TÉCNICO LISBOA

Deteção de incêndios florestais através da aprendizagem auto-supervisionada

Sara Alexandra Curado Fernandes

Dissertação para obtenção do Grau de Mestre em

Engenharia Eletrotécnica e de Computadores

Orientadores: Profª. Ana Catarina Fidalgo Barata
Prof. Alexandre José Malheiro Bernardino

Júri

Presidente: Prof. João Fernando Cardoso Silva Sequeira
Orientador: Profª. Ana Catarina Fidalgo Barata
Vogal: Prof. Hugo Pedro Martins Carriço Proença

Dezembro 2021

Declaração

Declaro que o presente documento é um trabalho original da minha autoria e que cumpre todos os requisitos do Código de Conduta e Boas Práticas da Universidade de Lisboa.

Agradecimentos

A presente dissertação traduz a conquista de várias etapas de um percurso académico de 5 anos, repleto de desafios e aprendizagens que contribuíram para o meu desenvolvimento pessoal. Durante este período pude contar com o apoio de inúmeras pessoas às quais quero deixar o meu sincero agradecimento.

Aos meus orientadores Professora Catarina Barata e Professor Alexandre Bernardino pelo apoio, partilha de conhecimentos e disponibilidade fundamentais para a conclusão desta dissertação. A todos os professores e alunos que fizeram parte das reuniões quinzenais, pela ajuda e partilha de ideias durante o desenvolvimento desta tese.

Aos meus pais, irmãos e avós, pelo incentivo, confiança e valores transmitidos durante toda a minha vida que ajudaram na superação deste desafio.

Aos meus colegas e amigos, com quem tive a oportunidade e privilégio de partilhar este percurso, pelo apoio e disponibilidade nos bons momentos e nos tempos mais difíceis.

À Junitec, pelos momentos de aprendizagem e convívio e por me incentivarem a procurar novos desafios e oportunidades.

À equipa de voleibol da AEIST, por me mostrar que com esforço e dedicação podemos alcançar os objetivos que traçamos.

Abstract

Wildfires are one of the most challenging disasters to control and are responsible for thousands of hectares burned, infrastructure destroyed, and lives lost every year all over the world. To develop robust systems capable of detecting and locating wildfires with high efficiency through supervised learning, it is necessary to acquire extensive labelled datasets. However, the number of publicly available images with associated annotations is low. Moreover, there are thousands of images related to forest fires available online, but without annotation. Furthermore, generating all the required labels can be time-consuming and costly. Therefore, this thesis hopes to take advantage of this unlabelled data by proposing a system that uses self-supervised learning to achieve the final goal of fire classification. The proposed methodology is divided into two phases. First, a network is trained to solve some tasks, different to the final task, using the unlabelled images. Afterwards, by combining part of the learned model with a small, labelled dataset, the final classifier is achieved. When comparing the models only trained with the small labelled dataset, the proposed methodology achieved a better performance, proving that there are advantages to using the unlabelled data available online.

Keywords

Self-supervised Learning; Deep Learning; Wildfire; Convolutional Neural Networks; UAVs;

Resumo

Os incêndios florestais são um dos desastres mais árduos de controlar, sendo por isso responsáveis por milhares de hectares ardidos, infraestruturas destruídas e vidas perdidas, todos os anos e em todo o mundo. Para desenvolver sistemas robustos capazes de detetar e localizar incêndios florestais com elevada eficácia através de aprendizagem supervisionada é necessário adquirir conjuntos de dados legendados extensos. Porém, a quantidade de imagens com anotações disponíveis é reduzida dificultando assim o desenvolvimento deste tipo de métodos. Por outro lado, existem milhares de imagens sem anotações disponíveis online, contudo, produzir as respetivas legendas é uma tarefa dispendiosa quer a nível de tempo e quer a nível monetário. Desta forma, esta tese pretende tirar partido das imagens não legendadas recorrendo à aprendizagem auto-supervisionada e cujo objetivo final é obter-se um modelo de classificação de fogo numa imagem. Em suma, foi desenvolvida uma metodologia que está dividida em duas fases. Primeiramente, é treinado um modelo a resolver determinadas tarefas, distintas da tarefa final, através das imagens não legendadas. Posteriormente, utilizando parte do modelo previamente ensinado em conjunto com os escassos dados legendados, treina-se o classificador final. Em comparação com modelos treinados apenas com um conjunto escasso de imagens legendadas, o sistema proposto atinge uma melhor performance, demonstrando a vantagem de utilizar as imagens não legendadas.

Palavras Chave

Aprendizagem Auto-supervisionada; Aprendizagem Profunda; Incêndios Florestais; Redes Neuronais Convolucionais; UAVs.

Conteúdo

Lista de Figuras

Lista de Tabelas

Acrónimos

ANN *Artifitial Neural Networks*

CNN *Convolutional Neural Network*

FC *Fully Connected*

FN Falso Negativo

FP Falso Positivo

HSV *Hue Saturation Value*

ReLU *Rectified Linear Unit*

RGB *Red Green Blue*

UAVs *Unmanned Aerial Vehicles*

VN Verdadeiro Negativo

VP Verdadeiro Positivo

1

Introdução

Conteúdo

Os incêndios florestais são, anualmente, responsáveis por milhares de hectares ardidos, infraestruturas destruídas e vidas humanas perdidas. Somente em Portugal continental, os incêndios florestais foram responsáveis por mais de 861 mil hectares ardidos, entre 2016 e 2020 [1].

A área florestal em Portugal ocupa 3.2 milhões de hectares, representando 36% da área total do território português. Assim, a monitorização de toda esta área é uma tarefa extremamente difícil, porém indispensável. Inicialmente, a monitorização dependia unicamente do Homem, através de guardas-florestais distribuídos pelas várias torres de vigilância. Mais recentemente, implementou-se um sistema de vigilância com recurso a câmaras fixas em torres espalhadas pela floresta, através do sistema CICLOPE [2]. Contudo, nem toda a área florestal é observada por essas câmaras pelo que a introdução de um sistema de monitorização em veículos aéreos é crucial para o acompanhamento de incêndios florestais.

1.1 Motivação e Objetivo

O projeto Firefront [1], no qual se insere esta tese, pretende automatizar a deteção, o acompanhamento e o combate de fogos florestais e eventuais reacendimentos e, desta forma, ajudar as equipas de combate aos incêndios a extinguir os fogos.

Tirando partido da tecnologia e, mais especificamente, da visão por computador e aprendizagem profunda, pretende-se automatizar a deteção de incêndios florestais, pois esta pode ter um papel crucial no desenvolvimento dos mesmos, ajudando a minimizar os danos provocados pelos fogos. Estas tecnologias têm mostrado o seu contributo nas mais diversas áreas, auxiliando na resolução de tarefas complexas.

No que diz respeito à deteção de incêndios, é benéfico que estas técnicas sejam implementadas em veículos aéreos tripulados ou não tripulados, para que toda a área florestal possa ser observada. Ademais, estes veículos devem estar equipados com uma câmara para captar imagens da floresta e, consequentemente, de eventuais fogos.

Infelizmente, o fogo, ao contrário de outro tipo de objetos, tem uma forma irregular, pelo que a aprendizagem de um sistema baseado em visão por computador é uma tarefa difícil. Para combater este obstáculo é benéfico utilizar um conjunto de dados extenso, todo ele legendado. Apesar de existirem milhares de imagens com fogo e relacionadas com incêndios florestais, os conjuntos de imagens com anotações disponíveis são escassos e contém poucos dados. Para obter as respetivas anotações é necessário fazê-lo manualmente o que implica um grande consumo de tempo e um custo monetário elevado. De acordo com [3], para anotações ao nível da imagem, demora-se cerca de 10 segundos por imagem, incrementando para 6 minutos e 41 segundos quando se pretende anotações ao nível do

[1] http://www.firefront.pt/

pixel.

Com intuito de superar as adversidades enunciadas, sugere-se o desenvolvimento de uma metodologia que aproveite os milhares de imagens disponíveis *online* sem que seja necessário fazer algum tipo de anotação manual. Assim, tem-se como objetivo elaborar um sistema que permita classificar a presença de fogo numa imagem utilizando uma metodologia baseada em aprendizagem auto-supervisionada, como ilustrado na Figura 1.1. Neste tipo de abordagem, continua a ser necessário a utilização de imagens legendadas, porém em quantidades inferiores.

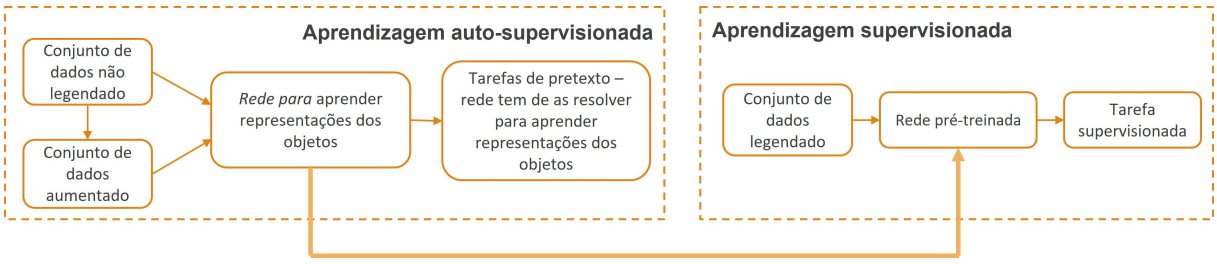


Figura 1.1: Funcionamento da aprendizagem auto-supervisionada.

1.2 Estrutura do documento

Esta tese está dividida em seis capítulos. O Capitulo 2 apresenta uma descrição de alguns conceitos teóricos e ainda a revisão da literatura. A metodologia proposta com a explicação das várias etapas é exposta no Capítulo 3. No Capítulo 4, são apresentadas todas as componentes utilizadas para treinar e avaliar o sistema proposto, incluindo as diversas experiências efetuadas. O Capítulo 5 mostra os resultados obtidos nas várias experiências. Finalmente, o Capítulo 6 apresenta as principais conclusões do trabalho proposto e ainda sugestões para implementar no futuro.

2

Contexto Teórico e Literatura

Conteúdo

Este capítulo começa por introduzir alguns conceitos teóricos cruciais para a compreensão dos temas abordados ao longo da dissertação. Inicialmente, é feita uma introdução às redes neuronais artificiais e às redes neuronais convolucionais responsáveis pela aprendizagem profunda. De seguida, procede-se com uma breve explicação sobre a aprendizagem auto-supervisionada, apresentando os principais objetivos e diferentes tipos de abordagens possíveis da mesma. Finalmente, são expostos os principais trabalhos desenvolvidos na área de deteção de incêndios e os métodos mais relevantes da aprendizagem auto-supervisionada.

2.1 Contexto teórico

2.1.1 Redes Neuronais Artificiais

As redes neuronais artificiais, do inglês *Artifitial Neural Networks* (ANN), têm como inspiração o cérebro humano [4], reproduzindo os sinais que os neurónios emitem entre si. A principal vantagem das redes neuronais traduz-se na obtenção automática das características mais relevantes, sem necessitar de supervisão humana. Estas redes estão organizadas em múltiplas camadas e cada uma contém um conjunto de neurónios artificiais. Tipicamente, as redes têm uma camada de entrada, uma ou mais camadas escondidas e uma camada de saída. Cada neurónio liga-se a todos os neurónios da camada seguinte e tem associado um peso e um limiar. Dependendo se o valor atribuído ao neurónio é superior ao do limiar, o mesmo pode estar ativo enviando os dados para a camada seguinte da rede, ou inativo e neste caso deixa de transmitir os dados para a camada seguinte. Cada neurónio é representado pela expressão matemática

$$y = \phi(\sum_{i=1}^{n} w_i x_i), \tag{2.1}$$

onde x_i é a entrada, w_i o peso associado a cada neurónio i da camada anterior e ϕ a função de ativação. Quando a função ϕ não é utilizada, ou corresponde a uma função linear, a entrada de cada camada será uma função linear da saída da camada anterior, pelo que a rede não é capaz de identificar padrões complexos nos dados. Por esta razão, a utilização de funções de ativação não lineares é crucial de forma a melhorar a habilidade da rede a ajustar os dados. Existem diversas funções de ativação que permitem a aprendizagem da rede, estando representadas na Tabela 2.1 algumas das funções de ativação mais comuns [4].

Treino

Quando se inicia uma rede neuronal, esta tem os seus pesos, w_i, inicializados aleatoriamente, pelo que os resultados obtidos numa fase inicial não são adequados. Assim, é necessário submeter a rede a um treino para que os pesos dos neurónios sejam ajustados à tarefa pretendida. Para realizar esses ajustes, recorre-se a uma função de custo e a rede tem como objetivo minimizar o custo, isto é, erro

Tabela 2.1: Funções de ativação.

Função de ativação	Expressão	Descrição
Linear	$f(x) = x$	Função linear também conhecida por função identidade; Não altera a soma ponderada da entrada, retornando diretamente o valor desta.
ReLU	$f(x) = \max(0, x)$	Função não linear; É simples, fácil de calcular e permite a retro propagação do gradiente, no entanto quando a entrada é nula ou negativa a aprendizagem torna-se mais difícil.
Sigmoid	$f(x) = \frac{1}{1+e^{-x}}$	Apresenta a forma de um 's' e tem valores de saída compreendidos entre 0 e 1.
Tanh	$f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	Apresenta uma forma semelhante à função de ativação sigmoid, porém os valores de saída variam entre -1 e 1.

entre a predição da rede, $(\hat{y}_i)$, e o resultado esperado, (y_i). Esta função é desenhada com base na tarefa que se pretende resolver com intuito de garantir que a rede devolve os resultados previstos. A atualização dos pesos e dos filtros é feita de acordo com o valor obtido pela função de custo e através da retro propagação gradiente descendente utilizando algoritmo de otimização [5].

A tabela 2.2 apresenta duas das funções de custo mais comuns e utilizadas para o desenvolvimento desta tese.

Tabela 2.2: Funções de custo.

Função de custo	Expressão
Entropia-Cruzada Binária	$\mathcal{L}_{BCE} = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log \hat{y}_i + (1-y_i) \cdot \log(1-\hat{y}_i)$
Erro Quadrático Médio	$\mathcal{L}_{MSE} = \frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y}_i)^2$

2.1.2 Redes Neuronais Convolucionais

As redes neuronais convolucionais, do inglês *Convolutional Neural Network* (CNN), são uma categoria de redes neuronais que têm vindo a revelar-se extremamente úteis e eficazes para tarefas como

deteção de objetos, classificação e segmentação de imagens. O êxito deste tipo de redes provém de três aspetos [6]. Por um lado, passa-se a ter ligações locais, em que cada neurónio está apenas ligado a um pequeno conjunto de neurónios da camada seguinte, reduzindo, significativamente, o número de parâmetros que a rede deve aprender assim como a velocidade do treino. O número de parâmetros é ainda mais reduzido com a partilha de pesos entre ligações. Finalmente, a introdução de uma camada *pooling* explora o princípio da correlação local, submetendo a entrada a um processo de compressão reduzindo a dimensão dos dados sem perder a informação mais relevante.

O algoritmo das redes CNN pode ser dividido em duas partes. A primeira corresponde à extração dos mapas de características e a segunda é responsável por executar a tarefa que foi imposta, por exemplo, classificação ou segmentação.

Camadas

As redes neuronais convolucionais são compostas por múltiplos blocos que podem conter camadas convolucionais, camadas *pooling* e ainda camadas totalmente ligadas, do inglês *Fully Connected* (FC). Os dois primeiros tipos de camadas estão associados à extração dos mapas de características enquanto que a última é responsável pela tarefa em concreto. De uma forma geral, as arquiteturas dos modelos que usam redes CNN para classificação têm o formato representado na Figura 2.1.

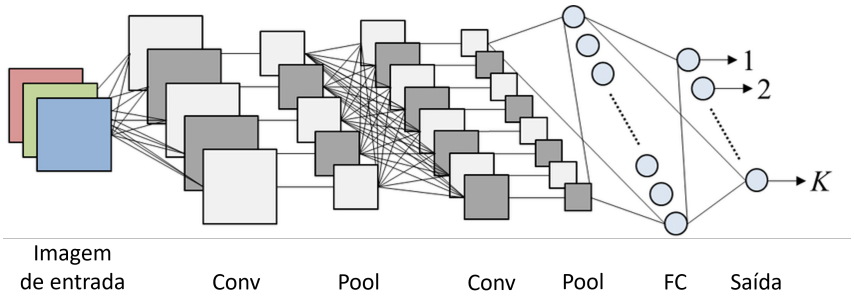


Figura 2.1: Visão geral de uma rede CNN. (Imagem adaptada de [7])

1. Camadas Convolucionais

   As camadas convolucionais são uma componente fundamental das redes CNN, sendo sempre a primeira camada das mesmas. Cada camada consiste num conjunto de filtros convolucionais, também conhecidos por *kernels* [8]. Estes filtros correspondem a uma matriz com vários pesos, cujos valores iniciais são atribuídos aleatoriamente. À medida que a rede vai avançando no treino estes pesos são ajustados de forma a ser possível extrair-se as principais características. Através destes filtros e recorrendo à operação convolução são extraídos os mapas de características da imagem de entrada. A operação consiste no produto escalar entre a imagem e um filtro. Em suma, o *kernel* percorre a imagem na sua totalidade, horizontalmente e verticalmente, e à medida

que o filtro passa por determinado segmento da imagem estes são multiplicados e somados de forma a resultar num único escalar. A Figura 2.2 apresenta um exemplo da operação descrita.

É de realçar que as primeiras camadas da rede deste tipo são responsáveis por extrair as características de alto nível, tais como cantos e arestas dos objetos e as últimas procuram por padrões e texturas.

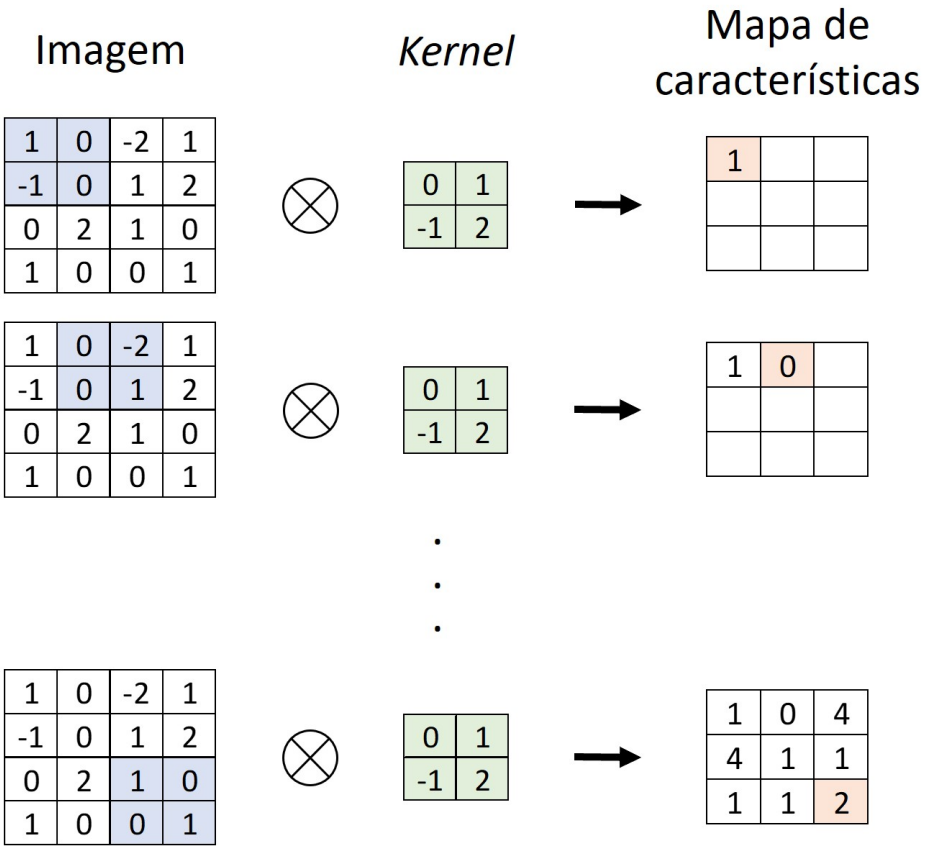


Figura 2.2: Funcionamento da camada de convolução.

Geralmente, utiliza-se a função de ativação *Rectified Linear Unit* (ReLU) para introduzir uma componente não linear à operação e assim permitir que a rede aprenda mais rapidamente e através de conjuntos de dados mais extensos e complexos.

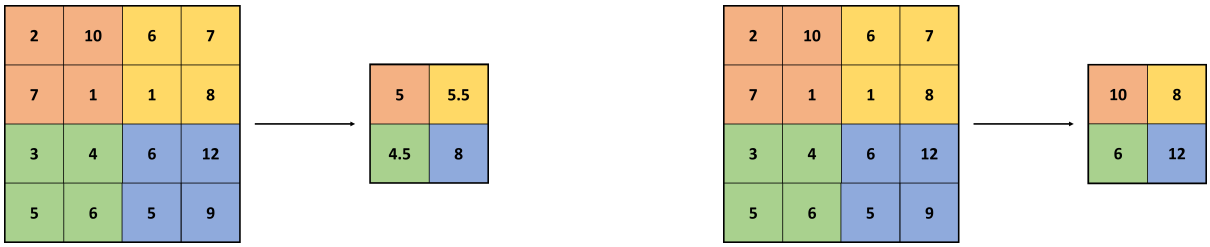
Os mapas de características resultantes de uma determinada camada convolucional memorizam a posição dessas características na entrada, pelo que pequenas alterações das posições, por exemplo, rotação e espelhamento, resulta em mapas de características totalmente distintos. Para que a rede não memorize, a saída da camada convolucional deve ser submetida a um processo de compressão de forma a reter os aspetos mais importante dos mapas de características mas não tão detalhado pois pode não ser benéfico para a tarefa a executar. Uma das formas de executar a compressão é aplicar uma camada *pooling*.

2. Camadas *Pooling*

   As camadas *pooling* são aplicadas a seguir a uma camada convolucional e tem como finalidade tornar a rede invariante no espaço, isto é, invariante a translações e distorções da entrada, reduzindo a dimensão da matriz de entrada sem perder informação relevante [9]. Assim, mesmo havendo variações da localização das características detetadas pela camada de convolução, o mapa de características resultante da camada *pooling* é invariante a translações locais. Cada mapa de características de uma camada *pooling* está ligado ao mapa de características da camada de convolução correspondente.

   Existem diversos tipos de camadas *pooling*, sendo as mais tradicionais:

   - *Average pooling* - Calcula o valor médio de cada segmento do mapa de características, Figura 2.3a
   - *Max pooling* - Calcula o valor máximo de cada segmento do mapa de características, Figura 2.3b



(a) *Average pooling* **(b)** *Max pooling*

Figura 2.3: Exemplos de camadas *pooling*.

3. Camadas Totalmente Ligadas

   Uma vez obtido os mapas de características através de um conjunto de camadas convolucionais e de *pooling*, estes são convertidos num vetor de forma a efetuar a classificação da imagem. Para a classificação são utilizadas uma ou mais camadas totalmente ligadas, cuja principal diferença face às camadas convolucionais está relacionada com a ligação entre os neurónios de duas camadas consecutivas. No caso da camada totalmente ligada todos os neurónios estão ligados à camada anterior.

Classificação

A classificação corresponde ao processo de prever a que classe pertencem os dados de entrada. Para efetuar a classificação, a rede CNN termina com uma camada totalmente ligada e o número de

neurónios é idêntico ao número total das diferentes categorias a classificar. Se na imagem estiver presente uma determinada classe, o neurónio correspondente apresentará uma probabilidade elevada.

2.1.3 Aprendizagem Auto-supervisionada

A aprendizagem auto-supervisionada é um subconjunto da aprendizagem não supervisionada que tem vindo a ganhar interesse nos últimos anos. Tal deve-se ao facto de tirar partido de uma grande quantidade de imagens não legendadas que são geradas constantemente, ao contrário da aprendizagem supervisionada cujas imagens têm de estar legendadas para se treinar os modelos.

O treino de uma rede, geralmente CNN, através da aprendizagem auto-supervisionada, consiste na resolução de uma ou mais tarefas de pretexto, de forma a aprender as características visuais de um objeto [10]. Para resolver a tarefa são geradas, automaticamente, legendas a partir do conjunto de dados e com base na informação necessária para resolver a tarefa escolhida. Desta forma, a rede aprende de uma forma semelhante à aprendizagem supervisionada, na medida em que existem legendas, contudo estas não requerem anotações por parte do humano, reduzindo-se, significativamente, o tempo e custo necessários para legendar as imagens.

Perante a rede treinada, a componente correspondente ao *encoder*, isto é, camadas convolucionais e *pooling*, é extraída para ser utilizada na tarefa supervisionada final pretendida, por exemplo, classificação ou segmentação. Existem duas formas de utilizar os pesos extraídos do *encoder* pré-treinado. Por um lado, fazendo *transfer learning* que consiste na utilização dos pesos determinados para uma dada tarefa de pretexto, congelando os pesos das camadas do *encoder* quando se pretende resolver a tarefa final. Em oposição, *fine-tuning* corresponde à utilização dos pesos previamente aprendidos exclusivamente como inicialização do encoder da nova rede, retreinando os mesmos para a tarefa final. De um modo geral, os *encoders* desempenham um papel fundamental na aprendizagem auto-supervisionada, pois são responsáveis por compactar a imagem de entrada, retornando a sua representação no espaço latente sob a forma de um vector [11] que será utilizado para a fase supervisionada.

Categorias

De acordo com [12], a aprendizagem auto-supervisionada pode ser dividida em três categorias, geração, contrastiva e geração-contrastiva. As principais diferenças entre as três categorias correspondem à arquitetura dos modelos bem como à tarefa que têm de executar. Do ponto vista probabilístico, dada a distribuição conjunta $P(X;Y)$ da entrada X e a legenda Y, os métodos de geração determinam $P(X|Y=y)$ enquanto que os métodos contrastivos calculam $P(Y|X=x)$.

A Figura 2.4 apresenta as operações e componentes gerais das diferentes categorias.

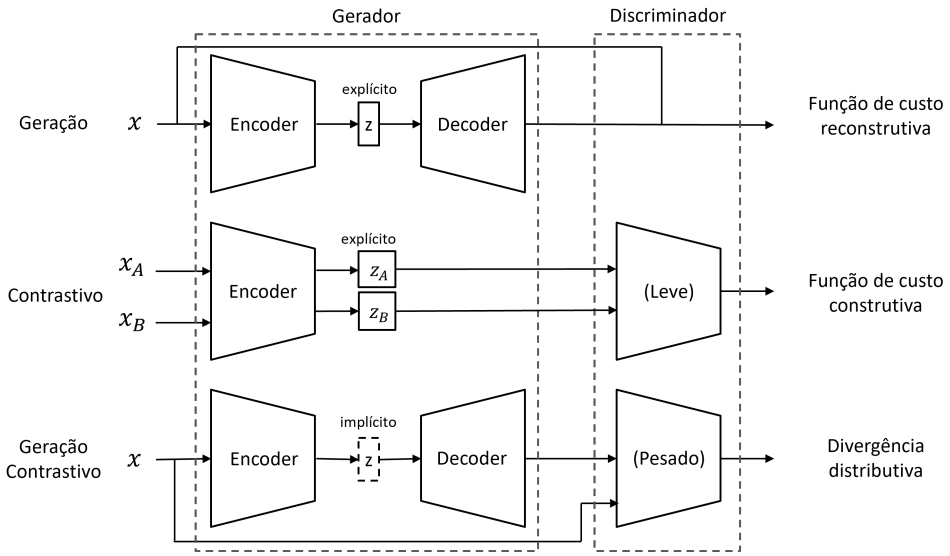


Figura 2.4: Comparação das diferentes categorias da aprendizagem auto-supervisionada. (Imagem adaptada de [12])

1. Métodos de Geração

   Recorrendo à nomenclatura da Figura 2.4, os métodos baseados na geração utilizam um *encoder* para obter o vetor com a representação no espaço latente, z, da entrada x. De seguida, para reconstruir x através de z é utilizado um *decoder*, resultando $\hat{x}$. Finalmente, a partir de uma função de custo reconstrutiva, compara-se a saída $\hat{x}$ com a entrada x e atualizam-se os pesos do *encoder* e *decoder*.

   A função de custo, nos métodos de geração, é calculada no espaço da saída, isto é, a partir das legendas. Um exemplo mais simples deste tipo de métodos é a utilização de um *autoencoder* [13] para reconstruir o sinal de entrada, sem aplicar qualquer tipo de distorção ao mesmo. A função de custo é determinada comparando a imagem gerada pela rede, $\hat{x}$, e sua legenda correspondente x.

   Por outro lado, uma tarefa mais complexa corresponde à adição de ruído ao sinal de entrada e o objetivo é que o *autoencoder* retorne a entrada sem o ruído [14], tornado o modelo mais robusto à introdução desta perturbação.

2. Métodos Contrastivos

   Os métodos contrastivos, tal como o nome indica, envolvem a aprendizagem das representações através da comparação entre dois elementos. Este tipo de aprendizagem pode ainda ser dividido em duas subcategorias [12].

   Por um lado, existem os métodos de contraste global-local, cujo objetivo é determinar a relação

entre as características locais da amostra e o contexto global da representação. A aprendizagem pode ser feita de acordo com a predição da posição relativa entre as características da imagem ou através da maximização da informação mútua.

Por outro lado, os métodos global-global estudam a relação entre as representações globais de duas amostras. Mais concretamente, são definidos pares positivos e pares negativos, como apresentado na Figura 2.4. Para se obter os pares positivos e negativos, é necessário aumentar os dados através de aplicação de transformações às imagens. Desta forma, os pares positivos correspondem às réplicas provenientes da mesma imagem original e os restantes são designados por pares negativos. Nestes métodos, a função de custo é calculada no espaço latente, ou seja, utilizando os vetores z_A e z_B que contêm as representações correspondentes às imagens de entrada x_A e x_B.

3. Métodos Geração-Contrastivos

   Este tipo de metodologia, também designado por *Adversarial Learning*, é uma alternativa aos métodos de geração na medida em que aprende a reconstruir a distribuição dos dados originais em vez dos próprios dados, tentando minimizar a divergência distributiva [12]. Por outras palavras, o processo de treino deste tipo de metodologia resume-se na geração de amostras fictícias e o discriminador tenta distingui-las das amostras reais através de comparações entre as diversas amostras, reais e fictícias. Ademais, os métodos *adversarial* utilizam um *autoencoder*, tal como nos métodos de geração e em oposição aos métodos contrastivos que apenas incluem a componente *encoder*. Tal arquitetura favorece a tarefa de geração, mas torna a componente de contraste mais desafiante na medida em que o *decoder*, ao reconstruir os dados de entrada, contém informação em excesso e para estas tarefas apenas é necessário aprender a informação que é diferente entre as amostras para distinguir as mesmas.

2.2 Literatura

Nesta secção está presente um resumo dos métodos mais relevantes na aprendizagem auto-supervisionada e ainda uma revisão da literatura sobre deteção de fogos.

2.2.1 Metodologias para Aprendizagem Auto-supervisionada

Os métodos contrastivos são aqueles que apresentam resultados mais significativos e ao nível dos métodos supervisionados.

Na categoria de métodos contrastivos global-local, os autores de [15] desenvolveram uma tarefa que tem como objetivo identificar a rotação aplicada à imagem de entrada, recebendo como entrada a

imagem transformada e cuja legenda é a rotação que lhe foi aplicada. Para que a rede consiga identificar a rotação, esta necessita de aprender a localizar os objetos mais salientes da imagem, reconhecer a orientação de cada um dos objetos e relacionar com a orientação dominante dos vários objetos. A Figura 2.5 mostra o processo da resolução da tarefa.

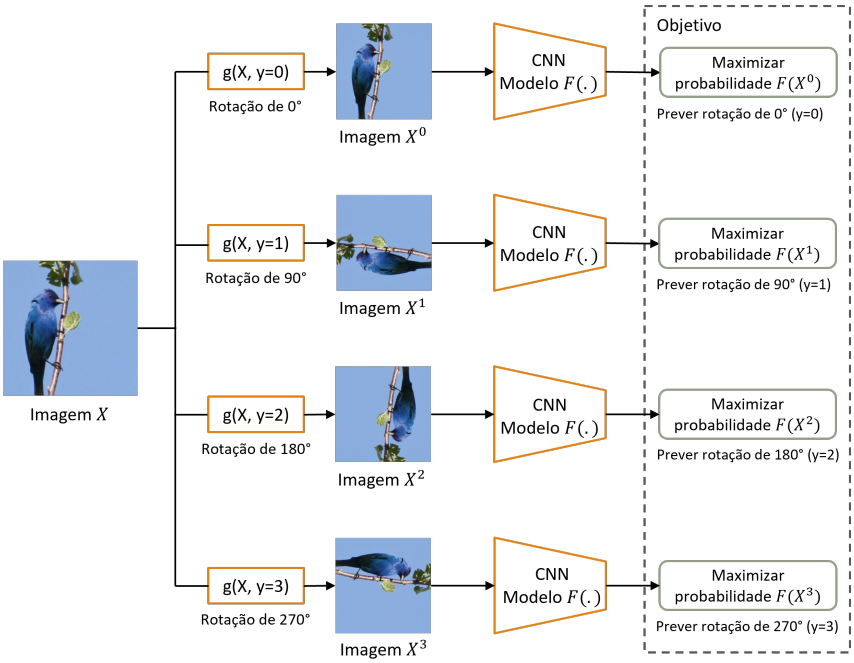


Figura 2.5: Ilustração da resolução da tarefa proposta por [15]. (Imagem adaptada)

No que diz respeito aos métodos contrastivos global-global, o método SimCLR [16], propõem que a cada imagem das N amostras do *batch*, sejam aplicadas duas transformações distintas, tal como ilustra a Figura 2.6, obtendo-se, assim, $2N$ réplicas no total. Através de comparações entre os vetores z_i e z_j o algoritmo deve retornar um valor de semelhança elevado caso sejam um par positivo ou baixo se coincidirem com um par negativo. Este método, apesar de conseguir resultados interessantes, tem a desvantagem de necessitar de *batches* com dimensões elevadas e consequentemente de conjuntos de treino extensos.

No caso de SwAV proposto em [17], este não recorre a comparações entre pares de imagens, baseando se numa abordagem por *clusters*. Inicialmente, é feito o aumento de dados através de transformações de tal forma que cada imagem tenha diversas réplicas no conjunto final de treino. Através de uma rede CNN, f_θ, obtêm-se as respetivas representações. Cada vetor de representações é utilizado como entrada de uma outra rede composta por uma única camada FC. Esta camada, denominada por camada de protótipos, não contem nenhum tipo de não linearidade, determinando apenas o produto escalar entre os protótipos, isto é, pesos da camada e o vetor de representações. Os pesos da camada são atualizados durante a retro propagação do gradiente. O resultado do produto escalar

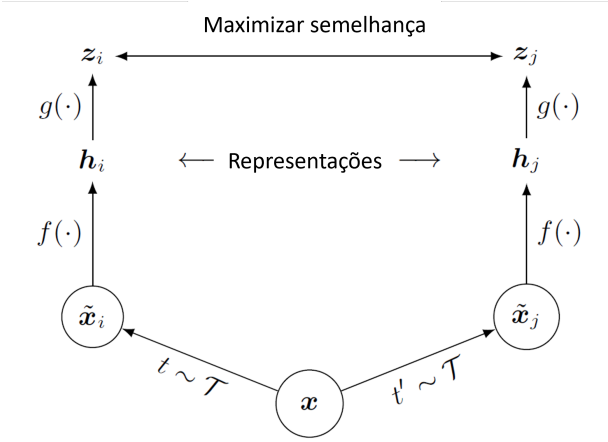


Figura 2.6: Metodologia proposta em [16]. (Imagem adaptada)

corresponde ao código associado a uma dada réplica. A intuição por detrás deste método resume-se à atribuição dos mesmos códigos às réplicas provenientes da mesma imagem original. Para garantir a consistência dos códigos atribuídos, a função de custo é calculada entre a representação de uma réplica e o código atribuído a outra versão da mesma imagem. Por outras palavras, se duas réplicas da mesma imagem contêm informação semelhante então tem de ser possível prever o código de uma através da representação da outra.

Finalmente, os autores de Barlow Twins [18] desenvolveram um método contrastivo semelhante ao SimCLR. Também neste método existem pares positivos e pares negativos obtidos de igual forma mantendo-se o mesmo objetivo, ou seja, aproximar as representações dos pares positivos e repelir as representações dos pares negativos. No entanto, desenvolveram uma outra função de custo que dispensa de *batches* de grandes dimensões para atingir bons resultados. Em suma, a função de custo é calculada através da matriz de correlação cruzada entre os vetores das representações do *batch* e é composta por dois termos. O primeiro corresponde ao termo de invariância que se assegura que a rede seja robusta às transformações. O segundo termo corresponde à redução da redundância certificando-se que as representações de pares distintos são independentes. Desta forma, o objetivo é que a matriz de correlação cruzada seja idêntica à matriz identidade.

2.2.2 Trabalhos desenvolvidos para deteção de fogos

Nos últimos anos, tem-se vindo a procurar soluções para identificar incêndios precocemente recorrendo a processamento de imagem, visão computacional e aprendizagem profunda e evitando, assim, que a monitorização das florestas seja feita exclusivamente por humanos.

Métodos tradicionais

Os primeiros métodos a aparecer e que utilizavam reconhecimento de imagem para a deteção de

incêndio baseavam-se na cor, no movimento e na posição espacial e temporal, isto porque, estes atributos são os mais relevantes para detetar o fogo. Existem inúmeros métodos que tiram partido do histograma de cor para identificar se os *pixels* são de fogo ou não, porém o espaço de cor utilizado varia [19–21]. Definindo os limiares apropriados é possível identificar se um determinado pixel é ou não fogo.

A maioria dos métodos tradicionais utiliza um sistema de vigilância para recolher as imagens. A principal vantagem de utilizar as câmaras fixas é a extração do fundo, o que facilita a deteção das regiões candidatas a fogo. O sistema CICLOPE [2] aplica este tipo de metodologia para monitorizar cerca de 1.300.000 hectares de floresta em Portugal, detetando o fumo através de três algoritmos que se baseiam em limiares e na comparação entre imagens simultâneas. Contudo, estes sistemas têm a desvantagem de não conseguir monitorizar toda a floresta uma vez que as câmaras estão fixas.

Por outro lado, a utilização de satélites que permitiria observar toda a floresta, revela algumas limitações tais como custos elevados, dependência nas condições climáticas (presença de nuvens e outros fatores atmosféricos) e uma baixa resolução, não permitindo a captura de imagens detalhadas [22].

Por último, tem-se vindo a desenvolver metodologias que permitam a utilização de veículos aéreos não tripulados, do inglês *Unmanned Aerial Vehicles* (UAVs), devido ao fácil manobramento, ao vasto alcance e por proporcionarem maior segurança na captação de imagens [22]. O método proposto em [23] recorre a UAVs para adquirir imagens e vídeo e, através de métodos de baseados na cor, classificar cada pixel como sendo ou não fogo. Porém, nas regiões em redor do fogo, onde a tonalidade é semelhante às cores deste, o número de falsos positivos é elevado.

Apesar de uma maneira geral os métodos tradicionais apresentarem uma eficácia elevada, este tipo de abordagem depende fortemente de informações fornecidas pelos autores, tais como definição dos limiares e caracterização dos parâmetros da câmara. Ademais, por norma têm uma taxa de falsos positivos elevada, possivelmente fruto de variações dos parâmetros da câmara e do meio ambiente.

Métodos baseados em aprendizagem profunda

Os métodos que recorrem à aprendizagem profunda extraem, automaticamente, as características do objeto que são mais relevantes para a identificação do mesmo, o que resulta numa melhor prestação face ao tipo de métodos referidos anteriormente. A grande maioria dos métodos que recorrem à aprendizagem profunda para detetar fogo são supervisionados, na medida em que necessitam de anotações feitas por humanos para treinar os modelos.

Em [24], são comparadas diferentes arquiteturas CNN, tais como AlexNet [25], VGG16 [26] e SqueezeNet [27] mostrando que todas conseguem detetar as chamas corretamente. Os autores sugerem ainda a utilização da informação temporal para extrair mais características do fogo, porém as melhorias apresentadas não são significativas. Para o treino do classificador foi utilizado um conjunto

de dados todo ele legendado e com mais de 160 mil imagens, tendo atingido uma eficácia de 96,78% para a rede AlexNet.

Os autores de [28] apresentam uma abordagem que permite detetar e localizar o fogo numa imagem e que pode ser implementado em UAVs. A sua estratégia consiste num classificador a nível global da imagem e um classificador a nível do *patch*, sendo que ambos os classificadores partilham a mesma rede neuronal profunda. A rede utilizada é a AlexNet pré-treinada utilizando o conjunto de imagens ImageNet [29], uma vez que o conjunto de imagens que os autores dispõem para treinar a rede de raiz é reduzido contendo 178 imagens dividindo-se em 12460 *patches*. O método proposto atinge uma eficácia de 97,3% e uma taxa de falsos positivos de 1,2%. No entanto, estes resultados são tendenciosos, uma vez que o conjunto de dados é reduzido e, desta forma, os *patches* usados para testes são semelhantes aos de treino já que são provenientes das mesmas imagens.

Os autores de [30] sugerem fazer um pré-processamento para ajudar a eliminar aspetos irrelevantes da imagem, aperfeiçoando a deteção de informação relevante do fumo e fogo. As operações utilizadas para pré-processamento consistem na comparação de histogramas e a filtragem de uma imagem através do método média da vizinhança, sendo que ambas remetem para uma manipulação da cor da imagem.

A desvantagem dos métodos supervisionados é a necessidade de um extenso conjunto de dados legendados, uma vez que apesar de existirem milhares de imagens disponíveis *online*, a porção de imagens legendadas é reduzida. Essas legendas podem ser ao nível global da imagem para a tarefa de classificação, ou ao nível do *pixel* por exemplo para tarefas de segmentação, sendo que a obtenção destas anotações requer mais tempo e detalhe comparativamente com as primeiras.

Atualmente existem trabalhos que recorrendo a imagens legendadas ao nível da imagem, conseguem resolver a tarefa de segmentação. Em [3] foi desenvolvida uma metodologia que não necessita de imagens legendas ao nível do *pixel* para treinar a rede de segmentação. Este método está dividido em duas componentes, a primeira consiste no treino de uma rede para classificar se existe fogo numa imagem utilizando imagens com anotações ao nível da imagem. Caso a imagem contenha, efetivamente, fogo esta é submetida à rede de segmentação. Contudo, continua a ser necessário uma grande quantidade de imagens legendadas para treinar a rede de classificação.

Desta forma, esta tese tem como objetivo desenvolver uma metodologia que reduza a quantidade de imagens legendadas necessária para o treino da rede de classificação. Assim, propõe-se o desenvolvimento de um sistema que consiga aproveitar os milhares de imagens disponíveis *online* sem que seja necessário fazer anotações à grande maioria das imagens. Utilizando uma metodologia que se baseia na aprendizagem auto-supervisionada para pré-treinar a rede, consegue-se superar a performance de modelos treinados exclusivamente com imagens legendadas. Por outro lado, uma vez que a técnica utilizada não recorre a metodologias como extração do fundo para detetar fogo, o algoritmo

pode ser implementado em sistemas fixos, como torres de vigilância, ou em veículos aéreos permitindo uma melhor cobertura das florestas.

3

Metodologia

Conteúdo

Neste capítulo, são apresentados os métodos desenvolvidos para obter o sistema final que fará a classificação das imagens face à presença ou não de fogo. Em primeiro lugar, é feito um breve resumo das várias componentes deste sistema e de seguida prossegue-se com a explicação mais detalhada de cada uma.

3.1 Visão geral

No âmbito desta tese, pretende-se desenvolver um sistema capaz de identificar fogo em imagens captadas numa perspetiva aérea e através de uma câmara *Red Green Blue* (RGB). A conceção deste sistema está dividido em duas fases, tal como pode ser verificado na Figura 3.1.

Primeiramente, a fase auto-supervisionada tem como objetivo fazer um pré-treino do modelo através da resolução de tarefas de pretexto úteis para aprendizagem das características do fogo e do meio que o rodeia. Para tal são utilizadas imagens não legendadas relacionadas com incêndios florestais.

Para uma melhor aprendizagem das representações são aplicadas às imagens do conjunto de treino transformações de cor e geométricas tais como distorção da cor ou pequenas rotações antes de serem usadas como entrada da rede. Estas transformações favorecem a aquisição de um conjunto de treino mais extenso e diversificado, permitindo que a rede seja invariante a ligeiras alterações da cor, da orientação e do posicionamento dos objetos.

A segunda fase corresponde à tarefa de classificação que, utilizando parte do modelo pré-treinado, tem como finalidade identificar se existe fogo numa determinada imagem. Para esta etapa é utilizado um outro conjunto de imagens mais reduzido, neste caso legendadas.

Figura 3.1: Visão geral do desenvolvimento do sistema.

3.2 Fase auto-supervisionada

Para esta fase são desenvolvidas três tarefas de pretexto para pré-treinar o modelo a extrair as representações do fogo e outros objetos relacionados com incêndios florestais.

Inicialmente, é utilizada uma tarefa de pretexto pertencente ao conjunto de métodos de geração. Esta tarefa consiste na compressão de uma imagem e de seguida reconstruí-la através da utilização de um *autoencoder*.

De seguida, implementam-se dois métodos contrastivos, primeiro a metodologia SimCLR proposta em [16] e, posteriormente, o processo Barlow Twins apresentado em [18], ambos expostos no Capítulo 2.

3.2.1 Método de geração

Entre as tarefas de geração selecionou-se a reconstrução de imagem como tarefa de pretexto. Esta tarefa envolve geração de imagens, utilizando uma arquitetura *autoencoder* [13]. Os *autoencoders* têm uma arquitetura que permite comprimir a imagem de entrada de forma a obter a sua representação no espaço latente e de seguida reconstruir a mesma imagem de entrada através da representação de baixa dimensão. Deste modo, o objetivo da rede é tornar a saída idêntica à entrada. A Figura 3.2 apresenta as várias componentes do *autoencoder*. A componente *encoder* é responsável pela compressão da imagem de forma a obter a representação no espaço latente, representação essa que servirá mais tarde para resolver a tarefa supervisionada. Por sua vez, o *decoder* tem como função reconstruir a imagem partindo dessa representação.

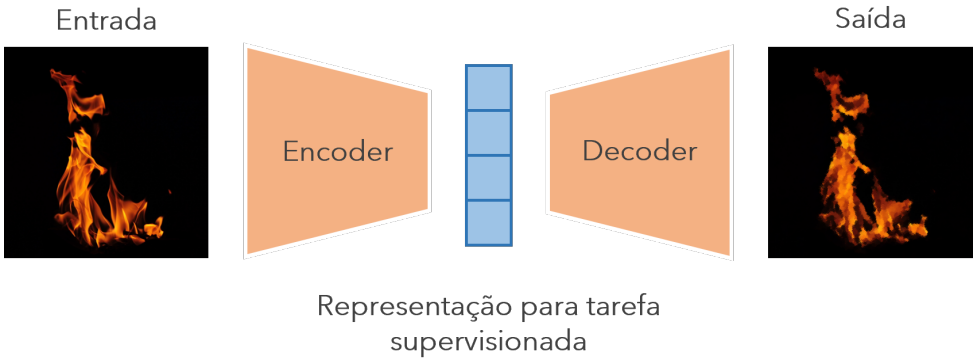


Figura 3.2: Componentes da arquitetura *autoencoder* para resolver a tarefa de pretexto reconstrução de imagem.

Para a aprendizagem deste tipo de redes continua a ser necessário algum tipo de legendas, porém estas não requerem anotações por parte do humano. Na prática, a legenda associada a cada uma das imagens é a própria imagem, pelo que a rede tem de comparar a imagem de saída do *decoder* com a imagem de entrada do *encoder*. Esta comparação é executada com recurso à função de custo

Erro Quadrático Médio exibida na Tabela 2.2 e para que a rede aperfeiçoe o seu desempenho tem que minimizar este erro.

De forma a que a rede seja a exposta a mais imagens e mais diversificadas, é feito um aumento do número de imagens recorrendo à aplicação de transformações, antes do treino da rede. Assim, para cada imagem é possível ter mais do que uma versão no conjunto final de treino. Esse conjunto final é constituído obrigatoriamente pelas imagens originais e pode conter uma ou mais versões transformadas das várias imagens originais.

A arquitetura do *autoencoder* utilizada para esta tarefa de pretexto encontra-se na Tabela 3.1. A componente *encoder* tem no total 10 camadas, 5 convolucionais e as restantes 5 são de *pooling*. No caso do *decoder*, este é composto por 6 camadas no total, das quais 5 são camadas convolucionais transpostas e uma convolucional.

Tabela 3.1: Arquitetura *autoencoder*.

Componente	**Camada**	**Entrada**	***Kernel***	**Função de ativação**	**Saída**
Encoder	Convolução	224x224x3	3x3	ReLU	224x224x32
	Average pooling	224x224x32	2x2	-	112x112x32
	Convolução	112x112x32	3x3	ReLU	112x112x32
	Average pooling	112x112x32	2x2	-	56x56x32
	Convolução	56x56x32	3x3	ReLU	56x56x64
	Average pooling	56x56x64	2x2	-	28x28x64
	Convolução	28x28x64	3x3	ReLU	28x28x64
	Average pooling	28x28x64	2x2	-	14x14x64
	Convolução	14x14x64	3x3	Tanh	14x14x64
	Average pooling	14x14x64	2x2	-	7x7x64
Decoder	Convolução transposta	7x7x64	3x3	ReLU	14x14x64
	Convolução transposta	14x14x64	3x3	ReLU	28x28x64
	Convolução transposta	28x28x64	3x3	ReLU	56x56x64
	Convolução transposta	56x56x64	3x3	ReLU	112x112x32
	Convolução transposta	112x112x32	3x3	ReLU	224x224x32
	Convolução	224x224x32	3x3	ReLU	224x224x3

3.2.2 Métodos contrastivos

Os métodos contrastivos, tal como o nome indica, envolvem a aprendizagem das representações através da comparação entre duas imagens. Tanto o método SimCLR como Barlow Twins funcionam à base de aproximar as representações de pares positivos e afastar os pares negativos. Para se obter pares positivos e negativos, são aplicadas a cada imagem do *batch* duas distorções diferentes, aumentando o número de imagens totais para o dobro. Assim, os pares positivos são duas réplicas obtidas a partir da mesma imagem original, sendo o resto considerado pares negativos. É de notar que estas transformações são aplicadas por época, pelo que durante o treino as transformações aplicadas às imagens não são constantes resultando em diferentes pares ao longo das várias épocas.

Implementação de SimCLR

A metodologia SimCLR tem como base a comparação entre duas imagens e determinar se estas são semelhantes ou distintas. Em termos de aprendizagem de máquina são necessários três aspetos:

- Exemplos de imagens semelhantes e distintas;
- Capacidade transformar uma imagem em representações;
- Capacidade para quantificar se duas imagens são semelhantes.

O método proposto em [16] põe em prática estes três tópicos sem que seja necessário qualquer tipo de supervisão humana. A Figura 3.3 apresenta o processo proposto pelos autores de SimCLR e está dividido em quatro etapas.

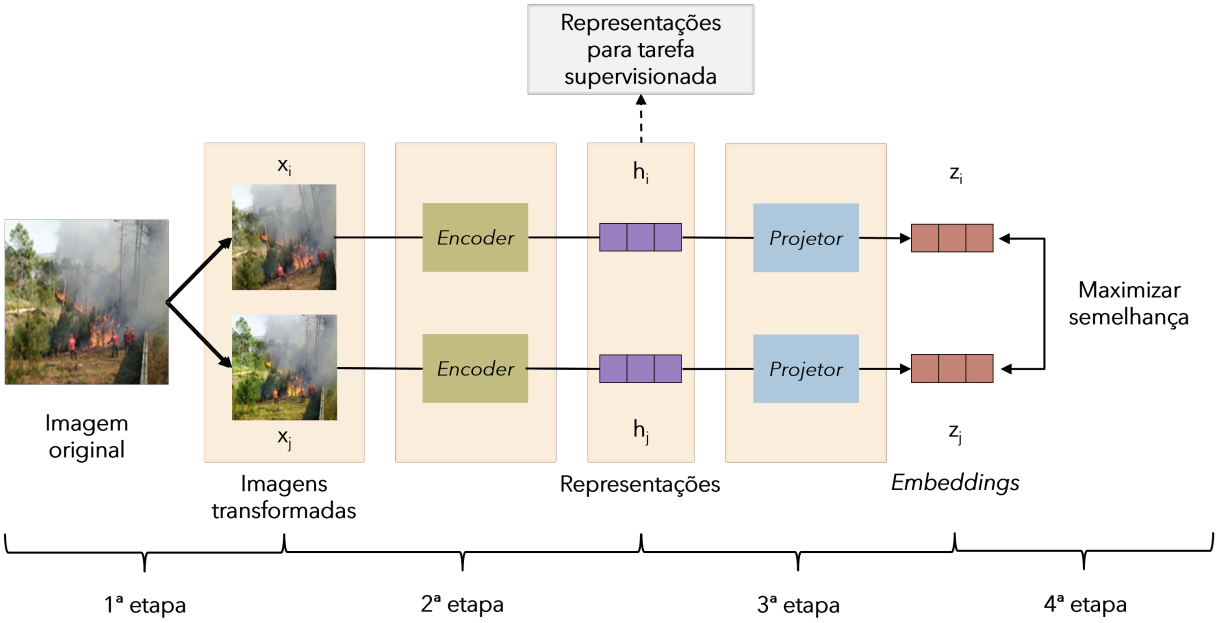


Figura 3.3: Etapas da metodologia SimCLR [16]. A transformação aplicada a x_i foi rotação e a x_j aplicou-se manipulação da cor. (Imagem adaptada [31])

A primeira etapa corresponde à aplicação de transformações às imagens de entrada. Para cada imagem do *batch* são aplicados dois conjuntos de transformações, de forma a gerar duas réplicas, x_i e x_j, de cada imagem original. Por outras palavras, se o *batch* tiver N imagens então obter-se-á um total de $2N$ imagens no *batch* final. As réplicas obtidas através da mesma imagem original são consideradas um par positivo e, pelo contrário, duas réplicas geradas a partir de imagens originais distintas correspondem a um par negativo. Desta forma, cada imagem do *batch* final tem 1 par positivo e $2(N-1)$ pares negativos, pelo que no total tem-se $2N(2N-1)$ pares, uma vez que o par x_i e x_j é considerado diferente do par x_j e x_i.

Os autores sugerem a utilização de diversas transformações, tais como corte e redimensionamento, converter a imagem em tons da escala de cinza, distorção da cor, oclusão parcial da imagem, entre outras. Contudo, as transformações aplicadas nesta tese devem estar enquadradas com o objetivo final. Isto porque, por exemplo, no caso das transformações de cor, esta pode eliminar a influência que a cor pode ter, permitindo que o aspeto espacial tenha uma maior importância [32]. Porém, a cor é uma característica distintiva muito importante do fogo e a manipulação excessiva dos canais de cor de uma imagem pode levar a exemplos irrealistas. Por exemplo, no caso do fogo e do fumo, estes apresentam um aspeto espacial semelhante e a principal característica que os distingue é a cor, pelo que a conversão da imagem para a escala de cinzas pode levar a uma aprendizagem errada destes dois objetos. No entanto, existem objetos que, devido à cor que apresentam podem ser confundidos com o fogo como é o caso de telhados, carros de bombeiros e por do sol.

Assim, a nível da cor foram efetuadas ligeiras manipulações dos canais de tonalidade, H, e saturação, S. Ademais, recorreu-se ainda ao corte e redimensionamento, pequenas rotações e espelhamento horizontal. A utilização dos vários tipos de transformações é um dos parâmetros estudados, pelo que não é constante ao longo das várias experiências.

A etapa seguinte diz respeito à conversão de x_i e x_j nas respetivas representações, h_i e h_j, através de um *encoder*. Posteriormente, estas representações irão ser utilizadas durante a tarefa supervisionada selecionada, ou seja, classificação.

A terceira etapa permite obter os *embeddings*, z_i e z_j, recorrendo a uma camada de projeção. Segundo os autores de SimCLR, a introdução de um projetor não linear é uma mais valia, aumentando a eficácia em 10% em contraste à não utilização de uma camada de projeção, ou seja, utilizando as representações h_i e h_j diretamente.

A quarta e última etapa corresponde ao cálculo da função de custo e está dividida em duas partes. Em primeiro lugar é calculada a semelhança entre dois *embeddings* do *batch* em análise de acordo com

$$\mathrm{sim}(z_i, z_j) = \frac{z_i^T z_j}{\|z_i\| \, \|z_j\|}. \tag{3.1}$$

Idealmente, a semelhança de duas réplicas da mesma imagem será elevada e, em oposição, os

pares negativos terão um valor reduzido. Por fim, é calculado o custo através da função de custo Entropia Cruzada Normalizada (NT-Xent) [16] entre dois pares positivos através de

$$\ell_{i,j} = -\text{sim}(z_i, z_j) + \log\left(\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp\left(\frac{\text{sim}(z_i, z_j)}{\tau}\right)\right) \tag{3.2}$$

onde τ é o parâmetro de temperatura com valor de $0,1$ visto ser o que apresenta melhores resultados segundo os autores e i e j correspondem a um par positivo. Esta função de custo contém duas componentes, a primeira corresponde ao termo de semelhança e a segunda representa o termo contrastivo.

É de realçar que para alcançar um melhor desempenho da rede a executar esta tarefa é conveniente utilizar *batches* de grandes dimensões. De acordo com os autores, os melhores resultados obtidos foram para *batches* com 8192 imagens. Tal deve-se ao facto de quanto maior o *batch*, maior o número de pares negativos, facilitando assim a convergência do algoritmo. Ademais, verificou-se que quanto mais épocas utilizadas para treinar a rede menor a discrepância de resultados entre os modelos treinados com *batches* de menor dimensão. Por último, a função de custo é favorecida quando são utilizados *embeddings* de pequenas dimensões.

Assim, devido ao reduzido número de imagens presente no conjunto de treino, a quantidade de imagens em cada *batch* foi 128. Desta forma, para compensar a dimensão do *batch*, foram utilizadas 200 épocas. A arquitetura utilizada durante a resolução desta tarefa está dividida em duas componentes, um *encoder* e um projetor. De forma a ser possível comparar esta metodologia com as restantes no que diz respeito ao pré-treino da rede para classificação, manteve-se a arquitetura do *encoder* constante para a resolução das várias tarefas de pretexto. Assim o *encoder* é idêntico ao apresentado na Tabela 3.1. Por sua vez, o projetor é composto por uma camada de vetorização, responsável por converter cada uma das representações num vetor e duas camadas totalmente ligadas cada uma com 64 unidades de neurónios separadas por uma camada de ativação ReLU.

Implementação de Barlow Twins

A tarefa Barlow Twins está enquadrada nos métodos contrastivos e foca-se na redução da redundância entre as representações. A Figura 3.4 ilustra as várias etapas desta tarefa. Em primeiro lugar, são obtidas de forma aleatória duas versões distintas, x_i^A e x_i^B, para cada imagem x_i do *batch*, aplicando dois conjuntos de transformações. Estas transformações coincidem com as aplicadas durante a resolução do método SimCLR. Por sua vez, os pares positivos são divididos em dois conjuntos, A e B, de forma a que cada imagem do conjunto A tenha o respetivo par positivo no conjunto B e as restantes imagens do conjunto B são os respetivos pares negativos. De igual modo, cada imagem do conjunto B tem um único par positivo no conjunto A e as restantes imagens do conjunto A são os seus pares negativos. Neste caso, os autores consideram o par x_i^A e x_i^B idêntico ao par x_i^B e x_i^A, pelo que é apenas contabilizado uma vez. Desta forma, para um *batch* com N amostras, cada imagem terá um

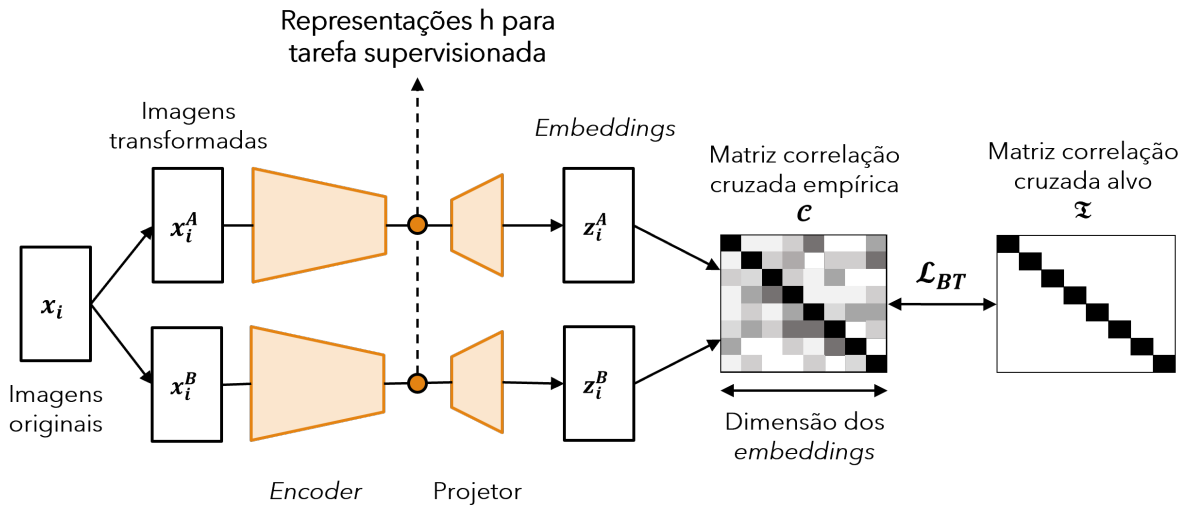


Figura 3.4: Visão geral de Barlow Twins [18]. (Imagem adaptada)

par positivo e $N-1$ pares negativos, sendo que no total tem-se N^2 pares.

De seguida, através de uma rede CNN são determinadas os *embeddings* no espaço latente, z_i^A e z_i^B, de cada uma das imagens distorcidas. Finalmente, é calculada a matriz de correlação cruzada, $\mathcal{C}$, entre duas saídas da rede, z_i^A e z_j^B, através de

$$\mathcal{C}_{ij} \triangleq \frac{\sum_b z_{b,i}^A z_{b,j}^B}{\sqrt{\sum_b (z_{b,i}^A)^2}\sqrt{\sum_b (z_{b,j}^B)^2}}. \tag{3.3}$$

onde b é o indíce do *batch*.

O objetivo desta tarefa é que a matriz $\mathcal{C}$ seja a matriz identidade pelo que as representações das duas imagens de entrada devem ser semelhantes, minimizando a redundância entre componentes desses vetores.

Desta forma, é utilizada a função de custo desenvolvida pelos autores de Barlow Twins e que consta na Equação (3.4). A função é composta por duas parcelas, a primeira corresponde ao termo da invariância que tem como objetivo tornar a rede robusta às transformações e a segunda representa o termo de redução da redundância pelo que procura tornar as representações independentes.

Assim, a função de custo é dada por:

$$\mathcal{L}_{BT} \triangleq \sum_i (1-\mathcal{C}_{ii})^2 + \lambda \sum_i \sum_{j \neq i} \mathcal{C}_{ij}^2 \tag{3.4}$$

onde λ é uma constante positiva e corresponde ao compromisso entre os dois termos da função de custo. De acordo com os autores de [18], o melhor valor para este parâmetro é 0,005, tendo sido este o valor utilizado nesta tese.

A principal vantagem desta metodologia face a SimCLR traduz-se no facto de não ser necessário utilizar *batches* de tamanhos elevados [18]. Esta vantagem é especialmente relevante na medida em que o conjunto de dados disponível para treinar o modelo é reduzido pelo que não é possível ter *batches* de grandes dimensões. Para esta tarefa foram utilizadas 128 imagens em cada *batch* e treinou-se a rede durante 75 épocas.

A arquitetura utilizada para esta tarefa está dividida em duas partes. A primeira componente correspondente ao *encoder* é, mais uma vez, idêntica à apresentada na Tabela 3.1. A camada de projeção por sua vez é composta por uma camada de vetorização responsável por converter as representações em vetores, três camadas totalmente ligadas, sendo que a seguir a cada uma das duas primeiras camadas vem uma camada de normalização do *batch* e uma camada de ativação ReLU. Para as camadas totalmente ligadas, os autores sugerem a utilização de camadas com elevado número de neurónios, porém o conjunto de treino não é extenso podendo levar a sobre ajuste (overfitting). Deste modo, a dimensão selecionada para as camadas totalmente ligadas foi 1024 unidades de neurónios.

3.3 Fase supervisionada

A tarefa supervisionada corresponde à classificação de uma imagem como contendo ou não fogo. Neste caso, são utilizadas imagens com as respetivas legendas ao nível da imagem sobre a existência de fogo. Ao contrário das tarefas de pretexto, no caso da classificação não foi aplicado nenhum tipo de transformações às imagens, tendo sido utilizadas as originais.

Para a resolução desta tarefa é utilizada a componente *encoder* da rede pré-treinada por uma das tarefas da aprendizagem auto-supervisionadas mencionadas, anteriormente. Apesar da arquitetura ser idêntica nas várias tarefas, o valor dos pesos difere consoante o tipo de tarefa de pretexto. Ao *encoder* é adicionada a componente de projeção que contem no total duas camadas e cujos pesos são inicializados de forma aleatória. A primeira camada é uma *global average pooling* responsável por calcular a média espacial dos mapas de características e a segunda corresponde a uma camada totalmente ligada com um neurónio. A Tabela 3.2 apresenta a arquitetura do classificador utilizado.

Tabela 3.2: Arquitetura para classificador de fogo.

Componente	**Camada**	**Entrada**	***Kernel***	**Função de ativação**	**Saída**
Encoder	Convolução	224x224x3	3x3	ReLU	224x224x32
	Average pooling	224x224x32	2x2	-	112x112x32
	Convolução	112x112x32	3x3	ReLU	112x112x32
	Average pooling	112x112x32	2x2	-	56x56x32
	Convolução	56x56x32	3x3	ReLU	56x56x32
	Average pooling	56x56x64	2x2	-	28x28x64
	Convolução	28x28x64	3x3	ReLU	28x28x64
	Average pooling	28x28x64	2x2	-	14x14x64
	Convolução	14x14x64	3x3	Tanh	14x14x64
	Average pooling	14x14x64	2x2	-	7x7x64
Projeção	*Global Average Pooling*	7x7x64	-	-	64
	Totalmente ligada	64	-	Sigmoid	1

4

Experiências

Conteúdo

Este capítulo começa por apresentar os conjuntos de imagens utilizadas em cada uma das fases, as métricas aplicadas para avaliar e comparar os classificadores obtidos nos vários testes, as configurações do ambiente experimental onde foi desenvolvido o sistema e ainda os parâmetros de treino de cada uma das redes utilizadas. No final, são expostas as várias experiências executadas para comparar e avaliar a performance dos classificadores.

4.1 Conjunto de Dados

De forma a desenvolver o modelo pretendido, é necessário criar um conjunto de imagens extenso e com diversidade para cada uma das fases. As imagens utilizados durante o treino e avaliação dos modelos treinados de forma supervisionada e auto-supervisionada foram recolhidas de várias fontes. No total adquiriram-se 2500 imagens, das quais 692 através do portal dos Bombeiros de Portugal [33] e 1808 do conjunto de imagens criado por um dos trabalhos do projeto Firefront [3].

A Figura 4.1 ilustra alguns exemplos de imagens pertencentes a ambas as fontes e com a respetiva legenda. Uma vez que as imagens tinham dimensões distintas foi necessário fazer um redimensionamento para que todas tivessem a mesma dimensão final. Para as imagens retiradas do portal Bombeiros de Portugal, foi necessário efetuar um recorte central de forma remover o logótipo e moldura antes de se realizar o redimensionamento.

As imagens foram divididas em dois conjuntos distintos, um para cada uma das fases, estando a distribuição das mesmas apresentadas na Tabela 4.1.

Tabela 4.1: Distribuição das imagens para cada uma das fases.

Nome	**Fase**	**# Imagens**	**Fogo [%]**	
A	Auto-supervisionada	2000	-	
B	Supervisionada	500	Positivo	70
			Negativo	30

A discrepância de imagens adquiridas para cada uma das fases está relacionada com o tipo de legendas necessárias para treinar e avaliar os modelos. Durante a fase auto-supervisionada, as imagens não requerem qualquer tipo de anotação manual para treinar os modelos. No entanto, o mesmo não acontece para o conjunto destinado a treinar e avaliar o classificador durante a fase supervisionada, já que neste caso, são necessárias legendas, ao nível da imagem.

Para o treino das redes ao longo da fase auto-supervisionada, divide-se o conjunto em duas partes, a primeira para treino com 90% das imagens e a segunda para validação com as restantes 10%. Esta

(a) Fogo: Positivo [33]

(b) Fogo: Positivo [3]

(c) Fogo: Negativo [33]

(d) Fogo: Negativo [3]

Figura 4.1: Exemplos de imagens do conjunto de dados.

divisão permite monitorizar a evolução do custo e garantir que a rede não está a sobre ajustar. Uma vez que o objetivo desta fase é apenas inicializar os pesos da componente *encoder* do classificador, não é necessário avaliar a performance da rede durante esta fase através de uma terceira partição do conjunto.

Para a segunda fase, isto é, treino e validação do classificador, recorreu-se à técnica validação cruzada (*cross validation*), visto que a quantidade de imagens disponíveis era reduzida. Este procedimento permitiu, por um lado, averiguar se o classificador estava ou não a sobre ajustar e por outro comparar a performance dos vários classificadores treinados. Esta técnica requer a divisão das imagens em várias pastas e, em cada iteração, uma das pastas serve para avaliar a performance e as restantes para treinar o modelo. O número de iterações é dado pelo número de pastas totais. Definiu-se que seriam 10 pastas e assim em cada iteração dispõe-se de 450 imagens para treino (90%) e de 50 para testes (10%). Por outro lado, a distribuição das amostras por cada pasta foi feita aleatoriamente. Dado que o conjunto de dados não é balanceado, foi determinado o número de amostras de cada classe e fornecida à rede a informação sobre os pesos de cada classe para cada iteração.

4.1.1 Aumento de dados

Durante a fase auto-supervisionada estudou-se a influência de aplicar transformações às imagens de forma a aumentar a diversidade do conjunto de treino. Estas transformações variaram entre transformações de cor e geométricas.

- **Transformações de cor**

  Procurou-se variar os canais de tonalidade (H) e de saturação (S), convertendo as imagens para o espaço *Hue Saturation Value* (HSV) e variando cada um dos canais aleatoriamente. A variação do canal de tonalidade permite ao modelo aprender as formas e arestas dos objetos enquanto a alteração no canal de saturação permite ter mais diversidade no que diz respeito à luminosidade com que as imagens são captadas. É de realçar que apesar da rede não se poder basear exclusivamente na cor, esta não deixa de ser uma das principais características do fogo, pelo que as perturbações aplicadas às imagens devem ser ligeiras, especialmente no canal da tonalidade. O principal objetivo da introdução desta transformação é simular diferentes exemplos de tonalidades que os objetos, e em particular o fogo, podem ter. Isto porque, dependendo da altura do dia e das condições meteorológicas em que as imagens são captadas, as cores que os objetos apresentam podem variar.

- **Transformações geométricas**

  Este tipo de transformações é especialmente útil quando se quer que a rede aprenda sem se basear na posição do objeto na imagem. No caso da captura de imagens aéreas das florestas, o fogo, assim como outros objetos, não ocupam sempre a mesma localização e orientação. Desta forma, a rede deve ser exposta a diferentes perspetivas de modo que não se baseie unicamente nestes dois aspetos. Começou-se por averiguar o impacto de aplicar rotações à imagem, contudo estas devem ser pequenas para não fornecer à rede exemplos irrealistas. Explorou-se ainda outras transformações, como o corte e redimensionamento e ainda espelhamento horizontal.

A Tabela 4.2 apresenta as propriedades das transformações aplicadas e a Figura 4.2 contém um exemplo de cada transformação utilizada.

Tabela 4.2: Lista de possíveis transformações aplicadas às imagens durante a fase auto- supervisionada.

Transformações	Descrição
Cor	Aplica-se distorção no canal H com uma variação aleatória até 20%. Aplica-se distorção no canal S com uma variação aleatória até 30%.
Rotação	Aplica-se rotação entre 1 a 20º de forma aleatória.
Espelhamento Horizontal	Aplica-se espelhamento horizontal.
Corte e Redimensionamento	Corte de um quadrado da imagem com dimensão aleatória entre 100 a 200 *pixels* e redimensionamento para dimensão inicial da imagem.

(a) Original

(b) Cor

(c) Rotação

(d) Espelhamento Horizontal

(e) Corte e Redimensionamento

Figura 4.2: Exemplos de transformações aplicadas às imagens.

4.2 Métricas

Para uma avaliação correta dos modelos de classificação, é necessário definir as métricas mais apropriadas.

4.2.1 Matriz de Confusão

A matriz de confusão permite visualizar o resumo da performance do modelo para as várias classes. A Tabela 4.3 apresenta o formato da matriz de confusão para uma classificação binária.

Um Verdadeiro Positivo (VP) e um Falso Positivo (FP) ocorrem quando o algoritmo deteta fogo numa imagem (evento positivo) de forma correta (a imagem tem de facto fogo) ou incorreta (a imagem não tem

Tabela 4.3: Exemplo de uma matriz de confusão para classificação binária.

		Classe prevista	
		Sem Fogo	Com Fogo
Classe real	Sem Fogo	VN	FP
	Com Fogo	FN	VP

fogo), respetivamente. Em oposição, um Verdadeiro Negativo (VN) e um Falso Negativo (FN) sucedem-se quando o inverso se verifica, isto é, o algoritmo classifica uma imagem como não contendo fogo, de forma correta (a imagem de facto não tem fogo) ou incorreta (a imagem tem fogo), respectivamente. Através desta matriz é possível calcular as métricas eficácia, precisão, sensibilidade e *F1 Score*.

4.2.2 Eficácia

A eficácia indica a performance geral do modelo, apresentando a percentagem de amostras que classificou corretamente e é calculada de acordo com

$$Eficácia = \frac{VP + VN}{VP + VN + FP + FN}. \tag{4.1}$$

4.2.3 Precisão

A precisão indica quão precisas são as verdadeiras previsões positivas (VP) em relação a todas as previsões classificadas como positivas $(VP+FP)$. Assim, quanto maior for o valor da precisão menor a quantidade de falsos positivos classificados pelo modelo. A precisão é determinada segundo a equação

$$Precisão = \frac{VP}{VP + FP}. \tag{4.2}$$

4.2.4 Sensibilidade

A sensibilidade coincide com a percentagem de verdadeiros positivos (VP) entre todas as amostras que são positivas $(VP + FN)$ e é dada por

$$Sensibilidade = \frac{VP}{VP + FN}. \tag{4.3}$$

4.2.5 *F1 Score*

A métrica *F1 Score* corresponde à média harmónica entre a precisão e a sensibilidade. Esta métrica é especialmente útil dado que o conjunto de dados não está balanceado, havendo muito mais casos positivos do que negativos, como se pode averiguar na Tabela 4.1. Esta métrica é calculada através de

$$F1\ Score = \frac{VP}{VP + \frac{1}{2} \cdot (FP + FN)}. \tag{4.4}$$

4.2.6 Teste estatístico: Teste-t

As métricas acima descritas permitem avaliar a performance dos vários modelos desenvolvidos. Porém, para ser possível tirar conclusões com confiança sobre os desempenhos dos classificadores deve-se recorrer a um teste estatístico. Os testes estatísticos, também conhecidos por testes de hipóteses, correspondem a uma ferramenta de estatística que permite interpretar os resultados observados e determinar algum dos classificadores tem melhor performance do que o outro. Estes testes possibilitam confirmar ou rejeitar hipóteses/ previsões que se faz sobre os resultados, calculando a probabilidade dos mesmos serem coincidência. Por outras palavras, estes testes permitem confirmar ou rejeitar uma teoria sobre os resultados obtidos nas experiências.

Para uma melhor compreensão destes testes estatísticos é necessário compreender os conceitos Hipótese nula, H_0, Hipótese alternativa, H_1, valor-p e nível de significância, α. A Hipótese nula deve ser assumida como verdadeira no início do teste e indica que não existe nenhuma diferença entre os resultados. Em oposição, a Hipótese alternativa corresponde à alternativa de H_0 e que se pretende provar como verdadeiro. O valor-p é a probabilidade de os resultados não rejeitarem H_0, sendo que quanto mais perto de zero maior a probabilidade de se rejeitar H_0. Para se saber quando se deve rejeitar a Hipótese nula é necessário definir o valor do nível de significância antes do cálculo do valor-p que por norma toma os valores 1%, 5% ou 10%. Nesta tese, adotou-se 5% como nível de significância.

Existem múltiplos testes estatísticos dependendo do tipo de dados em análise [34]. Por outro lado, a seleção do teste estatístico deve ser efetuada antes de começar a análise para garantir que os resultados não são influenciados. Neste caso, os dados a serem observados correspondem aos resultados *F1 Score* de cada uma das pastas durante a validação cruzada e quer-se averiguar se a diferença entre as performances médias dos dois classificadores distintos é zero. Estes dados são contínuos e independentes pelo que se selecionou o teste estatístico Teste-t. Porém, para que os resultados do teste possam ser considerados é necessário garantir que existe homogeneidade entre os dois conjuntos de dados, ou seja, a diferença entre as variâncias dos dois conjuntos tem de ser próxima de zero [35].

Desta forma, recorre-se a um teste auxiliar, teste Levene, que tem como finalidade averiguar a semelhança entre as variâncias de dois conjuntos. Este teste é semelhante ao Teste-t na medida em que existe uma Hipótese nula e quer-se averiguar se esta é ou não rejeitada. Neste caso, H_0 consiste na igualdade entre variâncias de dois grupos de dados e se o valor-p for inferior a 5 % conclui-se que H_0 deve ser rejeitada, não se podendo aceitar o resultado do Teste-t. Por outro lado, se o valor-p for superior a 5 % está garantida a homogeneidade entre os dois conjuntos de dados pelo que é possível aplicar-se o Teste-t.

Uma vez garantida a homogeneidade dos conjuntos, procede-se com o calculado do Teste-t. Este teste estatístico tem como Hipótese nula que a diferença entre as médias dos dois conjuntos ser nula. Para determinar o resultado deste teste recorre-se à expressão

$$t = \frac{\overline{x}_1 - \overline{x}_2}{\sqrt{s^2(\frac{1}{n_1} + \frac{1}{n_2})}} \tag{4.5}$$

onde $\overline{x}_1$ e $\overline{x}_2$ são as médias dos dois conjuntos de dados e n_1 e n_2 correspondem ao total de amostras de cada conjunto. A variável s corresponde ao erro padrão conjunto e é dada por

$$s = \sqrt{\frac{(n_1 - 1)(\frac{\sigma_1}{\sqrt{n_1}})^2 + (n_2 - 1)(\frac{\sigma_2}{\sqrt{n_2}})^2}{n_1 + n_2 - 1}} \tag{4.6}$$

onde σ_1 e σ_2 correspondem às variâncias dos respetivos conjuntos.

Através do teste estatístico determina-se o valor-p e caso este seja inferior a 5% assume-se que a média dos dois classificadores é de facto diferente e não fruto de uma coincidência.

4.3 Configuração do Ambiente Experimental

Para o desenvolvimento do sistema proposto, compilou-se o código através das componentes do Google Colab [36]. O processador utilizado foi o Intel(R) Xeon(R) CPU @ 2.30GHz, a componente GPU usada foi NVIDIA-SMI 470.74 e a RAM máxima disponível era 12 Gb.

O ambiente de desenvolvimento foi o Python 3.7.12 e foram ainda utilizadas as bibliotecas Tensorflow [37] e Keras [38].

A utilização deste ambiente de trabalho apresentou algumas limitações no desenvolvimento do sistema tais como impossibilidade de utilizar mais do que 5000 imagens no total para treinar os modelos e incrementar a dimensão do *batch*. Ao incrementar-se algum destes parâmetros, verificava-se a terminação da sessão, perdendo-se os progressos alcançados.

4.4 Treino das redes

O objetivo final é desenvolver um classificador que identifique se uma imagem contém fogo ou não. Dependendo da tarefa e da fase em que se encontra, existem diferentes propriedades do treino das redes que têm de ser ajustadas. A arquitetura de cada uma das redes utilizadas foi apresentada no Capítulo 3. Nesta secção são apresentados os restantes parâmetros para o treino da rede, tais como dimensão do *batch*, número de épocas e otimizadores.

Para a primeira tarefa de pretexto selecionada, isto é, reconstrução de imagem através de um *autoencoder* utilizaram-se os parâmetros apresentados na Tabela 4.4.

Tabela 4.4: Parâmetros de treino do *autoencoder* durante a resolução da tarefa de reconstrução de imagem.

Otimizador	**Função de custo**	***Batch***	**Épocas**
Adam()	Erro Quadrático Médio	75	60

No caso da resolução da tarefa proposta em SimCLR, utilizaram-se os parâmetros apresentados na Tabela 4.5. Os autores sugerem a utilização do otimizador *Layer-wise Adaptive Rate Scaling* (LARS). Este otimizador é extremamente útil quando os *batches* apresentam grandes dimensões. No entanto, uma vez que o tamanho do *batch* é reduzido, este otimizador não é benéfico. Assim optou-se pela utilização do otimizador *Adam* com os valores padrão deste. Adicionalmente, utilizou-se para a taxa de aprendizagem um algoritmo de decaimento, conhecido por *cosine decay schedule*, permitindo evitar mínimos locais.

Tabela 4.5: Parâmetros de treino da rede durante a resolução da tarefa SimCLR.

Otimizador	**Função de custo**	***Batch***	**Épocas**
Adam()	Entropia Cruzada Normalizada	128	200

Para a tarefa proposta em [18], os parâmetros utilizados durante o treino da rede encontram-se na Tabela 4.6. Mais uma vez é sugerido a utilização do otimizador LARS, no entanto, pelo mesmo motivo, foi selecionado o otimizador Adam com os valores padrão à exceção da taxa de aprendizagem. Também os autores de Barlow Twins exploraram a utilização do mesmo algoritmo para a taxa de aprendizagem, pelo que foi implementado o *cosine decay schedule* para o treino da rede.

Tabela 4.6: Parâmetros de treino da rede durante a resolução da tarefa Barlow Twins.

Otimizador	**Função de custo**	***Batch***	**Épocas**
Adam()	Barlow Twins Loss	128	70

Finalmente, para a tarefa supervisionada foi também utilizado o otimizador Adam, com os valores padrão das várias variáveis. A Tabela 4.7 permite visualizar os parâmetros de treino do classificador final.

Tabela 4.7: Parâmetros de treino do classificador.

Optimizador	Função de custo	*batch*	Épocas
Adam()	Entropia Cruzada Binária	32	75

4.5 Experiências

Nesta secção são apresentadas as diversas experiências realizadas que servem de comparação e validação das diferentes componentes do sistema proposto.

Para que seja possível averiguar a utilidade de usar aprendizagem auto-supervisionada para o pré-treino de um classificador é imprescindível determinar o classificador referência. Neste caso, essa referência é alcançada quando se utiliza exclusivamente aprendizagem supervisionada para treinar o classificador. Desta forma, recorrendo apenas ao conjunto de imagens B da Tabela 4.1 na sua totalidade e sem aplicar qualquer tipo de transformação, treina-se um classificador inicializado de forma aleatória. A Tabela 4.8 apresenta os resultados médios da eficácia e de *F1 score* das várias iterações e os respetivos desvios padrão.

Tabela 4.8: Performance do classificador de referência.

Eficácia média [%]	F1 Score médio [%]
$82,8\ (\pm 3,9)$	$87,0\ (\pm 3,6)$

A Figura 4.3 mostra alguns exemplos de imagens classificadas pelo classificador de referência. É de realçar que o classificador considera uma imagem como exemplo positivo se a probabilidade for igual ou superior a 0,5 e, em posição, esta é considerada um exemplo negativo se a probabilidade for inferior a 0,5.

Nas imagens da Figura 4.3 são observados cinco VP, para 4.3a, 4.3b, 4.3c, 4.3f e 4.3h, três FN para 4.3d, 4.3e e 4.3g, dois VN para 4.3i e 4.3l e, finalmente, dois FP para 4.3j e 4.3k. Os exemplos considerados como FP revelam que o classificador depende muito da cor para considerar se uma imagem contem ou não fogo. Ambos os exemplos considerados como FP contêm tons semelhantes ao do fogo pelo que foram classificados como positivos. Por outro lado, verifica-se que o classificador tem alguma dificuldade em identificar fogo quando este apresenta uma pequena dimensão na imagem. Dos cinco exemplos em que o fogo tem um tamanho reduzido, ou seja, Figuras 4.3d, 4.3e, 4.3f, 4.3g e 4.3h, o classificador de referência apenas consegue identificar fogo em duas das imagens, isto é, Figuras 4.3f e 4.3h.

(a) Probabilidade = 0,9 **(b)** Probabilidade = 1,0 **(c)** Probabilidade = 1,0 **(d)** Probabilidade = 0,4

(e) Probabilidade = 0,3 **(f)** Probabilidade = 0,6 **(g)** Probabilidade = 0,1 **(h)** Probabilidade = 0,9

(i) Probabilidade = 0,1 **(j)** Probabilidade = 0,9 **(k)** Probabilidade = 0,7 **(l)** Probabilidade = 0,1

Figura 4.3: Exemplos de imagens classificadas pelo classificador de referência.

4.5.1 Experiência A: Comparação entre as diferentes tarefas de pretexto e classificador de referência

Após a implementação das três diferentes tarefas pretexto, isto é, reconstrução de imagem, SimCLR e Barlow Twins, é relevante perceber qual a que traz mais benefícios para a classificação final. Para tal, selecionam-se os casos em que se obteve melhores performances dos classificadores pré-treinados para cada uma das tarefas de pretexto. Para as três situações utilizou-se o conjunto B para treinar os classificadores finais.

4.5.2 Experiência B: Comparação entre classificadores obtidos via *transfer learning* e *fine-tuning*

Esta experiência tem como objetivo averiguar se existe diferença em manter as camadas do *encoder* congeladas, treinando apenas a camada FC (*transfer learning*) ou se é preferível utilizar os pesos do *encoder* previamente aprendidos como inicialização e treinar tanto as camadas do *encoder* como a

camada FC (*fine-tuning*). Para tal, são utilizados os *encoders* pré-treinados que obtiveram melhor performance em cada uma das tarefas de pretexto e compara-se as situações em que se faz *fine-tunning* e *transfer learning*. Para ambas as situações utilizou-se o conjunto B para treinar os classificadores.

4.5.3 Experiência C: Influência do número de imagens originais durante a resolução da tarefa reconstrução de imagem

O principal objetivo desta experiência é perceber se existe vantagem em tirar partido das inúmeras imagens disponíveis *online*. Assim, pretende-se averiguar se existe um melhor desempenho do classificador ao incrementar o número de imagens utilizadas durante a fase auto-supervisionada. Ademais, dado que as tarefa contrastivas necessitam de uma grande quantidade de imagens e sendo 2000, por si só, um conjunto reduzido, não se analisou o impacto de diminuir ainda mais o número de imagens utilizadas inicialmente para este tipo de tarefas. Desta forma, para esta experiência foi apenas utilizada a tarefa de pretexto reconstrução de imagem.

Assim, decidiu-se variar o número de imagens para pré-treino entre 500, 1000 e 2000, todas pertencentes ao conjunto de dados A da Tabela 4.1 e sem aplicar qualquer tipo de transformações às imagens.

Para esta experiência fez-se *fine-tuning* dos modelos pré-treinados para os diferentes conjuntos de imagens. Para treinar e avaliar os classificadores utilizou-se mais uma vez o conjunto B da Tabela 4.1.

4.5.4 Experiência D: Influência da utilização de transformações para aumentar o conjunto de treino durante a tarefa reconstrução de imagem

Nesta experiência, pretende-se analisar o efeito de aumentar o número de imagens de treino recorrendo à utilização de transformações. Neste caso, cada imagem do conjunto inicial pode ter mais do que uma réplica, isto é, a original (obrigatoriamente) e eventuais transformadas, no conjunto final de treino. Para tal, parte-se de 1000 imagens iniciais e aumenta-se este número em $\{1,5;\ 2;\ 4;\ 5\}$ vezes. Mais uma vez recorre-se à tarefa de geração proposta na metodologia para compreender se o aumento do conjunto de treino através de transformações influencia a performance do classificador final.

O estudo sobre a influência das transformações a aplicar incide na utilização de um tipo de transformação exclusivamente e ainda na junção de várias transformações em simultâneo no mesmo conjunto de treino. As transformações base encontram-se descritas na Tabela 4.2, tendo-se definido os diferentes agrupamentos de transformações da seguinte forma:

(i) **Cor**

Neste caso apenas é aplicada a transformação Cor às imagens selecionadas para tal.

(ii) **Rotação**

Às imagens selecionadas é aplicada exclusivamente a transformação Rotação.

(iii) **Cor ou Rotação**

Perante as imagens selecionadas, aplica-se a metade a transformação Cor e às restantes a transformação Rotação. Desta forma, o conjunto de treino tem imagens transformadas de duas maneiras diferentes.

(iv) **Cor + Rotação**

Nesta situação, aplica-se às imagens selecionadas as transformações Cor e Rotação simultaneamente.

(v) **Espelhamento, Corte e Redimensionamento**

Para as imagens selecionadas são aplicadas, simultaneamente, as transformações Espelhamento Horizontal, Corte e Redimensionamento.

(vi) **Cor, Rotação e originais**

Neste caso, cada imagem do conjunto inicial tem 4 réplicas, a original, transformada de Cor, transformada de Rotação e ainda transformada de Cor e Rotação simultaneamente.

(vii) **Tudo**

Este agrupamento de transformações é semelhante ao de (vi) porém tem 5 réplicas de cada imagem em vez de 4. Para além das versões apresentadas, adiciona-se uma outra réplica com transformações simultâneas de Corte e Redimensionamento e Espelhamento Horizontal.

Em todas as situações acima referidas, após o pré-treino da rede, faz-se *fine-tuning* do *encoder* e treina-se as camadas do *encoder* assim como a camada totalmente ligada adicionada. Para o treino do classificador são utilizadas as 500 imagens do conjunto B sem qualquer transformação.

4.5.5 Experiência E: Influência da utilização de transformações durante tarefa Barlow Twins

Esta experiência resume-se em compreender qual o impacto que a aplicação de diferentes agrupamentos de transformações durante o pré-treino da rede tem no classificador final.

Desta forma, analisou-se a influência de cada transformação para a resolução da tarefa proposta em Barlow Twins. Neste caso, a própria tarefa implica ter de aplicar transformações a todas as imagens de cada *batch* para que cada imagem original tenha duas réplicas com transformações distintas. Realça-se que as duas versões transformadas a partir da mesma imagem original são divididas em dois conjuntos diferentes, A e B (Figura 3.4). Para esta experiência utilizou-se agrupamentos de transformações semelhantes aos acima descritos.

(i) **Cor e Originais**

O conjunto A tem as versões transformadas de cor e o conjunto B as versões originais.

(ii) **Rotação e Originais**

À semelhança do agrupamento Cor, o conjunto A tem as versões transformadas com rotação e o conjunto B são as versões originais.

(iii) **Cor ou Rotação**

Às réplicas do conjunto A foi aplicada a transformação Cor enquanto no conjunto B encontram-se as versões transformadas com rotação.

(iv) **Cor + Rotação e Originais**

As imagens do conjunto A correspondem às versões transformadas, simultaneamente, de Cor e Rotação e o conjunto B coincide com as versões originais das imagens.

(v) **Espelhamento + Corte + Redimensionamento e Originais**

No conjunto A estão as versões transformadas simultaneamente com Espelhamento horizontal, Corte e Redimensionamento, enquanto que o conjunto B corresponde às versões originais.

(vi) **Sem Cor**

Tanto o conjunto A como o conjunto B foram obtidos aplicando às imagens originais as transformações Rotação, Corte e Redimensionamento e Espelhamento Horizontal.

(vii) **Tudo**

Neste agrupamento, os conjuntos A e B foram gerados através da aplicação simultânea de todas as transformações da Tabela 4.2.

Não se justifica averiguar a influência das transformações para a tarefa SimCLR uma vez que os *encoders* resultantes desta tarefa são os que apresentam piores desempenhos quando utilizados para inicialização da componente *encoder* do classificador. Tal deve-se ao facto de o conjunto de dados para treino ser reduzido e desta forma não é possível utilizar *batches* de grandes dimensões como sugerido pelos autores de [16].

5

Resultados

Conteúdo

Este capítulo apresenta os resultados obtidos e a respetiva análise das várias experiências descritas no Capítulo 4.

5.1 Experiência A: Comparação entre as diferentes tarefas de pretexto e classificador de referência

A fim de determinar se existe vantagem em utilizar aprendizagem auto-supervisionada para pré-treino da rede, comparam-se os melhores modelos de cada tarefa de pretexto estudada, ou seja, reconstrução de imagem, SimCLR e Barlow Twins, com o classificador de referência. Para qualquer uma das tarefas, os melhores resultados foram obtidos quando se fez *fine-tuning* das redes pré-treinadas. A Tabela 5.1 apresenta as performances do classificador de referência e dos melhores classificadores obtidos para cada tarefa de pretexto.

Para a tarefa reconstrução de imagem, o classificador com melhor desempenho foi obtido quando pré-treinado recorrendo à transformação Cor para aumentar o conjunto de treino. Neste caso, o conjunto inicial continha 1000 imagens às quais se aplicou a transformação Cor. Assim cada imagem inicial tem, exatamente, duas réplicas no conjunto de treino final, uma original e uma transformada, ou seja, um total de 2000 imagens de treino. No caso das tarefas de geração, as transformações são aplicadas antes do início do treino pelo que as imagens são constantes ao longo das várias épocas.

Para as duas tarefas de contraste, SimCLR e Barlow Twins, é necessário que cada imagem do *batch* inicial tenha duas versões distintas no *batch* final, através da aplicação de transformações. Para ambas as tarefas, o classificador com melhor desempenho foi obtido de forma semelhante, isto é, utilizando o mesmo agrupamento de transformações e a mesma quantidade de imagens. Assim, para se obter as duas réplicas aplicou-se simultaneamente transformação de cor e rotação, para que, no final, metade das versões fossem as imagens originais e as restantes imagens fossem transformadas com cor e rotação simultaneamente. Para ambas as tarefas foram utilizadas as 2000 imagens do conjunto A da Tabela 4.1. Em ambos os métodos contrastivos, os *batches* contêm 128 imagens o que significa que são necessárias 16 iterações para cobrir o conjunto inicial de imagens. As transformações são aplicadas por *batch* e não ao conjunto total de imagens diretamente, como é o caso da tarefa de geração, pelo que as diferentes épocas têm diferentes pares. O número total de amostras por época é dado pela soma do número total de pares numa época e, desta forma, para SimCLR tem-se $2 \cdot 128^2 \cdot 16 = 524.288$ amostras e para Barlow Twins existem $128^2 \cdot 16 = 262.144$ amostras no final de cada época.

Os resultados da Tabela 5.1 sugerem que as tarefas de pretexto reconstrução de imagem e Barlow Twins conseguem alcançar melhores resultados comparativamente com o classificador de referência. No entanto, é necessário recorrer-se ao teste estatístico Teste-t para se poder concluir que estas tarefas proporcionam melhores desempenhos comparativamente com o classificador de referência.

Tabela 5.1: Performance do classificador de referência e dos melhores classificadores obtidos para cada uma das tarefas de pretexto.

Nome	Tarefa de pretexto		# Amostras por Época	Eficácia média [%]	*F1 Score* médio [%]
	# Imagens Inicial	Transformação			
Referência	-	-	-	$82,8\ (\pm 3,9)$	$87,0\ (\pm 3,6)$
Reconstrução de imagem	1000	Cor	2000	$87,2\ (\pm 4,5)$	$90,9\ (\pm 3,1)$
SimCLR	2000	Cor + Rotação	524.288	$83,0\ (\pm 5,6)$	$86,9\ (\pm 5,2)$
Barlow Twins	2000	Cor + Rotação	262.144	$86,0\ (\pm 5,7)$	$89,5\ (\pm 5,0)$

Através dos resultados da Tabela 5.2, apenas é possível afirmar que a tarefa de pretexto reconstrução de imagem permitiu alcançar um classificador com um desempenho melhor do que o classificador de referência, visto que apenas neste caso o valor-p é inferior a 5%. Por norma as tarefas de contraste conseguem alcançar melhores resultados comparativamente com as tarefas de geração. Contudo, dado que a quantidade de dados disponíveis para treino das redes é reduzida, torna-se complicado gerar tantos pares negativos e a rede tem mais dificuldade em cumprir o objetivo. Por outro lado, as imagens utilizadas para o treino acabam por ser semelhantes na medida em que os objetos presentes são comuns em muitas das imagens, como é o caso das árvores, do fumo e do fogo. No entanto, os autores de SimCLR e Barlow Twins utilizam um conjunto de dados mais diversificado [29], contendo imagens de animais, carros e outros objetos, que facilita a tarefa contrastiva, conseguindo, assim, alcançar melhores performances comparativamente com outro tipo de métodos.

Tabela 5.2: Resultados dos testes estatísticos entre o classificador de referência e cada um dos classificadores obtidos com as diferentes tarefas de pretexto descritas na Tabela 5.1.

Classificador A	Classificador B	Teste-t	Valor-p
Referência	Reconstrução de imagem	$-2,4124$	$0,013$
Referência	SimCLR	$0,0524$	$0,479$
Referência	Barlow Twins	$-1,1819$	$0,126$

Tal como para o classificador de referência, são apresentados alguns exemplos de imagens classificadas para as redes obtidas para cada uma das tarefas de pretexto. As imagens classificadas corretamente pelo classificador de referência foram também classificadas acertadamente pelos três classificadores obtidos, como mostram as Figuras 5.1, 5.2 e 5.3, à exceção do exemplo da Figura 5.2f

em que o classificador obtido para a tarefa SimCLR não conseguiu identificar fogo na imagem. Em relação às imagens classificadas incorretamente, os classificadores obtidos tiveram comportamentos distintos.

(a) Probabilidade = 1,0 **(b)** Probabilidade = 1.0 **(c)** Probabilidade = 1.0 **(d)** Probabilidade = 0,4

(e) Probabilidade = 0,7 **(f)** Probabilidade = 0,6 **(g)** Probabilidade = 0,1 **(h)** Probabilidade = 0,8

(i) Probabilidade = 0,1 **(j)** Probabilidade = 0,0 **(k)** Probabilidade = 0,0 **(l)** Probabilidade = 0,0

Figura 5.1: Exemplos de imagens classificadas através do classificador obtido a partir da tarefa de pretexto reconstrução de imagem.

No caso do classificador gerado a partir da tarefa reconstrução de imagem, não se obteve nenhum FP para as imagens da Figura 5.1, o que mostra que este classificador não está totalmente dependente da cor para classificar uma imagem como contendo ou não fogo. Ademais, as Figuras 5.1c e 5.1j, que têm características muito semelhantes a nível da cor, foram ambas classificadas corretamente e com a máxima certeza, ou seja, o exemplo positivo foi classificado com probabilidade 1,0 e o exemplo negativo apresentou uma probabilidade de 0,0. Em relação aos exemplos FN, este classificador apenas não conseguiu identificar fogo em duas das cinco imagens em que o fogo tem dimensão reduzida.

A partir da Figura 5.2, verificou-se que todos os exemplos classificados erradamente pelo classificador de referência foram também incorretamente classificados pelo classificador obtido através da tarefa SimCLR, sendo que se obteve mais um FN comparativamente com o classificador de referência. Por

(a) Probabilidade = 0,9 **(b)** Probabilidade = 0.9 **(c)** Probabilidade = 1.0 **(d)** Probabilidade = 0,1

(e) Probabilidade = 0,2 **(f)** Probabilidade = 0,4 **(g)** Probabilidade = 0,1 **(h)** Probabilidade = 0,6

(i) Probabilidade = 0,1 **(j)** Probabilidade = 1,0 **(k)** Probabilidade = 0,9 **(l)** Probabilidade = 0,1

Figura 5.2: Exemplos de imagens classificadas através do classificador obtido a partir da tarefa de pretexto SimCLR.

outro lado, as probabilidades dos exemplos classificados de forma incorreta pelo classificador SimCLR estão mais afastadas do resultado esperado, comparativamente com o classificador de referência.

Finalmente, para o classificador obtido através da tarefa Barlow Twins, apenas se verificou três imagens classificadas incorretamente. Tal como para o classificador obtido para a tarefa reconstrução de imagem, este classificador não apresentou nenhum FP. Desta forma, concluiu-se que a rede não recorre apenas ao histograma da cor para classificar as imagens, visto que, mais uma vez, as Figuras 5.3c e 5.3j têm tonalidades muito semelhantes apesar de uma ter fogo e a outra não e a rede consegue classificar as imagens de forma correta. Por outro lado, ao contrário do classificador da tarefa reconstrução de imagem, as probabilidades não são tão extremas, isto é, 0,0 quando se trata de um exemplo negativo e 1,0 para exemplos positivos. Neste caso, as Figuras 5.3j e 5.3k apresentam probabilidades de 0,4 e 0,3, respetivamente, ao contrário das Figuras 5.1j e 5.1k. Em relação aos três exemplos FN, verificou-se, mais uma vez, que o classificador demonstra dificuldades em identificar fogo quando este apresenta dimensões reduzidas na imagem.

(a) Probabilidade = 1,0 **(b)** Probabilidade = 1.0 **(c)** Probabilidade = 0.9 **(d)** Probabilidade = 0,2

(e) Probabilidade = 0,2 **(f)** Probabilidade = 0,7 **(g)** Probabilidade = 0,1 **(h)** Probabilidade = 0,8

(i) Probabilidade = 0,1 **(j)** Probabilidade = 0,4 **(k)** Probabilidade = 0,3 **(l)** Probabilidade = 0,1

Figura 5.3: Exemplos de imagens classificadas através do classificador obtido a partir da tarefa de pretexto Barlow Twins.

5.2 Experiência B: Comparação entre classificadores obtidos via *transfer learning* e *fine-tuning*

Com esta experiência pretende-se comparar os resultados entre dois classificadores pré-treinados com a mesma tarefa de pretexto, porém um é obtido fazendo *fine-tuning* e o outro *transfer learning*. Para esta experiência utilizou-se as redes pré-treinadas que obtiveram melhores desempenhos descritas na Secção 5.1.

Pelos resultados das Tabelas 5.3 e 5.4 pode-se concluir que ao fazer-se *fine-tuning* em vez de *transfer learning* consegue-se alcançar uma melhor performance dos classificadores. Isto significa que, apesar de a rede previamente aprendida servir para inicializar os classificadores, os pesos aprendidos não são totalmente adequados para a tarefa de classificação necessitando de alguns ajustes.

Por outro lado, observando a coluna *transfer learning* da Tabela 5.3, a tarefa Barlow Twins aparenta proporcionar uma melhor performance comparativamente com a tarefa reconstrução de imagem. Este

Tabela 5.3: Comparação entre a performance dos melhores classificadores para cada uma das tarefas de pretexto quando é feito *transfer learning* e *fine-tuning*.

	Tarefa de pretexto			*Transfer Learning*	*Fine-tuning*
Nome	**# Imagens Inicial**	**Transformação**	**# Amostras por Época**	***F1 Score* [%]**	***F1 Score* [%]**
Reconstrução de imagem	1000	Cor	2000	$70,5\ (\pm 4,1)$	$90,9\ (\pm 3,1)$
SimCLR	2000	Cor + Rotação	524.288	$66,9\ (\pm 7,2)$	$86,9\ (\pm 5,2)$
Barlow Twins	2000	Cor + Rotação	262.144	$74,1\ (\pm 4,7)$	$88,4\ (\pm 5,0)$

Tabela 5.4: Resultados dos testes estatísticos entre os classificadores obtidos fazendo *fine-tuning* e *transfer learning*.

Classificador A	**Classificador B**	**Teste-t**	**Valor-p**
Reconstrução de imagem *Fine-tuning*	Reconstrução de imagem *Transfer learning*	$11,896$	$1,24 \times 10^{-}9$
SimCLR *Fine-tuning*	SimCLR *Transfer learning*	$6,723$	$4,26 \times 10^{-}6$
Barlow Twins *Fine-tuning*	Barlow Twins *Transfer learning*	$6,741$	$2,63 \times 10^{-}6$

resultado está relacionado com o classificador ter classificado a maioria das imagens das pastas de validação como positivas. Mais concretamente, das 146 imagens negativas do conjunto B da Tabela 4.1, 83 foram classificadas como positivas, o que corresponde a uma taxa de falsos positivos de $56,8$ %. No caso da tarefa reconstrução de imagem, o classificador resultante também tem uma taxa de falsos positivos elevada, $43,2$ %, porém significativamente inferior à obtida pelo classificador de Barlow Twins. É de realçar que estas taxas são consideravelmente reduzidas quando é feito *fine-tuning* das redes, obtendo-se $16,4$ % para Barlow twins e $17,9$ % para reconstrução de imagem.

5.3 Experiência C: Influência do número de imagens originais durante a resolução da tarefa reconstrução de imagem

O principal objetivo desta experiência é perceber se existe vantagem em utilizar os milhares de imagens disponíveis *online* para pré-treinar o modelo.

Para tal, treinou-se três *autoencoders* a reconstruir imagens, utilizando 500, 1000 e 2000 imagens do conjunto A da Tabela 4.1 e sem se aplicar nenhuma transformação. Após a resolução desta tarefa, a componente *encoder* de cada uma das redes foi extraída e utilizada para a tarefa de classificação. Procurou-se fazer *fine-tuning* em vez de *transfer learning* do *encoder* aprendido durante a fase auto-supervisionada, uma vez que são obtidos melhores resultados.

Tabela 5.5: Performance dos classificadores em função do número imagens utilizadas para a resolução da tarefa de pretexto reconstrução de imagem e sem se aplicar qualquer tipo de transformação às imagens.

Tarefa de pretexto		**Eficácia média [%]**	**F1 Score médio [%]**
Nome	**# Imagens pré-treino**		
Referência	-	$82,8\ (\pm 3,9)$	$87,0\ (\pm 3,6)$
Reconstrução de imagem	500	$84,0\ (\pm 4,5)$	$88,1\ (\pm 3,2)$
Reconstrução de imagem	1000	$84,2\ (\pm 2,6)$	$88,3\ (\pm 2,4)$
Reconstrução de imagem	2000	$84,2\ (\pm 4,3)$	$88,5\ (\pm 3,0)$

A partir dos resultados apresentados na Tabela 5.5 não se pode concluir que o aumento do número de imagens originais para o treino da rede durante a fase auto-supervisionada seja vantajoso ou prejudicial para a inicialização dos pesos do *encoder* do classificador. Comparando os resultados dos três classificadores com o classificador de referência, o desempenho dos classificadores pré-treinados para as várias quantidades de imagem aparenta superar o do classificador de referência. Contudo para ser possível confirmar esta afirmação é necessário efetuar o teste estatístico Teste-t.

A Tabela 5.6 mostra que nenhum dos classificadores obtidos supera com certeza o classificador de referência, visto que o valor-p não é inferior a 5% em nenhuma das situações. No entanto, da experiência A soubesse que ao pré-treinar a rede com 2000 imagens, 1000 originais e 1000 transformadas de cor, o classificador resultante alcançou uma performance significativamente superior ao classificador de referência. Desta forma, o aumento do número de imagens pode ser favorável para o pré-treino, contudo não é linear e é preciso averiguar mais detalhadamente que transformações podem ajudar na aprendizagem das características do fogo. A Secção 5.4 mostra a influência de aumentar o conjunto de dados através de diferentes transformações para o pré-treino da rede.

Tabela 5.6: Resultados dos testes estatísticos entre o classificador de referência e os classificadores obtidos para diferentes quantidades de imagens durante o pré-treino. Para o pré-treino não se aplicou nenhuma transformação às imagens.

Classificador A	Classificador B	Teste-t	Valor-p
Referência	Reconstrução de imagem com 500	$-0,6562$	$0,260$
Referência	Reconstrução de imagem com 1000	$-0,8450$	$0,205$
Referência	Reconstrução de imagem com 2000	$-0,9280$	$0,183$

5.4 Experiência D: Influência da utilização de transformações para aumentar o conjunto de treino durante a tarefa reconstrução de imagem

Nesta experiência, analisou-se a influência de aumentar o número total de imagens para pré-treino recorrendo à aplicação de transformações. Neste caso, cada imagem do conjunto inicial tem pelo menos uma réplica no conjunto final, sendo uma a versão original da imagem e as eventuais versões transformadas. Tomou-se como ponto de partida 1000 imagens originais do conjunto A da Tabela 4.1 e a esta aplicou-se as diversas transformações descritas no capítulo anterior (4.5.4). Para os primeiros conjuntos de transformações, isto é, Cor, Rotação, Cor ou Rotação e Cor + Rotação, variou-se o rácio de imagens transformadas entre 50% e 100% obtendo-se 1500 e 2000 imagens totais de treino, respetivamente.

Pela Figura 5.4 verifica-se que as situações em que é aplicada transformação de cor a todas as imagens selecionadas, ou seja, no caso dos agrupamentos Cor e Cor + Rotação, existe uma melhoria da performance com o incremento do número de imagens para pré-treino, de 1000 (só originais) para 1500 e 2000 (originais e transformadas). Pelo contrário, a utilização de transformações que não sejam de cor não contribui para um aperfeiçoamento da performance dos classificadores resultantes apresentando resultados semelhantes ao do classificador pré-treinado apenas com as 1000 imagens originais.

De forma a aumentar ainda mais o conjunto de imagens para treino, combinou-se as várias transformações obtendo-se no total 4000 e 5000 imagens. Para tal foram aplicados os agrupamentos de transformações Cor, Rotação e Originais (vi) e Tudo (vii) apresentados no capítulo anterior (4.5.4). É de realçar que apesar de se conseguir alcançar um conjunto de treino com mais dados, não é garantido que o aumento de dados recorrendo a múltiplas transformações resulte numa melhor performance [32].

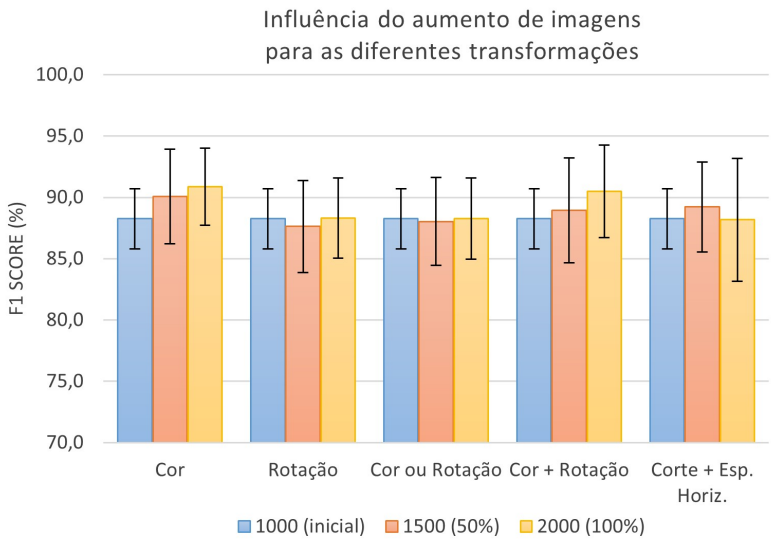


Figura 5.4: Performance dos classificadores pré-treinados para a tarefa reconstrução de imagem em função do aumento do número de imagens total através da aplicação de transformações a 50% e 100% das imagens do conjunto de treino inicial.

No caso de conjuntos de dados inicialmente reduzidos, a junção de diversas transformações como rotação, manipulação de cor e corte pode não impedir o modelo de sobre ajustar.

A Tabela 5.7 apresenta os desempenhos dos classificadores para cada um dos agrupamentos de transformações estudados. Neste caso, apesar de não se ter verificado sobre ajuste, também não se conseguiu alcançar um desempenho melhor comparativamente com o classificador pré-treinado com o agrupamento Cor. Desta forma, a melhor opção é optar por aplicar exclusivamente transformação de cor para duplicar o conjunto de treino uma vez que se consegue melhores resultados consumindo menos memória, GPU e tempo.

Tabela 5.7: Performance dos classificadores quando utilizadas 1000, 2000, 4000 e 5000 imagens recorrendo a diferentes agrupamentos de transformações para a resolução da tarefa de pretexto Reconstrução de imagem.

Agrupamento de transformações	**# Imagens Inicial**	**# Amostras por Época**	**Eficácia média [%]**	**F1 Score médio [%]**
Originais	1000	1000	$84,2\ (\pm 2,6)$	$88,1\ (\pm 3,2)$
Cor	1000	2000	$87,2\ (\pm 4,5)$	$90,9\ (\pm 3,1)$
Cor, Rotação e Originais	1000	4000	$86,0\ (\pm 3,9)$	$89,6\ (\pm 3,8)$
Tudo	1000	5000	$86,6\ (\pm 3,9)$	$90,2\ (\pm 3,2)$

5.5 Experiência E: Influência da utilização de transformações durante tarefa Barlow Twins

Esta experiência tem como finalidade analisar o efeito que os diferentes tipos de transformação, cor ou geométricas, têm quando utilizadas durante o treino da rede a resolver a tarefa Barlow Twins. Para tal exploram-se os diferentes agrupamentos referidos na Secção 4.5.5, ou seja, Cor e Originais, Rotação e Originais, Cor e Rotação, Cor + Rotação e Originais, Corte + Redimensionamento + Espelhamento e Originais, Sem Cor e Tudo.

Os resultados da Tabela 5.8 revelam que a aplicação exclusiva de pequenas rotações não permite uma boa inicialização dos pesos da rede. Contudo, quando combinada com outras transformações, como é o caso da Cor + Rotação, pode atingir resultados ao nível do desempenho do classificador de referência cuja eficácia é de $82,3 \pm 3,9$ % e o *F1 Score* é de $87,0 \pm 3,6$ %.

Tabela 5.8: Performance dos classificadores quando pré-treinados através da tarefa Barlow Twins e aplicando os vários agrupamentos de transformações. O número total de amostras por época é 262.144 para todas as situações da tabela.

Agrupamento de transformações	**Eficácia média [%]**	**F1 Score médio [%]**
Cor e Originais	$84,0\ (\pm 4,3)$	$88,4\ (\pm 3,1)$
Rotação e Originais	$82,2\ (\pm 8,3)$	$86,2\ (\pm 7,0)$
Cor e Rotação	$84,0\ (\pm 5,7)$	$87,7\ (\pm 5,4)$
Cor + Rotação e Originais	$86,0\ (\pm 5,7)$	$89,5\ (\pm 5,0)$
Corte + Redimensionamento + Espelhamento e Originais	$85,4\ (\pm 5,1)$	$89,0\ (\pm 4,6)$
Sem Cor	$83,8\ (\pm 6,6)$	$87,9\ (\pm 5,1)$
Tudo	$81,6\ (\pm 3,6)$	$86,2\ (\pm 3,1)$

6

Conclusão e Trabalho Futuro

Na área de deteção de incêndios, um dos principais desafios apontados é a falta de conjuntos de imagens legendadas para se treinar os modelos para as tarefas supervisionadas pretendidas, tais como classificação ou segmentação.

Para colmatar a escassez destes conjuntos, procurou-se desenvolver uma solução que tirasse partido dos milhares de imagens sem anotações disponíveis publicamente. Deste modo, desenvolveu-se uma metodologia que utiliza a aprendizagem auto-supervisionada, permitindo que a utilização das imagens não legendadas pudesse trazer benefícios para a tarefa final pretendida, neste caso, classificação.

O principal objetivo desta tese era determinar se este tipo de metodologia conseguia superar métodos que recorrem exclusivamente à aprendizagem supervisionada quando se tem poucos dados legendados para treino e avaliação dos modelos. Como tal, recorreu-se a três tarefas de pretexto distintas para averiguar o impacto que a aprendizagem auto-supervisionada podia ter na inicialização da rede de classificação. Para a tarefa de geração proposta, isto é, reconstrução de imagem, consegui-se superar o desempenho do classificador obtido exclusivamente através da aprendizagem supervisionada. Mais concretamente, verificou-se uma melhoria da métrica *F1 Score* em 3,9 %. Desta forma, é possível concluir que a utilização da aprendizagem auto-supervisionada para inicializar um classificador é uma mais-valia. No caso das tarefas contrastivas, apesar de não se verificar uma melhoria face ao classificador de referência, a tarefa Barlow Twins atinge uma performance ao nível do classificador de referência. O principal motivo que explica a não superação do classificador de referência incide no facto das redes necessitarem de mais imagens para resolverem as tarefas contrastivas propostas nesta tese.

Inicialmente, tinha-se como intenção a utilização de um conjunto de dados não legendados mais extenso. Contudo, tal não foi possível uma vez que o ambiente virtual onde foi desenvolvido o sistema não permitiu o aumento deste conjunto com prejuízo de terminar a sessão sem guardar os progressos alcançados.

Um fator que mostrou ser relevante para verificar a utilidade da aprendizagem auto-supervisionada foi a introdução de transformações. Procurou-se analisar a influência das diferentes transformações quando utilizadas individualmente ou em conjunto com outras. A transformação que revelou ter mais impacto foi a transformação Cor que, quando utilizada individualmente superou o desempenho do classificador de referência.

Nesta tese, apenas foi explorada a utilização da aprendizagem auto-supervisionada com finalidade de resolver a tarefa de classificação. Contudo, não está descartada a utilidade desta metodologia para resolver a tarefa supervisionada segmentação, pelo que deve ser explorada no futuro.

Numa fase inicial dos incêndios florestais, o fogo muitas vezes está oculto numa vista aérea devido à vegetação. No entanto, o fumo provocado pelo fogo rapidamente fica visível pelo que a deteção do mesmo pode ocupar um papel crucial na monitorização e acompanhamento de incêndios florestais. Futuramente, pode ser estudada a classificação e segmentação do fumo recorrendo à metodologia

proposta.

Espera-se que este trabalho sirva de motivação para aprofundar o estudo de metodologias que utilizem aprendizagem auto-supervisionada, já que ficou provado que esta pode superar métodos supervisionados quando existe escassez de dados legendados. Ademais, este tipo de aprendizagem está em fase de crescimento e, nesta tese, foi apenas abordado um pequeno conjunto de métodos. Porém, a literatura está em constante evolução pelo que poderá ser benéfico estudar outras técnicas.

Finalmente, com esta dissertação, espera-se ter desenvolvido uma metodologia que contribua para o projeto Firefront, auxiliando as forças de combate aos incêndios no acompanhamento dos fogos.

Bibliografia

[1] I. da Conservação da Natureza e das Florestas (ICNF). Incêndios rurais e área ardida – continente. Último acesso: 29.11.2020. [Online]. Available: https://www.pordata.pt/Portugal/Inc%c3%aandios+rurais+e+%c3%a1rea+ardida+%e2%80%93+Continente-1192

[2] M. Batista, B. Oliveira, P. Chaves, J. C. Ferreira, and T. Brandao, "Improved real-time wildfire detection using a surveillance system," in *IAENG*, 2019, pp. 520–526.

[3] B. Amaral, "Fire and smoke detection with weakly supervised methods," Master's thesis, Instituto Superior Técnico, 2021.

[4] S. Sharma and S. Sharma, "Activation functions in neural networks," *Towards Data Science*, vol. 6, no. 12, pp. 310–316, 2017.

[5] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv*, vol. abs/1609.04747, 2016.

[6] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21, 2021.

[7] A. Hidaka and T. Kurita, "Consecutive dimensionality reduction by canonical correlation analysis for visualization of convolutional neural networks," *Proceedings of the ISCIE International Symposium on Stochastic Systems Theory and its Applications*, vol. 2017, pp. 160–167, 12 2017.

[8] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," *Journal of big Data*, vol. 8, no. 1, pp. 1–74, 2021.

[9] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai *et al.*, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, pp. 354–377, 2018.

[10] L. Jing and Y. Tian, “Self-supervised visual feature learning with deep neural networks: A survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 4037 – 4058, 2020.

[11] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, “A survey on contrastive self-supervised learning,” *arXiv*, vol. abs/2011.00362, 2020.

[12] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang, “Self-supervised learning: Generative or contrastive,” *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2021.

[13] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[14] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” in *Proceedings of the 25th International Conference on Machine learning*, 2008, pp. 1096–1103.

[15] N. Komodakis and S. Gidaris, “Unsupervised representation learning by predicting image rotations,” in *International Conference on Learning Representations (ICLR)*, 2018.

[16] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” in *International Conference on Machine Learning*. PMLR, 2020, pp. 1597–1607.

[17] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, “Unsupervised learning of visual features by contrasting cluster assignments,” in *Thirty-fourth Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[18] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny, “Barlow twins: Self-supervised learning via redundancy reduction,” *arXiv preprint arXiv:2103.03230*, 2021.

[19] B. U. Toreyin and A. E. Cetin, “Online detection of fire in video,” in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–5.

[20] T. Çelik and H. Demirel, “Fire detection in video sequences using a generic color model,” *Fire Safety Journal*, vol. 44, no. 2, pp. 147 – 158, 2009.

[21] C. Yuan, Z. Liu, and Y. Zhang, “Aerial images-based forest fire detection for firefighting using optical remote sensing techniques and unmanned aerial vehicles,” *Journal of Intelligent & Robotic Systems*, vol. 88, no. 2-4, pp. 635–654, 2017.

[22] C. Yuan, Y. Zhang, and Z. Liu, “A survey on technologies for automatic forest fire monitoring, detection, and fighting using unmanned aerial vehicles and remote sensing techniques,” *Canadian journal of forest research*, vol. 45, no. 7, pp. 783–792, 2015.

[23] H. Cruz, M. Eckert, J. Meneses, and J.-F. Martínez, "Efficient forest fire detection index for application in unmanned aerial systems (UASs)," *Sensors*, vol. 16, no. 6, p. 893, 2016.

[24] J. Lehr, C. Gerson, M. Ajami, and J. Krüger, "Development of a fire detection based on the analysis of video data by means of convolutional neural networks," in *Iberian Conference on Pattern Recognition and Image Analysis*, 2019, pp. 497–507.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[27] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5 MB model size," *arXiv preprint arXiv:1602.07360*, 2016.

[28] Q. Zhang, J. Xu, L. Xu, and H. Guo, "Deep convolutional neural networks for forest fire detection," in *Proceedings of the 2016 International Forum on Management, Education and Information Technology Application*. Atlantis Press, 2016, pp. 568–575.

[29] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[30] Y. Chen, Y. Zhang, J. Xin, G. Wang, L. Mu, Y. Yi, H. Liu, and D. Liu, "UAV image-based forest fire detection approach using convolutional neural network," in *14th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, 2019, pp. 2118–2123.

[31] A. Chaudhary, "The illustrated simclr framework," 2020, Último acesso: 17.09.2021. [Online]. Available: https://amitness.com/2020/03/illustrated-simclr

[32] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, pp. 1–48, 2019.

[33] A. A. Bombeiros distrito guarda, "Bombeiros portugueses," Último acesso: 29.09.2021. [Online]. Available: http://www.bombeiros.pt/galeria/index.php

[34] S. Parab and S. Bhalerao, "Choosing statistical test," *International journal of Ayurveda research*, vol. 1, no. 3, p. 187–191, 2010.

[35] A. Field and G. Hole, *How to design and report experiments*. SAGE Publications, 2003, p. 132–135.

[36] E. Bisong, *Google Colaboratory*. Berkeley, CA: Apress, 2019, pp. 59–64.

[37] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[38] F. Chollet *et al.* (2015) Keras. [Online]. Available: https://github.com/fchollet/keras