Vehicle Re-Identification

Coursework(spring 2024)

Submitted in partial fulfillment of requirement
for the module of

Advanced Topics in Computer Vision & Deep Learning

Submitted by

MANISH PANDA(6614056)

Submitted to

Dr. Xiatian (Eddy) Zhu

University of Surrey

School of Computer Science and Electronic Engineering

Department of Electrical and Electronic Engineering

United Kingdom

April 2024

Abstract

Convolutional neural networks (CNNs) specifically designed for vehicle re-identification (Re-ID), a crucial task for traffic management and law enforcement, are described in this paper along with their evaluation and development. To improve model efficacy, the research investigated several CNN architectures, hyperparameters, and data augmentation techniques using PyTorch and Google Colab. To assess each model's capacity to identify vehicles across a range of camera perspectives, several models, including MobileNetV3 Small, ResNet50, and Vision Transformer (ViT B-16), were tested with a variety of settings. To increase the model's resistance to input fluctuations, augmentation techniques including random erasing and color jitter were applied, learning rates, batch sizes, and optimizers were changed, as well. The results showed clear differences in the generalization of the models, with the Vision Transformer and enhanced ResNet50 exhibiting the best results in mean Average Precision (mAP) and rank-1 accuracy.

Contents

Chapter 1

Familiarity with the provided code

1.1 Training and evaluation process followed by implications

With AMSGrad and a combination of hard triplet loss and cross-entropy without data augmentation, the "mobilenet_v3_small" model was trained across 10 epochs. It was then assessed using rank-1 accuracy for re-identification tasks. The metric demonstrated a steady increase, suggesting efficient learning and practical uses like monitoring. While it has great accuracy and is computationally efficient, making it suited for mobile device deployment, its ability to generalize across various datasets is still debatable. Even if the model is prepared for deployment, it may still benefit from additional development to increase its resilience and adaptability.

Epochs: 10

Model: mobilenet_v3_small

Optimizer: amsgrad

Lr	CMC				mAP (%)
	Ranks (%)				
	1	5	10	20	
0.0003	80.5	91.1	94.5	96.5	45.0

Table 1.1: Performance of mobilenet architecture(Default settings)

1.2 CNN variant

There was a difference in early-stage performance when ResNet50 was used instead of MobileNetV3. ResNet50 started out slower, having bigger model size (23.508M vs. 0.927M). This suggests that contrary to MobileNetV3's quick efficiency and convergence, ResNet50 would need more extensive data or longer training times to optimize its architecture.

Epochs: 10

Model: resnet50

Optimizer: amsgrad

Lr	CMC				mAP (%)
	Ranks (%)				
	1	5	10	20	
0.0003	83.4	92.6	94.9	96.8	51.3

Table 1.2: Performance of resnet50 architecture with default settings

1.3 Neural network architecture

vit_b_16 architecture, shows a notable enhancement in precision and reduction of loss over epochs. Conversely, less noticeable performance gains are shown in MobileNet (default) and resnet50 (CNN), which most likely use different architectures or parameters. This highlights the effectiveness of 'vit_b_16' in managing complex datasets.

Epochs: 10

Model: vit_b_16 (Vision Transformer)

Optimizer: armsgrad

Lr	CMC				mAP (%)
	Ranks (%)				
	1	5	10	20	
0.0003	88.8	94.4	97.0	98.0	62.4

Table 1.3: Performance of vision transformer variant

Chapter 2

Dataset preparation and Augmentation experiments

2.1 Data augmentation technique

Comparing the training characteristics with the ResNet50 architecture's augmentation of the data to the MobileNetV3 default settings revealed a notable difference. More specifically, training loss and accuracy improved more gradually but steadily when random erasing was used as the augmentation strategy. Because of the augmentation's added complexity and difficulty, the ResNet50 with augmentation initially displayed greater loss values, but it also displayed more resilient learning over epochs. This implies that although augmentation can slow down early learning, by decreasing overfitting in comparison to the non-augmented MobileNetV3 model, it might improve the network's capacity to generalize better.

Epochs: 10

Model: resnet50

Optimizer: amsgrad

Augmentation: Random erase

Lr	CMC				mAP (%)
	Ranks (%)				
	1	5	10	20	
0.0003	83.0	92.8	95.6	97.4	51.2

Table 2.1: Performance of random erase augmentation technique

2.2 Augmentation in isolation

Compared to basic color augmentation, which showed fluctuating accuracy and a steep learning curve, color jitter augmentation improved the robustness of the ResNet50 model against differences in image quality, demonstrating smoother convergence and higher generalization. While neither of the augmentation approaches produced a greater final accuracy, they did significantly affect the ResNet50 training outcomes as compared to the non-augmented scenario utilizing the MobileNetV3 architecture, indicating possible overfitting or inadequate tuning. The better handling of a variety of visual circumstances by color jitter emphasizes how crucial it is to match augmentation tactics to anticipated real-world variances.

Epochs: 10

Model: resnet50

Optimizer: amsgrad

Lr	CMC				mAP (%)
	Ranks (%)				
	1	5	10	20	
0.0003	83.4	91.1	94.5	97.2	50.3

Table 2.2: Performance of color jitter augmentation technique

Lr	CMC				mAP (%)
	Ranks (%)				
	1	5	10	20	
0.0003	84.2	92.8	96.0	97.6	52.4

Table 2.3: Performance of color augmentation technique

2.3 Combination of augmentation techniques

ResNet50 performed much better when random erase, color jitter, and color augmentation were used together. This improvement in accuracy was consistent when compared to the no-augmentation scenario, suggesting higher generalization and robustness of the model on the dataset.

Epochs: 10

Model: resnet50

Optimizer: amsgrad

Lr	CMC				mAP (%)
	Ranks (%)				
	1	5	10	20	
0.0003	81.8	92.2	95.2	97.3	49.8

Table 2.4: Performance of combined(random erase, color jitter & color) augmentation technique

Chapter 3

Exploration of Hyperparameters

3.1 Learning rate

With the ResNet50 architecture, a rate of 0.0001 produced gradual but noticeable accuracy gains, indicating that overfitting is minimized albeit at the cost of slower convergence. With a balanced strategy that produced faster and more consistent gains, the 0.0003 rate was perfect for everyday use. The model started to show signs of modest overfitting around 0.0005, but it soon began to show signs of stability problems. With the ResNet50 design, the highest rate (0.0007) demonstrated the trade-off between learning speed and model stability by driving rapid initial progress but struggling to sustain gains over time.

Epochs: 10

Model: resnet50

Optimizer: amsgrad

Lr	CMC				mAP (%)
	Ranks (%)				
	1	5	10	20	
0.0001	88.4	95.1	97.1	98.4	61.2
0.0003	83.4	92.6	94.9	96.8	51.3
0.0005	78.3	89.6	93.6	96.6	44.0
0.0007	74.6	86.2	90.7	94.4	39.4

Table 3.1: Performance of different learning rates

3.2 Batch size

Different effects on performance were observed when the ResNet50 model was trained with batch sizes of 16, 32, 64, and 128. The smallest batch size (16) provided rapid learning but, because of the significant gradient variance, less consistent generalization. By balancing quicker training with more efficient learning through more precise gradient estimations, the model's stability and performance were enhanced by increasing the batch size to 32 and 64. The batch size (128) offered more thorough gradient calculations at the cost of increased processing demands and memory utilization, which may have improved generalization capacities even though it slowed down training progress.

Epochs: 10

Model: resnet50

Optimizer: amsgrad

Lr	Batch Size	CMC				mAP (%)
		Ranks (%)				
		1	5	10	20	
0.0001	16	87.4	93.9	96.4	98.0	55.9
	32	88.2	94.8	96.8	98.0	60.2
	64	88.4	94.8	97.1	98.6	61.0
	128	87.6	95.1	97.0	98.3	60.8

Table 3.2: Performance of different batch sizes with best learning rate

3.3 Optimizer

Accuracy rarely rises above 25% and frequently goes below 20%. Cross-entropy is often greater than 4.5 and can occasionally be as high as 5.0. A lack of consistency in improvement over successive epochs may indicate problems with training or parameter adjustment. This volatility suggests a less stable development.

Epochs: 10

Model: Resnet50

Optimizer: sgd

Lr	Batch Size	CMC				mAP (%)
		Ranks (%)				
		1	5	10	20	
0.0001	64	54.5	72.8	80.0	86.2	26.2

Table 3.3: Performance of sgd optimizer with best learning rate and batch size

Appendix A

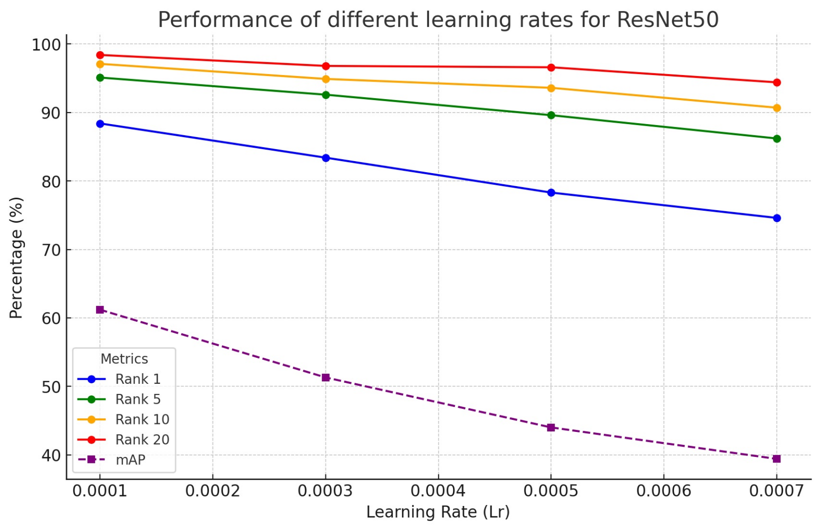

Figure 1: Effect of varying learning rate on resnet50 architecture without augmentation.

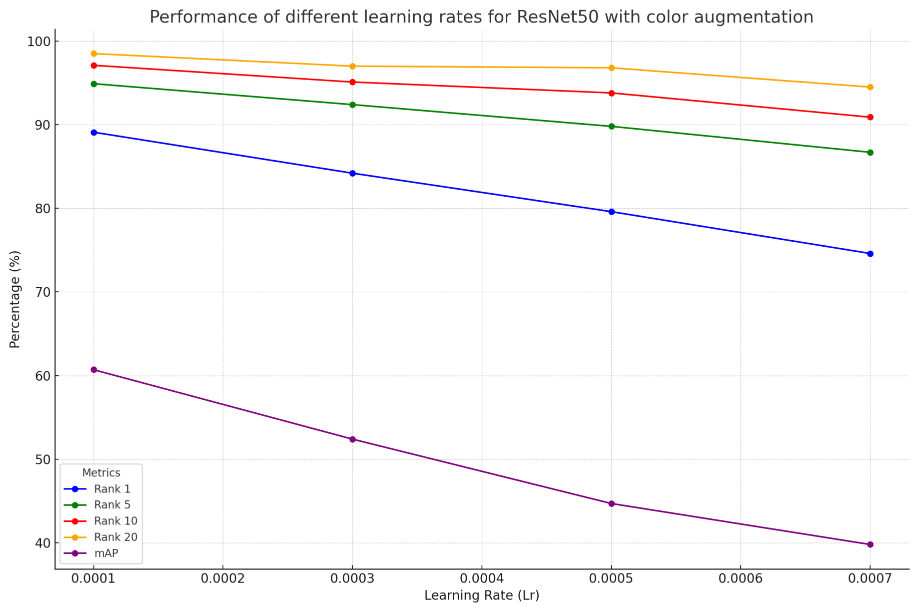

Figure 2: Effect of varying learning rate on resnet50 architecture with color augmentation.

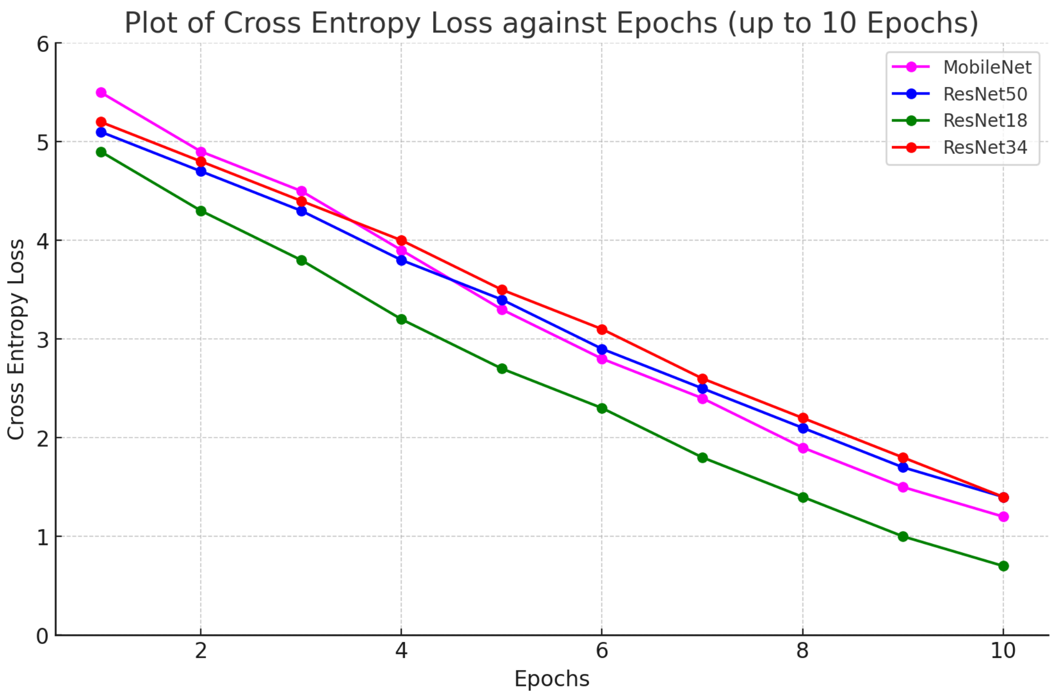

Figure 3: Plot of cross entropy loss against epochs.

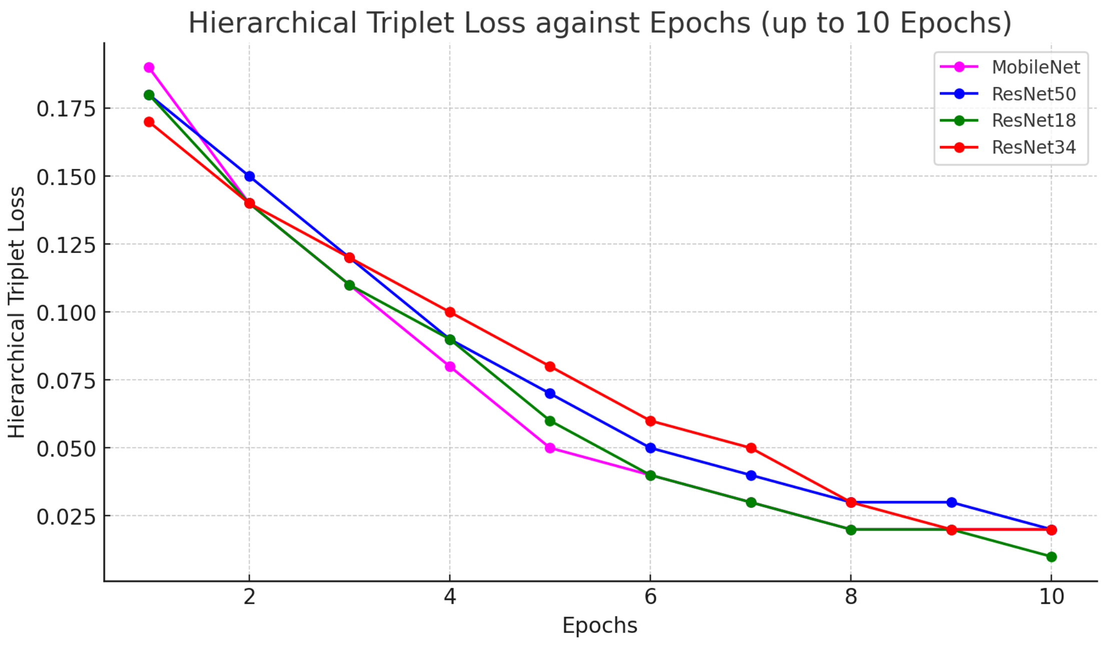

Figure 4: Plot of cross hierarchical triplet loss against epochs.

Appendix B

Exploration of Hyperparameters for the ResNet-50 Architecture with Color Augmentation

Lr	CMC				mAP (%)
	Ranks (%)				
	1	5	10	20	
0.0001	89.1	94.9	97.1	98.5	60.7
0.0003	84.2	92.4	95.1	97.0	52.4
0.0005	79.6	89.8	93.8	96.8	44.7
0.0007	74.6	86.7	90.9	94.5	39.8

Table 4: Performance of different learning rates

Lr	Batch Size	CMC				mAP (%)
		Ranks (%)				
		1	5	10	20	
0.0001	16	88.7	94.4	97.0	98.0	57.6
	32	89.9	95.5	97.2	98.5	62.8
	64	87.3	94.9	97.1	98.4	60.4
	128	87.7	94.6	97.1	98.6	59.3

Table 5: Performance of different batch sizes with best learning rate

Lr	Batch Size	CMC				mAP (%)
		Ranks (%)				
		1	5	10	20	
0.0001	32	61.1	76.9	82.6	88.8	30.8

Table 6: Performance of sgd optimizer with best learning rate and batch size

Appendix C

Exploration of Hyperparameters for mobilenet_v3_small architecture without Augmentation

Lr	CMC				mAP (%)
	Ranks (%)				
	1	5	10	20	
0.0001	88.4	95.1	97.1	98.4	61.2
0.0003	83.4	92.6	94.9	96.8	51.3
0.0005	78.3	89.6	93.6	96.6	44.0
0.0007	74.6	86.2	90.7	94.4	39.4

Table 7: Performance of different learning rates

Lr	Batch Size	CMC				mAP (%)
		Ranks (%)				
		1	5	10	20	
0.0001	16	87.4	93.9	96.4	98.0	55.9
	32	88.2	94.8	96.8	98.0	60.2
	64	88.4	94.8	97.1	98.6	61.0
	128	87.6	95.1	97.0	98.3	60.8

Table 8: Performance of different batch sizes with best learning rate

Lr	Batch Size	CMC				mAP (%)
		Ranks (%)				
		1	5	10	20	
0.0001	64	54.5	72.8	80.0	86.2	26.2

Table 9: Performance of sgd optimizer with best learning rate and batch size