta tese, onde se considerou para implementação a totalidade dos eventos.

O *dataset Sisfall* agrega uma miríade de dados, resultantes da simulação de ADL's bem como uma heterogeneidade no que se refere à faixa etária dos indivíduos envolvidos.

[9] Fonte para dataset: https://bitbucket.org/unipv_cvmlab/sisfalltemporallyannotated/

Estão incluídas um total de 38 entidades distintas: 23 indivíduos jovens e 15 idosos, que ao todo simulam 34 atividades diferentes num contexto controlado - 19 ADLs e 15 quedas, com até 5 tentativas na maioria dos casos, totalizando 4502 registos completos. A técnica para aquisição desses dados foi possível graças à fixação de um dispositivo no corpo dos indivíduos, simulando dessa forma a fivela de um cinto. Esse dispositivo está equipado com dois acelerómetros diferentes e um giroscópio, registando os dados a uma frequência de 200 Hz - importa referir que o dispositivo não foi concebido para registar dados em tempo real.

Uma das particularidades do *dataset* de quedas é que os dados recolhidos estão no seu estado puro, não havendo qualquer destrinça no que se refere aos intervalos temporais associados aos eventos relevantes de interesse, nem tão pouco existindo uma anotação específica que inclua a sua identificação temporal.

Em (Musci et al., 2018), foi desenvolvida e disponibilizada uma versão modificada do *Sisfall*, contendo um sistema de anotação para a totalidade do *dataset*, onde para cada ficheiro original, se criou um concomitante, com a anotação correspondente em valor de 0, 1 ou 2 – conforme a circunstância desse momento. Essa anotação foi denominada de "*Sisfall Temporally Annotated*" e constitui-se como sendo uma melhoria do conjunto de dados *SisFall*, conforme descrito em (Musci et al., 2018).

3.1 - Estrutura do Dataset

Tal como indicado na secção anterior, o sistema denominado "Anotação Temporal" tem uma estrutura idêntica à do conjunto de dados original do *Sisfall*: cada estrutura representa um indivíduo distinto. Dentro de cada uma destas estruturas encontram-se os ficheiros correspondentes às atividades. Cada um destes ficheiros tem o mesmo número de linhas que os ficheiros originais e uma única coluna.

O valor é da coluna é:

"0" para BKG

"1" para ALERTA

"2" para FALL

Detalhando o significado de cada valor, foram definidas duas classes principais de eventos: quedas - FALL e alertas - ALERT, sendo estes últimos (os ALERT) de interesse, uma vez que podem, ou não, conduzir diretamente a uma queda efetiva.

O também denominado *Sisfall Enhanced*, define uma terceira classe de background, designada de BKG, que comporta todas as atividades que não estão relacionadas com quedas ou alertas – sejam elas tais como andar, saltar, subir escadas, sentar-se numa cadeira, etc.

Todos os pormenores sobre esta estratégia de rotulagem são discutidos em (Musci et al., 2018), e conduzem a uma abordagem mais precisa no treino e validação para uma rede RNN (Torti et al., 2019).

3.2 - Arquitetura da rede de referência

Como referido em (Musci et al., 2018), os autores propõem uma arquitetura de rede relativamente simples, adaptada de (Guillaume Chevalier, 2016). O núcleo da arquitetura de *deep learning* está representado na figura abaixo. A rede de referência é baseada em duas células LSTM.

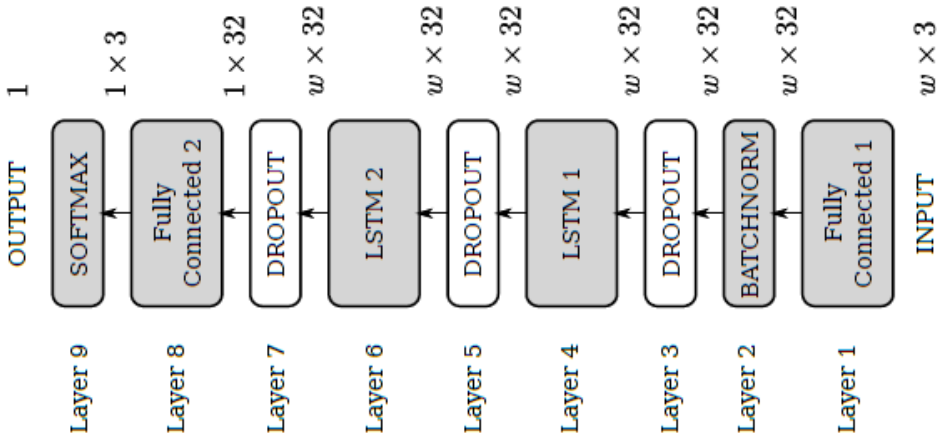


Figura 23 - Modelo de referência para implementação.

Fonte: (Musci et al., 2018)

Observa-se na Figura 23, a arquitetura do modelo da solução proposta em (Musci et al., 2018) baseada em (Guillaume Chevalier, 2016). O tamanho da entrada depende do número de características selecionadas (no caso de referência são utilizados 3 eixos do acelerómetro) com janelas de tamanho ***w*** a alimentar o modelo - conforme o modelo deste trabalho. O tamanho da saída depende do número de classes (ou seja, 3 - BKG, ALERT e FALL).

Importa referir que do modelo seguido se excluem algumas camadas, uma vez que não se justificam, na medida em que não se executa a implementação em hardware – pelo que o modelo se resume conforme a Figura 24 - os blocos representados com fundo branco estão ativos apenas durante a fase de treino – no modelo a treinar não serão considerados.

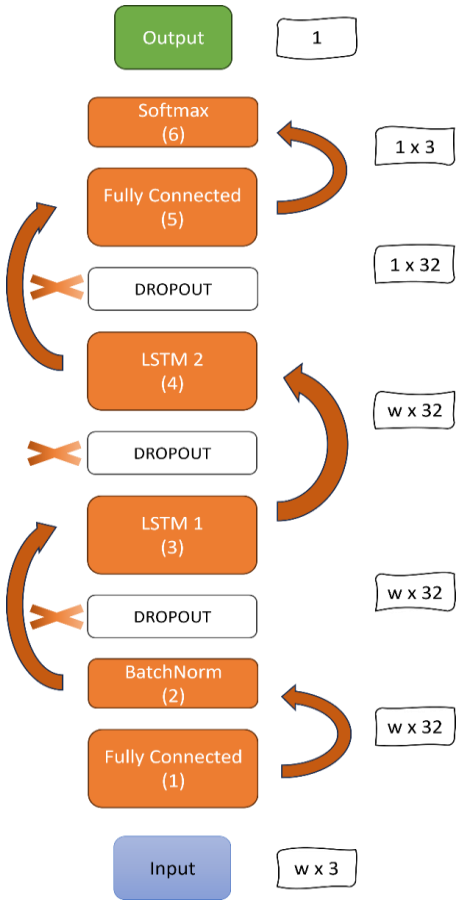


Figura 24 - Representação do modelo a treinar, com exclusão dos dropout.

A Figura 24 descreve as atividades da rede - as duas células LSTM estão ligadas em série (camadas 3 e 4), cada célula tem uma dimensão interna de 32 unidades e o pré-processamento é feito na camada 1 através de uma FC, desencadeando uma segunda FC – na camada 5, que recolhe a saída da célula LSTM na camada 4, que por sua vez alimenta uma camada final *SoftMax* (Nwankpa et al., 2018) (camada 6) que fornece a classificação das três classes já descritas. Para um maior detalhe pode ser consultada a Figura 30 - Esquema da rede proposta, com o detalhe para cada uma das fases.

Em relação à estrutura de rede proposta em (Guillaume Chevalier, 2016), a solução adotada neste trabalho inclui uma camada de normalização BATCHNORM – (Ioffe & Szegedy, 2015) (Camada 2) para regularizar os dados de entrada, normalizando as ativações em *mini-batches* durante o treino da rede, melhorando a estabilidade, a velocidade de treino e o desempenho geral do modelo.

3.3 - Treino da rede de referência

Utilizando o desdobramento temporal, a entrada é processada em subsequências, denominada de "janelas", com tamanho predefinido ***w***. Cada célula LSTM é colocada em cascata em exatamente ***w*** cópias de si mesma, e cada cópia recebe a saída da célula anterior.

O treino da rede foi realizado como descrito em (Torti et al., 2019) – como já referido, tendo como base os seguintes dados vindos do *dataset Sisfall* - foi adotada uma divisão típica com rácio para treino e teste da rede de 80% e 20% respetivamente: o conjunto de dados de treino incluiu 30 indivíduos (12 idosos), enquanto o conjunto de dados do teste incluiu 8 (3 idosos). Para este treino os autores referem ter tomado especial cuidado em evitar o chamado enviesamento de identidade – *identity bias*: as pessoas presentes no conjunto de treino não estão presentes no conjunto de testes e vice-versa – contrariando a opção técnica adotada por esta tese, conforme se apresentará na secção seguinte.

Tabela 5 - Resultados obtidos em (Torti et al., 2019).

	(Torti et al., 2019)	*Sisfall*
Sensitivity		
BKG	88,39	75,01
ALERT	91,08	68,15
FALL	98,73	75,79
Specificity		
BKG	97,85	92,52
ALERT	90,77	83,3
FALL	97,93	91,57
Accuracy		
BKG	93,12	83,77
ALERT	90,93	75,73
FALL	98,33	83,68

A Tabela 5 resume os resultados apurados com a abordagem tida nos dois trabalhos acima mencionados, referindo-se à validação experimental numa estação de trabalho. O quadro apresenta a *sensibilidade*, *especificidade* e *probabilidade de acerto* no conjunto de dados de teste para esta proposta, em comparação com o modelo base do *Sisfall*.

3.4 - A proposta baseada em Matlab

Conforme já referido, foi com base nos artigos já mencionados, que se construiu um modelo de inteligência artificial (IA) para a identificação e predição em situações de queda ou de iminente perigo. O processo, sua construção e os módulos que o compõem, serão detalhados abaixo. Esta abordagem obrigou a um pré-processamento do *Sisfall*, uma vez que para os 4502 registos havia alguns em falta, nomeadamente simulações dispersas entre alguns voluntários – o que potenciava enviesamento na recolha dos dados do modelo e a construção dos ficheiros que alimentam a rede.

3.4.1 - Enquadramento e pré-processamento

Em primeiro lugar - para o modelo, o passo inicial consistiu na criação de um ficheiro anotado (Matos-Carvalho et al., 2023) para a elaboração dos ficheiros que vão alimentar o processo de treino da rede. Os dados a aplicar têm origem no *dataset SisFall* já referido.

Em súmula, este conjunto de dados incorpora uma vasta gama de amostras simuladas em ambiente controlado - relativamente ao tamanho da amostragem e idade dos voluntários (38 participantes, incluindo 19 homens e 19 mulheres entre os 19 e 75 anos). Estão consideradas um conjunto de simulações com uma duração considerável para alimentar o modelo – cerca de 4.502 amostras entre 10 e 180 segundos, compreendendo 2.702 atividades da vida diária e 1.800 quedas, e uma diversidade de movimentos emulados: 19 classes de ADLs, incluindo atividades básicas e desportivas e 15 classes de quedas.

As atividades diárias têm códigos únicos que vão desde o D01 até D19. Estas são executadas por um período definido e em cada caso, ou são repetidas uma, ou cinco vezes. A Tabela 6 reflete o conjunto das atividades diárias executadas, seus códigos, tentativas e duração:

Tabela 6 - Tipos de atividades diárias *dataset Sisfall*, Fonte: (Sucerquia et al., 2017)

Código	Atividade	Tentativas	Duração
D01	Caminhada lenta	1	100 s
D02	Caminhada rápida	1	100 s
D03	Corrida lenta	1	100 s
D04	Corrida rápida	1	100 s
D05	Caminhada devagar a subir e a descer escadas	5	25 s
D06	Caminhada rápida a subir e a descer escadas	5	25 s
D07	Sentar-se devagar numa cadeira baixa, esperar um instante e levantar-se lentamente	5	12 s
D08	Sentar-se rapidamente numa cadeira baixa, esperar um instante e levantar-se rapidamente	5	12 s
D09	Sentar-se lentamente numa cadeira baixa, esperar um instante e levantar-se lentamente	5	12 s
D10	Sentar-se rapidamente numa cadeira baixa, esperar um instante e levantar-se rapidamente	5	12 s
D11	Sentar-se por um instante, tentar levantar-se e cair de volta na cadeira	5	12 s
D12	Sentar-se por um instante, deitar-se lentamente, esperar um momento e sentar-se novamente	5	12 s
D13	Sentar-se por um instante, deitar-se rapidamente, esperar um momento e sentar-se novamente	5	12 s
D14	Estar deitado de costas, mudar para a posição lateral, esperar um momento e mudar para de costas	5	12 s
D15	Em pé, dobrar lentamente os joelhos e levantar-se	5	12 s
D16	Em pé, dobrar-se lentamente sem dobrar os joelhos e levantar-se	5	12 s
D17	Em pé, entrar em um carro, permanecer sentado e sair do carro	5	25 s
D18	Tropeçar enquanto caminha	5	12 s
D19	Saltar suavemente sem cair (tentando alcançar um objeto alto)	5	12 s

À semelhança das ADL, são também classificadas as FALL, que têm uma formulação que vai desde F01 a F15. A duração das quedas é uniforme e de 15 segundos, com 5 tentativas para cada atividade desta natureza. A Tabela 7 reflete o conjunto das atividades de queda executadas, seus códigos, tentativas e duração:

Tabela 7 - Tipos de quedas *dataset Sisfall*, Fonte: (Sucerquia et al., 2017)

Código	Atividade	Tentativas	Duração
F01	Queda para a frente enquanto caminha causada por escorregão	5	15 s
F02	Queda para trás enquanto caminha causada por escorregão	5	15 s
F03	Queda lateral enquanto caminha causada por escorregão	5	15 s
F04	Queda para a frente enquanto caminha causada por tropeço	5	15 s
F05	Queda para a frente enquanto corre causada por tropeço	5	15 s
F06	Queda vertical enquanto caminha causada por desmaio	5	15 s
F07	Queda enquanto caminha, com uso das mãos em uma mesa para amortecer a queda, causada por desmaio	5	15 s
F08	Queda para a frente ao tentar levantar-se	5	15 s
F09	Queda lateral ao tentar levantar-se	5	15 s
F10	Queda para a frente ao tentar sentar-se	5	15 s
F11	Queda para trás ao tentar sentar-se	5	15 s
F12	Queda lateral ao tentar sentar-se	5	15 s
F13	Queda para a frente enquanto sentado, causada por desmaio ou adormecimento	5	15 s
F14	Queda para trás enquanto sentado, causada por desmaio ou adormecimento	5	15 s
F15	Queda lateral enquanto sentado, causada por desmaio ou adormecimento	5	15 s

Tanto as ADL como as FALL, existem porque se referem a atividades de indivíduos, que se apresentam com seguintes iniciais: SA e SE, onde SA, representando um conjunto de 23 adultos entre os 19 e os 30 anos e SE, representando um conjunto de 15 seniores entre os 60 e os 75 anos.

Tabela 8 - Idade, altura e peso dos participantes, Fonte: (Sucerquia et al., 2017)

	Sexo	Idade	Altura(m)	Peso (kg)
Idosos SE	Feminino	62–75	1.50–1.69	50–72
	Masculino	60–71	1.63–1.71	56–102
Adultos SA	Feminino	19–30	1.49–1.69	42–63
	Masculino	19–30	1.65–1.83	58–81

3.4.2 - Nomenclatura

Para entendermos a nomenclatura dos ficheiros, concatena-se o código de atividade, seguido do ID do indivíduo e por fim sequenciam-se as tentativas, pelo que:

<ADL OR FALL_CODE>_<SUBJECT_ID>_<TRIAL_NO>.txt

D01_SA01_R02.txt, significa tratar-se da tarefa de ADL#1 – "*Walking slowly*", pelo adulto #1, executando a tentativa número 2. Todos os adultos executam as atividades de ADL e FALL, mas os seniores apenas praticam as atividades ADL, com exceção para o SE06 que tem uma condição física que lhe permite executar todas as simulações. Ainda que executem as simulações de ADL, os seniores não executam as atividades D06, D13, D18 e D19, por recomendação médica. Para além destas condições, não sendo especificado, um conjunto de seniores não executaram algumas atividades por impedimento médico.

3.4.3 - Sensores utilizados

Os dados do *dataset sisfall* foram recolhidos através de três sensores – dois acelerómetros e um giroscópio, com uma frequência de amostragem de 200 Hz. Cada ficheiro contém nove colunas e um número diferente de linhas, dependendo do tamanho da simulação e do tempo que esta demora. A primeira coluna corresponde aos dados de aceleração no eixo dos X, medido pelo sensor ADXL345. A segunda coluna corresponde aos dados de aceleração no eixo dos Y, medido pelo sensor ADXL345. A terceira coluna corresponde aos dados de aceleração no eixo dos Z, medido também pelo sensor ADXL345. As restantes colunas referem-se aos sensores ITG3200 e MMA8451Q, que registam respetivamente os dados de aceleração. Estes dois sensores não são objeto de análise, sendo o foco nas três primeiras colunas e tudo o que envolva os seu cálculos. O sensor ADXL345 tem uma resolução de13 bits e amplitude de +-16g. Para a conversão dos dados de aceleração (AD) que estão em bits para gravidade, considerou-se o cálculo através da equação 5:

$$Aceleração[g]: \left[\left(\frac{2*amplitude}{2^{resolução}}\right)\right] * AD \qquad (5)$$

3.4.4 - Limpeza de dados

O conjunto de dados Sisfall original contém diversas incongruências. Para garantir a qualidade de dados foi necessário proceder a uma análise prévia e correspondente correção sempre que se justificasse.

Uma primeira situação, verificada no terceiro tipo de queda (queda lateral enquanto caminha causada por escorregão) do indivíduo número 13, na quarta tentativa (F03_SA13_R04), onde se pode ver na Figura 25, o pico inicial numa das variáveis, que por erro de leitura do sensor, provocou um pico momentâneo, que poderia enviesar o momento de análise – uma vez no processo de recolha de dados, é criada uma janela ***w*** em função do ponto máximo absoluto. A partir desse ponto, é extraída uma janela ***w*** de valores à esquerda e à direita para análise adicional relacionada com eventos de queda ou alerta.

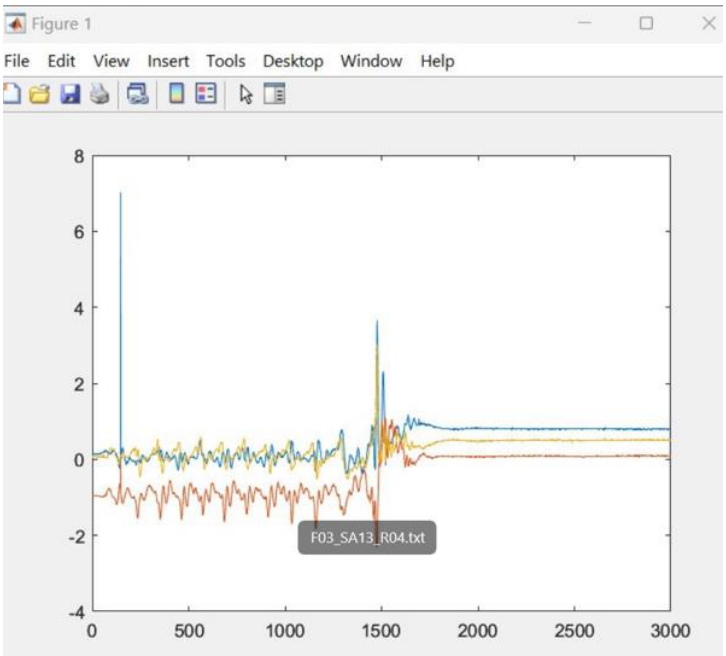


Figura 25 - Pico inicial, por erro no leitor numa queda do tipo F03.

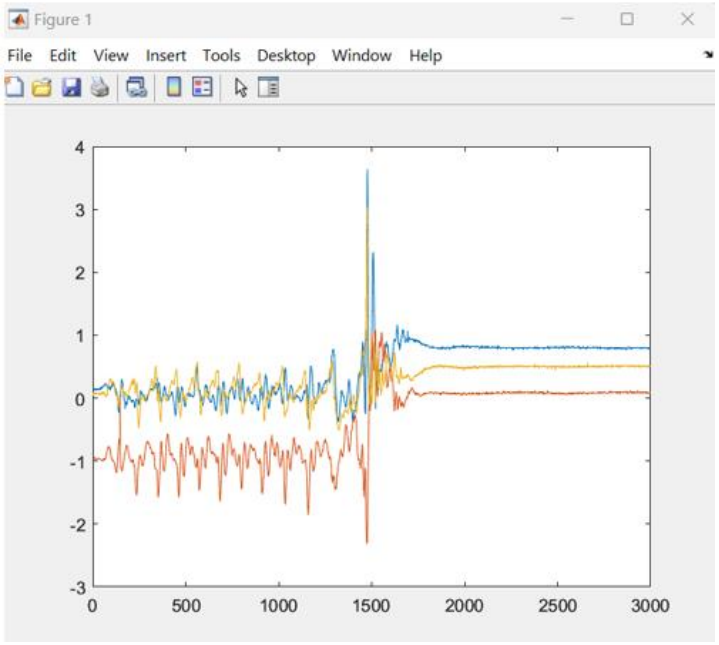


Figura 26 - Queda F03 já corrigido, consistente com a representação esperada.

Na Figura 26 observa-se a situação já corrigida, consistente com a representação esperada. O mesmo problema sucedeu com uma queda para trás enquanto caminha causada por escorregão, do indivíduo número 2, na quinta tentativa (F02_SA02_R05) e foi devidamente corrigida, como se pode ver no conjunto das Figura 27 e Figura 28.

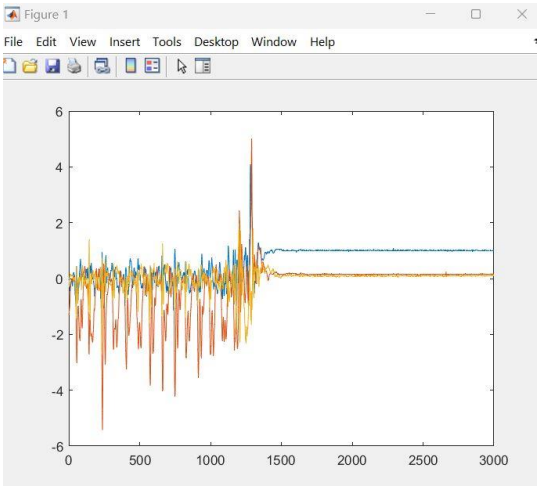


Figura 27 - Queda para trás enquanto caminha causada por escorregão.

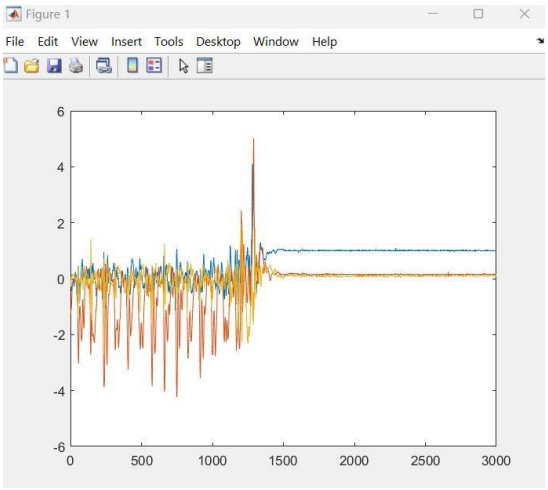


Figura 28 - F02 com os valores de aceleração corrigidos.

Importa relembrar que para as figuras apresentadas, a lista completa de quedas pode ser consultada na Tabela 7 - Tipos de quedas.

Para além destas situações que afetariam a qualidade de dados, foi necessário adicionar alguns ficheiros ao *dataset*, uma vez que estavam em falta no original. A lista pode ser observada na Tabela 9.

Sobre estas adições, convém referir que os ficheiros estavam igualmente em falta no *dataset enhanced*, o que indicia que quando este foi criado, o problema já existiria no *dataset Sisfall*. Não obstante essa situação se ter verificado, é importante referir que os ficheiros em falta estão todos relacionados com atividades diárias e nenhuma se enquadra numa queda. Essas atividades passam pela corrida rápida - D04, caminhada rápida a subir e a descer escadas - D06, sentar-se devagar numa cadeira baixa, esperar um instante e levantar-se lentamente - D07 e em pé, entrar em um carro, permanecer sentado e sair do carro - D17. Só duas das atividades - D04 e D07 são executadas por indivíduos idosos, sendo as restantes ações tidas por adultos cuja faixa etária permanece entre os 19 e 30 anos. Para cada atividade em falta e correspondente tentativa, foi utilizada a mesma atividade, preferencialmente pelo mesmo indivíduo numa das restantes tentativas ou, caso não tenha sido possível, pelo indivíduo imediatamente acima ou abaixo

A metodologia utilizada foi a seguinte: para cada atividade substituída no *Sisfall dataset*, o correspondente foi considerado para o *sisfall enhanced*.

Tabela 9 - Correções no *dataset Sisfall*

Sisfall e Sisfall enhanced em falta		**Descrição de origem**
D04_SE15_R01	←	D04_SE14_R01
D06_SA17_R01	←	D06_SA17_R02
D06_SA20_R02	←	D06_SA20_R01
D07_SE14_R04	←	D07_SE14_R03
D17_SA15_R01	←	D16_SA15_R05
D17_SA15_R02	←	D16_SA15_R05
D17_SA15_R03	←	D16_SA15_R05
D17_SA15_R04	←	D16_SA15_R05
D17_SA15_R05	←	D16_SA15_R05
D17_SA20_R02	←	D17_SA20_R01
F01_SA20_R03	←	F01_SA20_R02
F10_SA20_R04	←	F10_SA20_R03

3.4.5 - Ficheiro de anotação

Para a construção dos ficheiros de treino da rede, foi necessária a criação de um ficheiro de anotação, com origem em (Matos-Carvalho et al., 2023), adaptado ao contexto deste trabalho e parametrizado para contemplar as alterações constantes nos trabalhos de referência, alvos de analise. As alterações incidiram na personalização do ficheiro de anotação, permitindo enquadrar o cenário proposto de classificação do *sisfall enhanced* e, em paralelo, manter todas as funcionalidades já existentes.

Nesta fase foram replicados os cenários relevantes, que por sua vez irão alimentar os processos de treino da rede e por conseguinte permitir uma apreciação e teste através de matrizes de confusão, considerando cada uma das opções. A otimização da rede será tanto maior quanto menores forem os dados de entrada e melhor o desempenho da mesma.

Essas afinações têm lugar no processo de criação dos ficheiros que derivam do ficheiro de anotação, e mediante diversas combinações de cenários – 30 por cada segundo considerado, cerca de 150 no total, por cada dimensão de rede, conforme apresentado na Tabela 13 - Janelas estudadas e dimensões da rede consideradas. Nesse sentido, para replicar os resultados dos trabalhos de referência, foram consideradas diferentes opções com o objetivo de melhorar o desempenho do algoritmo. Algumas dessas opções passaram por alterar a taxa de amostragem dos dados, remoção de frequências indesejadas dos sinais de entrada bem como a sua normalização.

Foram testados diferentes intervalos de tempo – ***dt***, como forma de afinar o algoritmo relativamente à taxa de amostragem dos dados que podem - ou não, combinar com outros

fatores, mediante o seu desempenho. Esse processo é afinado tendo como base a frequência de 200 Hz e um intervalo inicial de 2 segundos – 400 pontos base – conforme se pode verificar pela Tabela 12 – Exemplo considerando um ***dt*** 2 e conjunto de frequências consideradas. Desta relação nascem múltiplas combinações uma vez que se consideram as frequências e a duração, até se obter uma afinação ótima do modelo - com menos pontos de análise, independentemente do tempo considerado.

Considerando os intervalos de tempo de **1,5, 2, 2,5, 3** e **4** segundos, conjugados com os restantes hiper parâmetros combinados no ficheiro de anotação, obtêm-se um conjunto de 150 unidades de ficheiro gerados pelo resultado da configuração, que servem o propósito de alimentar o treino subsequente da rede.

Na Tabela 12 pode ver-se uma representação de uma das combinações possíveis de hiper parâmetros que foram testadas para a afinação do modelo. No caso considerou-se uma janela de 2 segundos e para esse período foram testadas as 5 frequências definidas para analise. Para os 2 segundos e 200 Hz pode verificar-se a existência de 400 pontos por cada atividade. No caso de se considerar uma diminuição na frequência para metade, a análise passa a ter 200 pontos.

Tabela 10 - Janelas de tempo consideradas

Intervalo (dt) – em segundos				
1,5 s	**2 s**	**2,5 s**	**3 s**	**4 s**

Tabela 11 - Frequências e downsample aplicado aos dados

Frequência (fs) – em Hz				
1	**2**	**4**	**8**	**10**
200 Hz	**100 Hz**	**50 Hz**	**25 Hz**	**20 Hz**

Tabela 12 – Exemplo considerando um ***dt*** 2 e conjunto de frequências consideradas

Intervalo (dt) – de 2 segundos: e downsample para cada frequência:				
200 Hz	**100 Hz**	**50 Hz**	**25 Hz**	**20 Hz**
Pontos para treino por cada linha				
400	**200**	**100**	**50**	**40**

Para remover as frequências indesejadas dos sinais de entrada, poderá ser aplicado um filtro de passa-baixo – com detalhe na página seguinte – apenas se o resultado dessa escolha for benéfico para a otimização da rede. Por fim, para normalizar os dados de entrada do *sisfall dataset*, estão a considerar-se diferentes técnicas de normalização, como sejam a aplicação de transformações lineares que mantenham os valores entre os intervalos [0 - 1] e [-1 - 1] conforme Figura 29 – "*rescale 1 e 2*" e equações 6 e 7, bem como a normalização dos dados através da aplicação da média e desvio padrão, conforme explicado na página seguinte.

Detalhando o processo, é aplicado um filtro de *downsample*[10] (MathWorks, 2022), considerando os valores definidos na Tabela 11 - Frequências e downsample aplicado aos dados, onde a taxa de amostragem é diminuída pelo valor de entrada a considerar. No nosso caso, 1(200Hz) sem alterações, 2(100Hz), 4(50Hz), 8(25Hz) e 10(20Hz).

Quanto ao intervalo considerado para análise e apresentado na Tabela 10 - Janelas de tempo consideradas, tal como referido, será combinado com o resultado da aplicação das frequências que por sua vez produzirá uma taxa de amostragem. O resultado dessa aplicação será medido e avaliado o impacto na *probabilidade de acerto* poder ou não variar consideravelmente.

Em função da sua relevância, será aplicado um filtro de passa-baixo, para a remoção de frequências altas dos sinais. O filtro *Butterworth* é uma ferramenta versátil e útil em muitos domínios diferentes, desde a engenharia de áudio até à imagiologia médica. A sua resposta de frequência plana e a sua taxa de redução gradual fazem dele uma escolha popular para aplicações em que não é necessário um corte acentuado, e garantir que o sinal filtrado é reproduzido com precisão no tempo (*What Is a Butterworth RF Filter*, 2022). Será aplicado um filtro de ordem 4, com frequência de corte de 5Hz - o que significa que o filtro permitirá a passagem de sinais com frequências abaixo de 5Hz e atenuará sinais com frequências superiores a 5Hz.

[10] y = downsample (x, n) diminui a taxa de amostragem de x mantendo a primeira amostra e depois todas as n-ésimas amostras após a primeira. Se x for uma matriz, a função trata cada coluna como uma sequência separada.

Neste processo podem ser aplicadas igualmente as seguintes funções de normalização dos dados de entrada:

$$A = rescale(A) \quad (6)$$

$$A = rescale(A, -1,1) \quad (7)$$

A transformação linear considerada na equação 6, re-escala os valores de entrada, enquadrando-os entre 0 e 1. O mesmo sucede com os dados escalados pela equação 7, que os enquadra entre -1 e 1. Por último, podem ser normalizados os dados de entrada utilizando a média e desvio padrão. A normalização pode melhorar o desempenho de alguns algoritmos de *deep learning*, como redes neuronais, que tendem a funcionar melhor com dados normalizados, pelo que caso esta opção será considerada no caso de se verificar a existência de ganhos na eficiência global da rede.

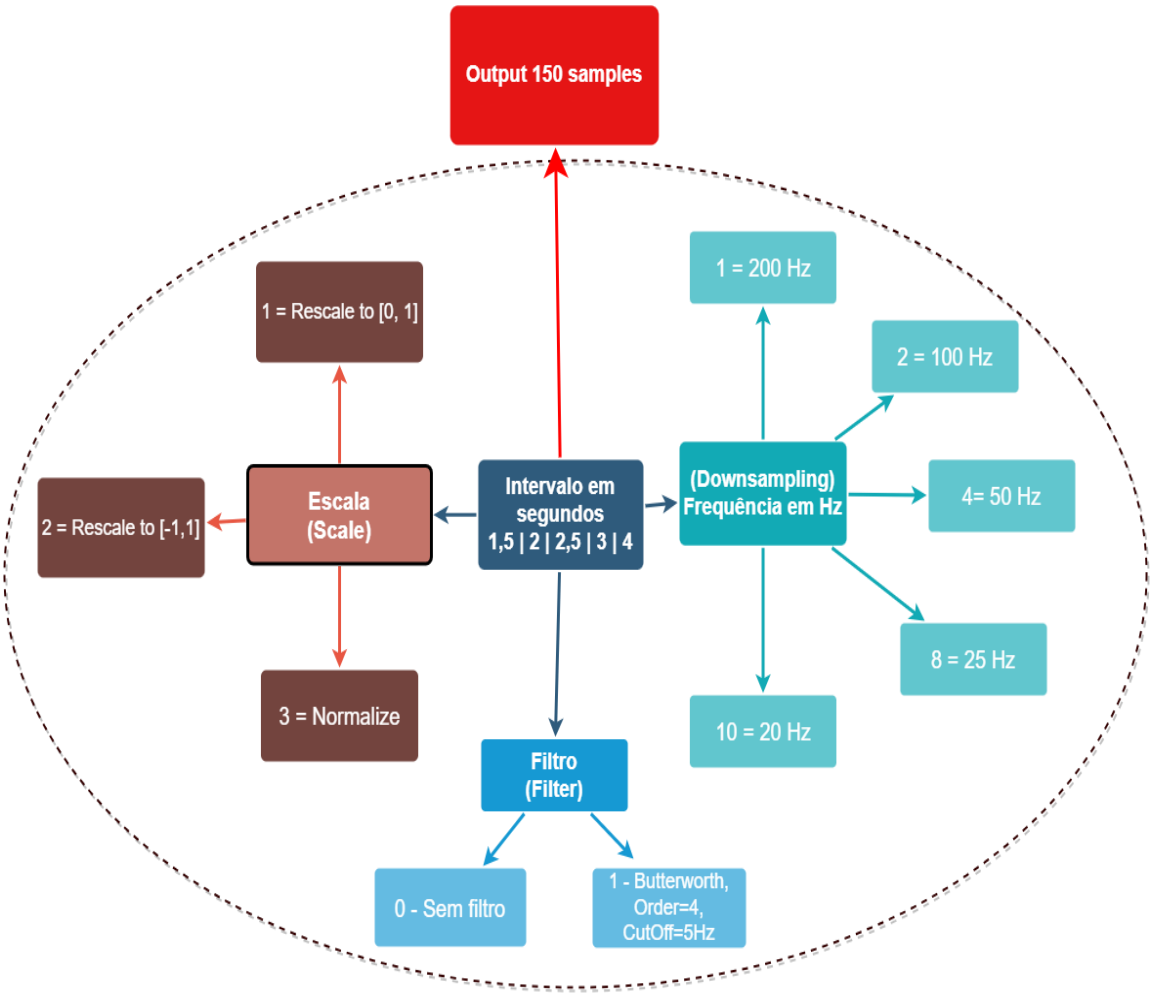


Figura 29 - Representação dos híper parâmetros a considerar.

3.4.6 - Arquitetura e treino

O treino da rede baseada no modelo (Torti et al., 2019) e (Musci et al., 2018), condiciona-se pelos dados de entrada que alimentam os algoritmos nas camadas ligadas. Neste trabalho, esses dados de entrada estão pré-processados pelo ficheiro de anotação e em 150 diferentes combinações, conforme o tamanho da entrada, dependendo do número de características selecionadas (no caso, os 3 eixos do acelerómetro) e do tamanho ***dt*** (em segundos) das janelas a introduzir no modelo. O tamanho da saída depende do número de classes (ou seja, BKG, ALERT e FALL). Todo o processo de treino da rede está descrito na Figura 30, onde para cada fase do treino se descrevem as atividades desenvolvidas.

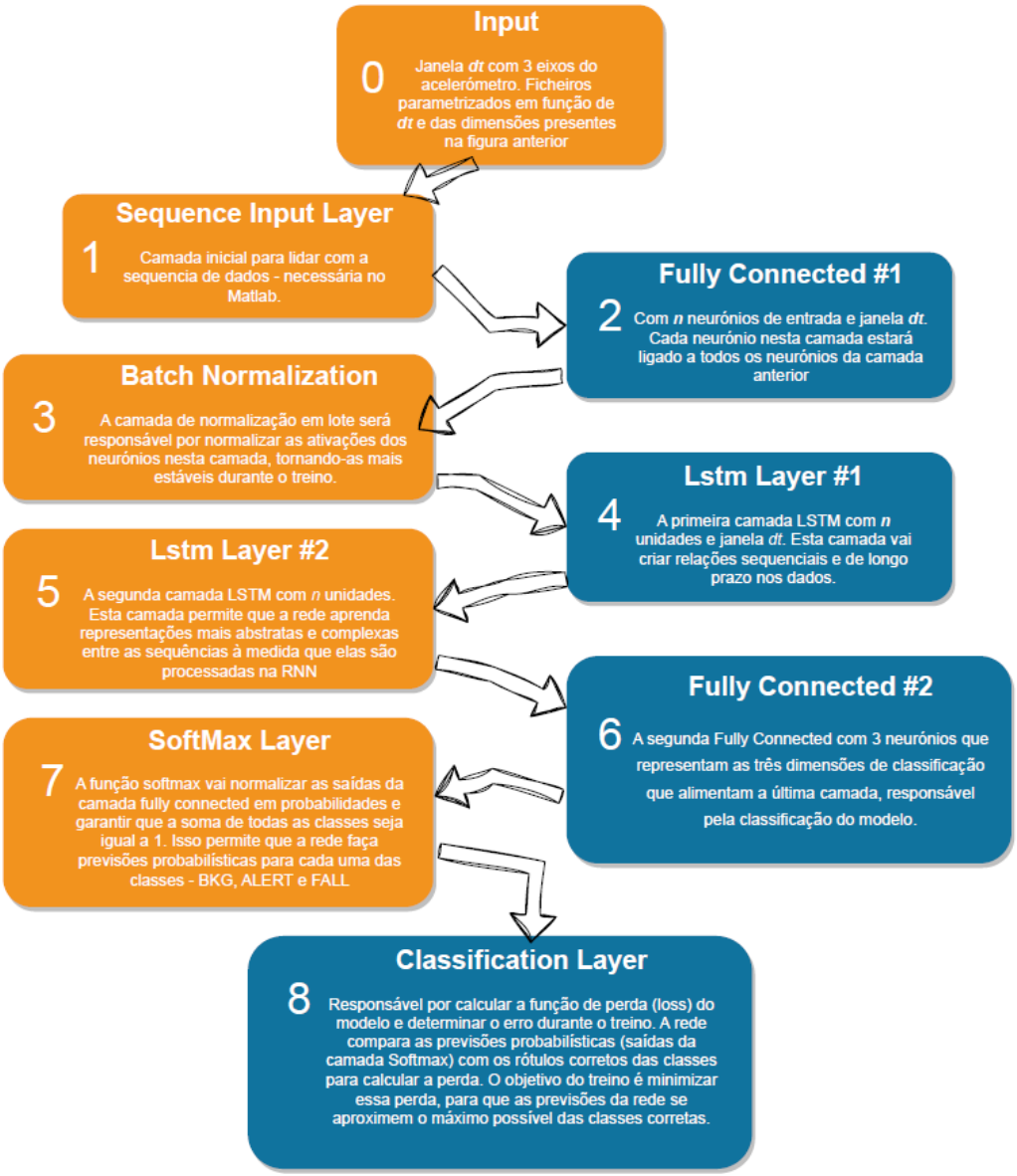


Figura 30 - Esquema da rede proposta, com o detalhe para cada uma das fases.

A rede de referência propõe a utilização de duas *fully connected* – FC e duas LSTM com paramentos fixos de 32 neurónios por camada, como pode ser observado na Figura 23.

Pretendeu-se testar a rede com uma amplitude maior no sentido de atingir a melhor *probabilidade de acerto* possível, pelo que se consideraram as seguintes hipóteses de estudo para as janelas de maior amplitude ***Δ***, nomeadamente um ***dt*** = 3 segundos e um ***dt*** = 4 segundos – uma vez que os resultados para um ***dt*** < 3 segundos ficaram aquém, não havendo necessidade de avaliação para os restantes cenários.

Tabela 13 - Janelas estudadas e dimensões da rede consideradas

Dimensão da rede	***Δ dt (segundos)***				
	1,5	2	2,5	3	4
20_20_80_3	-	-	-	x	x
32_32_32_3	x	x	x	x	x
40_40_60_3	-	-	-	x	x
60_60_40_3	-	-	-	x	x
80_80_20_3	-	-	-	x	x
100_100_10_3	-	-	-	x	x
150_150_75_3	-	-	-	x	x

A Tabela 13 pretende mostrar a representação do estudo, considerando os parâmetros e híper parâmetros – onde ***x*** corresponde a 30 *inputs* para treinar a rede (80% dados para treino e 20% para validação) para cada dimensão da rede em função de ***dt*** e restantes opções - conforme a Figura 29. A dimensão da rede representada na Figura 30, comporta uma primeira camada *fully connected*, seguida de duas LSTM's e uma última *fully connected*, sendo o cabeçalho e respetivos conteúdos de "Dimensão de rede" da Tabela 13 - Janelas estudadas e dimensões da rede consideradas, o reflexo da seguinte taxonomia: *FC1_LSTM1_LSTM2_FC2***.**

Tal como referido em (Torti et al., 2019), o treino de uma arquitetura deste tipo no conjunto de dados *Sisfall* requer cuidados específicos na criação de uma função de perda, capaz de gerir o desequilíbrio no conjunto de dados de rotulagem.

O facto de a classe BKG ser 40 vezes mais representada do que qualquer uma das restantes, fará com que os resultados referentes à *sensibilidade* e *especificidade* para as classes ALERT e FALL sejam inconsistentes – o que não se verifica para a classe dominante BKG, como se poderá apurar na apresentação dos resultados, no capítulo 4 - Discussão de resultados.

Para assegurar a melhor afinação dos dados no modelo, foi necessário testar alguns parâmetros até chegar a um ponto de equilíbrio – no qual a rede é potenciada no processo de treino e no output, de acordo com as expetativas; com enfase na *probabilidade de acerto* global e na suavização da *função de perda*.

Figura 31 – Opções tomadas para afinação do modelo para efeitos de treino e validação.

Podem observar-se as opções tomadas na Figura 31, onde diversas configurações são afinadas entre si para o treino da rede (Islam et al., 2020), e sua resposta face a outras opções - entretanto abandonadas. Com detalhe, o batch size tomado e que melhor respondeu durante o treino, foi a divisão em lotes de 35 unidades para os 3503 exemplares, o que gerou sensivelmente 100 iterações. Foram testadas 15, 30 e 45 *epochs*, sendo que os resultados estabilizaram com a última métrica aplicada. A frequência de validação durante o treino estabeleceu-se nos 50 pontos, com dois momentos de validação por cada *epoch*. Foi aplicado um optimizador Adam (Islam et al., 2020) para ajustar os pesos da rede durante o treino. O algoritmo Adam é uma variação do gradiente descendente estocástico que adapta as taxas de aprendizagem para cada parâmetro individual da rede, permitindo um treino mais eficiente e rápido.

Por fim, foi aplicada uma função de *shuffle* aos dados de teste e validação – equações 8 e 9, a cada início de uma nova *epoch*, como forma de os randomizar.

$$t = randperm(length(XT)) \quad (8)$$

$$t = randperm(length(XV)) \quad (9)$$

No capítulo seguinte serão apresentados os resultados da formulação, sendo eles o corolário de múltiplas decisões, escolhas e restrições – seguindo o modelo de referência. É importante mencionar que se apresentarão os melhores resultados apenas – tal como já referido acima.

4 - Discussão de resultados

Os resultados a apresentar focam a sua análise num ***dt*** = 4 segundos, onde para cada amostra se considerou um conjunto de pontos, com uma amplitude – *MinLength*, (correspondendo ao número de pontos por registo), representada no seguinte quadro:

Tabela 14 - Representação dos pontos a considerar numa janela de 4 segundos.

(dt)	(Downsampling)	Hz	(MinLength)	Filter	Scale
4,00	1	200	800	1	1
4,00	2	100	400	1	1
4,00	4	50	200	1	1
4,00	8	25	100	1	1
4,00	10	20	80	1	1

Na Tabela 14 podemos observar o panorama com o resumo dos híper parâmetros ótimos para o treino da rede. Neste processo foi aplicado o filtro de *Butterworth* - já detalhado, de Ordem 4, e frequência de corte de 5Hz, bem como se escalaram (equação 6) os valores para estarem num intervalo compreendido entre 0 e 1. A Figura 29 permite verificar as escolhas envolvidas, nomeadamente a escala e filtro, que correspondem ao *rescale* aplicado aos dados e o filtro para a remoção de frequências mais altas, apresentado na secção 3.4.5 - Ficheiro de anotação

A combinação presente na Tabela 14 em conjunto com a aplicação do filtro e re-escalando os dados entre 0 e 1, produziu os resultados que se julga serem os mais equilibrados – de todas as escolhas e avaliações possíveis, como se pode ver abaixo, na Tabela 15, uma vez que as métricas apuradas foram as mais próximas do modelo de referência e as mais altas até esse momento.

4.1 - Rede de referência

Para a rede de referência de 4 segundos, utilizando a amplitude total de frequências – Tabela 14, considerando uma *Fully Connected* de 32 nós, duas LSTM ligadas com 32 camadas escondidas, finalizando numa FC com 3 saídas, os resultados obtidos foram os seguintes para uma dimensão de rede conforme a Tabela 13, "Dimensão da rede" (32_32_32_3) ∩ Δ ***dt*** = 4 segundos.

Tabela 15 - Probabilidade de acerto para os pontos considerados na Tabela 13

Pontos	BKG	ALERT	FALL	Global Accuracy
80	93,839%	93,547%	94,677%	92,609%
100	93,527%	93,041%	93,959%	92,388%
200	93,951%	93,656%	95,237%	92,673%
400	93,726%	93,362%	95,278%	92,528%
800	93,526%	93,157%	94,072%	92,407%

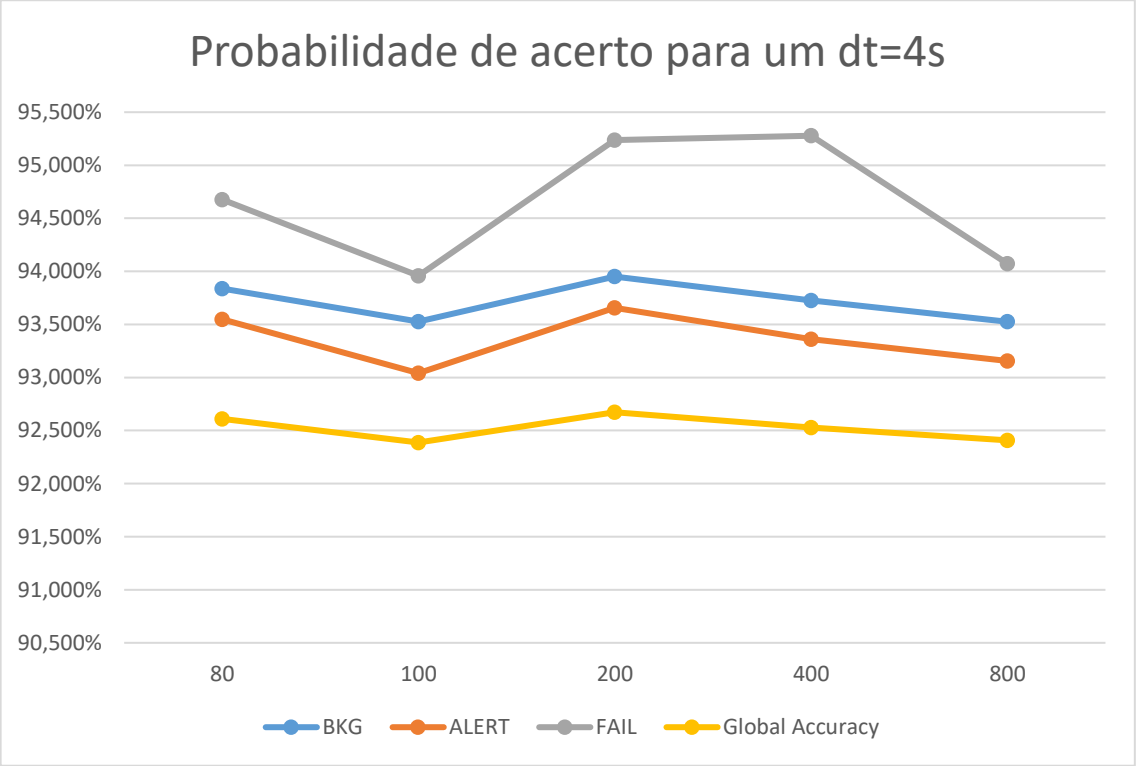


Figura 32 - Probabilidade de acerto para um dt=4segundos.

Pode verificar-se na Figura 32 uma relação de proximidade nos dados apurados sendo que a quantidade de pontos (por diminuição da frequência aplicada) – para uma janela de 4 segundos a introduzir no modelo, sugere não alterar significativamente os resultados, o que poderá conferir ao modelo uma redução na sua dimensão. A redução na quantidade de dados é benéfica para a modelação, sobretudo quando se a lida com múltiplas dimensões.

Possível melhoria no desempenho do modelo: A aplicação do filtro *Butterworth* terá filtrado ruídos indesejados ou detalhes de alta frequência, não tão relevantes para a previsão de quedas. Isso pode resultar num melhor desempenho do modelo.

Neste sentido, importa apresentar as métricas do treino da rede de dimensão (32_32_32_3) ∩ Δ ***dt*** = 4 segundos para os 80 pontos, bem como as de validação. Apresenta-se igualmente o estudo através da matriz de confusão, onde se pode avaliar as dimensões da *probabilidade de acerto* para cada uma das classes, bem como a *especificidade*, *sensibilidade* e *precisão*.

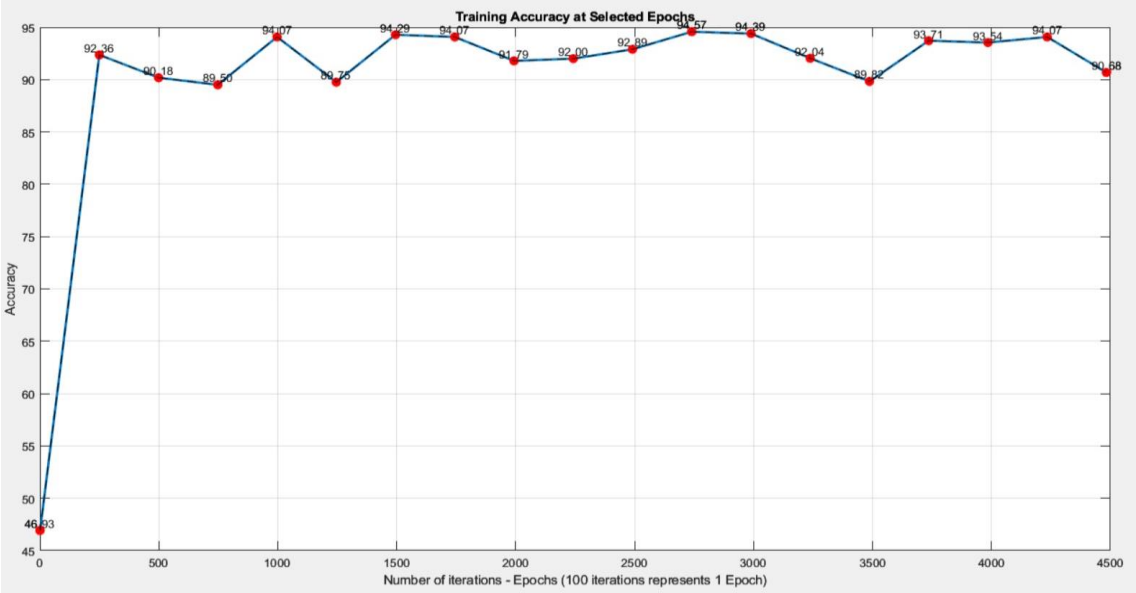


Figura 33 – Rede de 80 pontos com resultados do treino da *probabilidade de acerto*.

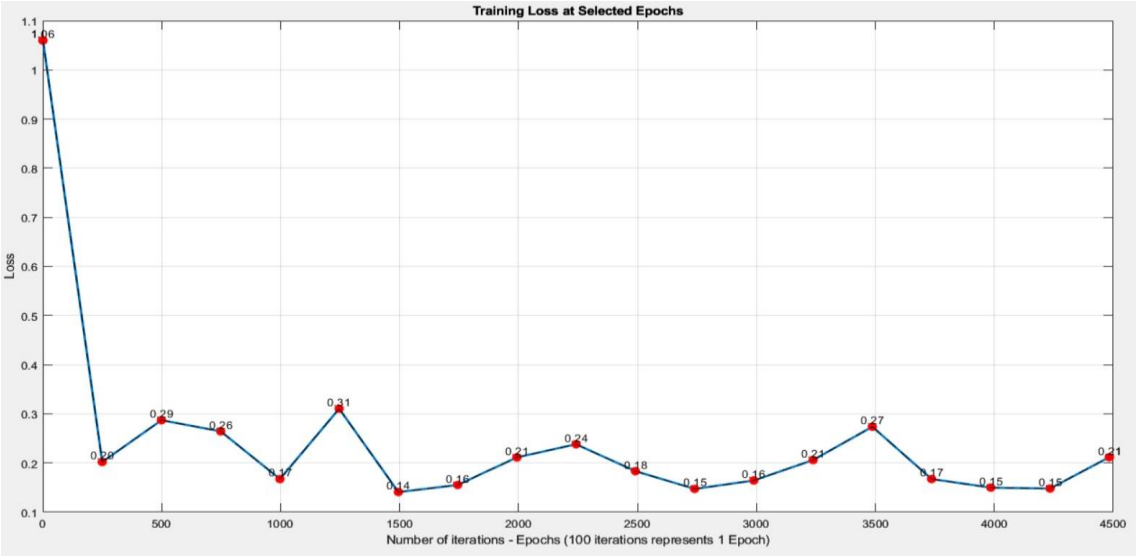


Figura 34 – Rede de 80 pontos com resultados do treino da *perda – training loss*.

A Figura 33 e Figura 34 apresentam o percurso e desempenho da rede numa fase de treino, nas dimensões da sua *probabilidade de acerto* bem com da *perda*. O valor final para cada uma delas foi respetivamente de 90.93% e 0.20%. Não obstante a relevância destes resultados, importa avaliar os resultados da validação do modelo. Os resultados apurados são apresentados na Tabela 16.

Tabela 16 – Resultados para a validação da *perda* e *probabilidade de acerto*

Final Validation Loss	Final Validation Accuracy
0.19%	92.54%

Quando os resultados da *perda* e *probabilidade de acerto* apresentam valores idênticos tanto na fase de treino como na validação – como se pode verificar pela Tabela 14 e Figura 37, poderá inferir-se que o modelo não sofre de *overfitting* (Decuyper et al., 2019). Isso pode ser interpretado como um sinal positivo em relação à qualidade do modelo que está a ser desenvolvido, mas também é importante considerar alguns pontos adicionais:

- **Boa Generalização** (Lecun et al., 2015): A consistência dos resultados entre as fases de treino e validação sugere que o modelo - específico para o *dataset Sisfall*, é capaz de aprender padrões e relações dos dados de treino, conseguindo aplicar esse conhecimento a dados não familiares. A generalização é um objetivo fundamental no desenvolvimento de modelos de *machine learning*, incluindo redes neuronais, porque o objetivo principal é criar modelos que possam fazer previsões o mais exatas possíveis em situações do mundo real.

- **Modelo Adequado**: Se o desempenho do modelo é igualmente constante em ambas as fases, isso sugere que a arquitetura e os híper parâmetros do modelo estão ajustados de forma coerente.

No entanto, importa referir que embora essa consistência seja um bom sinal, ela não garante que o modelo esteja otimizado.

Outra forma de apreciar os resultados é a utilização de matrizes de confusão, pois estas apresentam uma perspetiva do quão afinado o modelo está, uma vez que é uma ferramenta rigorosa na análise do desempenho do modelo de classificação criado. Vai permitir uma representação visual das previsões feitas pela rede e também a comparação com os valores reais das classes.

A matriz de confusão vai ser especialmente útil no caso em estudo para avaliar o problema da classificação com múltiplas classes, sobretudo quando existe uma classe maioritária e negativa, com maior representatividade em relação às restantes.

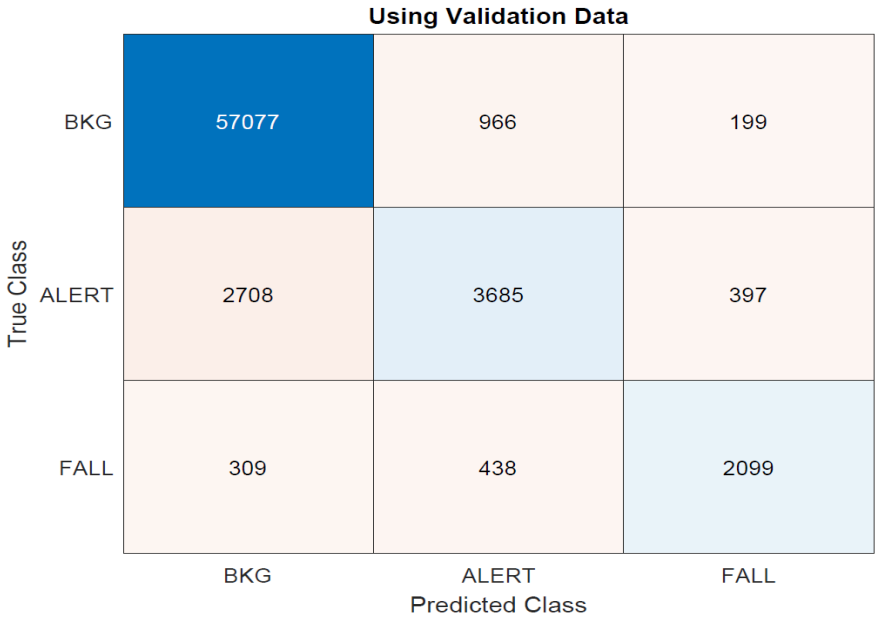


Figura 35 - Rede de 80 pontos com resultados da matriz de confusão.

Metrics Summary

Metric	Class	Value
Accuracy	BKG	0.94
	ALERT	0.93
	FALL	0.98
Sensitivity	BKG	0.98
	ALERT	0.54
	FALL	0.74
Specificity	BKG	0.69
	ALERT	0.98
	FALL	0.99

Figura 36 - Rede de 80 pontos com métricas da matriz de confusão.

Pode verificar-se pelos resultados apurados, apresentados na Figura 35 e Figura 36 (com o foco inicial na *probabilidade de acerto - accuracy*), que o modelo classifica todas as classes com grande rigor. A *sensibilidade* - também conhecida como taxa de verdadeiros positivos, mede a capacidade do modelo em identificar corretamente instâncias positivas. No caso específico, observa-se que a *sensibilidade* do modelo é alta para a classe BKG, mas relativamente baixa para as classes ALERT e FALL.

Quando há um desequilíbrio significativo na distribuição das classes, e neste caso há uma prevalência de valores para a classe de BKG – cerca de 40 vezes mais representativa do que as restantes, pode significar a existência de um desvio em baixa da métrica *sensibilidade* para as classes minoritárias (ALERT e FALL). O desequilíbrio de classes vai afetar os resultados de *sensibilidade* - visível na Figura 36, uma vez o cálculo é feito com base nos verdadeiros positivos e falsos negativos – equação 2 sendo que o desequilíbrio pode levar a um número maior de falsos negativos, o que afeta negativamente esta métrica nas classes ALERT e FALL. Esta circunstância foi identificada, mas não obstante, não foi alvo de tratamento aprofundado nesta tese.

A *especificidade* - também conhecida como taxa de verdadeiros negativos, mede a capacidade de o modelo identificar corretamente instâncias negativas – equação 3. Verifica-se que a classe BKG apresenta um valor mais baixo, uma vez que o modelo ainda classifica com insegurança os falsos positivos para esta classe, com enfase para os FP erradamente classificados como BKG mas que seriam na verdade eventos de ALERT. Pela dimensão que a classe BKG representa – 40 vezes mais, compreende-se que haja uma fronteira ténue entre o que é uma atividade diária e um eventual alerta que, pode não significar uma queda efetiva adiante, daí o valor de especificidade ser mais baixo do que o desejável. No entanto, entender essas nuances é crucial para ajustar o modelo e melhorar o seu desempenho no futuro.

A Tabela 17 resume todas as métricas apuradas para a rede (32_32_32_3) ∩ Δ ***dt*** = 4 segundos, conforme matriz da Tabela 13 - Janelas estudadas e dimensões da rede consideradas.

Tabela 17 – Resumo das métricas apuradas para a rede de referência.

Treino e Validação	
Accuracy durante treino	90,93
Perda durante treino	0,2
Validação final Accuracy	92,54
Validação final Perda	0,19
Matriz de Confusão	
Accuracy	
BKG	0,94
ALERT	0,93
FALL	0,98
Sensitivity	
BKG	0,98
ALERT	0,54
FALL	0,74
Specificity	
BKG	0,69
ALERT	0,98
FALL	0,99

4.2 - Versão mais otimizada da rede e comparação com a rede de referência

De seguida, interessa verificar as diferenças entre os resultados apresentados acima, comparando o modelo de rede (32_32_32_3) ∩ Δ ***dt*** = 4 segundos com a versão mais otimizada da rede (150_150_75_3) ∩ Δ ***dt*** = 4 segundos, previsto e testado conforme definido na matriz da Tabela 13 - utilizando os mesmos híper parâmetros.

Tabela 18 - Resumo das métricas apuradas para a rede mais otimizada.

Treino e Validação	
Accuracy durante treino	94,93
Perda durante treino	0,14
Validação final Accuracy	93,11
Validação final Perda	0,19
Matriz de Confusão	

Accuracy	
BKG	0,94
ALERT	0,94
FALL	0,98
Sensitivity	
BKG	0,98
ALERT	0,62
FALL	0,77
Specificity	
BKG	0,76
ALERT	0,97
FALL	0,99

A Figura 37 apresenta os valores obtidos para as duas redes e permite verificar o delta entre os resultados. As métricas isoladas apresentaram-se na Tabela 17 e Tabela 18, referentes às redes (32_32_32_3) ∩ Δ ***dt*** = 4 segundos e (150_150_75_3) ∩ Δ ***dt*** = 4 segundos.

Tamanho da rede

	32_32_32_3	Delta (%)	150_150_75_3
		Treino e Validação	
Accuracy durante treino	90,93	4,40%	94,93
Perda durante treino	0,2	-27,90%	0,14
Validação final Accuracy	92,54	0,62%	93,11
Validação final Perda	0,19	0,11%	0,1902
		Matriz de Confusão	
Accuracy			
BKG	0,94	0%	0,94
ALERT	0,93	1%	0,94
FALL	0,98	0%	0,98
Sensitivity			
BKG	0,98	0%	0,98
ALERT	0,54	15%	0,62
FALL	0,74	4%	0,77
Specificity			
BKG	0,69	10%	0,76
ALERT	0,98	-1%	0,97
FALL	0,99	0%	0,99

Figura 37 - Resumo e comparação para as duas redes, ambas com dimensão de 80 pontos, 4 segundos e 45 *Epochs*.

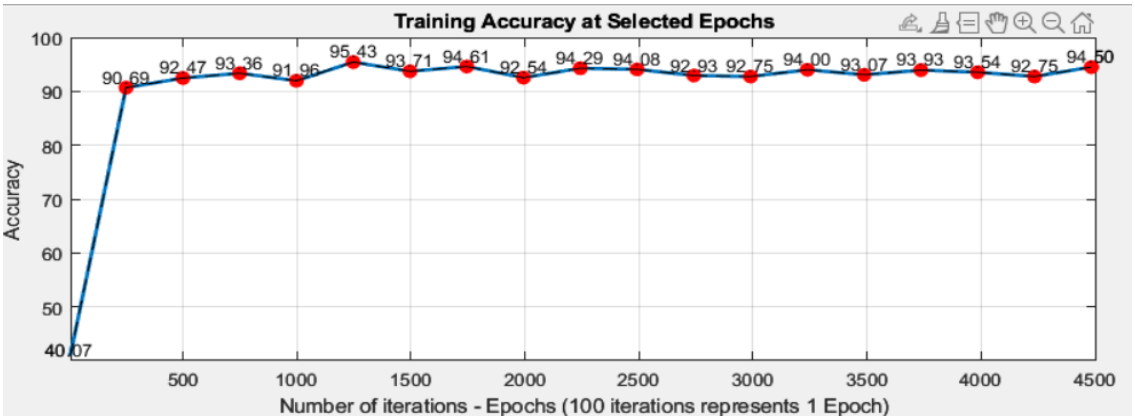


Figura 38 – Rede mais otimizada de 80 pontos com resultados do treino da *probabilidade de acerto – Accuracy*.

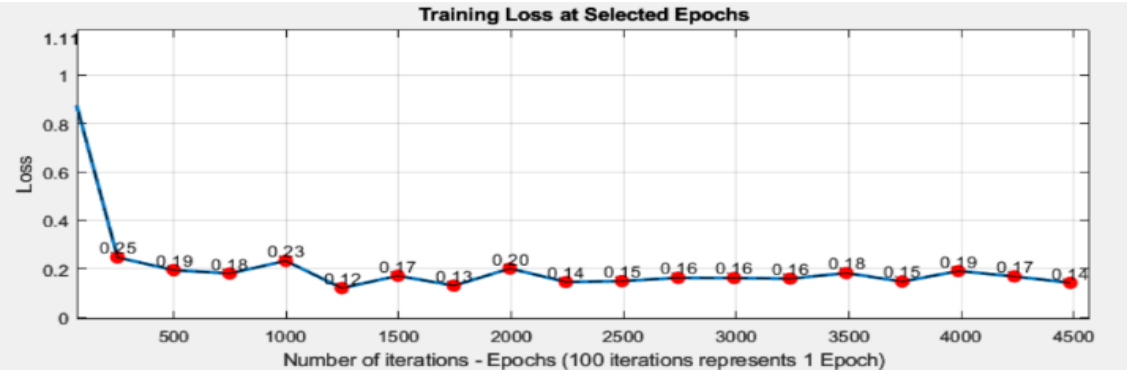


Figura 39 - Rede mais otimizada de 80 pontos com resultados do treino da *perda – training loss.*

Os resultados apresentados na Figura 38 e Figura 39, refletem uma ligeira alteração global dos valores, onde se verifica um aumento na performance do treino da perda, com ganho de 27,90% relativamente à rede (32_32_32_3) ∩ Δ dt = 4 segundos, e melhor valor na sensibilidade para as classe FALL em 4% e ALERT na ordem dos 15%.

Podem avaliar-se estes resultados como positivos, se se considerar o ponto de partida, onde se verificavam valores de *probabilidade de acerto* na ordem dos 84% com 15 *Epochs* - antes de quaisquer otimizações e acertos tidos nos parâmetros da rede, conforme se verifica pelas métricas iniciais presentes nos sucessivos testes:

Tabela 19 - Amostra testada para 15, 30 e 45 *Epochs*, numa fase inicial

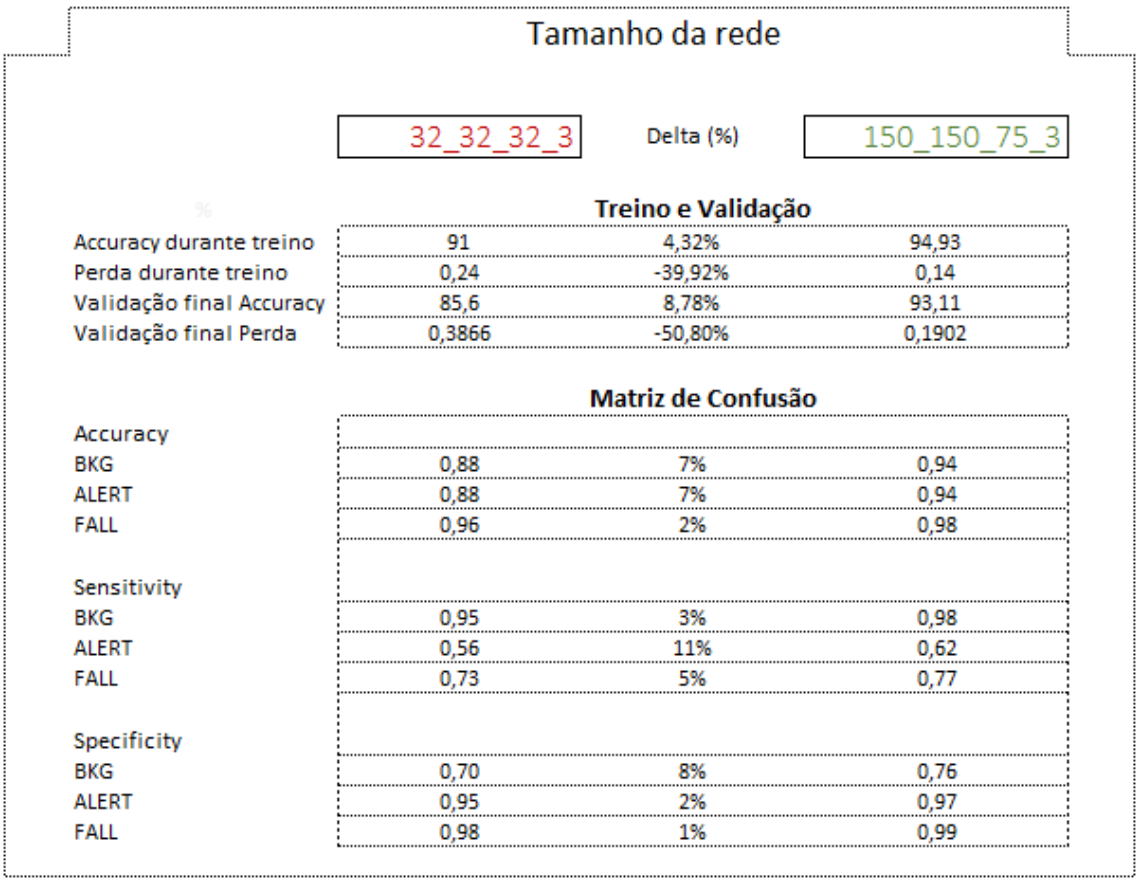
miniBatchSize	ValidationFrequency	Epochs	AC%	LOSS%
35	50	15	84,86	0,38
35	50	30	85,53	0,42
35	50	45	85,60	0,39

A Tabela 19 representa uma amostra testada para 15, 30 e 45 *Epochs*, sem otimização dos parâmetros de rede e com híper parâmetros não afinados, com intervalo ***dt*** = 2 segundos, diminuição de frequência de 40 pontos por registo e aplicação de escala de normalização, onde AC% corresponde à *probabilidade de acerto* global do modelo após validação, e LOSS%, aos dados de validação relativos à *perda* na função de custo.

4.3 - Comparativo entre versão inicial de 2 s. e versão otimizada

Por fim, na Figura 40, reflete-se o quadro resumo da amostra (32_32_32_3) e ***dt*** = 2 segundos, sem otimização dos parâmetros de rede e com híper parâmetros não afinados, comparando-a com o melhor desempenho alcançado.

Detalhando, importa referir que os dados se referem a amostras com 45 *Epochs* e no caso desta, ela é constituída por 40 pontos de dimensão por registo e uma rede de (32_32_32_3) ∩ Δ ***dt*** = 2 segundos - Tabela 13 - Janelas estudadas e dimensões da rede consideradas, o que significa tratar-se de uma rede com híper parâmetros não afinados e com os parâmetros padrão, retratados na Figura 23 - Modelo de referência para implementação. Por oposição, a rede compara-se com a versão mais otimizada do trabalho e que melhor desempenho atingiu, de todas as opções testadas, descrita na secção 4.2 - Versão mais otimizada da rede e comparação com a rede de referência, sendo uma rede (150_150_75_3) ∩ Δ ***dt*** = 4 segundos.



Tamanho da rede

	32_32_32_3	Delta (%)	150_150_75_3
Treino e Validação			
Accuracy durante treino	91	4,32%	94,93
Perda durante treino	0,24	-39,92%	0,14
Validação final Accuracy	85,6	8,78%	93,11
Validação final Perda	0,3866	-50,80%	0,1902
Matriz de Confusão			
Accuracy			
BKG	0,88	7%	0,94
ALERT	0,88	7%	0,94
FALL	0,96	2%	0,98
Sensitivity			
BKG	0,95	3%	0,98
ALERT	0,56	11%	0,62
FALL	0,73	5%	0,77
Specificity			
BKG	0,70	8%	0,76
ALERT	0,95	2%	0,97
FALL	0,98	1%	0,99

Figura 40 - resumo e comparação entre as duas redes, sendo a primeira de 40 pontos e 2 segundos, com a segunda de 80 pontos e 4 segundos e ambas com 45 *Epochs*.

Pode concluir-se que o modelo tem evolução em todas as dimensões e o comportamento das métricas depois de otimizadas refletem bem a afinação do modelo como um todo. Como corolário dos resultados, conclui-se com a tabela de referência, apresentada em (Torti et al., 2019), concatenada com os resultados deste trabalho, para que se possa entender a performance deste modelo face aos demais, bem como a comparação com o modelo de dados clássico *Sisfall*.

Observa-se pela Tabela 20, que a abordagem tida neste trabalho obteve ganhos comparados relativamente ao modelo de dados clássico *Sisfall* e também relativamente à abordagem tida nos trabalhos de referência. Tendo sido assumida a totalidade dos dados *Sisfall*, verifica-se por essa via um desvio nos valores relativos à *sensibilidade*, nas classes ALERT e FALL, uma vez que se potencia um número maior de falsos negativos, afetando negativamente a performance da métrica associada, não esquecendo o facto de a classe BKG ser 40 vezes mais representado do que as restantes classes.

Tabela 20 - Comparação com tabela de referência, que justificou o estudo desta tese

	(Torti et al., 2019)	*Sisfall*	Esta Tese	% (Torti et al., 2019)	% Sisfall
Sensitivity					
BKG	88,39	75,01	97,57	10,39	30,08
ALERT	91,08	68,15	62,09	-31,83	-8,89
FALL	98,73	75,79	77,23	-21,78	1,90
Specificity					
BKG	97,85	92,52	75,58	-22,76	-18,31
ALERT	90,77	83,30	97,33	7,23	16,84
FALL	97,93	91,57	98,99	1,08	8,10
Accuracy					
BKG	93,12	83,77	94,45	1,43	12,75
ALERT	90,93	75,73	93,81	3,17	23,87
FALL	98,33	83,68	98,08	-0,25	17,21

Não obstante esta evidência, aprecia-se ganhos significativos relativamente ao modelo clássico *Sisfall*, tendo a *probabilidade de acerto* melhorado com as afinações introduzidas com reflexo positivo em todas as categorias. A comparação com (Torti et al., 2019) para a mesma métrica, mantém a tendência, apenas deprecia 0,25% para a categoria Fall, uma vez que o modelo desta tese não ultrapassou os 98,08% contra os 98,33% da referência.

5 - Conclusão e trabalho futuro

Neste capítulo serão apresentadas as conclusões relativamente ao trabalho desenvolvido, discorrendo-se sobre os resultados obtidos, bem como se discutem as propostas de trabalho futuro, concluindo-se deste modo o estudo proposto.

5.1 – Enquadramento para os resultados

Neste trabalho foi proposta uma rede baseada em *deep learning*, alimentada por informações de sensores portáteis - sob a forma de um *dataset* preparado para o efeito e adaptado para conter um modelo de classificação de três dimensões.

Foram executadas atividades de análise e implementados algoritmos de inteligência artificial para a deteção de quedas em idosos com base na medição de aceleração, e tendo em conta a sua futura implementação em equipamentos de baixo poder de computação.

Foram estudadas redes recursivas com memória de curto prazo ou LSTM bem como redes convolucionais. Foi efetuada a análise do atual estado da arte, executada a implementação e análise de performance das topologias de rede consideradas as mais promissoras. Foram igualmente estudados algoritmos cuja implementação foi executada em plataformas computacionais de baixo poder de computação e analisados os resultados.

Implementou-se uma rede neuronal baseada em LSTM, sobre a qual foram aplicadas técnicas no sentido de melhorar o desempenho dos sistemas propostos. Foram testadas múltiplas camadas LSTM considerando janelas temporais mais alargadas, que na sua dimensão de processamento conseguiram manter uma performance acima dos sistemas de referência.

5.2 – Avaliação do desempenho

Tendo em conta o modelo de classificação de três dimensões, os resultados apurados nesta abordagem demonstraram-se satisfatórios, com métricas aperfeiçoadas relativamente ao modelo clássico *Sisfall* e com níveis de *probabilidade de acerto, sensibilidade* e *especificidade* em linha com os melhores resultados dos sistemas em análise – conforme se pode avaliar pela Tabela 20 - Comparação com tabela de referência, que justificou o estudo desta tese.

O modelo com a maior otimização desta tese atingiu um máximo de probabilidade de acerto no treino de 94,93% e 0,14% de *valor de perda*, sendo os valores finais de validação de 93,11% e 0,19% respetivamente, podendo concluir-se que existiu evolução em todas as dimensões e o comportamento das métricas depois de otimizadas refletiram bem a afinação do modelo como um todo – considerando a rede (150_150_75_3) ∩ Δ ***dt*** = 4 segundos.

5.3 – Trabalho futuro

Como tarefas para o futuro, no sentido de dar sequência a esta tese de mestrado, pretende-se formalizar as seguintes atividades:

Submeter o *dataset Sisfall enhanced* a uma rede *Gated Recurrent Unit* GRU e avaliar o comportamento da rede e seus resultados, comparando-os com a rede LSTM implementada nesta tese.

Estudar o problema do balanceamento do *dataset Sisfall enhanced*, e avaliar a distribuição de dados globais, uma vez que a classe predominante negativa de BKG surge com cerca de 40 vezes maior prevalência relativamente às demais, atenuando os resultados positivos relativamente à *sensibilidade* e *especificidade*. A estratégia poderá passar por balancear o número de amostras de quedas e eventos de atividades ADL, uma vez que o *dataset* estrutura-se por 60% de ADL's e cerca de 40% de eventos de queda, sendo que desses eventos, a janela com o máximo absoluto prevê um desvio à esquerda e direita de 80% de dados de ADL até ao evento de pico – que se considera com sendo uma queda ou alerta.

Por fim, depois de avaliada a performance de cada uma das soluções anteriores, a passagem a uma etapa de implementação do modelo numa unidade microcontrolador (MCU), onde se estabelecem um conjunto de atividades possíveis, desde a escolha da plataforma de hardware, a conversão para um formato compatível – por exemplo a conversão para o TensorFlow Lite, seguindo-se a otimização do modelo, geração do código, e outas atividades para as quais necessariamente haverá uma preparação que não tem cabimento neste momento.

Bibliografia

Abdelhedi, S., Baklouti, M., Bourguiba, R., & Mouine, J. (2016). Design and implementation of a fall detection system on a Zynq board. *Proceedings of IEEE/ACS International Conference on Computer Systems and Applications, AICCSA*, *0*. https://doi.org/10.1109/AICCSA.2016.7945775

Ajerla, D., Mahfuz, S., & Zulkernine, F. (2019). A real-time patient monitoring framework for fall detection. *Wireless Communications and Mobile Computing*, *2019*. https://doi.org/10.1155/2019/9507938

Alqahtani, A., Xie, X., Deng, J., & Jones, M. W. (2018). A Deep Convolutional Auto-Encoder with Embedded Clustering. *Proceedings - International Conference on Image Processing, ICIP*, 4058–4062. https://doi.org/10.1109/ICIP.2018.8451506

Casilari, E., Lora-rivera, R., & García-lagos, F. (2020). A study on the application of convolutional neural networks to fall detection evaluated with multiple public datasets. *Sensors (Switzerland)*, *20*(5). https://doi.org/10.3390/s20051466

Charte, F., Rivera, A. J., Martínez, F., & del Jesus, M. J. (2019). Automating Autoencoder Architecture Configuration: An Evolutionary Approach. In J. M. Ferrández Vicente, J. R. Álvarez-Sánchez, F. de la Paz López, J. Toledo Moreo, & H. Adeli (Eds.), *Understanding the Brain Function and Emotions* (pp. 339–349). Springer International Publishing.

Cola, G., Avvenuti, M., Piazza, P., & Vecchio, A. (2017). Fall Detection Using a Head-Worn Barometer. In P. Perego, G. Andreoni, & G. Rizzo (Eds.), *Wireless Mobile Communication and Healthcare* (pp. 217–224). Springer International Publishing.

Decuyper, M., Stockhoff, M., Vandenberghe, S., -, al, & Ying, X. (2019). *An Overview of Overfitting and its Solutions You may also like Artificial neural networks for positioning of gamma interactions in monolithic PET detectors Analysis of overfitting in the regularized Cox model An Overview of Overfitting and its Solutions*. 22022. https://doi.org/10.1088/1742-6596/1168/2/022022

Dong, G. (2018). A Review of the Autoencoder and Its Variants. *IEEE Geoscience and Remote Sensing Magazine*, *6*(4), 92–92. https://doi.org/10.1109/mgrs.2018.2877953

Fafoutis, X., Marchegiani, L., Elsts, A., Pope, J., Piechocki, R., & Craddock, I. (2018). Extending the battery lifetime of wearable sensors with embedded machine learning. *IEEE World Forum on Internet of Things, WF-IoT 2018 - Proceedings*, *2018-Janua*, 269–274. https://doi.org/10.1109/WF-IoT.2018.8355116

Ge, C., Gu, I. Y. H., & Yang, J. (2018). Co-Saliency-Enhanced Deep Recurrent Convolutional Networks for Human Fall Detection in E-Healthcare. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, *2018-July*, 1572–1575. https://doi.org/10.1109/EMBC.2018.8512586

Guillaume Chevalier. (2016). *LSTM-Human-Activity-Recognition*. https://github.com/guillaume-chevalier/LSTM-Human-Activity-Recognition

Hammerla, N. Y., Halloran, S., & Plötz, T. (2016). Deep, convolutional, and recurrent models for human activity recognition using wearables. *IJCAI International Joint Conference on Artificial Intelligence*, *2016-Janua*, 1533–1540.

He, J., Zhang, Z., Wang, X., & Yang, S. (2019). A low power fall sensing technology based on fd-cnn. *IEEE Sensors Journal*, *19*(13), 5110–5118. https://doi.org/10.1109/JSEN.2019.2903482

Hossain, F., Ali, M. L., Islam, M. Z., & Mustafa, H. (2016). A direction-sensitive fall detection system using single 3D accelerometer and learning classifier. *2016 International Conference on Medical Engineering, Health Informatics and Technology (MediTec)*, 1–6. https://doi.org/10.1109/MEDITEC.2016.7835372

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *32nd International Conference on Machine Learning, ICML 2015*, *1*, 448–456.

Islam, M., Tayan, O., Islam, R., Islam, S., Nooruddin, S., Kabir, M. N., & Islam, R. (2020). Deep Learning Based Systems Developed for Fall Detection: A Review. *IEEE Access*, *8*, 166117–166137. https://doi.org/10.1109/ACCESS.2020.3021943

Laboratory, S. A. (n.d.). *UniMiB SHAR*. Retrieved December 4, 2023, from https://www.sal.disco.unimib.it/technologies/unimib-shar/

Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

Lou, Z., Wang, L., Jiang, K., Wei, Z., & Shen, G. (2020). Materials Science & Engineering R Reviews of wearable healthcare systems : Materials , devices and system integration. *Materials Science & Engineering R*, *140*(November 2019), 100523. https://doi.org/10.1016/j.mser.2019.100523

Mariano, D. (2021). Métricas de avaliação em machine learning: acurácia, sensibilidade, precisão, especificidade e F-score. *BIOINFO - Revista Brasileira de Bioinformática e Biologia Computacional*, *2019*. https://doi.org/10.51780/978-6-599-275326-15

MathWorks. (2022). *Decimation — decrease sample rate by integer factor - MATLAB decimate - MathWorks United Kingdom*. MathWorks Help Center. https://www.mathworks.com/help/signal/ref/downsample.html

Matos-Carvalho, J. P., Correia, S. D., & Tomic, S. (2023). Sensitivity Analysis of LSTM Networks for Fall Detection Wearable Sensors. *Proceedings - 2023 6th Conference on Cloud and Internet of Things, CIoT 2023*, 112–118. https://doi.org/10.1109/CIoT57267.2023.10084906

Miikkulainen, R., Liang, J., Meyerson, E., Rawal, A., Fink, D., Francon, O., Raju, B., Shahrzad, H., Navruzyan, A., Duffy, N., & Hodjat, B. (2018). Evolving deep neural networks. *Artificial Intelligence in the Age of Neural Networks and Brain Computing*, 293–312. https://doi.org/10.1016/B978-0-12-815480-9.00015-3

Muro-de-la-Herran, A., García-Zapirain, B., & Méndez-Zorrilla, A. (2014). Gait analysis methods: An overview of wearable and non-wearable systems, highlighting clinical applications. *Sensors (Switzerland)*, *14*(2), 3362–3394. https://doi.org/10.3390/s140203362

Musci, M., De Martini, D., Blago, N., Facchinetti, T., & Piastra, M. (2018). *Online Fall Detection using Recurrent Neural Networks*. *Dl*. https://doi.org/10.1109/TETC.2020.3027454

Ngu, D. A. H. H., Yan, D. Y., Chang, D. J., & Milton, D. J. G. (n.d.). *SmartFall*. Retrieved November 4, 2023, from https://smartfall.github.io/

Nguyen Gia, T., Sarker, V. K., Tcarenko, I., Rahmani, A. M., Westerlund, T., Liljeberg, P., & Tenhunen, H. (2018). Energy efficient wearable sensor node for IoT-based fall detection systems. *Microprocessors and Microsystems*, *56*(November 2017), 34–46.

https://doi.org/10.1016/j.micpro.2017.10.014

Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). *Activation Functions: Comparison of trends in Practice and Research for Deep Learning*. 1–20. http://arxiv.org/abs/1811.03378

Nyan, M. N., Tay, F. E. H., & Murugasu, E. (2008). A wearable system for pre-impact fall detection. *Journal of Biomechanics*, *41*(16), 3475–3481. https://doi.org/10.1016/j.jbiomech.2008.08.009

Pena Queralta, J., Gia, T. N., Tenhunen, H., & Westerlund, T. (2019). Edge-AI in LoRa-based health monitoring: Fall detection system with fog computing and LSTM recurrent neural networks. *2019 42nd International Conference on Telecommunications and Signal Processing, TSP 2019*, 601–604. https://doi.org/10.1109/TSP.2019.8768883

PORDATA. (2021). *PORDATA - Indicadores de envelhecimento segundo os Censos*. Indicadores de Envelhecimento Segundo Os Censos. https://www.pordata.pt/portugal/indicadores+de+envelhecimento+segundo+os+censos-525

Rajagopalan, R., Litvan, I., & Jung, T. P. (2017). Fall prediction and prevention systems: Recent trends, challenges, and future research directions. In *Sensors (Switzerland)* (Vol. 17, Issue 11, p. 2509). Multidisciplinary Digital Publishing Institute. https://doi.org/10.3390/s17112509

Rangamani, A., Mukherjee, A., Basu, A., Arora, A., Ganapathi, T., Chin, S., & Tran, T. D. (2018). Sparse Coding and Autoencoders. *IEEE International Symposium on Information Theory - Proceedings*, *2018-June*, 36–40. https://doi.org/10.1109/ISIT.2018.8437533

Rawat, W., & Wang, Z. (2017). Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review. *Neural Computation*, *29*(9), 2352–2449. https://doi.org/10.1162/NECO_a_00990

Serra, R., Knittel, D., Croce, P. Di, & Peres, R. (2016). Activity Recognition With Smart Polymer Floor Sensor: Application to Human Footstep Recognition. *IEEE Sensors Journal*, *16*(14), 5757–5775. https://doi.org/10.1109/JSEN.2016.2554360

sisfall temporally annotated. (n.d.). Retrieved September 5, 2023, from

https://bitbucket.org/unipv_cvmlab/sisfalltemporallyannotated/src/master/

Solbach, M. D., & Tsotsos, J. K. (2017). Vision-Based Fallen Person Detection for the Elderly. *Proceedings - 2017 IEEE International Conference on Computer Vision Workshops, ICCVW 2017*, *2018-Janua*, 1433–1442. https://doi.org/10.1109/ICCVW.2017.170

Sucerquia, A., López, J. D., & Vargas-Bonilla, J. F. (2017). SisFall: A fall and movement dataset. *Sensors (Switzerland)*, *17*(1). https://doi.org/10.3390/s17010198

Tasoulis, S. K., Mallis, G. I., Georgakopoulos, S. V., Vrahatis, A. G., Plagianakos, V. P., & Maglogiannis, I. G. (2019). Deep learning and change detection for fall recognition. *Communications in Computer and Information Science*, *1000*, 262–273. https://doi.org/10.1007/978-3-030-20257-6_22/COVER

Torti, E., Fontanella, A., Musci, M., Blago, N., Pau, D., Leporati, F., & Piastra, M. (2018). Embedded real-time fall detection with deep learning on wearable devices. *Proceedings - 21st Euromicro Conference on Digital System Design, DSD 2018*, 405–412. https://doi.org/10.1109/DSD.2018.00075

Torti, E., Fontanella, A., Musci, M., Blago, N., Pau, D., Leporati, F., & Piastra, M. (2019). Embedding Recurrent Neural Networks in Wearable Systems for Real-Time Fall Detection. *Microprocessors and Microsystems*, *71*, 102895. https://doi.org/10.1016/j.micpro.2019.102895

Usmani, S., Saboor, A., Haris, M., Khan, M. A., & Park, H. (2021). Latest research trends in fall detection and prevention using machine learning: A systematic review. *Sensors*, *21*(15), 1–23. https://doi.org/10.3390/s21155134

Vavoulas, G., Chatzaki, C., Malliotakis, T., Pediaditis, M., & Tsiknakis, M. (2016). The MobiAct dataset: Recognition of activities of daily living using smartphones. *ICT4AWE 2016 - 2nd International Conference on Information and Communication Technologies for Ageing Well and e-Health, Proceedings*, 143–151. https://doi.org/10.5220/0005792401430151

Vavoulas, G., Pediaditis, M., Spanakis, E. G., & Tsiknakis, M. (2013). The MobiFall dataset: An initial evaluation of fall detection algorithms using smartphones. *13th IEEE International Conference on BioInformatics and BioEngineering, IEEE BIBE 2013*.

https://doi.org/10.1109/BIBE.2013.6701629

Veeriah, V., Zhuang, N., & Qi, G. J. (2015). Differential recurrent neural networks for action recognition. *Proceedings of the IEEE International Conference on Computer Vision*, *2015 Inter*, 4041–4049. https://doi.org/10.1109/ICCV.2015.460

Voulodimos, A., Doulamis, N., Doulamis, A., & Protopapadakis, E. (2018). Deep Learning for Computer Vision: A Brief Review. *Computational Intelligence and Neuroscience*, *2018*. https://doi.org/10.1155/2018/7068349

Wang, L., Peng, M., & Zhou, Q. F. (2019). Fall Detection Based on Convolutional Neural Networks Using Smart Insole. *2019 5th International Conference on Control, Automation and Robotics, ICCAR 2019*, 593–598. https://doi.org/10.1109/ICCAR.2019.8813332

Wang, S., Chen, L., Zhou, Z., Sun, X., & Dong, J. (2016). Human fall detection in surveillance video based on PCANet. *Multimedia Tools and Applications*, *75*(19), 11603–11613. https://doi.org/10.1007/s11042-015-2698-y

What is a Butterworth RF Filter. (2022). Electronics Notes. https://www.collimator.ai/reference-guides/what-is-a-butterworth-low-pass-filter

Xu, J., He, Z., & Zhang, Y. (2019). CNN-LSTM Combined Network for IoT Enabled Fall Detection Applications. *Journal of Physics: Conference Series*, *1267*(1). https://doi.org/10.1088/1742-6596/1267/1/012044

Yao, G., Lei, T., & Zhong, J. (2019). A review of Convolutional-Neural-Network-based action recognition. *Pattern Recognit. Lett.*, *118*, 14–22.

Yhdego, H., Li, J., Morrison, S., Audette, M., Paolini, C., Sarkar, M., & Okhravi, H. (2019). Towards musculoskeletal simulation-aware fall injury mitigation: Transfer learning with deep cnn for fall detection. *Simulation Series*, *51*(5). https://doi.org/10.23919/SpringSim.2019.8732857

Zhang, Q., Zhang, M., Chen, T., Sun, Z., Ma, Y., & Yu, B. (2019). Recent advances in convolutional neural network acceleration. *Neurocomputing*, *323*, 37–51. https://doi.org/10.1016/j.neucom.2018.09.038

Zhang, Q., & Zhu, S. (2018). Real-time Activity and Fall Risk Detection for Aging

Population Using Deep Learning. *2018 9th IEEE Annual Ubiquitous Computing, Electronics and Mobile Communication Conference, UEMCON 2018*, 1055–1059. https://doi.org/10.1109/UEMCON.2018.8796672

Zhang, Z., Ma, X., Wu, H., & Li, Y. (2019). Fall detection in videos with trajectory-weighted deep-convolutional rank-pooling descriptor. *IEEE Access*, *7*, 4135–4144. https://doi.org/10.1109/ACCESS.2018.2887144

Glossário

Acelerómetro	Um acelerómetro é um instrumento usado para detetar e medir acelerações ou vibrações
ADL	As Atividades da Vida Diária (AVDs) englobam todas as atividades da vida cotidiana, as quais têm um valor, um significado concreto e um propósito para cada pessoa. As ocupações são centradas na identidade e nas capacidades de cada pessoa, e influenciam modo como cada um ocupa seu tempo e toma decisões.
ADXL345	Um sensor de aceleração de alta resolução, que é frequentemente utilizado para medir a aceleração em três eixos.
Apache Flink	O Apache Flink é uma framework e um *engine* (motor) de processamento distribuído para computações com estado sobre fluxos de dados limitados e ilimitados. Foi concebido para funcionar em todos os ambientes de clusters comuns, efetuar cálculos à velocidade da memória e em qualquer escala.
Auto-encoder	Um auto-encoder (AE) é um tipo específico de rede neuronal artificial não supervisionada que fornece compactação e outras funcionalidades em *machine learning*. O uso específico do auto-encoder é usar uma abordagem de *feed-forward* para reconstituir uma saída de uma entrada. A entrada é compactada e, em seguida, enviada para ser descompactada como saída, que geralmente é semelhante à entrada original. Essa é a natureza de um auto-encoder - que entradas e saídas semelhantes são medidas e comparadas para obter resultados de execução.
Bluetooth	Tecnologia que permite ligar e transferir dados entre equipamentos eletrónicos através de sinais de rádio.
Camada convolucional	Esta camada extrai recursos dos dados de entrada e devolve-os para a próxima camada.
CNN	No contexto de inteligência artificial e *machine learning*, uma rede neuronal convolucional (CNN do inglês Convolutional Neural network ou ConvNet) é uma classe de rede neuronal artificial do tipo *feed-forward.*

Confusion Matrix	A matriz de confusão é uma tabela que permite extrair métricas que auxiliam na avaliação de modelos de *machine learning* para classificação — quando a variável resposta é categórica
Continuous Wavelet Transform	Técnica de análise de sinais que permite visualizar as diferentes frequências presentes num sinal ao longo do tempo.
CUSUM - cumulative sum	Técnica estatística que monitoriza a soma acumulada de desvios em relação a um valor de referência.
Datasets	Bases de dados estruturadas que são utilizadas para treinar, validar e testar modelos de computação.
Edge computing	A computação edge permite que os dispositivos em localizações remotas processem dados na "periferia" da rede, seja pelo dispositivo ou por um servidor local. E, quando os dados precisam de ser processados no *datacenter* central, apenas os dados mais importantes são transmitidos, minimizando, assim, a latência.
FD-CNN	Técnica que aplica convoluções neuronais a sinais não uniformes ou irregulares.
Fog Computing	Computação em névoa ou em inglês: fog computing (também chamada computação em neblina, computação de periférica ou nevoeiro, que deriva da denominação *fogging*), consiste na alocação do poder de processamento mais perto do limite da rede. Portanto é uma arquitetura de computação descentralizada onde dados, cálculos, comunicações, armazenamentos, medições, aplicações e gestão são distribuídos no local mais adequado e eficiente - entre a fonte de dados e a nuvem.
Fold cross-validation	A validação cruzada é um método de avaliação utilizado em *machine learning* para descobrir até que ponto o modelo de aprendizagem consegue prever o resultado de dados não utilizados. É um método fácil de compreender, funciona bem para uma amostra de dados limitada e também oferece uma avaliação menos tendenciosa, o que o torna numa escolha popular.

FPGA	Em inglês *field programmable gate array*, é um dispositivo semicondutor que é largamente utilizado para o processamento de informações digitais. Foi criado pela Xilinx Inc., e teve o seu lançamento no ano de 1985 como um dispositivo que poderia ser programado de acordo com as aplicações do utilizador (programador).
Framework	Uma estrutura ou conjunto de ferramentas que auxiliam no desenvolvimento de software ou outras soluções.
Fully connected	Um tipo de arquitetura de rede neuronal em que cada neurónio numa camada está ligado a todos os neurónios na camada seguinte.
Gating	O controlo do fluxo de informação numa rede neural ou sistema, muitas vezes usando mecanismos como portas.
Giroscópio	Um sensor que mede a velocidade de rotação de um objeto em torno de um eixo.
Hidden layers	Camadas intermédias numa rede neuronal entre a camada de entrada e a camada de saída, onde ocorre o processamento dos dados.
Intel PXA255	Um processador da Intel frequentemente utilizado em dispositivos portáteis e sistemas *embedded*.
ITG3200	Um sensor de giroscópio de três eixos que mede a taxa de rotação em torno de três eixos e é comumente usado em aplicações de orientação e controlo
Kinematics	O estudo do movimento de corpos sem considerar as forças envolvidas.
k-Nearest Neighbors (kNN)	o KNN tenta classificar cada amostra de um conjunto de dados avaliando sua distância em relação aos vizinhos mais próximos. Se os vizinhos mais próximos forem maioritariamente de uma classe, a amostra em questão será classificada nesta categoria

LPWAN Uma rede de comunicação sem fios desenhada para transmitir dados a longas distâncias com baixo consumo de energia.

LSTM Um tipo de rede neuronal recursiva usada para modelar sequências e dependências temporais - séries temporais

Machine Learning Ramo da Inteligência Artificial que se foca em desenvolver algoritmos que permitem aos sistemas aprender a partir de dados.

Magnetómetro Um sensor usado para medir campos magnéticos.

MbientLab Uma empresa que desenvolve hardware e software para aplicações de Internet das Coisas (IoT) e dispositivos Wearable.

MetaMotionR Uma plataforma de sensores portátil que pode ser usada para recolher dados diversos, como movimento, aceleração e outros parâmetros, frequentemente utilizada em aplicações de saúde e movimento.

MMA8451Q Um sensor de aceleração de três eixos que mede a aceleração em três direções e é frequentemente usado em dispositivos eletrónicos para detetar movimento e orientação.

MobiAct Um conjunto de dados de sensores de movimento recolhidos a partir de dispositivos móveis.

MobiFall Um conjunto de dados de sensores de movimento para deteção de quedas..

MPU6050 Um sensor de movimento de seis eixos, medindo aceleração e velocidade angular, frequentemente usado em dispositivos eletrónicos como smartphones.

Pordata A Pordata é uma base de dados sobre Portugal contemporâneo com estatísticas oficiais e certificadas sobre o país e a Europa, dividida num amplo conjunto de temas como a população, educação, saúde, entre outros. Esta está disponível para todos os cidadãos, é gratuita, de informação rigorosa e isenta. Toda a sua informação provém de entidades oficiais, tais como o Instituto Nacional de Estatística ou o Eurostat. O total das fontes

	utilizadas está neste momento perto de 60. Todos os dados são apresentados de uma forma anual e, sempre que possível, remontam a 1960.
RNN	Uma rede neuronal recorrente (RNN) é um tipo de rede neuronal artificial que utiliza dados sequenciais ou dados de séries temporais. Estes algoritmos de deep learning são normalmente utilizados para problemas temporais, como a tradução de línguas, o processamento de linguagem natural (NLP), o reconhecimento de voz e a legendagem de imagens; estão incorporados em aplicações populares como o Siri, a pesquisa por voz e o Google Translate.
SensorTile	Uma plataforma que integra sensores em dispositivos compactos, frequentemente utilizados para prototipagem e desenvolvimento de soluções IoT.
Séries temporais	Uma sequência de pontos de dados ordenados no tempo, frequentemente usada para análise de padrões e previsões.
SHAR	Dataset que contém dados de sensores de atividade humana recolhidos a partir de smartphones.
SHIMMER	Uma plataforma de sensores Wearable projetada para capturar uma variedade de sinais biomédicos.
SISFALL	É um dataset que contém informações sobre atividades e quedas humanas, utilizado para avaliar algoritmos de deteção de quedas.
SmartFall	Uma aplicação ou sistema projetado para detetar e responder a quedas, muitas vezes integrando tecnologias de sensores.
Softmax	Uma função matemática frequentemente usada em redes neuronais para converter um vetor de valores em probabilidades.
SVM	Um algoritmo de *machine learning* usado para classificação e regressão.
Tensorflow	Uma biblioteca de código aberto usada para desenvolver e treinar modelos de *machine learning*.

Texas Instruments MSP430	Uma família de microcontroladores de baixa potência frequentemente usados em dispositivos.
URFD	Um conjunto de dados para deteção de quedas humanas, frequentemente usado em R & D de algoritmos de deteção.
Wearable	Dispositivos eletrónicos que podem ser usados como acessórios ou roupas, frequentemente projetados para monitorizar atividades físicas e de saúde.
Wide generalization	Tem como objetivo fazer com que um modelo de rede neuronal funcione bem numa ampla variedade de cenários e dados, não apenas nos quais foi treinado.
ZigBee	Um protocolo de comunicação sem fio de curto alcance frequentemente usado em dispositivos IoT.
ZigBee Transceivers	Dispositivos que permitem a comunicação sem fio de acordo com o protocolo ZigBee.