Redes Neuronais Espaciais e Temporais para a Compreensão de Vídeo em Sistemas Embebidos

Paulo Renato Borges Duarte

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática
(2º ciclo de estudos)

Orientador: Prof. Doutor Luís Filipe Barbosa de Almeida Alexandre
Co-orientador: Prof. Doutor João Carlos Raposo Neves

junho de 2023

Declaração de Integridade

Eu, Paulo Renato Borges Duarte, que abaixo assino, estudante com o número de inscrição M11324 do 2º Ciclo de Engenharia Informática da Faculdade de Engenharias, declaro ter desenvolvido o presente trabalho e elaborado o presente texto em total consonância com o Código de Integridades da Universidade da Beira Interior.

Mais concretamente afirmo não ter incorrido em qualquer das variedades de Fraude Académica, e que aqui declaro conhecer, que em particular atendi à exigida referenciação de frases, extratos, imagens e outras formas de trabalho intelectual, e assumindo assim na íntegra as responsabilidades da autoria.

Assinado por: **PAULO RENATO BORGES DUARTE**
Num. de Identificação: 15909528
Data: 2023.06.13 10:57:10 +0100

Universidade da Beira Interior, Covilhã 12/06/2023

Agradecimentos

À minha família, agradeço por toda a ajuda e apoio incondicional que me deram ao longo de todo o meu percurso académico. A dedicação incansável que demonstraram durante toda a minha jornada académica foi um pilar fundamental para que eu pudesse superar todos os desafios e alcançar os meus objetivos.
À minha namorada, Mariana, quero expressar a minha gratidão por todo o apoio, paciência e por ter acreditado sempre em mim. Obrigado por me encorajares sempre face aos desafios existentes e por todas as vitórias que celebraste comigo.
Ao professor Luís Alexandre, quero agradecer por toda a orientação, apoio e dedicação ao longo do meu percurso no desenvolvimento desta dissertação. O seu vasto conhecimento, a sua disponibilidade e prontidão em esclarecer todas as minhas dúvidas foram fundamentais para o sucesso da minha dissertação.
Ao professor João Neves, quero agradecer a sua valiosa contribuição como meu professor co-orientador. A sua experiência e conhecimento da área foram fundamentais para ampliar o meu saber.
À DeepNeuronic, quero expressar a minha gratidão pela concessão da bolsa de investigação que me permitiu desenvolver este projeto no contexto da minha dissertação de mestrado. Para além do suporte financeiro, gostaria de expressar a minha admiração pela iniciativa em apoiar projetos académicos de investigação. Apoiado por CENTRO-01-0247-FEDER-113023 - DeepNeuronic.

Resumo

A deteção e classificação de ação humana em vídeo são, hoje em dia, tarefas de extrema importância da área de Visão Computacional. Tal importância é atribuída a estas tarefas devido à necessidade de detetar atividade criminosa ou situações de perigo, tornando possível a prevenção e a rápida intervenção no caso de ocorrências das mesmas. Um problema subjacente à utilização desta tecnologia é, precisamente, o elevado poder computacional que lhe está associado, seja a treinar as redes de Aprendizagem Profunda ou na própria inferência. Os dispositivos usados para desempenhar as funções dos sistemas de vigilância são, sobretudo, dispositivos de baixo poder computacional, devido principalmente a fatores como: o elevado custo das placas gráficas e a sua dimensão. É aqui que surgem os problemas que esta dissertação se propõe a tentar resolver. Em virtude da impossibilidade da fase de treino de um modelo ser realizada nos próprios dispositivos e, dado tal processo não ser indispensável, uma vez que esta fase pode ser efetuada em dispositivos com elevado poder computacional, torna-se necessário otimizar o modelo para que este possa ter o menor tempo de inferência e tamanho com a melhor taxa de acertos.
Para tentar solucionar este problema, este projeto visa explorar diversas técnicas/métodos de otimização, tais como: fazer uso das camadas convolucionais separáveis, quantização, *knowledge distillation*, entre outros; assim como criar métodos ou algoritmos que possam ser adicionados ou substituam parte de uma rede.

Palavras-chave

Visão computacional, redes de aprendizagem profunda, deteção de objetos, classificação em vídeo, dispositivos de baixo poder computacional, otimização

Abstract

The detection and classification of human action in video are, nowadays, extremely important tasks in the field of Computer Vision. Such importance is attributed to these tasks due to the need to detect criminal activity or dangerous situations, making prevention and quick intervention possible in the event of occurrences. A problem underlying the use of this technology is precisely the high computational power associated with it, whether training Deep Learning networks or inference itself. The devices used to carry out the functions of surveillance systems are, above all, devices with low computational power, mainly due to factors such as: the high cost of graphics cards and their size. It is here that the problems that this dissertation proposes to try to solve arise. Due to the impossibility of the training phase of a model being carried out on the devices themselves and, given that such a process is not essential, since this phase can be carried out on devices with high computational power, it becomes necessary to optimize the model so that it may have the shortest inference time and size with the best hit rate.
To try to solve this problem, this project aims to explore several optimization techniques/methods, such as: making use of separable convolutional layers, quantization, *knowledge distillation*, among others; as well as creating methods or algorithms that can be added to or replace part of a network.

Keywords

Computer vision, deep learning networks, object detection, video classification, low-power devices, optimization

Índice

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

CFAM	*Channel Fusion and Attention Mechanism*
CNN	*Convolutional Neural Network*
CSP	*Cross Stage Partial*
FLOP	*Floating Point Operation*
FPN	*Feature Pyramid Network*
GFLOP	*Giga Floating Point Operation*
GIoU	*Generalized Intersection over Union*
GPU	*Graphics Processing Unit*
IoT	*Internet of Things*
IoU	*Intersection over Union*
LFB	*Long-term Feature Bank*
mAP	*mean Average Precision*
MLP	*Multilayer Perceptron*
NMS	*Non-maximum supression*
RoI	*Region of Interest*
SGD	*Stochastic gradient descent*
UBI	Universidade da Beira Interior
ViT	*Vision Transformers*
WOO	*Watch Only Once*
YOLO	*You Only Look Once*
YOWO	*You Only Watch Once*

Capítulo 1

Introdução

Este documento descreve o trabalho desenvolvido ao longo do ano para a unidade curricular de dissertação inserida no plano curricular de mestrado em Engenharia Informática na Universidade da Beira Interior. Neste capítulo é introduzido o problema (secção 1.1), os objetivos (secção 1.2) e a estrutura do documento (secção 1.3).

1.1 Problema

A deteção e a classificação de ações humanas em vídeo é uma atividade que, hoje em dia, tem um elevado custo computacional, sendo que os resultados obtidos pelos modelos ainda estão longe de serem os ideais. Esta atividade permite detetar ações através da criação de *bounding boxes* em volta, assim como a sua respetiva classificação, sendo possível às máquinas perceber que tipo de atividade está a ser feita e onde, tendo em conta não só a atividade em si, como o seu contexto espacial, a interação com pessoas ou objetos e até a sua duração.
A deteção e classificação de ações humanas em vídeo é cada vez mais utilizada em diversos cenários, tais como: na vigilância de espaços públicos, na análise de imagem médica, na navegação em veículos autónomos, no controlo de qualidade industrial e até em sistemas avançados de lançamento de mísseis na indústria militar.
Os modelos existentes atualmente ainda são bastante limitados, uma vez que não produzem os resultados desejados (ainda têm taxas de acerto bastante baixas) e o custo temporal de inferência ainda é bastante alto.
Com este problema identificado, serão explorados mecanismos existentes e criados novos para tentar reduzir o tempo de inferência e, simultaneamente, aumentar a taxa de acertos de classificação de ações dos modelos.
Por fim, o objetivo é obter uma rede nova que permita a dispositivos de baixo processamento reconhecer ações humanas corretamente e mais rapidamente.

1.2 Objetivos

Os principais objetivos desta dissertação são:

- Estudar e explorar os métodos de deteção e classificação de vídeo do estado da arte;
- Explorar as diversas técnicas de otimização de redes neuronais;
- Desenvolver e implementar mecanismos para otimizar redes;
- Analisar os resultados face aos mecanismos implementados.

1.3 Estrutura do Documento

Este relatório de dissertação é constituído pelos seguintes capítulos:

1. **Introdução** - este capítulo visa introduzir o tema desta dissertação, apresentar os principais objetivos e descrever a estrutura deste relatório;
2. ***Background* e Trabalhos Relacionados** - este capítulo descreve os artigos científicos estudados que tiveram um papel essencial para conhecer e compreender as redes, os conjuntos de dados existentes, as técnicas de otimização mais conhecidas e as métricas mais importantes neste contexto.
3. **Metodologia** - este capítulo visa explicar a escolha do modelo que serviu de base, assim como descrever as técnicas de optimização utilizadas e os métodos desenvolvidos;
4. **Resultados e Discussão** - este capítulo mostra os resultados e algumas conclusões que foram inferidas a partir dos mesmos;
5. **Conclusões e Trabalho Futuro** - este capítulo visa concluir as ideias inferidas ao longo do ano, assim como expor os procedimentos futuros a realizar neste projeto.

Capítulo 2

Background e Trabalhos Relacionados

Neste capítulo são explicados os conceitos principais sobre deteção e classificação em vídeo. Aqui, é elucidado no que consiste a deteção e a classificação em vídeo, descrito os principais conjuntos de dados utilizados, assim como alguns métodos do estado da arte e os principais métodos de otimização.

2.1 Visão Geral

A deteção e classificação em vídeo tem tido, nos últimos anos, bastante destaque na comunidade científica, devido às inovações significativas que têm sido feitas nesta área. Essas inovações incluem melhorias em diversas áreas, bem como a possibilidade de aplicação destes sistemas baseados em redes de deteção e classificação em vídeo em vários setores. A localização em vídeos é fortemente dependente da informação temporal, ao contrário da deteção de objetos em imagens. Isso significa que criar uma solução eficaz que possa extrair e relacionar características espaciais e temporais representa um desafio significativo para os investigadores dessa área. A deteção e a classificação em vídeo requerem a capacidade de compreender os movimentos e as interações dos objetos em movimento, bem como a capacidade de entender como esses objetos se movem ao longo do tempo. É preciso uma combinação de técnicas de processamento de imagem, análise de movimento e de aprendizagem automática para desenvolver uma solução eficaz para a deteção e classificação em vídeo.

2.2 Métricas

Ao desenvolver um projeto que envolva a otimização de uma rede, é fundamental medir o desempenho desta para avaliar se as alterações realizadas melhoram ou prejudicam o seu desempenho. Para isso, é importante definir as métricas a serem analisadas.
Neste projeto, as principais métricas selecionadas para avaliar o desempenho de cada uma das redes foram:

- a frame-mAP - mede a área da curva de precisão-recall das detecções de cada *frame*. Esta métrica é importante para avaliar a precisão das detecções em cada *frame*, o que é especialmente relevante em aplicações de processamento de vídeo em tempo real. Quanto maior for a taxa de acertos relativamente à frame-mAP, mais precisas e confiáveis são as deteções produzidas em cada *frame*;
- a video-mAP - foca-se em ações que ocorrem ao longo do tempo, representando a deteção de objetos em movimento. Esta métrica é calculada pela média de IoU (Intersection over Union) com o ground truth em todas as frames de um vídeo. Se o resultado for

superior a um determinado *threshold* e a classificação for feita corretamente, então a ação em movimento é considerada uma instância correta. Esta métrica é importante para avaliar a capacidade da rede em detetar objetos em movimento e reconhecer ações em vídeos;

- o tempo de inferência - é uma métrica importante que representa o tempo que a rede leva para realizar uma *forward propagation*. Esta métrica é fundamental em redes que exigem respostas em tempo real, como em sistemas de visão computacional. A redução do tempo de inferência é um dos principais objetivos deste projeto, tornando esta métrica crítica para avaliar o sucesso das otimizações implementadas;
- o número de parâmetros;
- o número de GFLOPs - representa o número total de operações executadas por um modelo, sendo que 1 GFLOPs $= 1 \times 10^9$ FLOPs;

Em suma, as métricas selecionadas para avaliar o desempenho da rede neste projeto foram a frame-mAP, a video-mAP, o tempo de inferência, o número de parâmetros e o número de operações. Estas métricas permitem avaliar a precisão das detecções em cada frame, a capacidade da rede em detectar e classificar ações em movimento, além de avaliar o desempenho em tempo real da rede.

2.3 Métodos de Otimização

O processo de inferência de um modelo de aprendizagem profunda é o processo de *forward propagation* que, dada uma entrada, devolve uma saída. De forma geral, a otimização da inferência é um fator crítico para tornar os modelos de aprendizagem profunda práticos e escaláveis.
Em particular, a otimização do tempo de inferência é especialmente importante quando se pretende utilizar estes modelos em dispositivos de baixo poder computacional, como dispositivos móveis ou sistemas embutidos em câmaras de videovigilância. Como destacado em [1], conhecer o tempo de inferência de um determinado modelo pode ser útil para fazer ajustes na arquitetura da rede de forma a otimizar a inferência e a torná-la mais eficiente.
Nesse sentido, existem várias técnicas clássicas de otimização que permitem a redução do tamanho do modelo e do número de operações realizadas. Estas técnicas visam reduzir a complexidade do modelo, minimizando o número de operações e de parâmetros necessários para realizar a inferência. Estas técnicas podem ainda incluir a simplificação da arquitetura da rede, a remoção de camadas redundantes ou a redução do número de canais de entrada ou de saída.
Além disso, a aplicação de técnicas de quantização nos pesos e nas ativações também podem ser utilizadas para reduzir o tamanho do modelo e o tempo de inferência. Essas técnicas permitem a representação de parâmetros de rede com menos bits, reduzindo o tamanho da memória necessária para armazenar o modelo e o número de operações necessárias para realizar a inferência.

Em geral, o objetivo de todas essas técnicas de otimização é reduzir o tempo de inferência, sem comprometer a qualidade da saída da rede. Com as técnicas de otimização adequadas, é possível reduzir o tempo de inferência de uma rede em segundos e até mesmo modificar drasticamente a saída do modelo.

2.3.1 Redução do Tamanho do Modelo

Quando se fala em modelos de aprendizagem profunda, o seu tamanho pode ser um fator crucial para a sua eficiência e viabilidade em diferentes aplicações. Modelos de grande dimensão consomem mais recursos computacionais, demoram mais tempo a serem carregados e treinados, e ocupam mais espaço em disco. Por esta razão, é comum recorrer a técnicas de redução de modelo para diminuir a sua dimensão, sem prejudicar significativamente o seu desempenho.

A redução do tamanho do modelo pode trazer diversos benefícios em diferentes contextos. Por exemplo, em dispositivos com poder computacional limitado, como os *smartphones* ou dispositivos IoT, a utilização de modelos mais leves pode ser essencial para permitir o seu funcionamento eficiente e reduzir o consumo de energia. Além disso, em sistemas que requerem um elevado tempo de resposta, como sistemas de reconhecimento de voz ou de tradução em tempo real, a utilização de modelos mais leves pode ser crítica para assegurar uma resposta rápida e em tempo útil.

Outros benefícios da redução de modelo incluem um menor tempo de importação, uma compilação mais rápida e um menor espaço em disco ocupado pelo modelo. Isto pode ser particularmente importante em aplicações em que o tempo de resposta é um fator crítico ou em que é necessário armazenar vários modelos diferentes em simultâneo.

Existem várias técnicas de redução de modelo disponíveis, desde a remoção de camadas ou neurónios redundantes até à utilização de técnicas de quantização. O objetivo final é obter sempre um modelo mais leve, sem prejudicar significativamente o seu desempenho, e que possa ser utilizado de forma eficiente em diferentes aplicações.

2.3.1.1 Quantização

A quantização é um processo utilizado em diversas áreas da computação, que tem como objetivo reduzir a complexidade dos dados manipulados, ao mapear valores pertencentes a um conjunto grande para um conjunto menor. Em outras palavras, a quantização consiste em aproximar um valor contínuo por um valor discreto, seja ele um número inteiro ou um número com menos casas decimais (por exemplo, transformar 10.8912312 em 10.9).

Na área de aprendizagem automática, a quantização é utilizada para reduzir a complexidade de modelos de aprendizado profunda, como redes neuronais. Ao reduzir a precisão dos parâmetros do modelo, é possível reduzir o seu tamanho e, consequentemente, o seu consumo de memória e poder computacional. Além disso, a quantização também pode melhorar o desempenho do modelo em dispositivos com recursos limitados, como smartphones e dispositivos de IoT.

2.3.1.2 *Knowledge Distillation*

O método de *Knowledge Distillation* é uma técnica de transferência de conhecimento que permite treinar modelos menores e mais simples com base no conhecimento adquirido por modelos maiores e mais complexos. Nesta técnica, um modelo maior e mais preciso é chamado de modelo *teacher*, e um modelo menor e computacionalmente menos exigente é chamado de modelo *student*. A ideia principal é que o modelo *student* possa aprender com o modelo *teacher*, sem a necessidade de processar toda a complexidade do modelo *teacher*. Em vez disso, o modelo *student* aprende apenas o que é considerado mais importante e relevante pelo modelo *teacher*.
O método de *Knowledge Distillation* é particularmente útil em cenários em que se pretende implementar modelos em dispositivos com recursos limitados. Nesses casos, é desejável ter modelos menores e mais eficientes que possam ser executados em tempo real, com menos recursos computacionais e de energia. Além disso, o método de *Knowledge Distillation* também pode ser usado para melhorar o desempenho do modelo *student* em tarefas específicas, como classificação de imagens, deteção de objetos, reconhecimento de fala, entre outras.
Para implementar o método de *Knowledge Distillation*, é necessário treinar o modelo *teacher* com um conjunto de dados e, de seguida, usar esse modelo para treinar o modelo *student*. Durante o treino, o modelo *teacher* fornece informações adicionais ao modelo *student*, normalmente na forma de probabilidades suaves, que o modelo *student* usa para ajustar os seus pesos. O processo de treino do modelo *student* é, portanto, uma combinação de aprendizagem supervisionada e aprendizagem com base em informações fornecidas pelo modelo *teacher*.

2.3.1.3 Partilha de Pesos

A partilha de pesos é uma técnica que permite reduzir significativamente o número de pesos que precisam de ser armazenados, o que é especialmente útil em sistemas com recursos limitados de hardware ou memória. Esta técnica consiste em partilhar os mesmos pesos entre vários neurónios, reduzindo assim a quantidade de informação a ser armazenada e diminuindo o tempo de treino da rede.
Uma das técnicas mais conhecidas para implementar esta técnica é o algoritmo K-Means. Este algoritmo é utilizado para agrupar neurónios que possuam pesos semelhantes num número menor de grupos ou *clusters*. Cada *cluster* representa um grupo de neurónios que partilham os mesmos pesos, em vez de ter pesos diferentes para cada neurónio. Isso reduz a quantidade de informação que precisa de ser armazenada para cada camada da rede.

2.3.2 Redução do Número de Operações

Quando se reduz o número de operações num modelo de aprendizagem profunda, estamos, na verdade, a substituir algumas operações por outras que são mais eficientes e consomem menos recursos computacionais. Existem várias técnicas que podem ser utilizadas para alcançar essa redução, sendo algumas delas: o *pooling*, as convoluções separáveis e o *model pruning*.

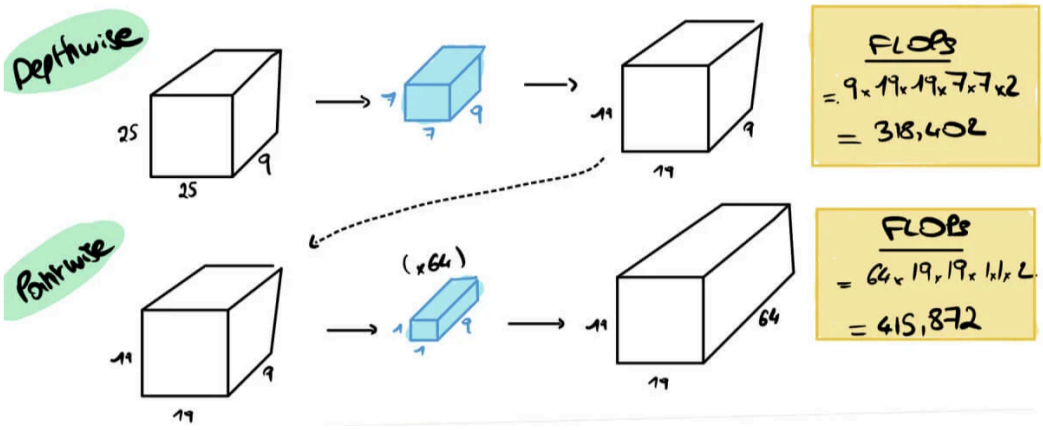

Figura 2.1: Funcionamento das camadas de convolução separáveis. Retirada de [1]

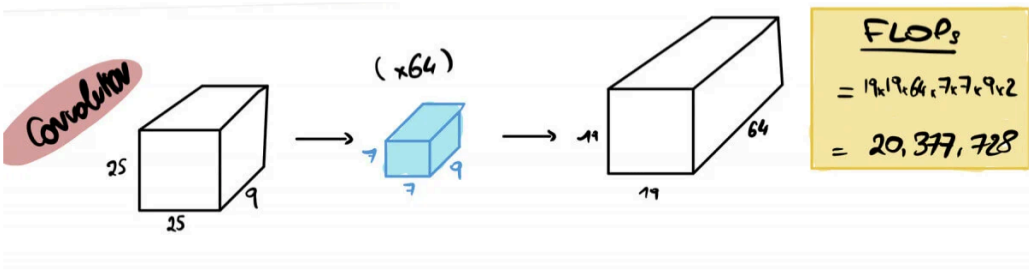

Figura 2.2: Funcionamento das camadas de convolução convencionais. Retirada de [1]

2.3.2.1 Convoluções Separáveis

As redes convolucionais 3D têm sido amplamente utilizadas na área de Visão Computacional 3D devido à sua capacidade em extrair características de alta dimensão a partir de dados 3D. No entanto, um problema comum é que o número de parâmetros tende a crescer exponencialmente quando são utilizadas camadas de convolução 3D 2.2, o que pode originar problemas devido à capacidade limitada do poder computacional disponível.

Para contornar este problema, podem-se utilizar camadas de convolução separáveis 2.1, as quais consistem em duas camadas convolucionais: uma camada de convolução *depthwise* e uma camada de convolução *pointwise*. Uma convolução *depthwise* é uma convolução normal que não aumenta a profundidade do mapa de características, o que contribui automaticamente para a diminuição do número de operações. Uma convolução *pointwise* é uma convolução 1×1, na qual cada filtro itera sobre cada pixel da imagem.

A utilização de convoluções separáveis pode reduzir significativamente o número de parâmetros necessários em comparação com convoluções 3D tradicionais, sem perda significativa de desempenho em tarefas de Visão Computacional 3D. Além disso, as convoluções separáveis podem melhorar a eficiência computacional e a velocidade de treino da rede. Portanto, é uma técnica amplamente utilizada para otimizar a arquitetura da rede e garantir que a rede seja adequada para dispositivos com baixo poder computacional.

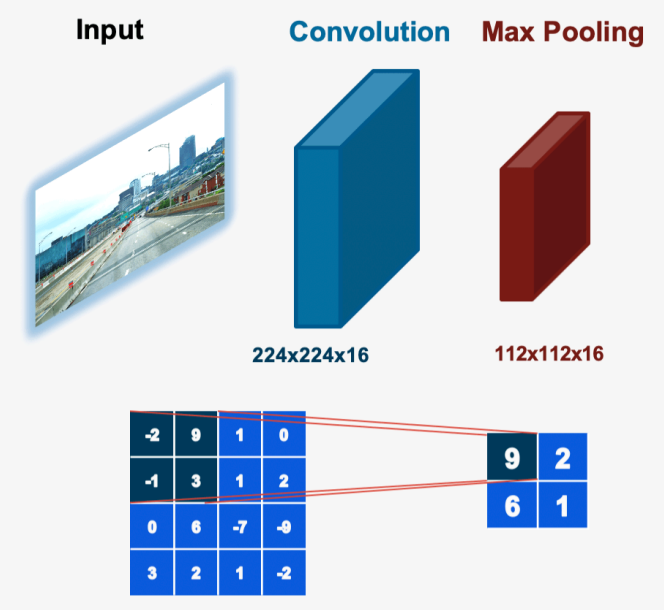

Figura 2.3: Técnica de Pooling. Retirada de [1]

2.3.2.2 *Pooling*

O *Pooling* 2.3 é uma técnica comum em redes neuronais convolucionais para reduzir a dimensionalidade dos mapas de características. A principal função do *Pooling* é diminuir o tamanho do mapa de características enquanto preserva as informações mais importantes. Em regra geral, essas camadas de *subsampling* são colocadas após as camadas de convolução.

Assim, o *Pooling* é uma técnica amplamente utilizada em modelos de Visão Computacional, com a finalidade de reduzir a complexidade computacional e o número de parâmetros da rede. Geralmente, esta técnica é aplicada em janelas com dimensões fixas, como por exemplo, 2×2, e são usados operadores simples, como a média ou o máximo, para agregar os valores das janelas num único valor representativo. Desta forma, o tamanho da representação é reduzido, mantendo as informações mais relevantes do mapa de características.

2.3.2.3 *Model Pruning*

Pruning 2.4 é uma técnica de otimização em redes neuronais que consiste na remoção dos parâmetros redundantes com o objetivo de preservar a precisão original da rede. O processo de *Pruning* começa por identificar os pesos que possuem maior importância e, em seguida, remover os pesos de menor importância.

A técnica de *Pruning* tem sido amplamente utilizada para reduzir a complexidade computacional de redes profundas, uma vez que uma grande quantidade de parâmetros pode aumentar significativamente o tempo de treino e a memória necessária para armazenar a rede.

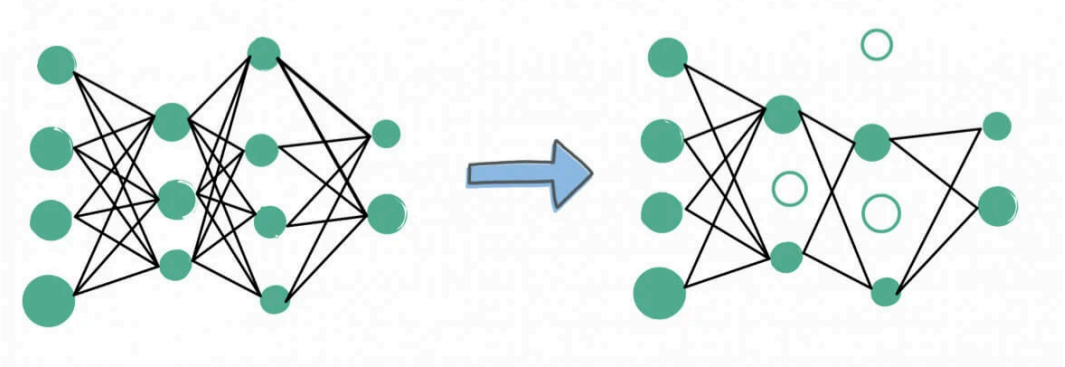

Figura 2.4: Técnica de *Model Pruning*. Retirada de [1]

2.4 Conjuntos de dados

Durante o desenvolvimento deste projeto, foram utilizados principalmente dois conjuntos de dados para fazer experiências com os métodos propostos: o UCF101-24 (secção 2.4.1) e o AVA (secção 2.4.2).

2.4.1 UCF101-24

O UCF101-24 é um subconjunto do conjunto de dados UCF101 [9], que contém vídeos realistas que representam diversas ações. Este subconjunto em particular contém 24 classes de ação diferentes e um total de 3207 vídeos. Cada vídeo pode apresentar mais de uma ação a ocorrer simultaneamente, com limites temporais e espaciais distintos, o que o torna um conjunto de dados bastante desafiante.

2.4.2 AVA

O AVA, por sua vez, contém 80 classes de ação anotadas em 430 vídeos de 15 minutos, resultando num total de 1.58 milhões de anotações de ação, já que cada pessoa pode estar associada a múltiplas ações. Esse conjunto de dados é particularmente desafiante, pois cada *frame* pode apresentar várias pessoas, cada uma com várias anotações.

Ambos os conjuntos de dados são utilizados para avaliar o desempenho da rede em termos de deteção e classificação de ações em vídeos, permitindo assim a validação da eficácia da rede e das técnicas utilizadas para otimização. A escolha desses conjuntos de dados é importante para a análise comparativa com outras redes do estado da arte e para a identificação de possíveis limitações ou pontos fortes da rede proposta.

2.5 Deteção de objetos

Deteção de objetos é, hoje em dia, uma tarefa de muita importância na área de Visão Computacional (*Computer Vision*) que consiste, essencialmente, em detetar instâncias de diversos objetos, tais como: pessoas, animais, carros, etc. Detetar pessoas em clipes de vídeo é, atualmente, muito usado em sistemas de vigilância.

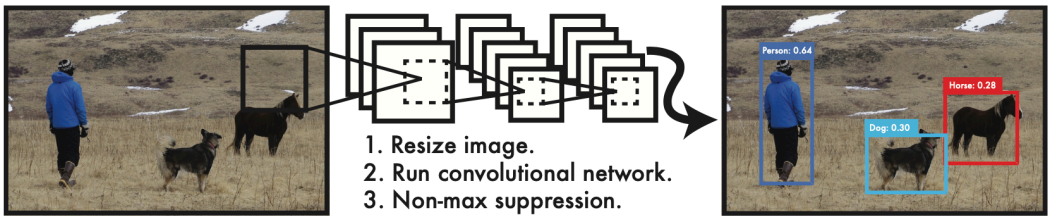

Figura 2.5: Arquitetura do YOLO. Imagem retirada de [2].

Com o rápido avanço tecnológico das técnicas baseadas em Aprendizagem Profunda, a deteção de objetos tem tido um crescimento enorme. Com as redes de Aprendizagem Profunda e com as GPUs a tornarem-se cada vez melhores e mais potentes, respetivamente, o desempenho na deteção de objetos tem tido melhorias significativas.
Existem, de forma geral, como é apresentado em [10], duas maneiras de detetar objetos, através de: técnicas de processamento de imagem ou de métodos de Aprendizagem Profunda.
Relativamente às técnicas de processamento de imagem, embora estas não necessitem de ter imagens anotadas (para o treino supervisionado), estão restritas a múltiplos fatores, tais como: à oclusão de objetos e até à iluminação e a sombras.
Relativamente aos métodos baseados em Aprendizagem Profunda, estes são significativamente mais robustos às fraquezas das técnicas de processamento de imagem, no entanto necessitam de uma enorme quantidade de dados para treino, têm um elevado custo na anotação de imagens e estão largamente limitados pelo poder computacional das GPUs.
Em suma, a deteção de objetos é uma área de grande relevância na Visão Computacional, e as técnicas baseadas em Aprendizagem Profunda têm impulsionado significativamente o desempenho nesta tarefa.

2.5.1 YOLO

Neste artigo [2] é apresentado o YOLO 2.5, uma rede neuronal que prevê *bounding boxes* e as respetivas probabilidades de classe a partir de imagens numa única avaliação.
Esta rede utiliza as características de toda a imagem para prever cada *bounding box*. Basicamente, este modelo divide a imagem de entrada numa grelha de $S \times S$, sendo que cada uma das células da grelha irá prever B *bounding boxes* com os respetivos valores de confiança, como se pode ver na figura 2.6. Caso os valores de confiança sejam próximos ou iguais a zero, significa que não existe nenhum objeto numa determinada célula da grelha. O valor de confiança ideal é aquele que for igual à IoU entre a *box* prevista e o *ground truth*.
Cada uma das *bounding boxes* consiste em 5 valores previstos:

- (x, y) que representa o centro da *box* relativamente aos limites da célula;
- a largura e a altura;
- o valor de confiança que corresponde à IoU entre a *box* prevista e o *ground truth*.

Para além disso, cada célula da grelha prevê C probabilidades condicionadas de classe, $Pr(Class_i|Object)$, sendo que estas são condicionadas à célula que contém um objeto.

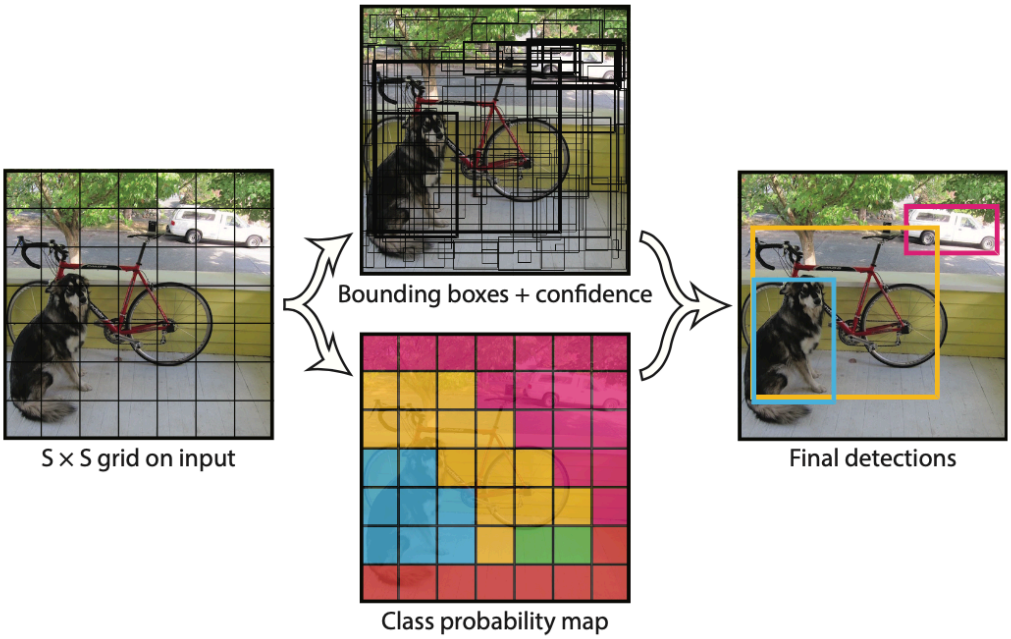

Figura 2.6: Funcionamento do modelo. Imagem retirada de [2].

Este modelo consiste em 24 camadas convolucionais seguidas por 2 camadas *fully connected*, sendo que as primeiras camadas convolucionais extraem características da imagem e que as camadas *fully connected* fazem a previsão da saída (as coordenadas e as probabilidades).

2.5.2 YOLOv5

O YOLOv5 [11] é um modelo de deteção de objetos desenvolvido pela Ultralytics e lançado em maio de 2020. O YOLOv5 apresenta várias melhorias em relação às versões anteriores do YOLO, como o uso de uma arquitetura baseada em CSP (cross stage partial) para reduzir o custo computacional, além de usar técnicas de aumento de dados para melhorar a capacidade de generalização do modelo.
O YOLOv5 é treinado em várias escalas de imagem, permitindo que este detecte objetos com diferentes tamanhos. Este modelo usa uma técnica chamada *mosaic data augmentation*, que combina várias imagens numa única imagem durante o treino, para melhorar a capacidade de generalização do modelo.
Relativamente ao desempenho, o YOLOv5 tem um desempenho muito bom em termos de velocidade e precisão. A versão YOLOv5x, a mais poderosa do modelo, pode atingir uma velocidade de 200 FPS com uma NVIDIA V100. Este método atingiu uma AP de 50.7% no conjunto de dados MS COCO, que é um conjunto de dados de referência para avaliação de modelos de deteção de objetos.

2.5.3 YOLOv8

O YOLOv8 [11] foi lançado em janeiro de 2023 pela Ultralytics, a mesma empresa que desenvolveu o YOLOv5. Embora ainda não exista um artigo científico publicado sobre esta versão do YOLO, já se sabe que a rede de suporte é idêntica à do YOLOv5.

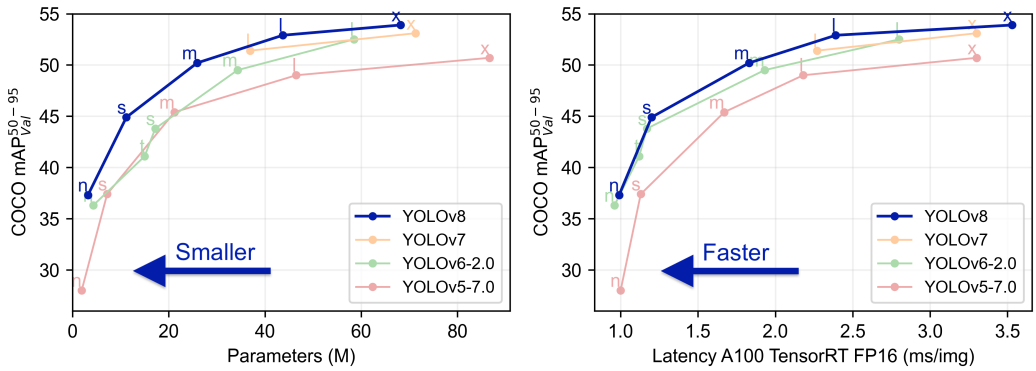

Figura 2.7: Comparação do desempenho das diferentes versões do YOLO. Imagem retirada de [3].

O YOLOv8 é apresentado em cinco versões escaláveis, desde o YOLOv8n (nano) até ao YOLOv8x (extra-large).
Relativamente aos resultados, foram realizados testes com o conjunto de dados MS COCO, sendo mostrado que o modelo YOLOv8x, com tamanho de imagem de 640 píxeis, conseguiu atingir uma AP de 53,9% (comparado com os 50,7% do YOLOv5 na mesma configuração), como se pode ver na figura 2.7, alcançando uma velocidade de 280 FPS com a NVIDIA A100 com TensorRT.
O código-fonte deste projeto foi disponibilizado pela empresa que o criou em [3]

2.6 Redes Convolucionais 3D

As redes de convolução 3D (CNNs 3D) são um tipo de modelo de aprendizagem profunda utilizada para análise de vídeos. Ao contrário das CNNs convencionais para processamento de imagem, as CNNs 3D são capazes de processar informações ao longo do tempo, utilizando volumes de dados tridimensionais.
As CNNs 3D são constituídas por camadas de convolução 3D, que são capazes de extrair características de cada frame do vídeo na dimensão espacial e temporal, permitindo que o modelo capture informações de movimento e a dinâmica da cena. Além disso, as CNNs 3D também possuem camadas de *pooling* 3D, responsáveis por reduzir a dimensionalidade dos mapas de características, e camadas *fully-connected*, que utilizam as informações extraídas pelas camadas de convolução para realizar a classificação da ação dos vídeos.
Estas redes são amplamente utilizadas em aplicações como o reconhecimento de ações humanas em vídeos, o reconhecimento de gestos, a análise de movimentos em desportos, entre outros. No entanto, devido à sua complexidade, a utilização de CNNs 3D pode ser computacionalmente exigente e requer uma grande quantidade de dados para que seja possível a rede aprender.

2.6.1 *Learning Spatiotemporal*

Neste artigo [12], os autores propuseram extrair características espaciais e temporais, através de uma *deep 3D ConvNet* com um conjunto de treino supervisionado em larga escala e novas

arquiteturas para conseguir obter o melhor desempenho em diferentes tipos de tarefas de análise de vídeo. As *3D ConvNets* conseguem extrair informação associada a objetos, cenas e ações num vídeo, o que as torna úteis em várias tarefas, dado que não se torna necessário ajustar o modelo para cada delas.
Os investigadores, em primeiro lugar, tentaram identificar uma boa arquitetura para as *3D ConvNets*. Uma vez que treinar redes de Aprendizagem Profunda em conjuntos de dados de vídeo de larga-escala é muito demorado, começaram por iniciar as suas experiências com um conjunto de dados de tamanho médio, o UCF101, para procurar a melhor arquitetura. De acordo com as investigações efetuadas com as *ConvNets* 2D, arquiteturas profundas com filtros de convolução 3×3 produzem melhores resultados; pelo que os investigadores mantiveram, o campo recetivo espacial, em 3×3, alterando apenas a profundidade temporal dos filtros de convolução 3D.
Para solucionar o problema proposto, os investigadores configuraram as redes treinadas para receber clipes de vídeo como entrada e prever as respetivas classes pertencentes a 101 ações diferentes. As *frames* do vídeo foram redimensionadas para 128×171, o que é aproximadamente metade da resolução das *frames* do conjunto de dados UCF101. Os vídeos foram divididos em clipes de vídeo de 16 *frames*. As redes têm 5 camadas de convolução e 5 camadas de *pooling* (cada camada de convolução é seguida de uma camada de *pooling*), 2 camadas *Fully-connected* e 1 camada de *softmax loss* para prever as classes de ação. Todas as camadas de convolução são preenchidas (ao nível espacial e ao nível temporal) e *stride* 1, portanto não há alteração de tamanho na entrada e na saídas dessas camadas. Todas as camadas de *pooling* são *max pooling* com tamanho de filtro $2 \times 2 \times 2$ (excepto na primeira camada) com *stride* 1, o que significa que o sinal de saída é reduzido por um fator 8 quando comparado com o sinal de entrada. A primeira camada de *pooling* tem tamanho de filtro $1 \times 2 \times 2$ com o intuito de não misturar o sinal temporal demasiado cedo e para satisfazer o comprimento do clipe de 16 *frames*. As duas camadas *Fully-connected* possuem 2048 outputs. As redes foram treinadas do zero usando *mini-batches* de 30 clipes, com uma taxa de aprendizagem inicial de 0.003. A taxa de aprendizagem foi dividida por 10 após cada 4 épocas. O treino foi de 16 épocas.
Após a configuração das redes, os investigadores variaram a sua arquitetura. Para isso, estes variaram a profundidade do filtro das camadas de convolução, mantendo todas as outras configurações comuns. Os autores experimentaram dois tipos de arquiteturas: profundidade temporal homogénea e profundidade temporal variável. Na profundidade temporal homogénea, todas as camadas de convolução têm a mesma profundidade temporal do filtro. Na profundidade temporal variável, a profundidade temporal do filtro muda através das camadas.

2.6.2 ShuffleNet V2

O ShuffleNet V2 [13] é um modelo que tem um desempenho muito competitivo e com exigências computacionais bastante reduzidas, sendo adequado para dispositivos com recursos computacionais limitados, como *smartphones* e dispositivos embutidos.
Este modelo baseia-se na operação de *shuffle*, possibilitando que as informações extraídas

de diferentes camadas sejam combinadas de forma eficiente, reduzindo assim a sua complexidade computacional. O modelo foi desenvolvido com blocos designados por *Shuffle Units*, cujo objetivo é superar as limitações associadas à estrutura das CNNs que são constituídas por camadas sequenciais, o que pode limitar a eficácia da aprendizagem. Estes blocos permitem que as características extraídas pelas camadas convolucionais sejam combinadas e partilhadas entre diferentes canais o que torna possível a captura de relações complexas entre as mesmas.
Os *Shuffle Units* dividem os canais de entrada em grupos mais pequenos e combinam as características entre si promovendo a obtenção de representações com maior diversidade. Este processo de combinação das características é realizado através de operações de convolução 1×1, concatenação e permutação.
Os investigadores realizaram diversas experiências que demonstraram que este modelo supera alguns métodos do estado da arte relativamente à complexidade computacional, mantendo um desempenho competitivo em tarefas como classificação de imagens. Desta forma, os autores mostram que este método oferece uma arquitetura eficiente para o desenvolvimento de modelos com desempenhos competitivos em dispositivos de baixo poder computacional.

2.7 Redes *Two-stages Two-backbone*

As redes Two-stages Two-backbone são uma abordagem utilizada para deteção de ações em vídeos. Estas redes são constituídas, como o nome indica, por duas fases: na primeira fase, o modelo é pré-treinado e ajustado ao conjunto de dados pretendido; na segunda fase, é selecionada uma *key-frame* a partir de um clipe de vídeo e esta será a entrada para o modelo de deteção resultante da primeira fase. Este modelo é responsável por prever as *bounding boxes* da ação na *frame* selecionada. Posteriormente, as *bounding boxes* previstas e o clipe de vídeo são utilizados como entrada de uma rede 3D, que será responsável por extrair características da região de interesse (RoI) para classificar a ação.
Ao contrário do que acontece com as redes End-to-end, onde as *bounding boxes* de ação e as respetivas classes são previstas diretamente dado um clipe de vídeo como entrada, as redes Two-stages Two-backbone separam a deteção da ação em duas tarefas distintas: localização e classificação. Esta abordagem apresenta desafios significativos devido à elevada complexidade por causa da fase de treino sequencial e à arquitetura constituída por dois modelos separados.
Embora esta abordagem apresente desvantagens em termos de eficiência, a utilização destas redes pode melhorar a precisão da deteção da ação em vídeos.

2.7.1 *SlowFast*

Este artigo [4] introduz o modelo *SlowFast* 2.8. Este modelo é constituído por dois métodos: o *Slow* e o *Fast*.
Relativamente ao *Slow*, este método serve para capturar a semântica espacial, sendo que funciona a uma taxa baixa de *frames*. Este método é utilizado porque o domínio espacial da

ação não varia muito, tornando-se este um excelente método a capturar o espaço da ação, assim como as suas características (tais como: as cores, as texturas, a iluminação, etc...).
Quanto ao método *Fast* este método serve para capturar o movimento associado à ação. Este método é importante ser utilizado porque consegue capturar rapidamente a variação do movimento presente (por exemplo: a ação de uma pessoa pode mudar rapidamente de correr para saltar); este método opera a uma taxa elevada de *frames* e a uma resolução temporal elevada.
O *SlowFast* é parcialmente inspirado nos estudos biológicos sobre as células ganglionares da retina do sistema visual primata. Estes estudos mostraram que estas células são constituídas por *P-cells* em $\sim 80\%$ da sua constituição e por *M-cells* em $\sim 15 - 20\%$. As *M-cells* são responsáveis pelas rápidas mudanças temporais, no entanto não são sensíveis relativamente ao espaço e à cor. As *P-cells* fornecem bastante detalhe espacial e respondem lentamente aos estímulos. Como os investigadores notaram, o modelo por eles criado funciona de forma similar: é constituído por dois métodos, um é responsável por capturar as rápidas mudanças do movimento e o outro por capturar as lentas, tal como podemos comparar com as *M-cells* e as *P-cells*; o método *Fast* é leve, tal como o pequeno rácio de *M-cells* no sistema visual.
As redes *SlowFast* têm uma arquitetura *single stream* que opera em duas taxas de *frames* diferentes. De forma geral, a arquitetura é constituída por um método *Fast* e um método *Slow* que são fundidos por conexões laterais e, por fim, formam a rede *SlowFast*.
O método *Slow* pode ser qualquer tipo de modelo convolucional que trabalhe com clipes de vídeo. A ideia principal é que o *stride* seja elevado nas *frames* de entrada. Um valor que os investigadores estudaram foi o 16.
Assim como o método *Slow*, o método *Fast* é uma rede convolucional com as propriedades seguintes:

Rácio elevado de *frames*. O método *Fast* trabalha com um *stride* baixo.

Características de resolução temporal elevadas. Não são usadas camadas de *down-sampling* temporais, assim como camadas de *pooling* temporal ou de convolução *time--strided*.

Baixa capacidade de representação do espaço. Esta característica pode ser interpretada como a fraca capacidade que este modelo tem de representar características espaciais. Os bons resultados sugerem que foi uma boa decisão por parte dos investigadores, reduzirem a capacidade de modelação espacial e aumentarem a capacidade de modelação temporal.

Os investigadores exploraram várias formas de diminuir a capacidade de modelação espacial do método *Fast* reduzindo a resolução espacial da entrada e removendo a cor. Como foi visto pelas experiências, estas versões têm bons resultados, sugerindo que um método *Fast* leve com menos capacidade espacial possa ser benéfico.

2.7.1.1 Conexões Laterais

Os dois métodos têm dimensões temporais diferentes, por isso as conexões laterais irão fazer uma transformação que os equipare. Os investigadores utilizaram uma conexão lateral

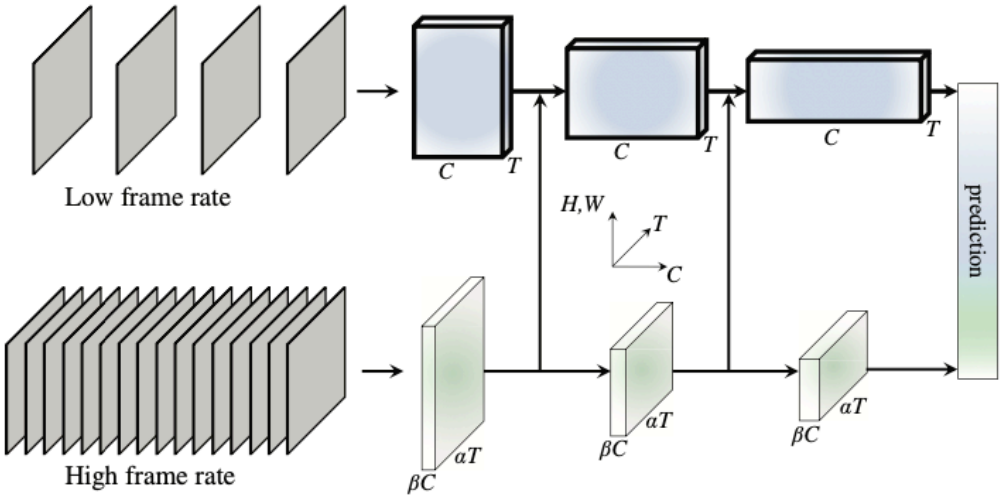

Figura 2.8: A rede *SlowFast*. Imagem retirada de [4].

unidirecional que funde as características do método *Fast* no método *Slow*.
Por fim, é executada uma camada de *pooling* em ambos os métodos. O resultado da execução desta camada irá gerar dois vetores que serão concatenados e servirão de entrada à camada de classificação *Fully-connected*.

2.7.2 *Context-Aware*

Neste artigo [5] é apresentada a rede Context-Aware 2.9. Esta rede de deteção de ação segue um paradigma popular de deteção de ação baseado na *frame*, contendo duas fases: a localização da ação e respetiva classificação. Para fazer a localização da ação é utilizado um detetor de pessoas a partir da *key frame* de um vídeo, obtendo assim um conjunto de *bounding boxes* 2D. Os investigadores repararam que a informação de contexto era muito importante para classificar com precisão as atividades humanas, então para além de usarem características locais de ação, utilizaram uma rede com parâmetros partilhados para extrair características de cena globais de todo o vídeo. Para capturarem informação de contexto de longo horizonte temporal utilizaram uma operação não local simplificada a partir de uma *long-term feature bank*. Por fim, as caracteríticas de ação, as características de cena globais e as caractertísticas de longo prazo são concatenadas para a classificação da ação.
Para o conjunto de dados AVA, as propostas de *boxes* são obtidas pela Faster RCNN com uma ResNeXt-101-FPN. Para o conjunto de dados JHMDB, são detetadas pela Faster RCNN com uma ResNet-50-FPN.
A rede de suporte utilizada é a rede I3D ResNet-50 com blocos não locais. Esta rede é pré-treinada com a ImageNet e expandida para uma rede 3D utilizando o método I3D. A seguir, o modelo é pré-treinado com o Kinectics-400 equipado com blocos não locais para a classificação de vídeo.
Para extrair as características de ação, as ações detetadas são cortadas e redimensionadas para que a rede apenas consiga visualizar o conteúdo das *bounding boxes*.
Usar apenas características de ação locais, faz com que o modelo tenha um desempenho inferior devido à falta de informação contextual. Assim, é utilizado um clipe de vídeo para

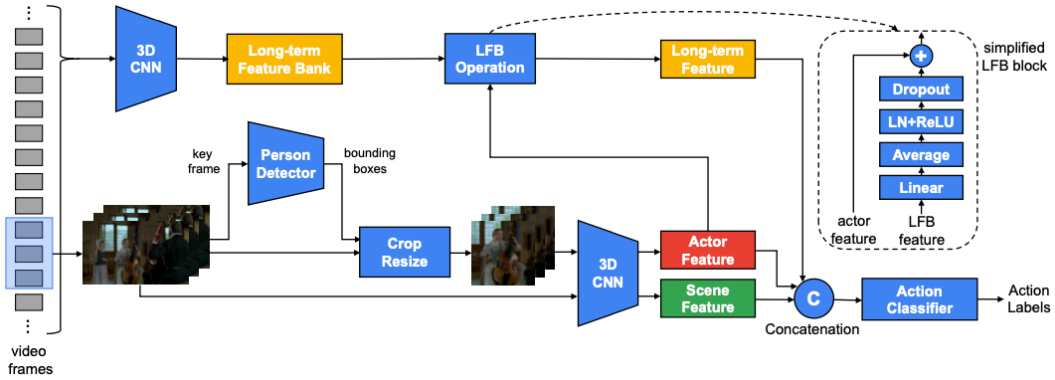

Figura 2.9: Arquitetura da Context-Aware RCNN. Imagem retirada de [5].

alimentar a rede convolucional 3D seguida de uma *global average pooling* para obter uma representação de um vetor de características de cena com dimensão de 2048. É usada uma rede com parâmetros partilhados para computar as características de ação e as características de cena.

Para extrair as características de longo prazo, os investigadores adotaram as arquitetura *Long-term Feature Bank* (LFB). A LFB executa com as características da ação centradas no clipe atual com um tamanho de janela de 61 segundos. A operação de LFB, que consiste em blocos LFB simplificados, é usada para extrair características de longo prazo tendo como entrada a LFB e as características de ação. Os investigadores pré-processaram estas duas entradas reduzindo a sua dimensão e fazendo *dropout*. São usados 3 blocos LFB e a saída é um vetor de características de longo prazo com dimensão 512.

Por fim, as características de ação, as características de cena e as características de longo prazo são concatenadas e enviadas para o classificador de ação. Para o conjunto de dados AVA é usada a *sigmoid loss* por classe, devido a ser um problema de multi-classe, e para o conjunto de dados JHMDB é usada a *softmax loss*.

2.8 Redes *End-to-end*

As redes de deteção de ação em vídeos End-to-end têm a capacidade de prever diretamente as *bounding boxes* de ação e as classes correspondentes a partir de um clipe de vídeo de entrada, sem a necessidade de extrair manualmente as características do vídeo. Isto permite que estes modelos processem clipes de vídeo em tempo real. Ao contrário dos métodos tradicionais de deteção de ação em vídeos, que exigem que os clipes de vídeo sejam processados por várias fases e algoritmos de processamento de imagem antes da deteção da ação, os modelos end--to-end têm a vantagem de processar um clipe de vídeo uma única vez.

Assim, os modelos end-to-end para deteção de ação em vídeos são capazes de aprender diretamente a partir de clipes de vídeos e prever as *bounding boxes* e classes correspondentes sem a necessidade de pré-processamento manual da entrada. Estas características tornam estes modelos adequados para aplicações em tempo real, como em sistemas de videovigilância, de deteção de movimento e de análise do comportamento humano.

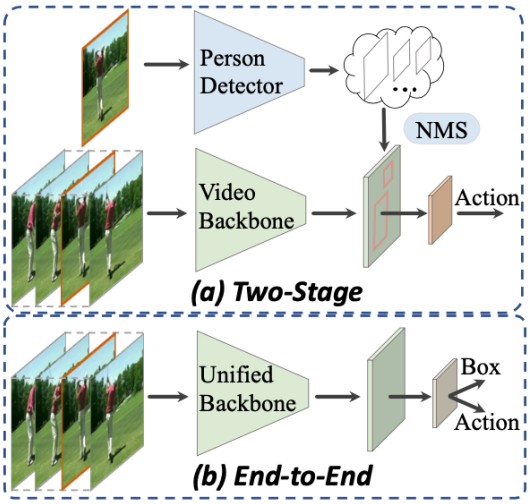

Figura 2.10: Em (a) está representada a arquitetura da maior parte das *frameworks* de deteção de ações em vídeos, normalmente utilizam duas redes separadas: uma rede 2D para localizar as ações em cada uma das *frames*, e uma rede 3D para classificar as ações de cada um dos clipes de vídeo. Em (b) está representada a rede criada pelos investigadores. Esta rede recebe apenas um vídeo de cada vez e produz diretamente a localização da ação e a respetiva classificação. Imagem retirada de [6].

2.8.1 *WOO*

Watch Only Once (WOO) 2.10 é uma rede de ponta-a-ponta que consegue detetar ações em vídeo [6]. A WOO consegue prever as coordenadas das *bounding boxes* e as probabilidades de cada uma das classes na classificação de vídeo.

Este método consiste em 3 componentes principais: uma rede de suporte unificada, uma incorporação da ação espaço-temporal e um mecanismo de fusão do conhecimento espaço--temporal.

Relativamente ao primeiro módulo, os investigadores desenvolveram a rede de suporte do modelo para executar tarefas específicas relativamente à localização da ação e à respetiva classificação. Este módulo é leve e é utilizado para isolar as características das *key frames* das características de todas as *frames* numa fase inicial. Ao longo da execução do modelo, as características das *key frames* vão interagindo cada vez mais com as *frames* vizinhas.

Relativamente ao segundo módulo, os investigadores observaram que esta rede de suporte unificada se comportava bem quanto à localização da ação, mas mantinha-se limitada quanto à sua respetiva classificação. Assim, propuseram uma incorporação da ação espaço-temporal e um mecanismo de interação entre eles, para tornar as características da classificação de ação mais discriminativas na perspetiva espacial e temporal.

Relativamente ao terceiro módulo, propuseram um mecanismo de fusão do conhecimento espaço-temporal para juntar o conhecimento espacial e temporal. As propriedades espaciais e temporais são combinadas através deste módulo para gerar características de ação para a sua classificação.

O modelo proposto consegue identificar diretamente as *bounding boxes* e as classes de ação dado um clipe de vídeo, sem qualquer tipo de pós-processamento (como por exemplo: *non--maximum supression (NMS)*).

Com o objetivo de extrair características da *key frame*, os investigadores adotaram a *Feature*

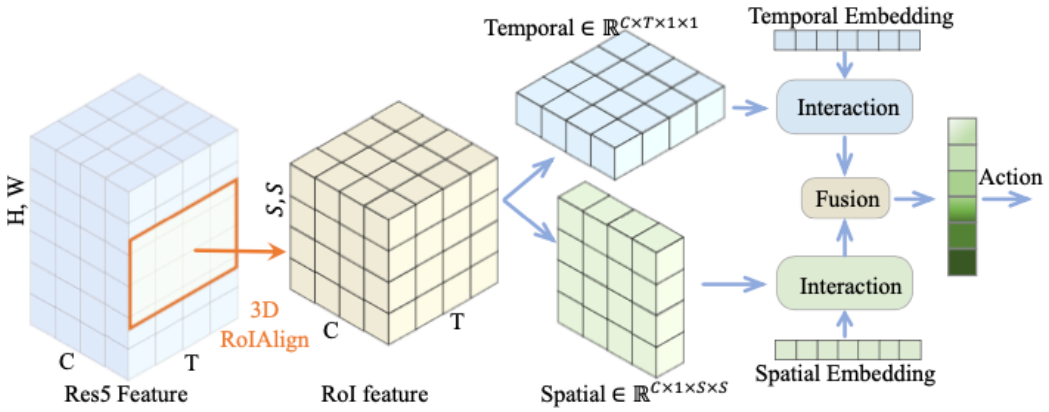

Figura 2.11: Arquitetura da classificação da ação. Dadas as características RoI de uma *box* específica de T *frames*, são geradas características espaciais e temporais. De seguida, é utilizada um módelo de interação para tornar as características mais discriminativas. Por fim, é gerada a previsão da classe da ação após o MLP receber as características espaciais e temporais fundidas como entrada. Imagem retirada de [6].

Pyramid Network (FPN). Assim, foi adotada uma representação de características hierárquica como fonte de características, o que traz muitas vantagens para a deteção de objetos. De seguida, as características da *key frame* usadas para a localização da ação são isoladas das características de todas as *frames* através da estrutura da FPN, desde uma fase inicial. Por fim, a utilização do módulo FPN reduz consideravelmente o número de parâmetros da rede e de FLOPs.

Assim que são recebidas as características hierárquicas geradas pela FPN, é possível prever as coordenadas das *bounding boxes* e o respetivo valor de confiança do modelo na *box* que contém a ação.

Quando são obtidas as *boxes* propostas pelo detetor de pessoas, é usado o RoIAlign para extrair as características espaciais e temporais de cada *box*. De seguida, estes dois tipos de características são fundidos para gerar a previsão final da classe de açãon, como está representado na figura 2.11.

Para obter as características espaciais, é realizada uma *global average pooling* ao longo da dimensão temporal (na saída da camada res5) para obter um mapa de características espaciais. A RoIAlign é aplicada a este mapa de características com N *boxes* propostas de ação, produzindo assim N características RoI espaciais.

Para obter as características temporais, extraem-se as características temporais de todas as *frames* da saída da camada res5. Como o maior foco é a extração da informação temporal, é aplicada uma *global average pooling* na dimensão espacial para extrair eficientemente as características RoI temporais.

Para obter características mais discriminativas (Embedding Interaction), foi criado um módulo de atenção para todas as características RoI.

2.9 Redes *One-stage Two-backbone*

As redes de deteção de ação One-stage Two-backbone são redes apresentam duas redes de suporte: uma para processamento 2D e outra para processamento 3D. Estas redes de su-

porte são treinadas simultaneamente. Embora esta abordagem tenha a vantagem de reduzir o tempo de treino da rede comparativamente às redes End-to-end, a sua complexidade ainda é elevada, pois há dois modelos separados a otimizar. O modelo 2D é responsável pela deteção de objetos em cada *frame* do vídeo, enquanto o modelo 3D é responsável por analisar a sequência temporal das *frames* para extrair informações sobre as ações em movimento. Apesar de sua complexidade, as redes One-stage Two-backbone têm apresentado resultados promissores na tarefa de deteção de ações em sequências de vídeo.

2.9.1 *You Only Watch Once: A Unified CNN Architecture for Real-Time Spatio-temporal Action Localization*

O YOWO [7] pode ser dividido em quatro partes, como é mostrado na figura 2.12: o branch 3D-CNN, o branch 2D-CNN, a CFAM e a regressão das *bounding boxes*.
O branch 3D-CNN é responsável por extrair as características espaciais e temporais, uma vez que obter representações do contexto e do movimento ao longo do tempo é crucial para compreender ações humanas. Desta forma, a informação é obtida através de convoluções no espaço e no tempo. A entrada desta rede é um clipe de vídeo composto por *frames* sucessivas ao longo da linha do tempo. A saída desta rede é um mapa de características no qual este é *squeezed* para resultar num mapa de características de dimensão igual ao mapa de características resultante do branch 2D-CNN.
O branch 2D-CNN é responsável por extrair as características 2D das *key frames* paralelamente à extração de características espaciais efetuada pelo 3D-CNN. A *key frame* selecionada para servir de entrada a esta rede é a *frame* mais recente do clipe de vídeo.
Após ambas as redes obterem a sua saída, estes mapas de características vão ser fundidos através da concatenação. Este mapa de características resultante irá conter a informação 3D e 2D e irá servir de entrada ao módulo CFAM.
O CFAM ou *Channel Fusion and Attention Mechanism* é baseado na matriz de Gram e tem um papel fundamental na fusão de características com origem em fontes diferentes, uma vez que a concatenação efetuada anteriormente negligencia por completo a relação entre a informação 3D e 2D. O mapa de características resultante deste módulo tem as características discriminadas e uma relação contextual das mesmas.
Relativamente à regressão das *bounding boxes*, o YOWO segue as mesmas diretrizes do YOLO.
Os conjuntos de dados utilizados pelos investigadores para avaliar o desempenho do YOWO foram: o UCF101-24, J-HMDB-21 e o AVA. Assim, os autores criaram uma nova arquitetura unificada que lhes permite modelar o contexto espaço-tempo de *frames* sucessivas para compreender ações humanas, enquanto se realiza a extração de informação espacial através de *key frames* para se conseguir localizar a ação.

2.9.2 *YOWO Plus: An Incremental Improvement*

O YOWO Plus [8] é o YOWO com algumas alterações que os autores fizeram para que este tivesse um desempenho superior. Os autores usaram exatamente a mesma rede 3D-Res-

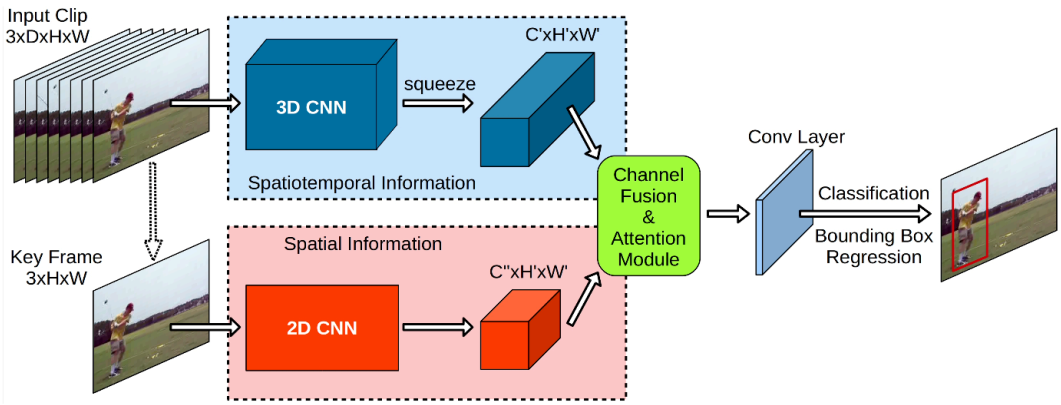

Figura 2.12: Arquitetura do YOWO. Imagem retirada de [7].

Next-101 e a YOLOv2 com pesos pré-treinados melhores da sua reimplementação. Também otimizaram a anotação das classes usada no YOWO original e substituíram a L1 *loss* pela GIoU *loss* para a regressão das *boxes*.

Relativamente à melhoria da rede de suporte - o YOWO tem duas redes de suporte para processar cada vídeo de entrada. A rede de suporte 2D serve para extrair as características espaciais das *key frames*, enquanto que a rede de suporte 3D serve para extrair as características espaço-temporais de cada um dos vídeos. Relativamente à rede de suporte 2D, dado que foi esta a modificada, os autores usaram pesos melhores do pré-treino efetuado com o COCO, conseguindo atingir resultados significativamente melhores do que o YOLOv2 oficial.

Relativamente à melhoria da anotação das classes - o YOWO original utiliza a YOLOv2 para processar as *key frames*, assim como a respetiva anotação das classes. O YOWO original calcula a IoU das 5 *bounding boxes* obtidas e apenas usa a *bounding box* com o valor mais elevado de IoU para calcular a *loss* de confiança, a *loss* de classificação e a *loss* de regressão das *boxes*. Ao contrário do YOWO original, nesta rede é calculada a IoU de 5 *anchor boxes* e não de *boxes* obtidas, de seguida as *anchor boxes* com IoU superior a 0.5 são atribuídas ao *ground truth*.

Relativamente à melhoria da função de *loss* - os autores usaram a mesma *loss* de confiança e a mesma *loss* de classificação. Relativamente à *loss* de regressão de *boxes*, os autores usaram a GIoU em vez da L1 *loss*.

O código-fonte deste projeto foi disponibilizado pelos autores em [14]

Para criar um detetor de ação mais leve, o YOWO Nano, os autores substituíram a 3D-ResNext-101 por uma ShuffleNet-v2, sendo que todas as restantes configurações são iguais às do YOWO Plus.

2.10 *Vision Transformers* (ViT)

Os *Vision Transformers* (ViT) [15] são modelos baseados na arquitetura de *Transformers* que, originalmente, foram criados para desempenhar tarefas de processamento de linguagem natural. Ao contrário dos modelos baseados em CNNs, que utilizam camadas de convolução para extrair características, os ViT dividem a imagem em *patches* e, em seguida, transfor-

mam esses *patches* numa sequência linear de *tokens* que é processada através de camadas de atenção.
Esta abordagem permite que a rede capture relações complexas entre os diversos píxeis que constituem a imagem, permitindo capturar interações locais e globais de cada píxel em relação aos píxeis vizinhos da imagem. Assim, os ViT são capazes de aprender representações mais sofisticadas e ricas das imagens, melhorando o desempenho de tarefas de classificação, deteção e segmentação.
Os ViT têm demonstrado grande escalabilidade e capacidade de lidar com conjuntos de dados de imagens extremamente grandes, como o ImageNet. De facto, em comparação com os métodos baseados em CNNs, os ViT apresentam desempenhos comparáveis e em alguns casos até superiores em tarefas de classificação de imagem. Além disso, eles também têm sido aplicados em outras áreas, como processamento de vídeo e reconhecimento de objetos em 3D, mostrando grande potencial como uma abordagem alternativa e complementar às CNNs.

2.10.1 Simple ViT

O Simple ViT [16] é uma versão simplificada do modelo ViT, projetado para fornecer uma alternativa mais leve e eficiente para realizar tarefas de visão computacional. Em vez de utilizar múltiplas camadas de *Transformers* e de atenção cruzada, o Simple ViT é constituído por apenas uma camada de *Transformers* e utiliza um MLP para a classificação. Este modelo simplificado é significativamente mais leve e, portanto, mais fácil de integrar e treinar em dispositivos com recursos computacionais limitados. No entanto, devido à sua simplicidade, o Simple ViT não é capaz de aprender representações tão ricas e complexas comparativamente com o ViT, e portanto, tem um desempenho inferior.
O código-fonte pode ser encontrado em [17]

2.10.2 3D ViT

O 3D ViT é uma extensão do bem-sucedido modelo ViT, e foi desenvolvido com o intuito de processar dados tridimensionais, nomeadamente vídeos e imagens médicas. Enquanto o ViT utiliza *patches* bidimensionais para representar imagens, o 3D ViT utiliza *patches* tridimensionais, possibilitando a captura de relações espaciais ao longo das três dimensões de um vídeo ou imagem médica. Além disso, o 3D ViT inclui uma camada de convolução 3D na primeira etapa da rede, que tem como função extrair características das entradas tridimensionais.
A utilização do 3D ViT apresenta uma abordagem promissora ao aplicar *Transformers* em tarefas de processamento de vídeo, permitindo assim a captura de informações de contexto mais complexas. O potencial deste modelo extende-se a diversas tarefas da visão computacional, tais como a classificação de ações em vídeos, o reconhecimento de objetos em imagens médicas e segmentação de objetos em imagens tridimensionais, entre outras. Assim, a incorporação da camada de convolução 3D e dos *patches* tridimensionais no modelo ViT podem torná-lo uma solução mais eficaz para lidar com entradas de dados tridimensionais do que

as redes baseadas em CNNs tradicionais.
O código-fonte pode ser encontrado em [17]

Capítulo 3

Método Proposto

Neste capítulo é abordada a metodologia durante o processo de investigação e desenvolvimento deste projeto, assim como dos métodos desenvolvidos. Aqui, são explicadas as razões que justificam a escolha do modelo a otimizar, assim como as técnicas/métodos que vão ser utilizados e desenvolvidos para posterior análise e comparação.

3.1 Visão Geral

O principal objetivo deste projeto consistiu em selecionar e otimizar uma rede de Aprendizagem Profunda que apresentasse um bom desempenho comparativamente a outras soluções disponíveis no estado da arte. Para atingir este objetivo, adotou-se uma abordagem sistemática baseada em no estudo do estado da arte, na experimentação e na análise de resultados.
O processo de seleção da rede de Aprendizagem Profunda foi planeado, tendo em consideração vários critérios, como o desempenho em tarefas semelhantes e a eficiência computacional. Após uma análise cuidadosa, foi selecionada uma rede que demonstrou ser particularmente adequada para a tarefa em questão.
Uma vez escolhida a rede, o próximo passo foi tentar melhorar o seu desempenho através da aplicação de métodos de otimização clássicos e de ideias inovadoras, resultantes do estudo do estado da arte. Para tal, foram implementados e testados vários métodos e técnicas de otimização, com o objetivo de identificar as configurações mais eficazes para a rede selecionada.

3.2 Modelo Selecionado

Como os principais objetivos deste projeto são a otimização de uma rede através da diminuição do tempo de inferência e do tamanho que o modelo ocupa em memória com a melhoria dos resultados obtidos pela mesma, foi necessário procurar uma rede leve, rápida e com bons resultados.
Ao longo do estudo de várias redes durante o desenvolvimento do estado da arte, houveram duas redes que se destacaram, tendo em conta os objetivos e as limitações do projeto: a YOWO Nano e a YOWO Plus [8]. Na verdade, estas redes são resultado do desenvolvimento e da correspondente melhoria da YOWO [2].
Através da melhoria da rede de suporte 2D, da atribuição de classes e da função de *loss*, conseguiu-se melhorar o desempenho da YOWO e, consequentemente, criou a YOWO Plus, como se pode ver em 2.9.2.
A única diferença que existe entre a YOWO Plus e a YOWO Nano é a sua rede de suporte 3D. A YOWO Plus utiliza a mesma rede de suporte 3D da YOWO que é a 3D-ResNext-101;

Tabela 3.1: Comparação entre os modelos com o *dataset* UCF101-24. Dados da tabela retirados de [8].

Modelo	GFLOPS(B)	Params(M)	frame-mAP	video-mAP
YOWO	43.7	121.4	80.4	48.8
YOWO Plus	43.7	121.4	84.9	50.5
YOWO Nano	6.0	72.7	81.0	49.7

Tabela 3.2: Comparação entre os modelos com o *dataset* AVA. Dados da tabela retirados de [8].

Modelo	FPS	mAP
YOWO	31	17.9
YOWO Plus	33	20.6
YOWO Nano	91	18.5

e a YOWO Nano utiliza a ShuffleNet-V2. Com apenas a alteração da rede de suporte 3D, foi possível diminuir o número de operações e o número de parâmetros da YOWO Nano e, consequentemente, diminuir muito significativamente o tempo de inferência. Para além disso com esta mudança desta rede de suporte, os resultados caíram ligeiramente, como se poderá ver nas seguintes tabelas 3.1 e 3.2.

3.3 Modelos

Foram desenvolvidos diversos modelos baseados no YOWO Nano, com o intuito de otimizar o desempenho do modelo base e alcançar os melhores resultados possíveis. Esses modelos foram submetidos a testes empíricos para avaliar a sua eficácia relativamente às métricas estabelecidas, de forma a identificar quais as configurações que apresentam os melhores resultados. Este processo de experimentação e ajuste é uma prática comum em muitas áreas de investigação em ciência da computação, com o objetivo de melhorar os modelos existentes e desenvolver soluções mais eficientes e precisas.

3.3.1 YOWO Nano

O YOWO Nano 3.1 é uma versão melhorada do YOWO, que usa a mesma rede neuronal e contém algumas mudanças para que este tenha um desempenho superior. A melhoria foi feita nas redes de suporte e na forma como as classes são anotadas. Além disso, a função de loss para a regressão das *boxes* também foi alterada. As mudanças incluem usar pesos pré-treinados melhores para as redes de suporte e substituir a *loss* utilizada para a regressão das *boxes*, em vez da L1 *loss*, utilizaram a GIoU *loss*. No que diz respeito à anotação das classes, em vez de calcular a IoU de 5 *boxes* obtidas, é calculada a IoU de 5 *boxes* de ancoragem as quais são atribuídas ao *groundtruth*. Isto resultou em resultados significativamente melhores em comparação com o YOLOv2 original. Para além disso, decidiram substituir a rede de suporte 3D pela Shufflenetv2, o que permitiu obter uma rede muito mais rápida e leve.

3.3.2 YOWO Nano com camadas de convolução separáveis

O modelo YOWO Nano com camadas de convolução separáveis 3.2, como o nome indica, resultou da incorporação de camadas de convolução separáveis no modelo YOWO Nano ori-

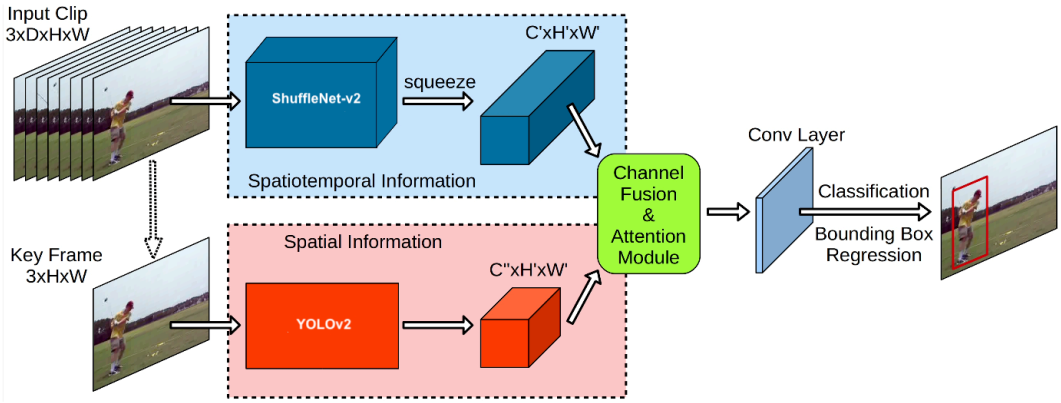

Figura 3.1: Arquitetura da YOWO Nano. Esta imagem foi modificada. Retirada de [7].

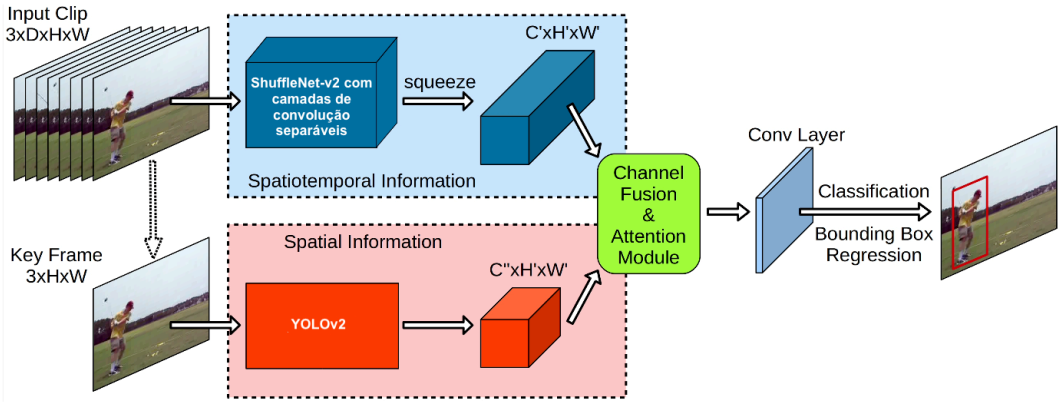

Figura 3.2: Arquitetura da YOWO Nano com camadas de convolução separáveis. Esta imagem foi modificada. Retirada de [7].

ginal. Esta rede foi desenvolvida com esta ideia em mente devido à investigação aprofundada das diversas técnicas clássicas de otimização. Durante esse estudo, observou-se que a utilização de camadas de convolução separáveis proporcionava uma redução significativa no número de operações realizadas pelo modelo. Portanto, considerando que os autores do YOWO original obtiveram sucesso ao pré-treinar os pesos da rede de convolução 2D, o YOLOv2, decidiu-se aplicar as camadas de convolução separáveis exclusivamente na rede de convolução 3D, a ShuffleNetv2. Esta abordagem visou aproveitar ao máximo os avanços alcançados pelos autores, que contribuíram significativamente para a melhoria do desempenho do YOWO. Ao adotar as camadas de convolução separáveis na rede de convolução 3D, o objetivo era diminuir o número de operações do YOWO Nano, o que consequentemente resultaria na diminuição do tempo de inferência, tentando manter os bons resultados do modelo base o mais possível. Desta forma, este modelo recém-desenvolvido poderia aproveitar os desenvolvimentos realizados pelos autores do YOWO Nano de forma a alcançar melhorias ao nível da eficiência computacional.

3.3.3 YOWO Nano com YOLOv8n

O modelo YOWO Nano com YOLOv8n 3.3 resulta da substituição da rede de suporte 2D, conhecida como YOLOv2, pela nova versão melhorada denominada YOLOv8n. Esta subs-

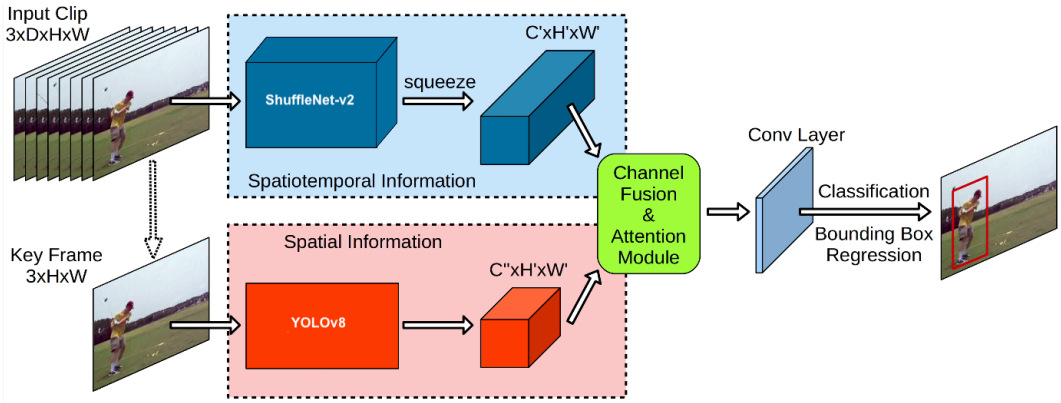

Figura 3.3: Arquitetura da YOWO Nano com YOLOv8n. Esta imagem foi modificada. Retirada de [7].

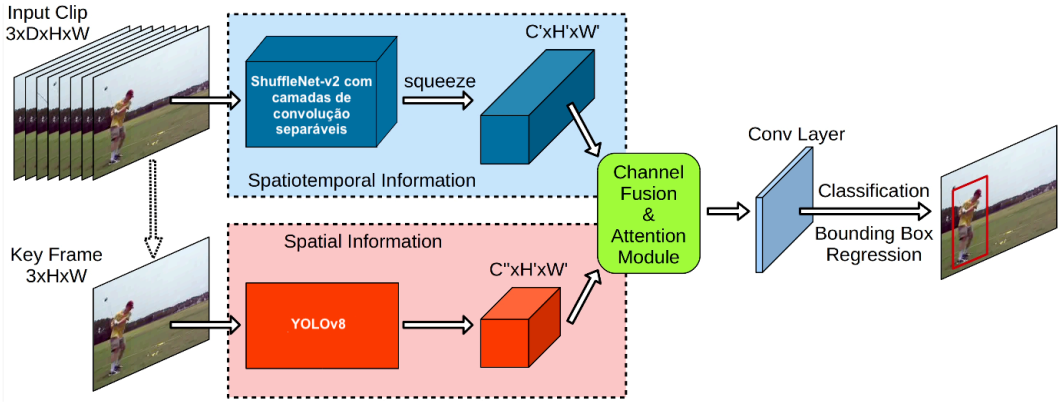

Figura 3.4: Arquitetura da YOWO Nano com YOLOv8 e camadas de convolução separáveis. Esta imagem foi modificada. Retirada de [7].

tituição tem como objetivo melhorar o desempenho do modelo, permitindo a deteção e a classificação de ações em tempo real.

3.3.4 YOWO Nano com YOLOv8n e camadas de convolução separáveis

O modelo YOWO Nano com YOLOv8n é um modelo que surge a partir da substituição da rede de suporte 2D, conhecida como YOLOv2, pela sua versão mais atualizada denominada YOLOv8n. Assim, o modelo YOWO Nano com YOLOv8n e camadas de convolução separáveis 3.4 foi criado a partir deste YOWO Nano com YOLOv8 aplicando, adicionalmente, camadas de convolução separáveis na rede de suporte 3D, designada por ShuffleNetv2. Estas alterações realizadas no modelo base visam a melhoria do desempenho e da precisão do modelo, proporcionando um sistema capaz para realizar tarefas de deteção e classificação de ação em vídeo.

A aplicação de camadas de convolução separáveis na rede de suporte 3D, ShuffleNetv2, foi realizada com o objetivo de reduzir a complexidade computacional e o número de parâmetros da rede, mantendo a capacidade da mesma aprender.

Desta forma, a ideia subjacente à criação deste modelo foi o de potenciar o desempenho e a eficiência da rede através da combinação da utilização da versão mais atualizada da YOLO com as camadas de convolução separáveis.

3.3.5 YOWO Nano com ViT 3D e Simple ViT

O modelo YOWO Nano com Vit-3D e Simple ViT resulta da substituição de ambas as redes de suporte. A substituição ocorre com o uso do Simple ViT em substituição à rede de suporte 2D, designada por YOLOv2, e o Vit-3D em substituição à rede de suporte 3D, denominada por ShuffleNetv2. Estas substituições têm como objetivo perceber de que forma é alterado o desempenho da rede, oferecendo uma abordagem totalmente baseada em *transformers*, eliminando assim o uso de CNNs na sua composição. Isto representa uma mudança significativa na abordagem tradicional no desenvolvimento de arquiteturas de redes de deteção e classificação de ação em vídeo.

Capítulo 4

Resultados e Discussão

Neste capítulo são explicadas as experiências que foram realizadas e mostrados os resultados obtidos de cada uma destas. Estas experiências visam, essencialmente, à diminuição do tempo de inferência e à melhoria dos resultados obtidos. Caso não se consigam atingir estes dois objetivos simultaneamente, a ideia é explorar o melhor *trade-off* entre os resultados e a otimização do modelo.
O modelo será executado num computador cujas especificações estão detalhadas na tabela 4.1.

4.1 Resultados

Após a conclusão do desenvolvimento dos modelos, foi conduzida uma análise comparativa de desempenho entre os modelos desenvolvidos e o estado da arte atual 3.3.
Para avaliar os diferentes modelos, foram definidas diferentes configurações para os mesmos ao longo do processo de experimentação. Essas configurações estão detalhadas nas tabelas 4.2, 4.4, 4.6 e 4.8.
Ao conduzir a análise comparativa, as principais métricas consideradas serão a frame-mAP, a video-mAP e o tempo de inferência. Outras métricas que também fazem parte da análise, embora com menor grau de relevância, são o número de parâmetros, o número de operações em GFLOPs e o tempo de treino. O conjunto de dados utilizado para estas experiências foi o UCF101-24.

4.1.1 Experiência 1

Em relação à primeira experiência realizada, optou-se por manter as mesmas configurações definidas pelos autores do YOWO Nano, com uma exceção importante: a não utilização do pré-treino da rede de suporte 3D, ShuffleNet v2. Esta decisão foi tomada devido ao facto de que alguns modelos não possuírem a rede de suporte 3D pré-treinada, o que colocaria esses modelos em desvantagem em relação aos restantes.
Embora a não utilização do pré-treino da rede de suporte 3D possa ter colocado alguns modelos em desvantagem relativamente ao estado da arte, esta escolha foi feita com o objetivo

Tabela 4.1: Especificações do *hardware* do servidor.

Hardware	Modelo
Placa Gráfica	Tesla T4
Processador	AMD EPYC 7V12 64-Core
Memóriam RAM	27Gi

Tabela 4.2: Configuração dos hiperparâmetros sujeitos a análise e comparação.

Hiperparâmetros	Valor
Taxa de aprendizagem base	1×10^{-4}
Otimizador	AdamW
Épocas	5
Pré-treino da rede 3D	Falso
Pré-treino da rede 2D	Verdadeiro
Scheduler da taxa de aprendizagem	MultiStepLR

de garantir uma avaliação justa e comparável entre os diferentes modelos. Desta forma, todos os modelos foram submetidos às mesmas condições de treino e teste, sem a utilização das vantagens provenientes do pré-treino.

Após a finalização do treino dos modelos, é importante fazer a análise das curvas de treino e de validação, uma vez que fornecem *insights* sobre o desempenho dos modelos durante o processo de aprendizagem. A capacidade dos modelos convergirem rapidamente indica uma adaptação eficiente aos dados de treino, enquanto que uma convergência mais lenta pode sugerir a necessidade de ajustar a taxa de aprendizagem ou o número de épocas. Além disso, a capacidade de fazer inferências precisas no conjunto de validação é um indicativo da capacidade dos modelos generalizarem em dados nunca vistos anteriormente.

Com base na figura 4.1, é possível observar as curvas de aprendizagem dos diferentes modelos, destacando o YOWO Nano e o YOWO Nano sep conv como aqueles que demonstram uma convergência mais rápida. As respetivas curvas de *losses* de treino são representativas de um processo de aprendizagem efetivo, indicando que os modelos estão a ajustar os seus pesos aos dados e progredindo ao longo do treino. O modelo YOWO Nano YOLOv8n parece estar a aprender, porém, de uma maneira diferente quando comparado aos modelos mencionados anteriormente. Por fim, o YOWO Nano YOLOv8n sep conv exibe uma convergência aparentemente lenta, conforme indicado no gráfico, sugerindo a possibilidade de solucionar este problema através do aumento da taxa de aprendizagem ou do número de épocas, permitindo assim que a aprendizagem aconteça durante mais tempo.

No que diz respeito às curvas de validação, é possível observar que os modelos YOWO Nano e YOWO Nano sep conv são os que melhor conseguem fazer inferências precisas sobre os dados do conjunto de validação. Um ponto interessante a notar é que o modelo que apresenta uma aprendizagem mais lenta, o YOWO Nano YOLOv8n sep conv, não é necessariamente aquele com a menor convergência durante o treino.

Após concluir a fase de treino, teste e avaliação de cada um dos modelos, tornou-se evidente que o modelo baseado em transformers apresentava um desempenho extremamente lento em comparação com os restantes, incluindo o modelo do estado da arte. Devido ao tempo de inferência deste ser excessivamente longo e ao baixo desempenho obtido, este modelo foi excluído das experiências e análises futuras.

Relativamente ao tempo de inferência, os restantes modelos demonstraram manter uma consistência quase constante, não sendo possível identificar um modelo ótimo que supere o estado da arte em termos de velocidade de inferência.

Relativamente aos resultados obtidos pelos modelos, apenas o modelo YOWO Nano sep conv demonstrou ter um desempenho competitivo em comparação com o estado da arte. Este mo-

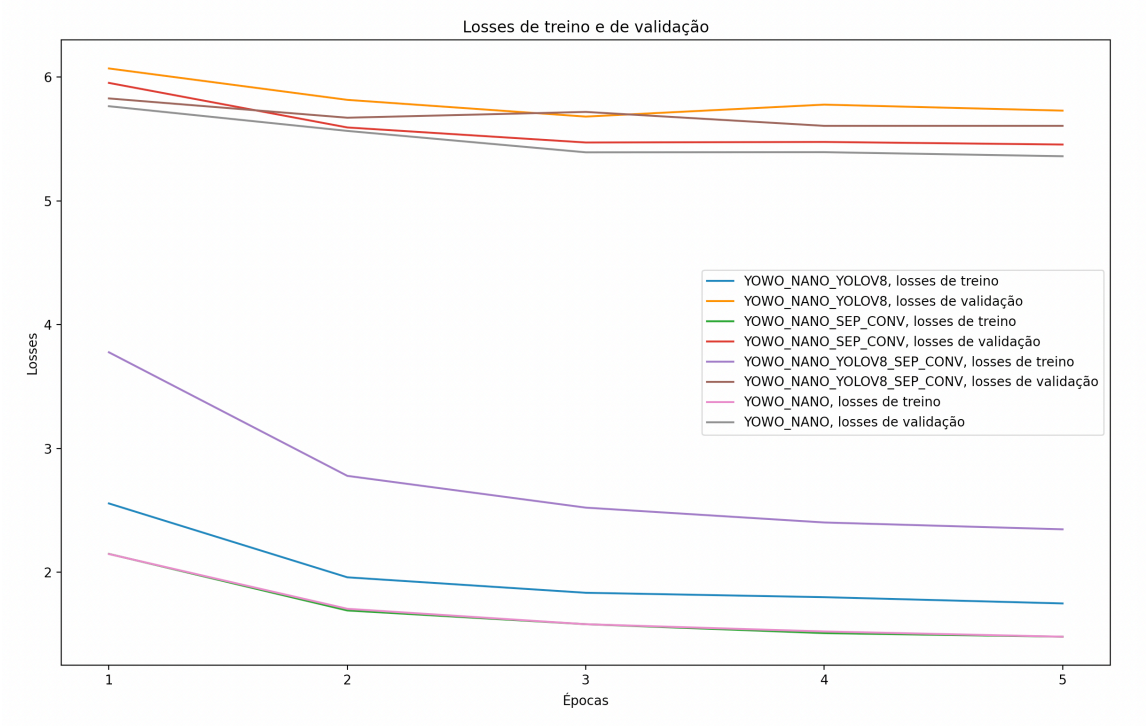

Figura 4.1: *Losses* de treino e de validação com as configurações definidas em 4.2.

delo obteve 47.15% em frame-mAP, em comparação com 49.01% do estado da arte, e 31.35% em video-mAP, em comparação com 31.74% do estado da arte. A métrica de video-mAP é particularmente importante, pois está relacionada com a deteção e a classificação do vídeo como um todo, em contraste com a frame-mAP, que avalia o desempenho em cada *frame* individualmente. Assim, a video-mAP assume maior relevância na análise comparativa.
O modelo YOWO Nano sep conv obteve resultados de video-mAP praticamente equivalentes ao estado da arte, ao mesmo tempo que apresentou uma complexidade de modelo inferior, com menos operações a serem realizadas.
Estes resultados indicam que, apesar do desafio de superar o estado da arte em termos de velocidade de inferência, o modelo YOWO Nano sep conv destaca-se ao alcançar um desempenho competitivo em relação à deteção e classificação em vídeos. A sua capacidade de atingir uma video-mAP próxima à do estado da arte, combinada com uma menor complexidade de modelo, sugere o seu potencial para aplicações em tempo real e dispositivos com recursos limitados.

4.1.2 Experiência 2

Na segunda experiência realizada, a taxa de aprendizagem foi aumentada (como pode ser visto na tabela 4.4). Esta alteração teve como objetivo investigar se um aumento da mesma poderia acelerar o processo de convergência dos modelos durante o treino. Uma vez que a taxa de aprendizagem determina a magnitude dos ajustes realizados aos pesos em cada iteração, a ideia seria permitir a redução do tempo necessário para obter resultados e assim otimizar o tempo de treino.
Após a fase de treino dos modelos, foi criado um gráfico com as curvas das *losses* de treino e

Tabela 4.3: Comparação do desempenho do modelo *baseline* e dos modelos desenvolvidos. Os hiperparâmetros sujeitos a análise e os respetivos valores podem ser visualizados em 4.2.

Modelo	Parâmetros (M)	GFLOPs (B)	Treino	Inferência (ms)	frame-mAP	video-mAP
YOWO Nano	72.78	6.07	18h 5m 2s	17.081	49.01%	31.74%
YOWO Nano sep conv	72.78	5.58	18h 29m 9s	17.082	47.15%	31.35%
YOWO Nano YOLOv8n	24.68	11.0	18h 1m 9s	17.140	32.21%	21.84%
YOWO Nano YOLOv8n sep conv	24.68	11.0	17h 53m 33s	17.418	21.15%	15.55%
YOWO Nano Simple ViT e ViT 3D	indef.	indef.	indef.	120.150	1.15%	0.76%

Tabela 4.4: Configuração dos hiperparâmetros sujeitos a análise e comparação.

Hiperparâmetros	Valor
Taxa de aprendizagem base	1×10^{-3}
Otimizador	AdamW
Épocas	5
Pré-treino da rede 3D	Falso
Pré-treino da rede 2D	Verdadeiro
Scheduler da taxa de aprendizagem	MultiStepLR

de validação de forma a perceber a influência que o aumento da taxa de aprendizagem teve.
Ao analisar o gráfico referente às curvas representativas das *losses* de treino e de validação (conforme mostrado na figura 4.2), podemos observar padrões semelhantes aos encontrados na experiência anterior. Novamente, os modelos YOWO Nano e YOWO Nano sep conv demonstram uma convergência mais rápida durante a fase de treino e, por outro lado, o modelo YOWO Nano YOLOv8n sep conv apresenta uma convergência mais lenta.
No que diz respeito às curvas que representam as *losses* de validação, o modelo YOWO Nano sep conv mostra um desempenho superior em comparação com os modelos restantes. É interessante notar que, desta vez, os modelos YOWO Nano e YOWO Nano YOLOv8n exibem curvas com *losses* superiores durante a fase de validação.
Um aspeto relevante a destacar é o impacto do aumento da taxa de aprendizagem de 1×10^{-4} para 1×10^{-3}. Observa-se que, no início das fases de treino e de validação, as *losses* são inferiores às mostradas na figura anterior 4.1. Este aumento da taxa de aprendizagem pode ter acelerado a convergência inicial dos modelos e levado a uma redução nas *losses*.
Mantendo o otimizador e aumentando a taxa de aprendizagem de 1×10^{-4} para 1×10^{-3}, os resultados diferiram bastante da análise realizada anteriormente.
Os modelos que apresentaram um desempenho competitivo anteriormente sofreram uma diminuição significativa relativamente ao seu desempenho. No entanto, um aspeto interessante a ser observado é que o modelo YOWO Nano sep conv obteve uma frame-mAP superior à do estado da arte, considerando as configurações especificadas na tabela 4.4.
É importante notar que o modelo YOWO Nano YOLOv8n não teve uma frame-mAP muito inferior à do estado da arte. Além disso, em comparação com a análise anterior 4.3, observa-se que o modelo YOWO Nano YOLOv8n sep conv manteve a frame-mAP, enquanto que a video-mAP foi superior.

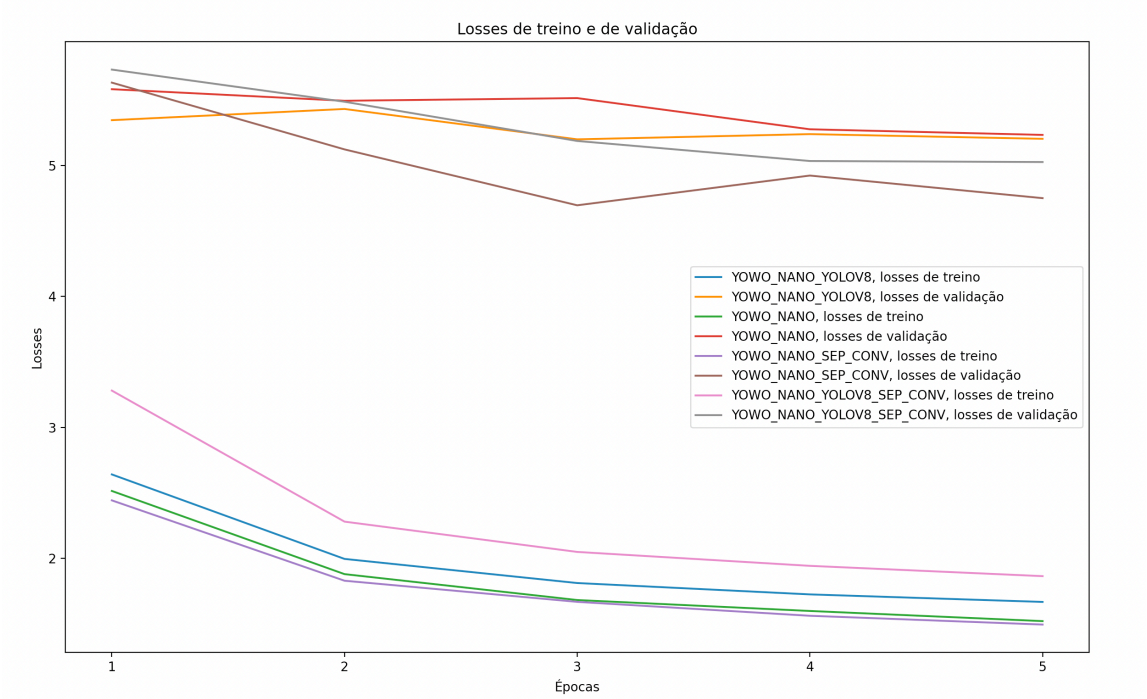

Figura 4.2: *Losses* de treino e de validação com as configurações definidas em 4.4.

Tabela 4.5: Comparação do desempenho do modelo *baseline* e dos modelos desenvolvidos. Os hiperparâmetros sujeitos a análise e os respetivos valores podem ser visualizados em 4.4.

Modelo	Parâmetros (M)	GFLOPs (B)	Treino	Inferência (ms)	frame-mAP	video-mAP
YOWO Nano	72.78	6.07	18h 5m 2s	17.081	36.31%	26.35%
YOWO Nano sep conv	72.78	5.58	18h 29m 9s	17.082	38.74%	24.86%
YOWO Nano YOLOv8n	24.68	11.0	18h 1m 9s	17.140	32.82%	18.23%
YOWO Nano YOLOv8n sep conv	24.68	11.0	17h 53m 33s	17.418	21.30%	17.37%

Estas descobertas revelam que, ao aumentar a taxa de aprendizagem, ocorrem mudanças significativas nos resultados dos modelos avaliados. Embora alguns modelos tenham sofrido uma queda no desempenho em relação ao estado da arte, o modelo YOWO Nano sep conv destacou-se ao alcançar uma frame-mAP superior, sendo que, por outro lado, alcançou um resultado inferior ao obtido com a taxa de aprendizagem da experiência anterior. Esta é uma indicação importante de que alterar a configuração e os hiperparâmetros dos modelos pode ter um enorme impacto nos resultados obtidos.

4.1.3 Experiência 3

Para a realização da experiência 3, foi decidido mudar o otimizador para tentar perceber se a escolha de um diferente teria muita influência nos resultados obtidos. Assim sendo, foi decidido mudar o AdamW para o SGD, como se pode ver na tabela 4.6.
O SGD é um dos otimizadores mais utilizados em aprendizagem automática. Este otimizador atualiza os pesos da rede utilizando uma taxa de aprendizagem fixa e amostras aleatórias

Tabela 4.6: Configuração dos hiperparâmetros sujeitos a análise e comparação.

Hiperparâmetros	Valor
Taxa de aprendizagem base	1×10^{-4}
Otimizador	SGD
Épocas	5
Pré-treino da rede 3D	Falso
Pré-treino da rede 2D	Verdadeiro
Scheduler da taxa de aprendizagem	MultiStepLR

do conjunto de treino. Esta abordagem estocástica pode introduzir uma certa quantidade de ruído no processo de otimização, mas também pode permitir que o modelo escape de mínimos locais e alcance uma solução melhor. O SGD foi escolhido devido a ser um otimizador menos complexo do que o AdamW, pelo que tem a vantagem de poder ser implementado em dispositivos de baixo poder computacional.

Ao examinar o gráfico referente às curvas das *losses* de treino e de validação (conforme apresentado no gráfico 4.3), torna-se evidente o impacto significativo da alteração das configurações dos modelos. Ao substituir o otimizador AdamW pelo SGD, juntamente com uma taxa de aprendizagem de 1×10^{-4}, conforme especificado na tabela 4.6, observa-se que tanto as *losses* de treino como as *losses* de validação são superiores em comparação com os gráficos apresentados nas experiências anteriores (gráficos 4.1 e 4.2), nos quais o otimizador utilizado era o AdamW. Esta observação indica que todos os modelos estão a aprender mais lentamente e apresentam um desempenho significativamente inferior na fase de inferência com o conjunto de validação.

Apesar deste cenário, é possível identificar um padrão: os modelos YOWO Nano e YOWO Nano sep conv são aqueles que convergem mais rapidamente, enquanto o modelo YOWO Nano YOLOv8n sep conv é o que apresenta a convergência mais lenta. Esta consistência no comportamento dos modelos ao longo das diferentes configurações sugere a existência de características intrínsecas às arquiteturas que influenciam a sua capacidade de aprender e se adaptar aos dados.

Ao realizar a modificação do otimizador para o SGD e reduzindo a taxa de aprendizagem para 1×10^{-4}, observamos que, no geral, o desempenho dos modelos é inferior em comparação com as configurações utilizadas nas experiências anteriores (conforme apresentadas nas tabelas 4.1 e 4.2).

Os modelos YOWO Nano e YOWO Nano sep conv apresentam desempenhos praticamente idênticos tanto em relação à frame-mAP quanto à video-mAP. Isto indica que, independentemente do otimizador e da taxa de aprendizagem, estes modelos conseguem manter um desempenho consistente e competitivo. Essa consistência pode ser atribuída às arquiteturas desses modelos, que parecem ser menos sensíveis às alterações das configurações.

Por outro lado, o modelo YOWO Nano YOLOv8n sep conv apresenta uma queda significativa do seu desempenho. Esta queda é tão drástica que torna impraticável a implementação deste modelo em qualquer sistema de deteção e classificação de ações em vídeos. Esta observação evidencia a sensibilidade deste modelo às configurações do otimizador e da taxa de aprendizagem, o que torna importante o planeamento das suas configurações a fim de obter resultados satisfatórios.

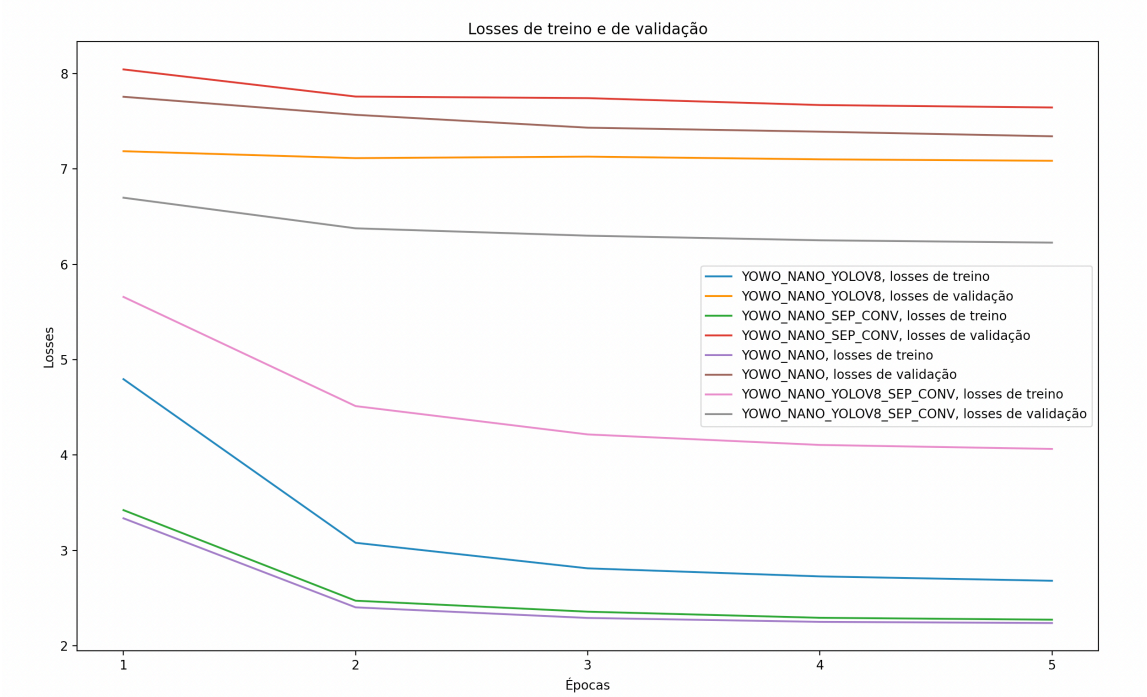

Figura 4.3: Gráfico das *losses* de treino e de validação com as configurações definidas em 4.6.

Tabela 4.7: Comparação do desempenho do modelo *baseline* e dos modelos desenvolvidos. Os hiperparâmetros sujeitos a análise e os respetivos valores podem ser visualizados em 4.6.

Modelo	Parâmetros (M)	GFLOPs (B)	Treino	Inferência (ms)	frame-mAP	video-mAP
YOWO Nano	72.78	6.07	18h 23m 50s	17.081	30.39%	25.36%
YOWO Nano sep conv	72.78	5.58	21h 23m 29s	17.082	29.37%	22.26%
YOWO Nano YOLOv8n	24.68	11.0	17h 59m 44s	17.140	25.58%	20.12%
YOWO Nano YOLOv8n sep conv	24.68	11.0	22h 46m 44s	17.418	1.96%	1.47%

4.1.4 Experiência 4

Na última experiência, a principal modificação foi feita na taxa de aprendizagem, como indicado na tabela 4.8. Esta decisão foi tomada com base nos resultados da experiência anterior, na qual foi observado que, ao utilizar o SGD com uma taxa de aprendizagem de 1×10^{-4}, as *losses* de treino e de validação foram consideravelmente mais altas comparativamente com as duas primeiras experiências. Para superar este desafio e obter uma convergência mais rápida das curvas, optou-se por aumentar a taxa de aprendizagem.

Com este aumento da taxa de aprendizagem, é esperado que as curvas das *losses* de treino e de validação mostrem uma convergência mais rápida, levando a um desempenho geral melhor dos modelos.

Ao analisar o gráfico 4.4, que corresponde às *losses* obtidas no treino e na validação, observamos uma semelhança notável com o gráfico 4.1, da primeira experiência. Esta semelhança indica que o uso do otimizador SGD resulta numa convergência mais lenta dos modelos comparativamente com o otimizador AdamW. No entanto, é importante destacar que ao aumen-

Tabela 4.8: Configuração dos hiperparâmetros sujeitos a análise e comparação.

Hiperparâmetros	Valor
Taxa de aprendizagem base	1×10^{-3}
Otimizador	SGD
Épocas	5
Pré-treino da rede 3D	Falso
Pré-treino da rede 2D	Verdadeiro
Scheduler da taxa de aprendizagem	MultiStepLR

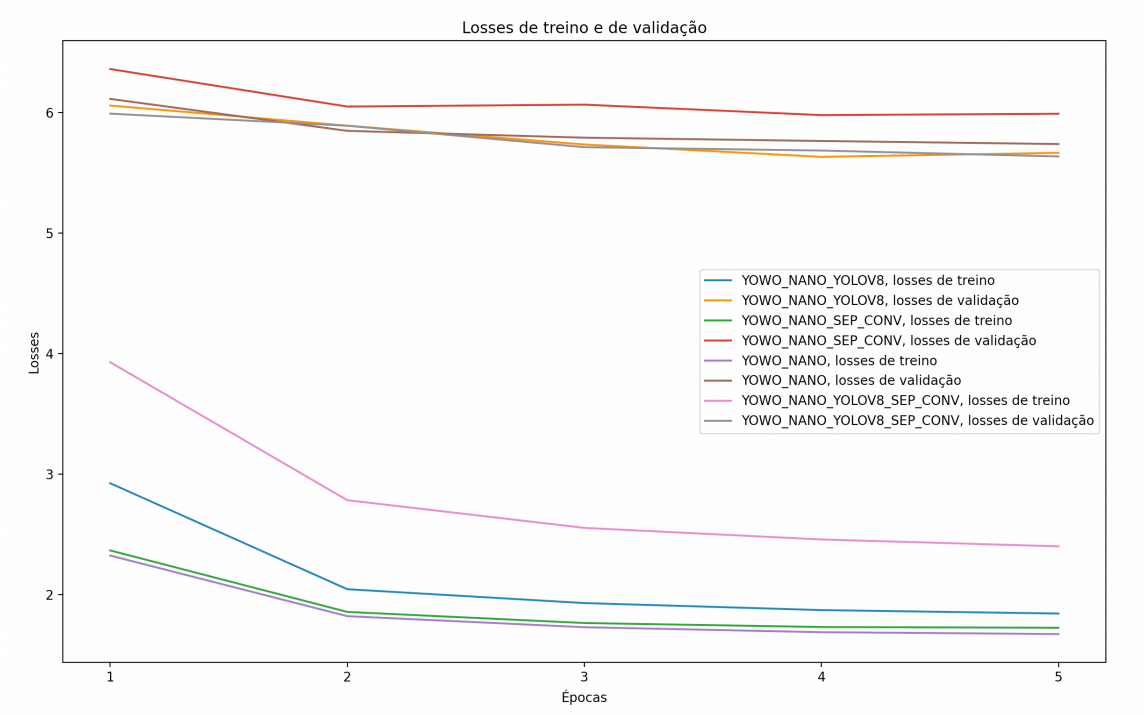

Figura 4.4: Gráfico das *losses* de treino e de validação com as configurações definidas em 4.8.

tar a taxa de aprendizagem, a convergência torna-se semelhante.

Apesar dos modelos convergirem mais rapidamente na fase de treino, estes não atingem o desempenho ideal na inferência com o conjunto de validação. Isto sugere que outros fatores além da taxa de aprendizagem podem estar a influenciar o desempenho dos modelos durante a fase de inferência. É possível que seja necessário ajustar outros hiperparâmetros (como o número de épocas) para obter melhores resultados.

Ao manter o otimizador SGD e ao aumentar a taxa de aprendizagem, observamos, no geral, uma melhoria do desempenho em comparação com os resultados apresentados na tabela 4.7.

Como visto nas experiências anteriores, o modelo YOWO Nano continua a apresentar um desempenho competitivo, mantendo-se como um dos melhores modelos avaliados. No entanto, o que chama a atenção é o desempenho do modelo YOLO Nano YOLOv8n, que surpreendentemente supera o desempenho do YOWO Nano sep conv pela primeira vez. Essa melhoria pode ser explicada pelo fato de que, com o otimizador SGD, o modelo YOLO Nano YOLOv8n é o segundo modelo com *losses* de validação mais baixas e com uma curva de aprendizagem que parece convergir durante o treino.

Tabela 4.9: Comparação do desempenho do modelo *baseline* e dos modelos desenvolvidos. Os hiperparâmetros sujeitos a análise e os respetivos valores podem ser visualizados em 4.8.

Modelo	Parâmetros (M)	GFLOPs (B)	Treino	Inferência (ms)	frame-mAP	video-mAP
YOWO Nano	72.78	6.07	18h 23m 50s	17.081	40.82%	30.64%
YOWO Nano sep conv	72.78	5.58	21h 23m 29s	17.082	32.35%	23.64%
YOWO Nano YOLOv8n	24.68	11.0	17h 59m 44s	17.140	36.33%	27.19%
YOWO Nano YOLOv8n sep conv	24.68	11.0	22h 46m 44s	17.418	5.48%	3.52%

4.2 Discussão

Após a conclusão das experiências realizadas com cada um dos modelos desenvolvidos, bem como a comparação com o modelo selecionado do estado da arte, é evidente que nenhum dos métodos desenvolvidos apresentou uma velocidade de inferência e/ou desempenho superior ao modelo YOWO Nano.

Estes resultados reforçam a escolha dos autores do YOWO Nano em relação à configuração definida, uma vez que proporciona os melhores resultados para o seu modelo, assim como para os modelos desenvolvidos ao longo do estudo com 5 épocas de treino.

É importante destacar que não foi explorada a execução dos modelos desenvolvidos ao longo de 20 épocas. Seria interessante executar o modelo YOWO Nano sep conv, dado que obteve uma frame-mAP superior ao estado da arte. Assim sendo, este poderia beneficiar do aumento do número de épocas de treino. Outra razão que sugere que este modelo pode ser promissor é o facto deste convergir ligeiramente mais rapidamente do que o estado da arte, conforme demonstrado no gráfico 4.2.

Em suma, existe a possibilidade de que ao treinar o modelo YOWO Nano sep conv com um maior número de épocas, se possa alcançar um desempenho superior ao estado da arte.

Capítulo 5

Conclusões e Trabalho Futuro

5.1 Conclusões

Durante o curso deste ano letivo, o estudo aprofundado dos diferentes métodos do estado da arte desempenhou um papel fundamental no desenvolvimento dos métodos propostos. Esta análise permitiu uma melhor compreensão das vantagens e das limitações de cada abordagem investigada.

Através do desenvolvimento de métodos baseados em redes convolucionais e em *transformers*, foi possível adquirir um vasto conhecimento nas áreas da Aprendizagem Profunda e da Visão Computacional. Além disso, surgiram diversos desafios associados à criação destes modelos, pelo que foi possível aprender como superá-los através de experiências e soluções inovadoras.

Embora os resultados das experiências realizadas com as redes desenvolvidas ao longo do ano não tenham superado os resultados do estado da arte, é importante destacar que o processo de estudo, desenvolvimento, experimentação e análise foi de grande valor. A pesquisa aprofundada e a melhoria contínua permitiram o desenvolvimento de habilidades técnicas sólidas na área, assim como de outras competências relevantes: autonomia, resiliência e capacidade analítica.

Por conseguinte, é evidente que, embora os resultados não tenham sido os esperados nas experiências realizadas, o conhecimento e as habilidades adquiridas durante este processo de investigação são inestimáveis. Estas competências irão contribuir para futuros empreendimentos académicos e profissionais, permitindo uma abordagem mais sólida e informada em futuras investigações.

5.2 Trabalho Futuro

Relativamente às experiências realizadas, seria interessante fazer um estudo sobre o desempenho de cada um dos modelos com um número de épocas e uma taxa de aprendizagem superiores e manipular os mapas de características resultantes das redes de suporte 2D e 3D de forma a extrair o maior número de características.

No contexto das experiências realizadas, surge um caminho interessante para investigação futura, que envolve a experimentação dos mesmos hiperparâmetros utilizados, assim como outros para avaliar o desempenho dos modelos. Relativamente aos hiperparâmetros utilizados seria interessante aumentar o número de épocas e a taxa de aprendizagem.

Um estudo que envolva um número maior de épocas permitiria uma análise mais aprofundada do desempenho dos modelos ao longo do treino. Desta forma, seria possível investigar

se o aumento do número de épocas resultaria num desempenho superior, permitindo assim que os modelos atingissem uma convergência maior.
Com o aumento da taxa de aprendizagem, os modelos poderiam atingir a convergência mais rapidamente, o que, aliado a um maior número de épocas podia resultar em desempenhos mais competitivos.
Adicionalmente, seria interessante explorar os mapas de características resultantes das redes de suporte 2D e 3D. Esta abordagem poderia potenciar a utilização das características mais relevantes e assim, possivelmente originar resultados superiores e até otimizar os modelos. Esta proposta é bastante desafiante devido à necessidade de investigar de que forma a manipulação dos mapas de características possa afetar o desempenho dos modelos, podendo ser necessário um *trade-off* entre a complexidade computacional e a melhoria dos resultados.

Bibliografia

[1] J. Cohen, "How to optimize a deep learning model for faster inference?" 2023 (acedido em Janeiro, 2023). [Online]. Available: https://www.thinkautonomous.ai/blog/deep-learning-optimization/ xiii, 4, 7, 8, 9

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real--time object detection," 2015. [Online]. Available: https://arxiv.org/abs/1506.02640 xiii, 10, 11, 25

[3] Ultralytics, "Ultralytics yolov8," 2023. [Online]. Available: https://github.com/ultralytics/ultralytics xiii, 12

[4] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," 2018. [Online]. Available: https://arxiv.org/abs/1812.03982 xiii, 14, 16

[5] J. Wu, Z. Kuang, L. Wang, W. Zhang, and G. Wu, "Context-aware rcnn: A baseline for action detection in videos," 2020. [Online]. Available: https://arxiv.org/abs/2007.09861 xiii, 16, 17

[6] S. Chen, P. Sun, E. Xie, C. Ge, J. Wu, L. Ma, J. Shen, and P. Luo, "Watch only once: An end-to-end video action detection framework," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, oct 2021, pp. 8158–8167. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/ICCV48922.2021.00807 xiii, 18, 19

[7] O. Köpüklü, X. Wei, and G. Rigoll, "You only watch once: A unified cnn architecture for real-time spatiotemporal action localization," 2019. [Online]. Available: https://arxiv.org/abs/1911.06644 xiii, 20, 21, 27, 28

[8] J. Yang, "Yowo plus: An incremental improvement," 2022. [Online]. Available: https://arxiv.org/abs/2210.11219 xv, 20, 25, 26

[9] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," 2012. 9

[10] G. Boesch, "Object detection in 2023: The definitive guide," 2023 (acedido em Janeiro, 2023). [Online]. Available: https://viso.ai/deep-learning/object-detection/ 10

[11] J. Terven and D. Cordova-Esparza, "A comprehensive review of yolo: From yolov1 to yolov8 and beyond," 2023. 11

[12] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," 2014. [Online]. Available: https://arxiv.org/abs/1412.0767 12

[13] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," 2018. 13

[14] J. Yang, “Yowo plus: An incremental improvement,” 2022. [Online]. Available: https://github.com/yjh0410/PyTorch_YOWO 21

[15] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, “An image is worth 16x16 words: Transformers for image recognition at scale,” 2021. 21

[16] L. Beyer, X. Zhai, and A. Kolesnikov, “Better plain vit baselines for imagenet-1k,” 2022. 22

[17] P. Wang, “Vision transformer - pytorch.” [Online]. Available: https://github.com/lucidrains/vit-pytorch 22, 23