Vehicle Re-Identification using CNN-based models

Gianluca Sanfilippo 1943663

Computer Vision – Eng. C.S. – Sapienza University of Rome

Outline

This project presentation is divided as follows:

first we briefly depict the Vehicle Re-Identification field, its challenges and its importance in modern days applications;

then, are introduced the most recent approaches and techniques to arrive to the state of the art systems and solutions adopted in the field;

now, the solution developed and proposed is described in all its details, from the method, to the used datasets, to the evaluation criteria and the model enhancements put in place;

at the end, the conclusions, the observations annotated during the development of the project, and considerations on the performance of the proposed solution.

1 The Problem of Vehicle Re-Identification

Vehicle Re-Identification (VeRI) is one of the core components of Urban Surveillance and is directly connected to the development of Intelligent Transportation Systems such as Autonomous Vehicles.
Despite this technology is quite new, in a few years it has gained much attention and it has seen vast improvements since its importance in modern scenarios is much considered by governments and Tech industries.
In fact, Vehicle Re-Id involves recognizing and matching vehicles across multiple cameras or views. This allows, for example, matching car models and in general classes of vehicles across a network of cameras in a city.

As it could seem an easy task in modern A.I., with Neural Network architectures, especially Convolutional Neural Networks (CNN), recognizing and matching vehicles across image frames provided by camera views is not a simple task.
In fact, many outer factors obstacle the process of feature recognition in a VeRI architecture: illumination and environmental variations, angle of target, occlusion and partial visibility. These are just a few of such factors which could easily deceive the recognition algorithm to be precise and accurate in its task.

All the aforementioned problems require researchers to optimize their models to be highly robust in order to maximize the precision of their systems.

2 State of the Art

During the past few years, many approaches and techniques have emerged in this area, here are depicted some of the most relevant:

1. **Deep Learning approaches**

   - **Convolutional Neural Networks (CNNs):**
     CNN are the backbone of most recent ReID methods. Trained for feature extraction and matching using large datasets of vehicle images (e.g. ResNet is a common architecture).

   - **Siamese Networks:**
     This approach implements twin networks to learn similarity between two images. The network is trained to minimize the distance between matching vehicles and maximize the distance between non-matching ones.

   - **Triplet Networks:**
     Similar to the previous one but extended. Takes as input three images: anchor image, positive image (same vehicle as anchor), negative image (different vehicle). The model learns to reduce the distance between the matching vehicles images while increasing the non matching ones distance.

2. **Attention Mechanisms**
   This mechanisms help the model to focus on important parts of the vehicle image, reducing the less useful features to be considered and therefore making the model more robust.
   In VeRI, this can be crucial, especially when vehicles are partially occluded.

   - **Spatial Attention:**
     Allows the model to focus on different parts of the vehicle, such as the wheels, logo, or windows, depending on what is most discriminative for the vehicle identity.

   - **Channel Attention:**
     Channel Attention helps the model to weigh the importance of different feature channels. For example, certain channels in the network may capture color information, while others capture texture or shape information. Thus, each channel may be useful in distinguishing vehicles.

3. **Cross-Domain Learning**

4. **Feature Fusion:**
   Combines different types of features, such as appearance-based ones (texture, color) and motion-based ones (speed, trajectory), to enhance VeRI performance. Often implemented with radar (LiDAR in autonomous vehicles).

5. **Metric Learning**
   In order to be effective in VeRI, it is of primary importance to learn the right distance metric to measure similarity between vehicle images.

   - **Contrastive Loss:**
     Often used with Siamese networks, minimizes the distance between matched pairs and maximizes the distance between mismatched pairs.

   - **Triplet Loss:**
     Usually used in triplet networks, this loss function aims to learn a discriminative metric that makes the distance between a vehicle and its matching image smaller, while increasing the distance to non-matching images.

6. **Generative Models**
   Generative Adversarial Networks (GAN) have been used to generate synthetic vehicle images to augment training data (data augmentation). Also implements GANs for generating domain specific transformations, for example, simulating images in different weather conditions and viewpoints to make the model more robust.

7. **End-to-End Solutions**
   Recently, the trend has leaned toward learning systems that take vehicle images from different cameras and output a unique identity for each vehicle. These systems are optimized to perform all the necessary tasks (feature extraction, matching, ranking) in one unified architecture.
   Often, End-to-End systems incorporate a mix of CNNs, attention mechanisms and metric learning.

3 The Solution Adopted

The solution adopted in the present project can be considered an End-to-End approach.
In the following sections are described all the characteristics of the VeRI system we have developed and, at the end of such depiction, will be examined the experimental results.

3.1 Implemented Datasets

In this project, we have implemented two datasets, one very large VRU and one smaller, but largely recognized and utilized in the field, VeRi-776.
Both datasets are available at the link in the README file of the project repository.

3.2 Proposed Method

The developed model loads a dataset chosen by the user and trains two CNN-based ReID backbones (ResNet-18 and ResNet-50) on such dataset, then learns discriminative vehicle embeddings using Cross-Entropy Loss (identity classification) and Batch-Hard Triplet Loss (metric learning).
ResNet-18 is much more lightweight with respect to ResNet-50. In fact, this last one is much deeper and, therefore, it should be able to better extract and match vehicle features.
A key point of the infrastructure is the P-K sampling, required for batch hard triplet loss. It takes batches of P identities with K images per identity.
Batch-Hard triplet loss, as explained before, is highly effective in learning discriminative embeddings (features) in each picture by selecting the hardest positive and hardest negative matching within each batch of sampled images, which ensures robust separation between visually similar vehicles.

To ensure reproducibility and a fair comparison we have set a fixed seed and also, if available, the system is defined to support GPU acceleration.

So far, we have implemented a complete Vehicle Re-Identification pipeline using CNN-based feature extractors, enhancing the ResNet architecture to a Strong Baseline ReID architecture with BNNeck (modern VeRI systems use this to separate metric and classification losses into two different feature spaces to stabilize training and improve metric learning, obtained by adding a batch normalization layer after global pooling layer) and, as depicted in the following section, we are able to compare performances in different settings by standard metrics.

The model is also enforced with Data Augmentation during training which eventually applies the following transformations:

- *Random Horizontal Flip* to simulate viewpoint changes
- *Color Jitter* to simulate illumination variations
- *Resize and Normalization*

This improves robustness against the already mentioned problems which can affect the images.

3.3 Enhancements - Attention Mechanism

To improve the model, using modern approaches, we have also implemented both Channel and Spatial Attention, already mentioned in the previous sections.
In the project it is disabled by default, it can be turned on with the *–attention* command-line flag.
This modification integrates into the existing architecture pipeline as depicted in the following images.

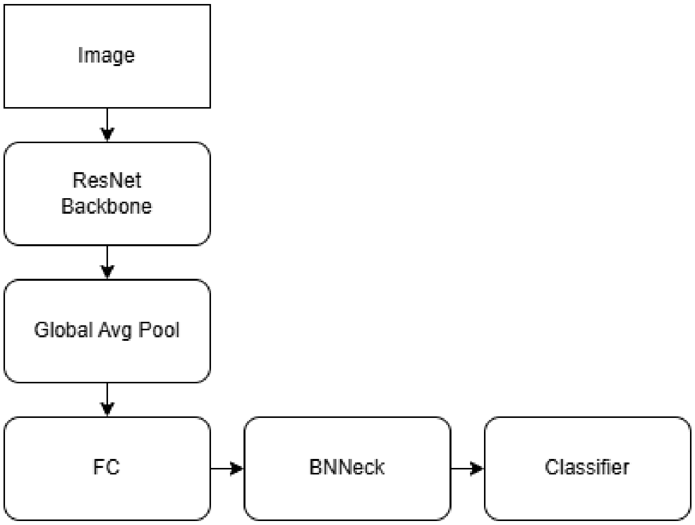

Figure 1: Architecture without attention mechanism

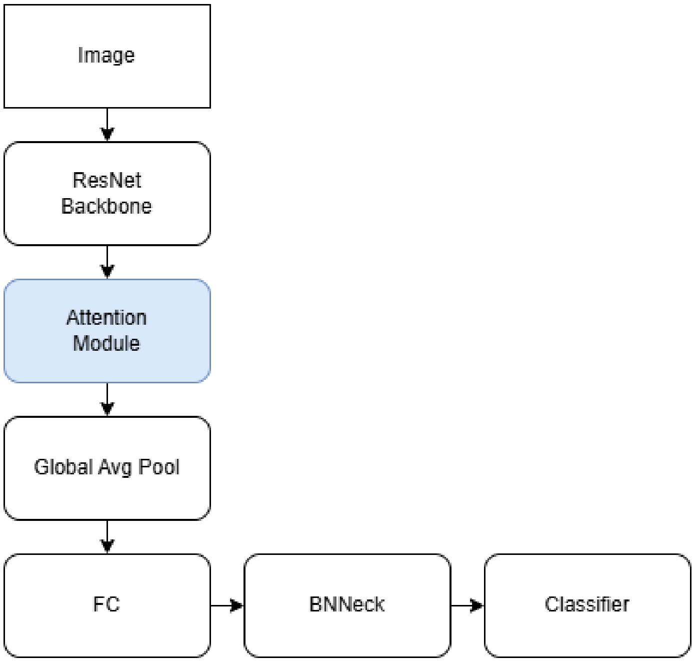

Figure 2: Architecture with attention mechanism

Attention mechanism is introduced after the CNN backbone. Specifically, a Convolutional Block Attention Mechanism (CBAM). Attention makes features more discriminative suppressing background noise, since Channel attention focuses on informative feature maps, for example vehicle color and lights, whereas Spatial attention focuses on vehicle regions like body, license plates and logos. By textbook, this mechanism usually guarantees 1 to 3 percent of improvement in precision, which is significant.

3.4 Model Evaluation

For each epoch, whose total number is defined in the *globals.py* file, is computed the average loss, which gives the necessary information to compute precision and accuracy in evaluation and to make comparisons for the ablation study.

The trained CNNs are saved on the disk to allow the user to evaluate them by testing, which is implemented using two different distance metrics: Cosine distance and Euclidean distance.
Cosine distance computes directional similarity and it is scale invariant.
Euclidean distance measures the absolute distance in the embedding space of the feature vector.
At the end of the training, the system configured in *Extended training evaluation*

mode is able to save plots of the performance metrics, measured at each step of the training (each epoch), which show the mean average precision (mAP) and accuracy (Rank-1) extracted by the CMC (Cumulative Matching Characteristics) curves.
The plots are based on cosine distance and it is used only for mapping the training process. The final comparison is done by running the program in evaluation mode, which gives us the metrics for both cosine and euclidean distances.

The evaluation process is implemented on different settings of the model proposed. In fact, we compared each characteristic of the system in different scenarios:
Attention mechanism enabled or not;
different backbone CNNs;
training with **different number of epochs**;
different datasets and relative partitions in train, query and gallery sets.

Besides this, as mentioned before, during evaluation we take in consideration both training process information on loss with plots on accuracy and precision, and pure testing mAP and Rank-1.

First, for simplicity, query and gallery sets of the test set were kept merged during evaluation, which falsely lead the Rank-1 accuracy to be equal to 1. In fact, given a certain image in the query set, it will find the same image also in the gallery set which will produce a perfect match (distance of feature vectors equals to zero for such image).
Furthermore, with merged sets for query and gallery, euclidean distances among feature vectors cannot be computed due to the amount of memory required to compute a full-set euclidean distance.
For clarity, here are shown metrics of the models trained without attention and evaluated with cosine distance and with **no split test set**, where is visible the data about Rank-1 being exactly equal to 1.

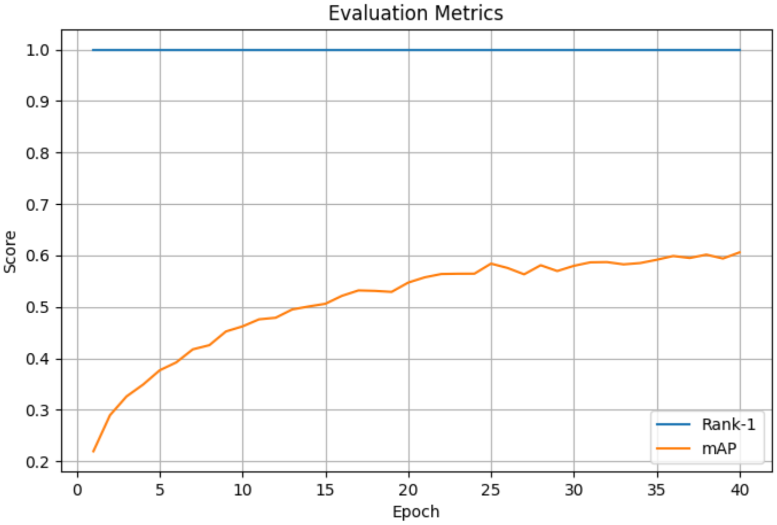

Figure 3: Metrics of ResNet-18 on VeRi-776 with Query == Gallery

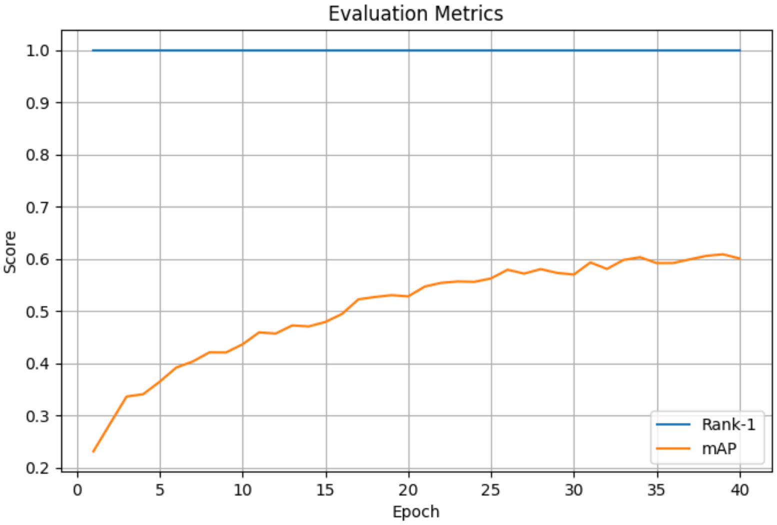

Figure 4: Metrics of ResNet-50 on VeRi-776 with Query == Gallery

4 Results and Conclusions

The experiments made to formalize our ablation study have been the following.

BackBone	Epochs	Dataset	Attention	Distance Metric	Results
ResNet-18	20	VRU		Cosine	mAP: 0.7757 Rank-1: 0.9592
ResNet-18	20	VRU		Euclidean	mAP: 0.7757 Rank-1: 0.9592
ResNet-18	20	VRU	Yes	Cosine	mAP: 0.7890 Rank-1: 0.9583
ResNet-18	20	VRU	Yes	Euclidean	mAP: 0.7890 Rank-1: 0.9583
ResNet-50	20	VRU		Cosine	mAP: 0.8167 Rank-1: 0.9742
ResNet-50	20	VRU		Euclidean	mAP: 0.8167 Rank-1: 0.9742
ResNet-50	20	VRU	Yes	Cosine	mAP: 0.8206 Rank-1: 0.9733
ResNet-50	20	VRU	Yes	Euclidean	mAP: 0.8206 Rank-1: 0.9733
ResNet-18	20 + 5	VRU		Cosine	mAP: 0.8033 Rank-1: 0.9683
ResNet-18	20 + 5	VRU		Euclidean	mAP: 0.8033 Rank-1: 0.9683
ResNet-18	20 + 5	VRU	Yes	Cosine	mAP: 0.8133 Rank-1: 0.9708
ResNet-18	20 + 5	VRU	Yes	Euclidean	mAP: 0.8133 Rank-1: 0.9708
ResNet-50	20 + 5	VRU		Cosine	mAP: 0.8494 Rank-1: 0.9842
ResNet-50	20 + 5	VRU		Euclidean	mAP: 0.8494 Rank-1: 0.9842
ResNet-50	20 + 5	VRU	Yes	Cosine	mAP: 0.8504 Rank-1: 0.9800
ResNet-50	20 + 5	VRU	Yes	Euclidean	mAP: 0.8504 Rank-1: 0.9800

Table 1: Evaluation of training on VRU dataset

BackBone	**Epochs**	**Dataset**	**Attention**	**Distance Metric**	**Results**
ResNet-18	40	VeRi-776		Cosine	mAP: 0.6085 Rank-1: 0.995
ResNet-18	40	VeRi-776		Euclidean	mAP: 0.6085 Rank-1: 0.995
ResNet-18	40	VeRi-776	Yes	Cosine	mAP: 0.5824 Rank-1: 0.995
ResNet-18	40	VeRi-776	Yes	Euclidean	mAP: 0.5824 Rank-1: 0.995
ResNet-50	40	VeRi-776		Cosine	mAP: 0.5866 Rank-1: 0.98
ResNet-50	40	VeRi-776		Euclidean	mAP: 0.5866 Rank-1: 0.98
ResNet-50	40	VeRi-776	Yes	Cosine	mAP: 0.5935 Rank-1: 0.98
ResNet-50	40	VeRi-776	Yes	Euclidean	mAP: 0.5935 Rank-1: 0.98

Table 2: Evaluation of training on VeRi-776 dataset

We can observe that the values concerning mAP and Rank-1 are equivalent for both cosine and Euclidean distances.
Since feature embeddings are L2-normalized during both training and evaluation, Cosine and Euclidean distances become monotonically equivalent and this explains why both mentioned performance metrics yield identical values.
Using L2-normalization is a standard practice in VeRI. In fact, it is implemented in real TransReID models and strong VeRI baselines.
Furthermore, the fact that we get such metrics to be equal means that the model is behaving correctly, learning the right metrics.

We also performed an evaluation on the VeRi-776 dataset using the models trained with the larger VRU dataset.

BackBone	**Epochs**	**Dataset**	**Attention**	**Evaluation**	**Results**
ResNet-18	20 + 5	VRU		VeRi-776	mAP: 0.2420 Rank-1: 0.9800
ResNet-18	20 + 5	VRU	Yes	VeRi-776	mAP: 0.2310 Rank-1: 0.9550
ResNet-50	20 + 5	VRU		VeRi-776	mAP: 0.2366 Rank-1: 0.9550
ResNet-50	20 + 5	VRU	Yes	VeRi-776	mAP: 0.2343 Rank-1: 0.9550

Table 3: Evaluation on VeRi-776 of training on VRU dataset

In the following pictures are reported all the training processes and their relative evaluation plots (only when extended training evaluation is enabled).

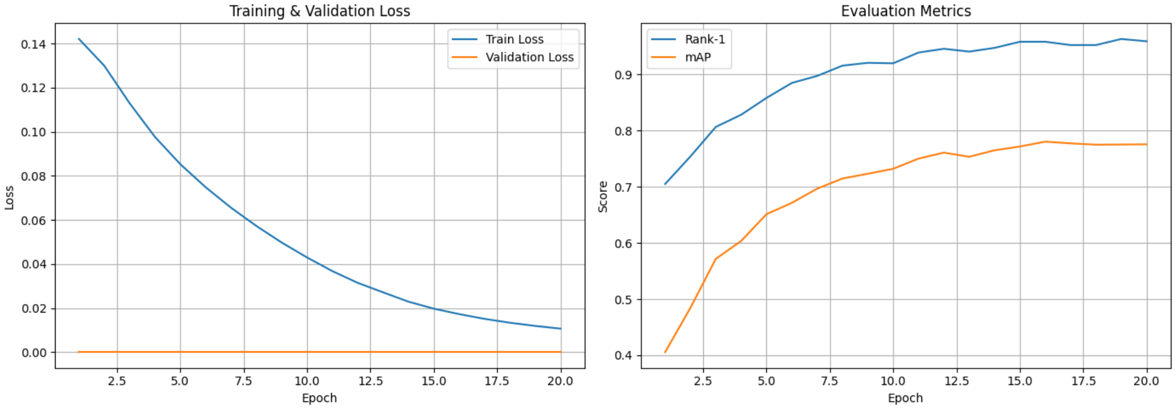

Figure 5: Training extended evaluation, VRU, ResNet-18, 20 epochs, no attention

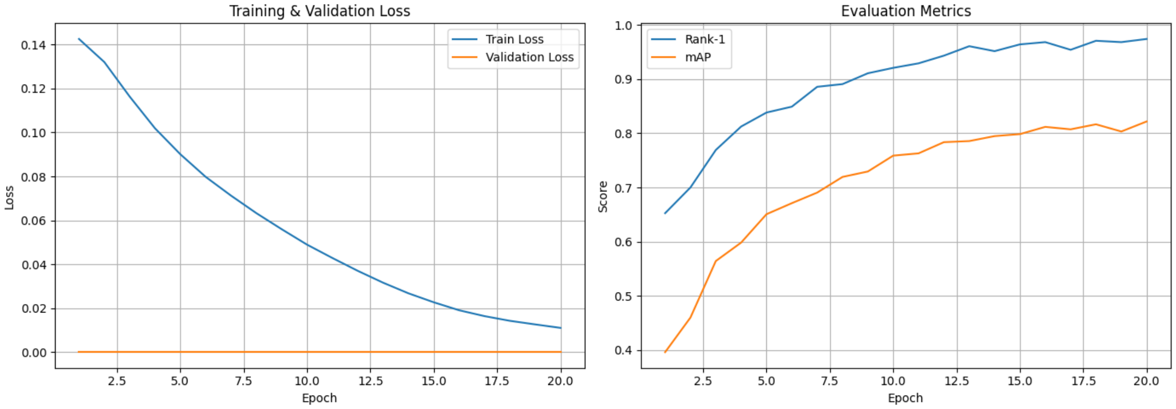

Figure 6: Training extended evaluation, VRU, ResNet-50, 20 epochs, no attention

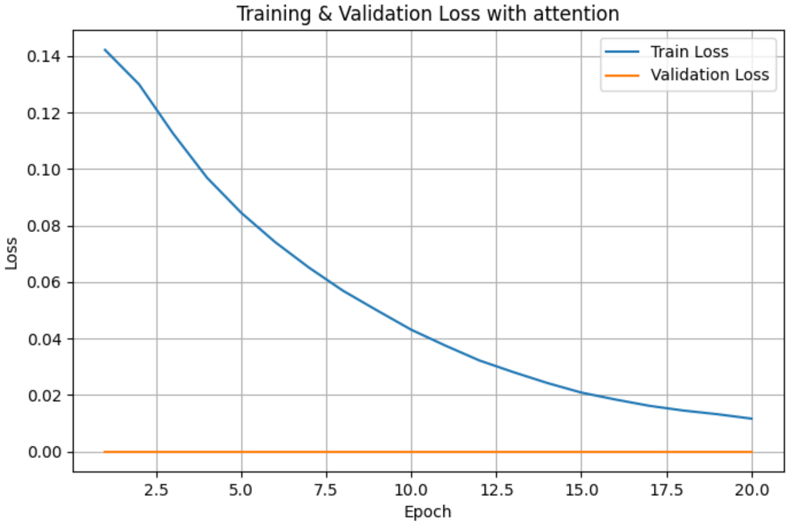

Figure 7: Training, VRU, ResNet-18, 20 epochs, with attention

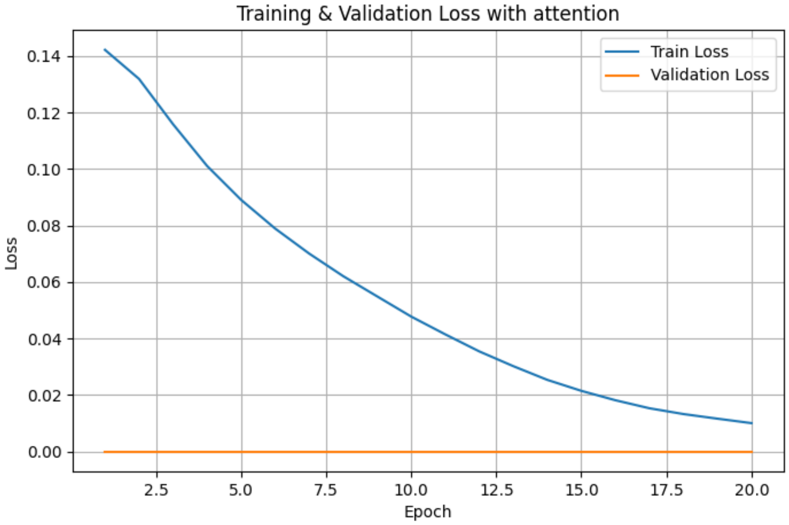

Figure 8: Training, VRU, ResNet-50, 20 epochs, with attention

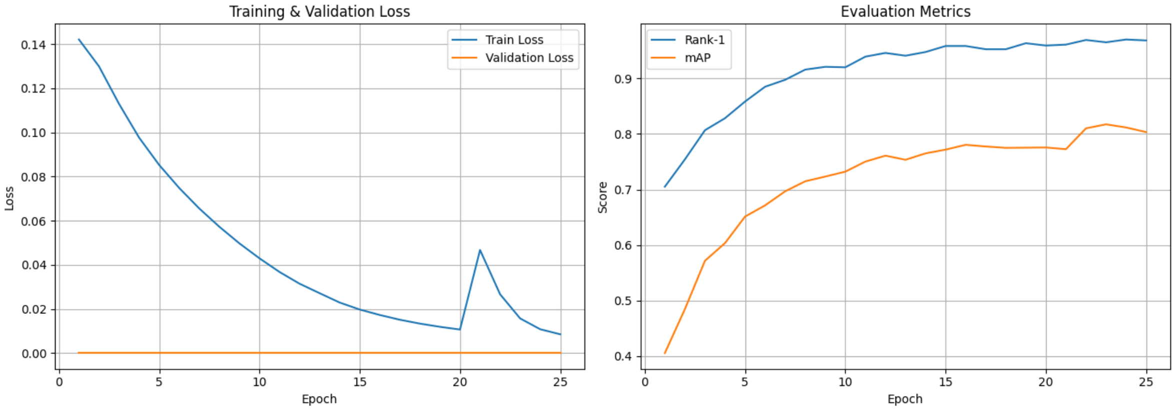

Figure 9: Training extended evaluation, VRU, ResNet-18, 20 + 5 new epochs, no attention

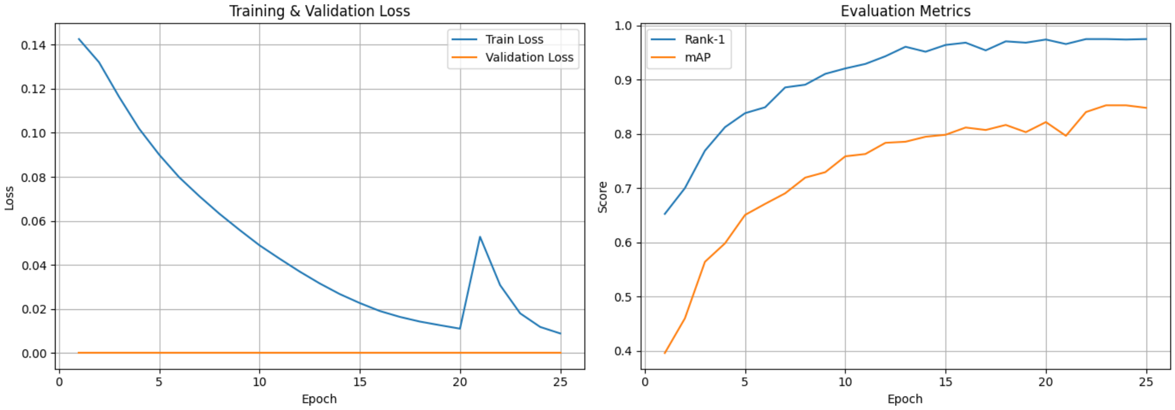

Figure 10: Training extended evaluation, VRU, ResNet-50, 20 + 5 new epochs, no attention

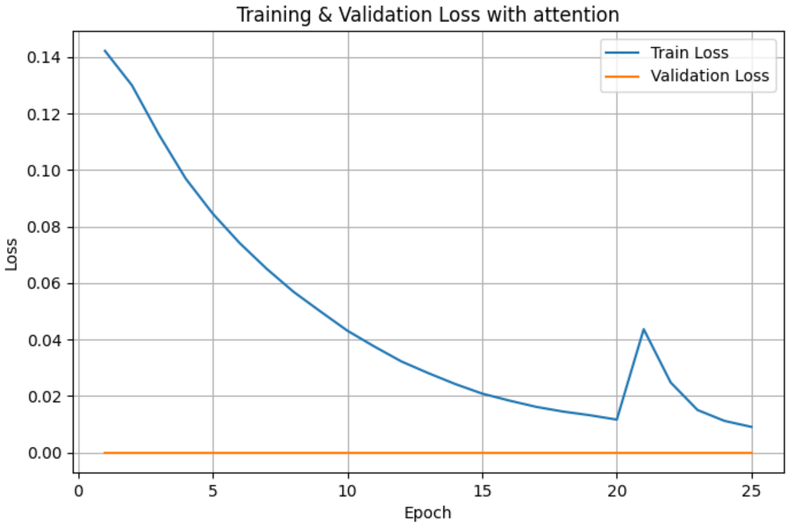

Figure 11: Training, VRU, ResNet-18, 20 + 5 new epochs, with attention

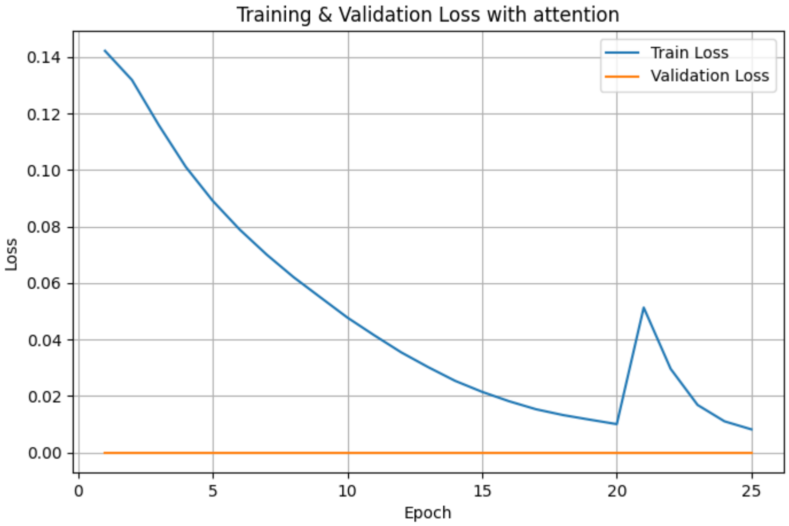

Figure 12: Training, VRU, ResNet-50, 20 + 5 new epochs, with attention

As per standard, we used mAP and Rank-1 as evaluation metrics.
In VeRI, the task is **retrieval**, not classification. More precisely, given a query image, extract its feature vector, compute distances between query and gallery images, sort gallery images by increasing distance.
In this scenario, Rank-k measures the probability that a correct match falls within the top-k sorted gallery images.
Thus, **Rank-1 means how often the first image sorted is of the same vehicle as the one of the query**.

However, it is important to consider as more relevant the value of **mAP**, since **it measures how well the gallery has been sorted with respect to the multiple correct matches**.

The following image summarize the collected statistics in a single plot to better visualize the effective numbers just analyzed obtained in the present experiment.

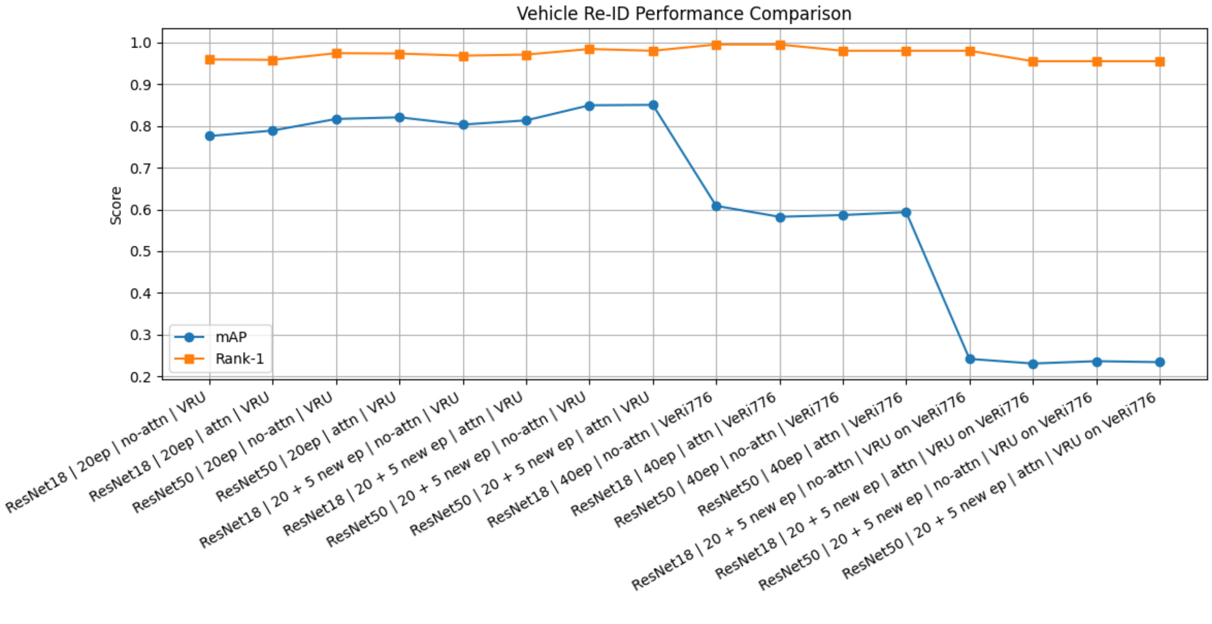

Figure 13: Comparison of results on different models and settings

To schematize the results obtained we divide the data we are analyzing in branches:

- **Training on VeRi-776 dataset**
  This dataset has 576 training identities. It is quite small to perform a satisfying training of the VeRI models.
  The results confirm it even with 40 epochs of training, since on average we scored about 99% in Rank-1 but just 60% in mAP, with minimum negative fluctuations when attention mechanism is enabled.
  Furthermore, the smaller Neural Network ResNet-18 outperformed the deeper ResNet-50.

  This indicates that the dataset is too small to correctly let the model learn discriminative features, and the more complex the model, the more noise is learned and overfitting incurs.

  Since we used for both networks the same parameters of EPOCHS, LR and dataset, this explains the lack of performance, which is almost comparable to the first VeRI systems introduced in 2019.

- **Training on VRU dataset**
  This dataset has 7086 training identities. Much larger than the previous one.
  Generally, ResNet-50 outperformed the smaller ResNet-18.
  With 20 epochs of training and no attention mechanism, for ResNet-18 and ResNet-50 respectively, we reached good results of about 96% - 97% of Rank-1 and 78% - 82% of mAP.
  When the attention mechanism is enabled, the fluctuation is slightly positive.

  Then we fine tuned the model with 5 additional epochs of training, totaling 25 epochs, **resetting the optimizer parameters** (Adam optimizer). This implied an immediate temporary negative spike in performance but increased the ultimate results since it allowed the model to uneven the embedding learned so far which may cause overfitting.
  The metrics reached the 80% - 84% in mAP without attention to top score 81% - 85% respectively for ResNet-18 and ResNet-50, again with an enhancement of 1-2% thanks to Channel and Spatial attention layer.

  Further improvements can be implemented by extending the number of epochs (also without resetting the optimizer parameters).

- **Training on VRU and evaluate on VeRi-776 dataset**
  The results revealed how the model is largely dependent on the training dataset.
  We scores a Rank-1 of about 96-98% with a mAP of just 23-24%.
  This enlightens how a cross-dataset evaluation does not generalize without additional training on the unseen image queries of the other dataset under evaluation.
  The best practice, in this case, would be to fine tune on VeRi-776 train split and successively evaluate on its respective test split.

Another aspect to consider for our study is the **computational efficiency** against the **feature quality** of the models we implemented.

If we consider the **training process**, ResNet-18 is by far the most efficient compared to the deeper one; with training evaluation enabled, ResNet-18 with 20 epochs and no attention took 16 hours to train while ResNet-50 took 38 hours.
Even though this is not very common, this can be particularly important when dealing with low performances devices and the model should undergo online training fine tuning.
On the other hand, if we consider the **evaluation process**, which is considerably faster than training, the differences in efficiency for the two Networks are not that evident, taking almost 3 seconds per identity for ResNet-18 and 5 seconds per identity for the deeper ResNet-50.

So, in conclusion, it is believed that the precision reached by ResNet-50 with attention enabled justifies the small additional latency required by the deeper network.

Modern top approaches have reached mAP values of maximum 93% just in the past two to three years.
Since, as just mentioned, we reached a top score of 85% mAP, this makes our model a great starting point for future enhancements and optimizations.

References

1. Course slides
2. V2ReID Vision-Outlooker-Based Vehicle Re-Identification
3. “Bag of Tricks for Re-ID” (Luo et al.)
4. AI generative models