UNIVERSIDADE D
COIMBRA

Aprendizegem Profunda Aplicada

Trabalho Pratico 1

Autor:
Gonçalo Bastos
Leonardo Cordeiro

Numero de Estudante:
2020238997
2020228071

Outubro 29, 2024

1 Parte 1

1.1 Exploratory Data Analysis (EDA)

Para compreender melhor o conjunto de dados CIFAR-10, realizámos uma análise exploratória que incluiu a visualização de imagens aleatórias de cada classe e a verificação da distribuição de classes nos conjuntos de treino e teste.

Visualização de Imagens por Classe

Na Figura 1, apresentamos uma amostra de imagens aleatórias de cada uma das 10 classes do conjunto de dados CIFAR-10. Esta visualização é essencial para adquirir uma perceção inicial sobre as características visuais das classes, que variam desde veículos (aviões, automóveis, camiões, etc.) até animais (gatos, cães, sapos, etc.).

Ao observar estas amostras, podemos identificar uma diversidade de contextos e posições das imagens, o que é benéfico para o modelo de aprendizagem profunda, pois ajuda a aumentar a robustez da rede em relação a variações nas características visuais. No entanto, dada a baixa resolução das imagens (32x32 pixels), algumas classes (e.g., pássaros e aviões) podem ser mais difíceis de distinguir visualmente, o que pode impactar o desempenho do modelo, especialmente nas camadas iniciais da rede convolucional, que dependem de texturas e formas de baixo nível.

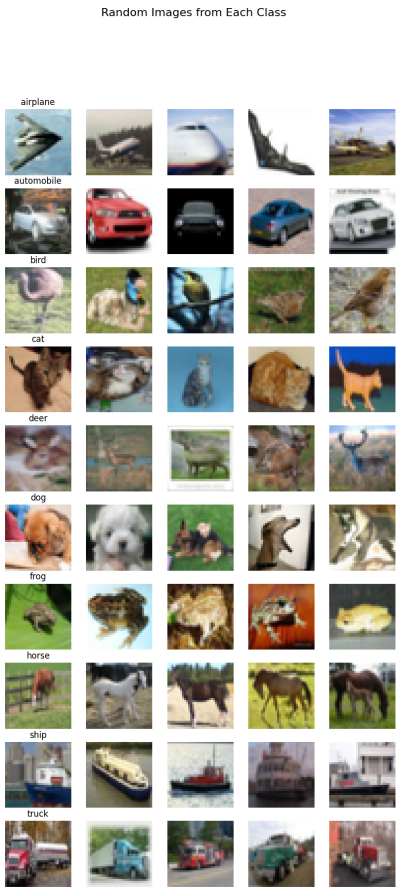

Figure 1: Imagens aleatórias de cada classe no dataset CIFAR-10.

Distribuição de Classes

Para verificar o equilíbrio das classes no conjunto de dados, analisámos a distribuição de classes tanto no conjunto de treino quanto no de teste, ilustradas nas Figuras 2 e **??**, respetivamente. Cada classe possui aproximadamente 5000 imagens no conjunto de treino e 1000 imagens no conjunto de teste, o que assegura um equilíbrio entre as classes e reduz o risco de enviesamento do modelo em relação a qualquer classe específica.

Este equilíbrio é vantajoso para o modelo, pois evita o problema de classes minoritárias, o que, em cenários de desequilíbrio, poderia fazer com que o modelo aprendesse padrões preferenciais de classes com maior frequência. No entanto, um ponto a considerar é a variabilidade intra-classe, como já observado na visualização de imagens aleatórias, o que pode desafiar o modelo a aprender representações consistentes para cada classe.

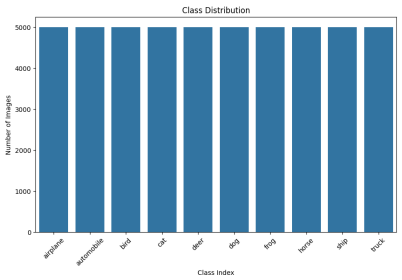

Figure 2: Distribuição de classes no conjunto de treino CIFAR-10.

Estatísticas do Conjunto de Dados

A normalização do dataset utilizando a média e o desvio-padrão dos valores de pixel em cada canal (RGB) é essencial para melhorar a aprendizagem do modelo. Estes dados permitem ajustar as imagens para que cada canal tenha uma distribuição com média zero e variância unitária, o que reduz a variabilidade indesejada entre as amostras e facilita a convergência durante o treino. Com uma distribuição mais uniforme dos valores de pixel, o modelo pode focar-se em aprender as características relevantes das imagens, como padrões e texturas, em vez de ser influenciado por variações de intensidade que não contribuem para a tarefa.

1.2 Modelo CNN Base

Primeiramente foi definido o modelo Base que corresponde à configuração da figura 1 do Assigment.

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1            [-1, 8, 32, 32]             224
            Conv2d-2           [-1, 32, 32, 32]           2,336
            Conv2d-3           [-1, 16, 32, 32]           4,624
            Linear-4                 [-1, 256]       4,194,560
           Dropout-5                 [-1, 256]               0
            Linear-6                  [-1, 64]          16,448
            Linear-7                  [-1, 10]             650
================================================================
Total params: 4,218,842
Trainable params: 4,218,842
Non-trainable params: 0
----------------------------------------------------------------
```

Figure 3: Base Model Summary

Treino

Foi defenida uma função de treino em que para cada epoch, o que ativa camadas específicas como dropout e batch normalization, para aumentar a robustez do modelo. Para cada batch de dados, os gradientes são levados a zero com o optimizer.zero_grad(). No forward pass o batch é paassado pelo modeolo, e a perda é calculado comparando as previsões com as labels reais. Por fim é efetuada o bacward pass e a atualização dos pesos, a retropropagação é usada para calcular os gradientes. Em seguida, o otimizador atualiza os pesos do modelo.

Função `evaluate_model`

A função evaluate_model avalia o desempenho do modelo no conjunto de teste, colocando-o em modo de avaliação para reduzir o uso de memória e acelerar o processo. Ela percorre o test_loader, obtém as previsões para cada lote e armazena-as para análise. Em seguida, gera uma matriz de confusão, que indica acertos e erros por classe, e um relatório de classificação com métricas como precisão, recall e F1-score, essenciais para entender o desempenho do modelo em cada classe. Esta função permite identificar limitações, como a dificuldade em distinguir classes visualmente semelhantes.

Analise de resultados: Matriz de Confusão e Relatório de Classificação

1. **(Accuracy)**: A Accuracy mede a proporção de previsões corretas em relação ao total de previsões feitas. É uma métrica geral que mostra o desempenho global do modelo.

$$\text{Accuracy} = \frac{\sum_{i=1}^{N} \text{VP}_i}{\sum_{i=1}^{N} (\text{VP}_i + \text{FP}_i + \text{FN}_i + \text{VN}_i)}$$

onde VP_i (verdadeiros positivos), FP_i (falsos positivos), FN_i (falsos negativos) e VN_i (verdadeiros negativos) representam, respectivamente, o número de previsões corretas e incorretas para cada classe i.

2. **Precisão (Precision)**: A precisão mede a proporção de previsões corretas entre todas as previsões feitas para uma determinada classe. Em termos matemáticos, para uma classe i:

$$\text{Precision}_i = \frac{\text{VP}_i}{\text{VP}_i + \text{FP}_i}$$

onde VP_i representa o número de previsões corretas para a classe i, e FP_i representa o número de vezes que outras classes foram incorretamente classificadas como a classe i.

3. **Recall (Sensibilidade)**: O recall mede a proporção de previsões corretas em relação ao total de exemplos reais de uma determinada classe. Matematicamente, para a classe i:

$$\text{Recall}_i = \frac{\text{VP}_i}{\text{VP}_i + \text{FN}_i}$$

onde FN_i representa o número de vezes que exemplos da classe i foram incorretamente classificados como pertencendo a outras classes.

4. **F1-score**: O F1-score é a média harmônica entre a precisão e o recall, proporcionando uma única métrica que considera tanto a exatidão quanto a abrangência do modelo para cada classe. É calculado como:

$$\text{F1-score}_i = 2 \cdot \frac{\text{Precisão}_i \cdot \text{Recall}_i}{\text{Precisão}_i + \text{Recall}_i}$$

O F1-score é particularmente útil em casos de classes desbalanceadas, pois pondera a precisão e o recall igualmente.

5. **Médias (macro e ponderada)**: As médias *macro* e *ponderada (weighted)* fornecem uma visão geral do desempenho do modelo em todas as classes. A média macro calcula a média simples das métricas para todas as classes:

$$\text{Macro Avg} = \frac{1}{N} \sum_{i=1}^{N} \text{Métrica}_i$$

enquanto a média ponderada considera o suporte (número de exemplos) de cada classe:

$$\text{Weighted Avg} = \frac{\sum_{i=1}^{N} \text{Métrica}_i \cdot \text{Suporte}_i}{\sum_{i=1}^{N} \text{Suporte}_i}$$

Observa-se uma precisão média ponderada de 61%, indicando que o modelo tem um desempenho mediano. A análise da matriz de confusão e do relatório de classificação revela que o modelo tem maior dificuldade em distinguir classes como `bird`, `cat` e `dog`, que apresentam características visuais semelhantes, especialmente com a resolução de 32x32 pixels das imagens CIFAR-10.

Classification Report:

	precision	recall	f1-score	support
airplane	0.66	0.57	0.61	1000
automobile	0.80	0.63	0.71	1000
bird	0.49	0.44	0.46	1000
cat	0.40	0.40	0.40	1000
deer	0.49	0.48	0.49	1000
dog	0.51	0.51	0.51	1000
frog	0.60	0.72	0.66	1000
horse	0.56	0.71	0.63	1000
ship	0.73	0.69	0.71	1000
truck	0.65	0.70	0.67	1000
accuracy			0.58	10000
macro avg	0.59	0.58	0.58	10000
weighted avg	0.59	0.58	0.58	10000

Figure 4: Classification Report Base Model

A Figura 5 apresenta a matriz de confusão, onde se observa que classes como `bird` e `cat` têm uma maior taxa de confusões, especialmente com outras classes de animais, achamos que a baixa resolução das imagens e a similaridade visual entre algumas classes podem ser fatores que dificultam o desempenho do modelo, e esta primeira análise sugere que o modelo pode beneficiar de ajustes adicionais.

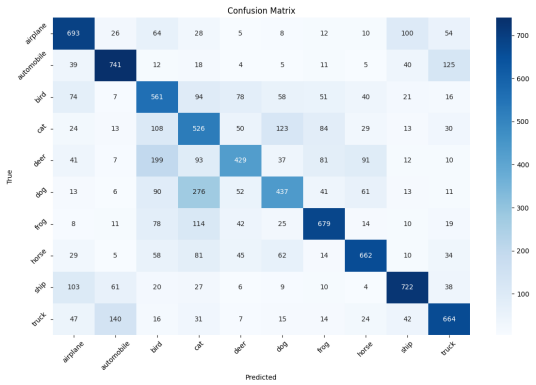

Figure 5: Matriz de Confusão para o Modelo CNN Base no Conjunto de Teste CIFAR-10.

Analise de resultados: Curvas de Treino e Validação

Na Figura 6, são apresentadas as curvas de perda e precisão para o conjunto de treino e validação ao longo das 25 epochs de treino.

- **Curvas de Perda:** Observa-se uma redução consistente na perda de treino, indicando que o modelo está a aprender de forma progressiva. A perda de validação também diminui, mas de forma mais lenta e estabiliza em torno do 15ª epoch, sugerindo que o modelo está próximo de alcançar a sua capacidade máxima de generalização com esta configuração de treino.
- **Curvas de Accuracy:** A accuracy de treino aumenta de forma contínua até cerca de 69%, enquanto a precisão de validação se estabiliza em torno de 61.6%. Esta diferença indica uma leve tendência de overfiting, em que o modelo aprende bem os dados de treino, mas tem um desempenho inferior em dados de validação. Técnicas adicionais de regularização, como aumento de dropout ou técnicas de aumento de dados, poderiam ajudar a melhorar a generalização.

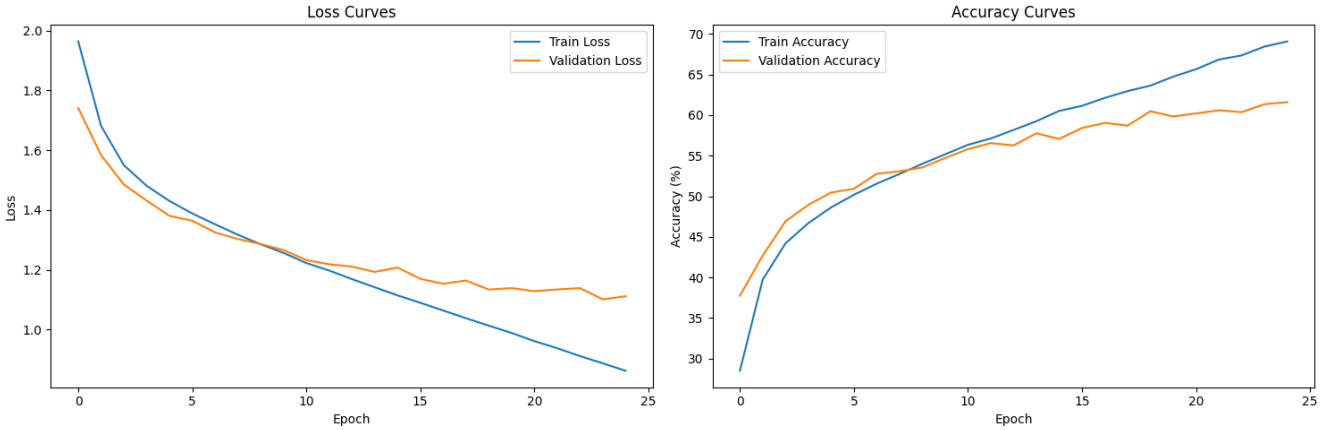

Figure 6: Curvas de Perda e Precisão para o Treino e Validação do Modelo CNN Base.

Esta análise revela que, embora o modelo apresente uma evolução positiva no treino, existem oportunidades para melhorar a sua capacidade de generalização, o que será explorado nas proximas seções.

1.3 Melhoria do Modelo CNN: Versão CNNV2

Após a análise do modelo CNN base, identificámos oportunidades para melhorar a generalização e reduzir o sobreajuste. Assim, criámos o modelo `CNNV2` com as seguintes modificações principais:

- **Aumento de Filtros e Camada Convolucional Extra:** Mais filtros foram adicionados (16-32-64-128 em vez de 8-32-16), com uma quarta camada convolucional para extração de características mais profunda.
- **Batch Normalization e Max-Pooling:** Aplicados após cada camada convolucional, estabilizam o treino e reduzem a dimensionalidade, ajudando a controlar o overfitting.

- **Camadas Fully Connected com Dropout:** Ajustadas para acomodar as mudanças de pooling, com Dropout aumentado para 0.4 para maior regularização.

Os hiperparâmetros utilizados para treinar o modelo `CNNV2` foram: **Learning Rate:** 0.001, **epochs:** 25, **Função de Perda:** `CrossEntropyLoss`, e **Otimizador:** `Adam`.

Os resultados finais do treino são os seguintes:

- **Loss de Treino:** 0.1583, com uma accuracy de treino de 94.44%.
- **Loss de Validação:** 1.0673, com uma accuracy de validação de 75.04%.

O relatório de classificação para o modelo `CNNV2` mostra uma melhoria significativa na precisão de validação, que passou de 61.6% no modelo base para 75.04%. Observa-se um desempenho mais equilibrado entre as classes, especialmente nas classes de veículos (`automobile` e `truck`), devido à maior profundidade e regularização do modelo. No entanto, classes como `cat` e `dog` ainda apresentam baixa distinção, indicando desafios na diferenciação de algumas classes de animais.

As curvas de perda e precisão do modelo `CNNV2` (Figura **??**) mostram sinais de overfitting. A perda de treino diminui consistentemente, enquanto a perda de validação estabiliza em torno de 1.0673 nas últimas epochs, sugerindo que o modelo memoriza os dados de treino sem generalizar bem. A precisão de treino atinge 94.44%, mas a precisão de validação fica em 75.04%, reforçando a dificuldade do modelo em generalizar para novos dados.

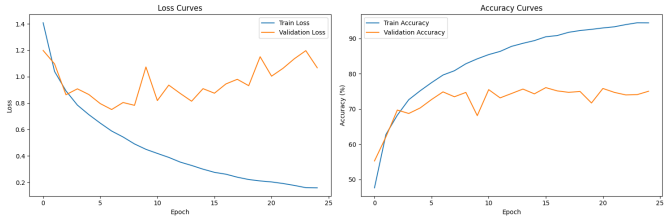

Figure 7: Curvas de Perda e Precisão para o Treino e Validação do Modelo CNNV2.

Os resultados das curvas de treino e validação mostram que, apesar das melhorias em relação ao modelo base, o modelo `CNNV2` continua a sofrer de overfitting. Para mitigar este problema, serão necessárias estratégias adicionais, como Data Augmentation mais extensivo, o ajuste fino da taxa de dropout, ou mesmo o uso de técnicas avançadas de regularização. Estas abordagens podem ajudar a reduzir a diferença entre as curvas de treino e validação, promovendo uma melhor capacidade de generalização do modelo.

1.4 Melhoria do Modelo CNN: Versão CNNV3

Com o objetivo de mitigar o overfitting identificado na versão `CNNV2` do modelo, foi desenvolvida a versão `CNNV3`. Esta nova versão incorpora um número adicional de camadas convolucionais e uma regularização mais robusta, com o intuito de melhorar a generalização do modelo e aumentar a precisão em dados de validação.

A `CNNV3` foi projetada com as seguintes melhorias e ajustes:

- **Cinco Blocos Convolucionais:** Em comparação com a `CNNV2`, foi adicionado um quinto bloco convolucional, com 512 filtros, seguido de Batch Normalization, Max-Pooling, e Dropout para tentar que o modelo capture características ainda mais complexas das imagens de entrada.
- **Taxas de Dropout Variáveis:** As taxas de dropout foram aumentadas progressivamente em cada bloco convolucional (de 0.2 até 0.5) para melhorar a regularização e prevenir o overfitting. O uso de diferentes taxas de dropout para cada camada contribui para uma regularização adaptativa, onde as camadas mais profundas, mais propensas ao overfitting, recebem uma taxa de dropout mais alta.
- **Weight Decay no Otimizador:** Foi aplicado um parâmetro de weight decay de 0.0001 no otimizador `Adam`, o que adiciona regularização L2 aos pesos da rede. Esta técnica contribui para a redução do overfitting, penalizando pesos excessivamente altos e promovendo um modelo mais robusto.

O modelo `CNNV3` foi treinado com os seguintes hiperparâmetros os mesmos hiperparametros do anterior e os resultados do modelo `CNNV3` indicam uma melhoria clara em relação às versões anteriores, com uma precisão de validação geral de 80%, conforme apresentado no relatório de classificação na Tabela 1. A precisão aumentou consideravelmente, principalmente nas classes `automobile`, `ship` e `truck`, sugerindo que o modelo conseguiu captar melhor as características distintivas dessas classes.

Table 1: Relatório de Classificação do Modelo CNNV3 no Conjunto de Teste CIFAR-10

Classe	Precisão	Recall	F1-Score	Suporte
airplane	0.84	0.83	0.83	1000
automobile	0.92	0.91	0.91	1000
bird	0.79	0.67	0.72	1000
cat	0.63	0.66	0.64	1000
deer	0.78	0.75	0.76	1000
dog	0.69	0.75	0.72	1000
frog	0.72	0.90	0.80	1000
horse	0.88	0.81	0.84	1000
horse	0.88	0.81	0.84	1000
ship	0.89	0.90	0.90	1000
truck	0.93	0.82	0.87	1000
Média Geral	**0.81**	**0.80**	**0.80**	**10000**

A análise das métricas de precisão, recall e F1-score mostra que o modelo `CNNV3` conseguiu reduzir os erros em classes mais difíceis, como `bird` e `dog`, e aumentou a precisão nas classes de veículos, como `automobile` e `truck`. No entanto, algumas classes de animais, como `cat` e `dog`, ainda apresentam valores de F1-score moderados, sugerindo que o modelo enfrenta desafios em distinguir características visuais semelhantes entre estas classes em algumas situações.

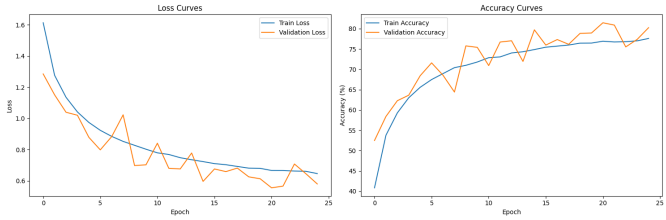

Figure 8: Curvas de Perda e Precisão para o Treino e Validação do Modelo CNNV3.

A Figura 8 mostra as curvas de perda e precisão para o treino e validação do modelo `CNNV3` ao longo das 25 epochs. Comparando com as versões anteriores, o modelo apresenta uma maior estabilidade nas curvas de validação e uma menor disparidade entre treino e validação, indicando que as técnicas de regularização aplicadas foram eficazes em reduzir o overfitting. A curva de perda de treino diminui de forma consistente, enquanto a perda de validação apresenta uma tendência de estabilização a partir das últimas epochs, em torno de 0.6. Esta estabilização, com menos oscilações em comparação com as versões anteriores, indica que o modelo conseguiu alcançar uma boa generalização para os dados de validação. A precisão de treino aumenta até aproximadamente 88%, enquanto a precisão de validação se estabiliza em torno de 80%. A proximidade entre as curvas de treino e validação sugere que o modelo está a generalizar melhor, beneficiando das camadas adicionais e das técnicas de regularização mais robustas.

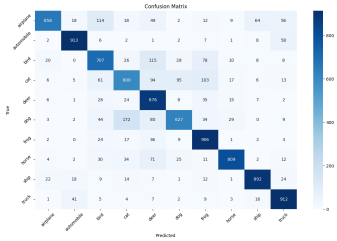

Figure 9: Confusion Matrix CNN V3

1.5 Análise Comparativa das Configurações de Hiperparâmetros

Para avaliar o impacto dos hiperparâmetros na generalização do modelo, testámos quatro configurações de taxa de aprendizagem (LR), dropout e weight decay no `CNNV3`. A Tabela 2 resume os principais resultados de cada configuração, considerando cenários de overfitting e underfitting.

Table 2: Resultados das Configurações de Hiperparâmetros no `CNNV3`

Configuração	LR	Dropout	Weight Decay	Val Acc (%)	Observações
Config 1	0.001	0.3	0.0001	83.13	Bom equilíbrio, menor overfitting
Config 2	0.0005	0.4	0.0005	82.23	Bom controle de overfitting, leve underfitting
Config 3	0.0001	0.3	0.0001	78.48	Underfitting devido à baixa LR
Config 4	0.0001	0.5	0.0005	70.69	Underfitting acentuado, excesso de regularização

Análise das Configurações: A configuração com **LR=0.001, Dropout=0.3, Weight Decay=0.0001** apresentou o melhor desempenho (83.13% de precisão de validação), mostrando bom equilíbrio entre precisão de treino e validação, com leve overfitting em classes complexas como `cat` e `dog`. O aumento do dropout e do weight decay nas Configurações 2, 3 e 4 ajudou a controlar o overfitting, mas resultou em underfitting, especialmente nas classes mais difíceis. Estas observações ajudam a guiar otimizações futuras para maximizar a precisão do modelo e minimizar problemas de overfitting e underfitting.

2 Parte II

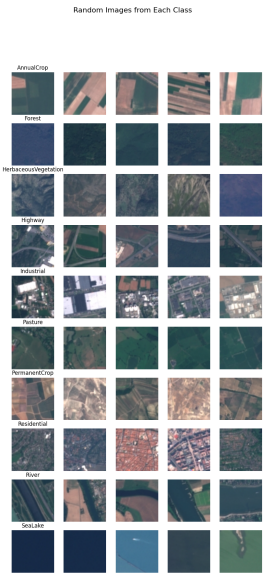

Figure 10: Eurosat Imagens de Exemplo

Na parte II do trabalho definimos os modelos AlexrNet e MobileNetV2 a partir dos modelos pré definidos do Pytorch. Quando os modelos são treinados de raiz os pesos são inicializados de forma aleatória, pelo que é necessário maior cuidado com os hiperparâmetros para treinar o modelo sem overfitting. O procedimento de treino usado foi igual ao implementado na parte I do trabalho.

Em transfer learning, os modelos são inicializados com pesos pré treinados com um grande dataset(ImageNet), isto permite que o modelo alavanque a aprendizagem de features, o que pode levar a um speedup do tempo de treino bem como melhorar a performance do modelo. Para usar o modelo com os pesos pré treinados devemos de congelar as camadas convolocionais de modo a preservar as features previamente aprendidas. Aléem disso temos de modificar a camada de classificação para corresponder ao número de classes do nosso dataset em específico, neste caso 10 classes.

2.1 Comparativo dos modelos com treino de raiz e com os pesos pré treinados

Todos os modelos treinados com os pesos pré treinados têm uma convergência mais rápida e começam com uma loss inicial mais baixa, o que indica um processo de aprendizagem mais eficiente. O modelo MobileNetV2 com os pesos pré treinados mostra uma boa generalização, com a accuracy de validação a corresponder ou mesmo a ultrapassar a accuracy de treino. O modelo com pior resultado foi a AlexNet treinada de raiz que tem algumas flutuações na curva de Loss, o que pode significar que o modelo tenha overfitting. O modelo MobileNetV2 alcançou maiores valores de accuracy e menores valores de perda. Com esta análise percebemos que o transfer learning usando pesos pré

treinados podem resultar em muitas vantagens e eficiência especialmente em casos com dados e recursos limitados.

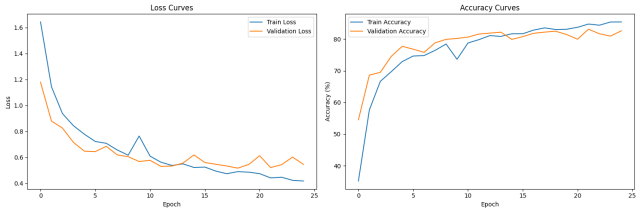

Figure 11: Curvas de Loss e Accuracy de AlexNet from scratch

Classification Report:

	precision	recall	f1-score	support
AnnualCrop	0.87	0.81	0.84	628
Forest	0.87	0.91	0.89	557
HerbaceousVegetation	0.73	0.78	0.75	581
Highway	0.87	0.73	0.79	478
Industrial	0.82	0.99	0.90	474
Pasture	0.86	0.66	0.75	437
PermanentCrop	0.61	0.76	0.68	492
Residential	0.96	0.90	0.93	643
River	0.76	0.71	0.74	488
SeaLake	0.95	0.94	0.94	622
accuracy			0.83	5400
macro avg	0.83	0.82	0.82	5400
weighted avg	0.83	0.83	0.83	5400

Figure 12: AlexNet from scratch - Classification Report

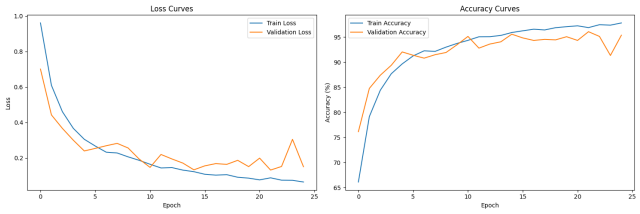

Figure 13: Curvas de Loss e Accuracy de MobileNetV2 from scratch

Classification Report:

	precision	recall	f1-score	support
AnnualCrop	0.94	0.92	0.93	578
Forest	0.98	0.98	0.98	560
HerbaceousVegetation	0.90	0.92	0.91	599
Highway	0.97	0.97	0.97	495
Industrial	0.96	0.98	0.97	534
Pasture	0.97	0.94	0.95	400
PermanentCrop	0.91	0.89	0.90	507
Residential	0.98	0.99	0.98	616
River	0.96	0.94	0.95	498
SeaLake	0.96	1.00	0.98	613
accuracy			0.95	5400
macro avg	0.95	0.95	0.95	5400
weighted avg	0.95	0.95	0.95	5400

Figure 14: MobileNetV2 from scratch - Classification Report

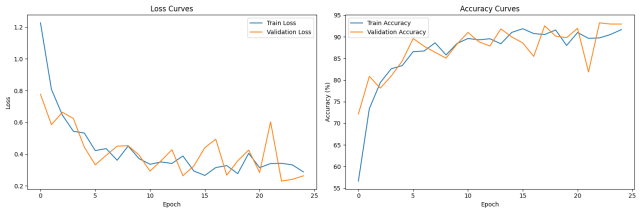

Figure 15: Curvas de Loss e Accuracy de AlexNet com pesos pré-treinados

Classification Report:

	precision	recall	f1-score	support
AnnualCrop	0.95	0.90	0.93	629
Forest	0.98	0.95	0.97	588
HerbaceousVegetation	0.85	0.95	0.90	624
Highway	0.96	0.88	0.92	491
Industrial	0.97	0.96	0.96	484
Pasture	0.84	0.95	0.89	363
PermanentCrop	0.95	0.79	0.86	492
Residential	0.98	0.99	0.98	646
River	0.84	0.92	0.88	506
SeaLake	0.98	0.98	0.98	577
accuracy			0.93	5400
macro avg	0.93	0.93	0.93	5400
weighted avg	0.93	0.93	0.93	5400

Figure 16: AlexNet com pesos pré-treinados - Classification Report

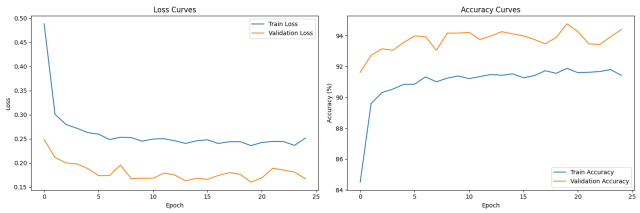

Figure 17: Curvas de Loss e Accuracy de MobileNetV2 com pesos pré-treinados

Classification Report:

	precision	recall	f1-score	support
AnnualCrop	0.93	0.95	0.94	601
Forest	1.00	0.99	0.99	608
HerbaceousVegetation	0.91	0.97	0.94	614
Highway	0.88	0.87	0.87	488
Industrial	0.96	0.97	0.96	508
Pasture	0.87	0.98	0.92	403
PermanentCrop	0.96	0.83	0.89	491
Residential	0.96	0.99	0.98	604
River	0.93	0.84	0.89	501
SeaLake	0.99	0.98	0.99	582
accuracy			0.94	5400
macro avg	0.94	0.94	0.94	5400
weighted avg	0.94	0.94	0.94	5400

Figure 18: MobileNetV2 com pesos pré-treinados - Classification Report

2.2 Middle Fusion Model

A Middle Fusion é uma técnica usada em deep learning para combinar representações de features de duas ou mais redes neuronais. Combinanddo as features de múltiplos redes pré treinadas conseguimos criar um modelo que

alavanca as vantagens de cada modelo, com isso prevemos alcançar uma maior generalização, aumentar o poder da representação e capturar múltiplas escalas de features devido à profundidade das convuloções. Neste caso foi defininda a classe MidleFusionModel que usa os modelos pré treinados da AlexNet e da MobileNetV2, foi removida as camadas de classificação pois não são necessárias nos modelos combinados. Foi necessário extrair as features dos dois modelos, porém para combinar os modelos ée necessário ter a mesma dimensão espacial, logo utilizámos a AddaptativeAvgPool2d para as features do MobileNetV2 corresponderem ao AlexNet. Uma vez que as dimensões estão ajustadas combinamos os mapas de features das duas redes. Ajustámos o classificador para corresponder ao nosso problema e, por último, congelámos as camadas de convolução das redes pré-treinadas, que faz com que retemos as features da ImageNet e focamos o nosso treino na nova classificação.

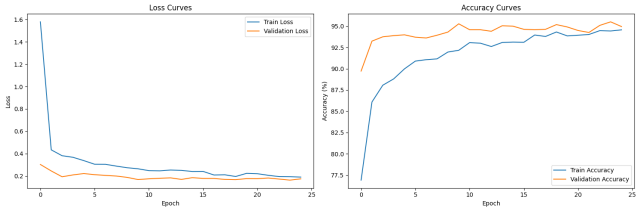

Figure 19: Curva de Loss e Accuracy - Middle Fusion Model

Classification Report:

	precision	recall	f1-score	support
AnnualCrop	0.95	0.94	0.94	633
Forest	0.98	0.99	0.98	588
HerbaceousVegetation	0.94	0.96	0.95	569
Highway	0.93	0.92	0.92	543
Industrial	0.99	0.96	0.98	483
Pasture	0.94	0.96	0.95	399
PermanentCrop	0.92	0.92	0.92	496
Residential	0.99	0.99	0.99	610
River	0.91	0.91	0.91	493
SeaLake	0.99	0.99	0.99	586
accuracy			0.95	5400
macro avg	0.95	0.95	0.95	5400
weighted avg	0.95	0.95	0.95	5400

Figure 20: Classification Report - Middle Fusion Model

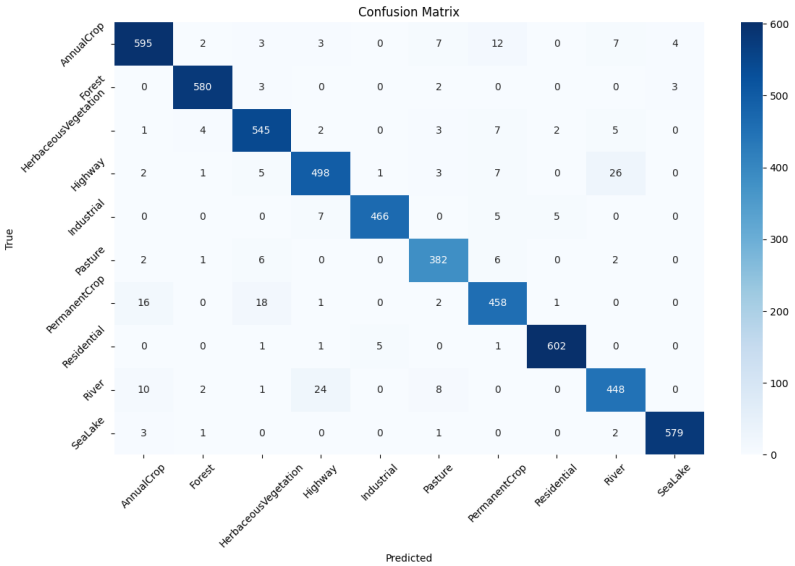

Figure 21: Confusion Matrix - Middle Fusion Model