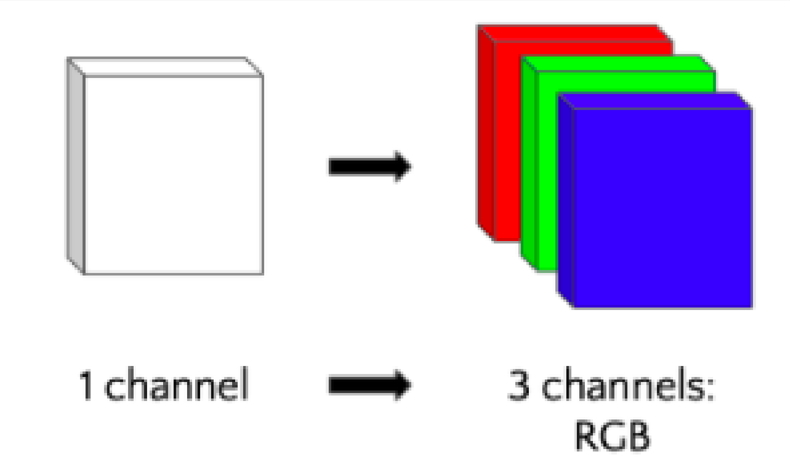
IMAGE RE-COLORING
USING
GENERATIVE ADVERSARIAL NETWORKS

Final Project for the 'Neural Networks and Deep Learning' Course

Filippo Canderle, Alberto Merotto, Cecilia Rossi

INTRODUCTION AND RELATED WORKS

The performed task was **IMAGE RE-COLORING** from Black and White (Grayscale) Images.



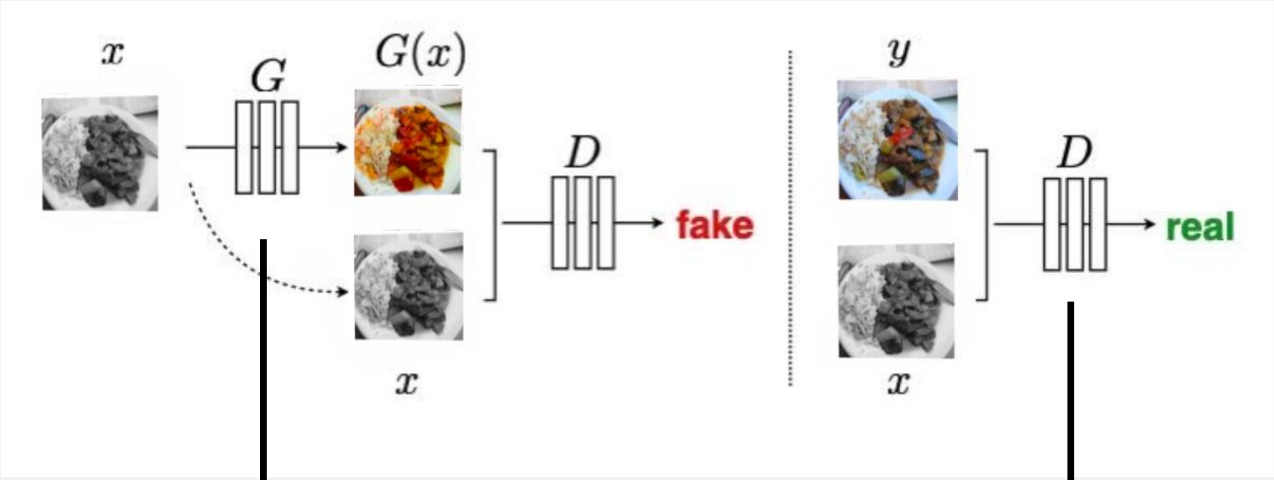
INTRODUCTION AND RELATED WORKS

Our basis was the work from Isola, Zhu et al. (2016), 'Image-to-Image Translation with Conditional Adversarial Networks'.

Other related works:
- CycleGAN
- StarGAN
- UNIT

The performed task was **IMAGE RE-COLORING** from Black and White (Grayscale) Images.

We tried to solve this problem relying on a modern approach, based on **(Conditional) Generative Adversarial Network (GAN and cGAN):**



Generator

Generates the **new re-colored images** from the grayscale ones, trying to trick the discriminator mapping from a latent distribution to the target one.

Discriminator

Aims to **classifies the images**, trying to discriminate if they are real (comes from a true distribution) or are generated.

APPROACH

=

in literature is known as:

PIX2PIX

⬇

Combination of
Regression based approaches (MAE) and
Implicit Density Estimation (BCE)

1) Mean Absolute Error (MAE) or L1-Loss

aims to reconstruct pixel wise the image, giving as target the original image.

$$\mathcal{J}(\theta) = \mathbb{E}_{x,y\sim p_{data}}[||f_\theta(x) - y||]$$

→ not assess joint statistics of the result, and therefore do not measure the very structure that structured losses aim to capture.

2) Implicit Density Estimation as Binary Cross Entropy (BCE)

the objective of a Conditional GAN is solving a min-max problem of the following loss:

$$\mathcal{J}(G, D) = \left[\mathbb{E}_{x\sim p_{data}}[\log D(x)] + \mathbb{E}_{z\sim p_z}[1 - \log D(G(z))]\right]$$

where, G tries to minimize this objective against an adversarial D that tries to maximize it, i.e.

$$G^* = argmin_G max_D \mathcal{J}(G, D)$$

APPROACH

1) Mean Absolute Error (MAE) or L1-Loss

aims to reconstruct pixel wise the image, giving as target the original image.

Even though GANs have initially been proposed using BINARY-CROSS ENTROPY (BCE), it has been shown that is not very stable, and that practically they work better with regression losses, as they provide a nicer gradient.

For this reason, we used MAE also for the classification error made by the discriminator:

$$\mathcal{J}(G,D) = \left[\mathbb{E}_{x,y\sim p_{data}}[\mathrm{L1}(G(x),y)] + \mathbb{E}_{x,y\sim p_{data}}[\log D(x,y)] + \mathbb{E}_{x,y\sim p_{data}}[1-\log D(1-D(x,G(x)))]\right]$$

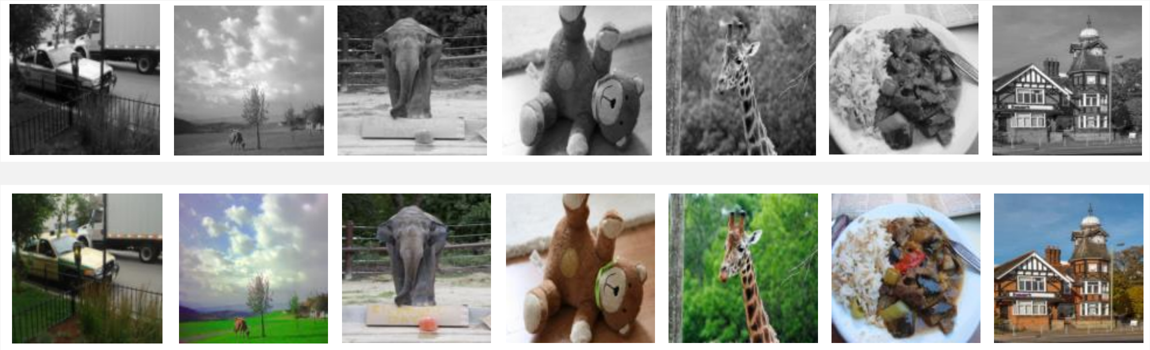
Implicit Density Estimation (BCE)

where, G tries to minimize this objective against an adversarial D that tries to maximize it, i.e.

$$G^* = argmin_G max_D \mathcal{J}(G,D)$$

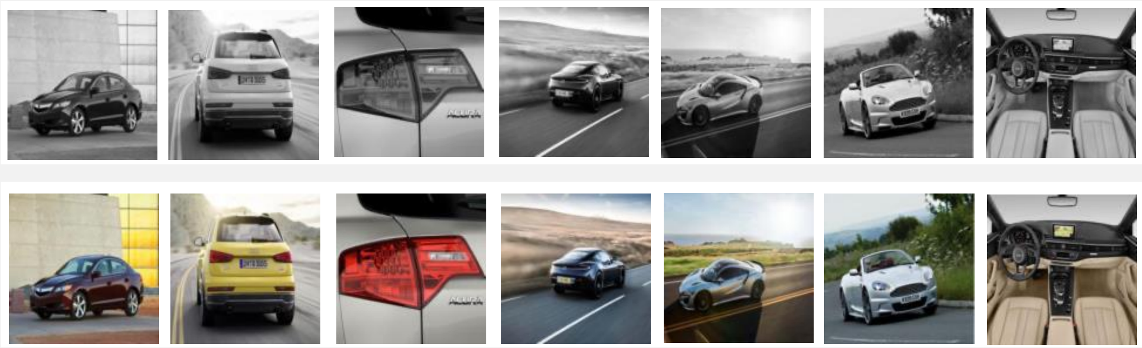
DATASETS

COCO DATASET



: a large-scale image recognition dataset for object detection, segmentation, and captioning tasks. We used it as a more Generic Dataset, on which we based our initial analysis.

CARS DATASET



: a 60k image dataset, obtained from Kaggle. We chose it since we wanted to compare the results obtained on representation of a simple object, that could be designed in slightly different ways and with different color possibilities.

DATASET PRE-PROCESSING

RESCALING

Firstly, we resized the images, bringing them to size **128x128 pixels.** In this way:

- We reduced the **computational cost**
- And maintained a **good quality** of the images, preserving visual features

FLIP

The, we randomly flipped the images, both **horizontally** and **vertically**, to increase the variability of the data.

DATALOADERS

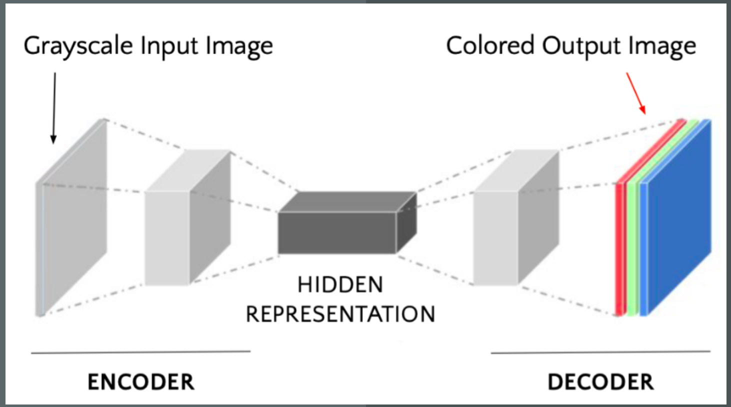
Finally, we organized both training and test data into **Python Dataloaders**, created in two different color spaces:

- RGB
- Lab

LEARNING FRAMEWORK
=
CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORK

as said before, it consists of 2 networks:

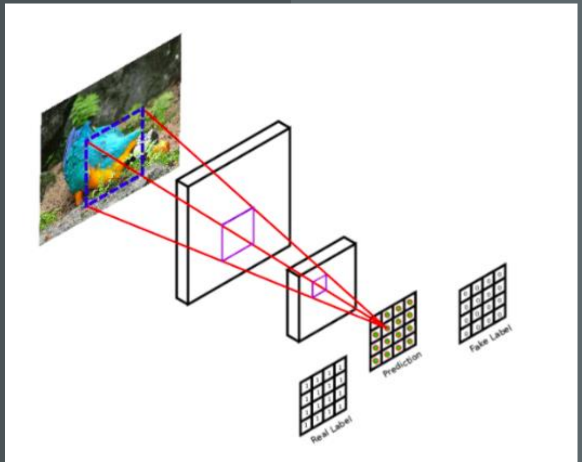
GENERATOR



We tested different popular architectures:

- **U-Net**
- **Autoencoder**
- **ResNet**

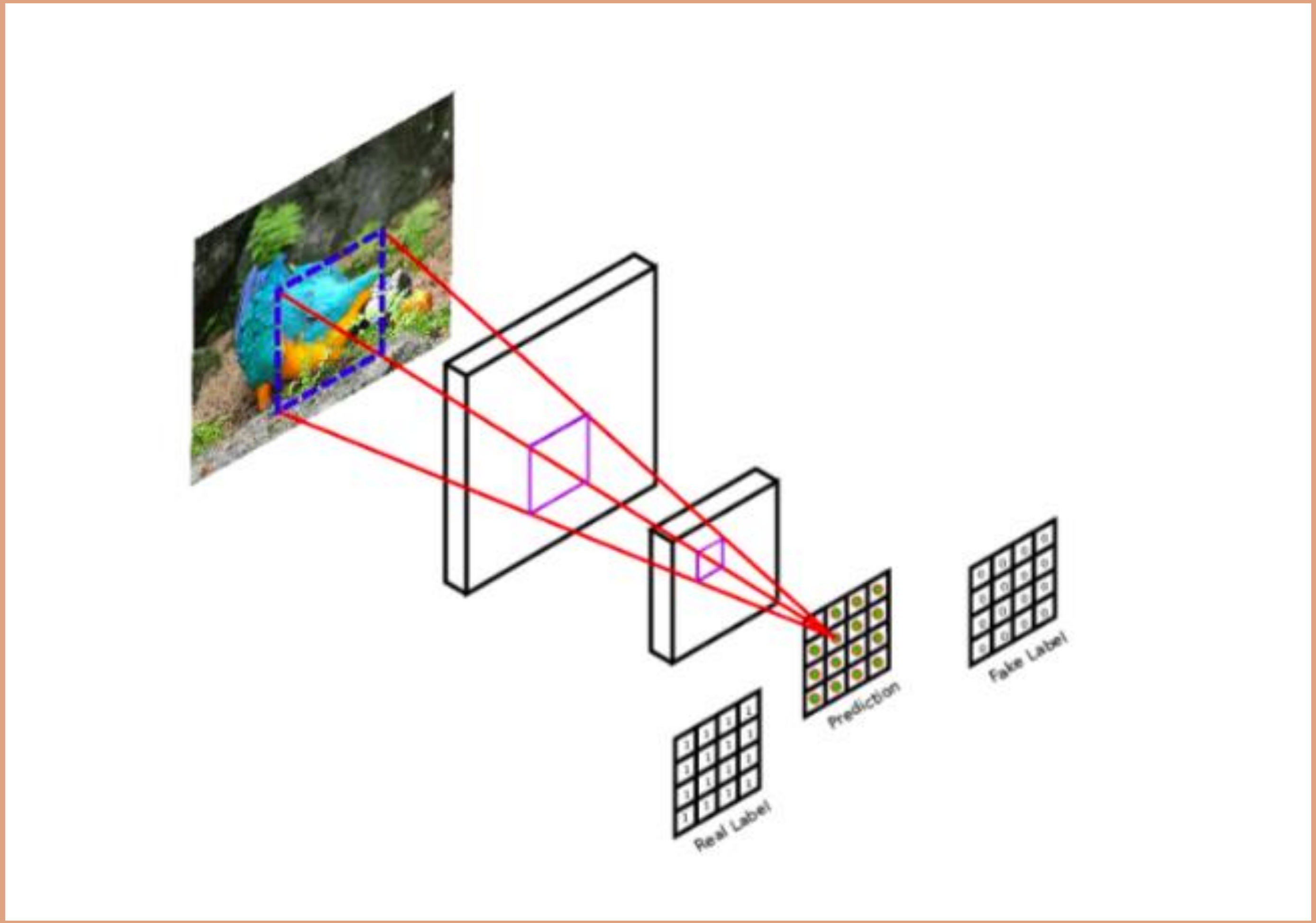
DISCRIMINATOR



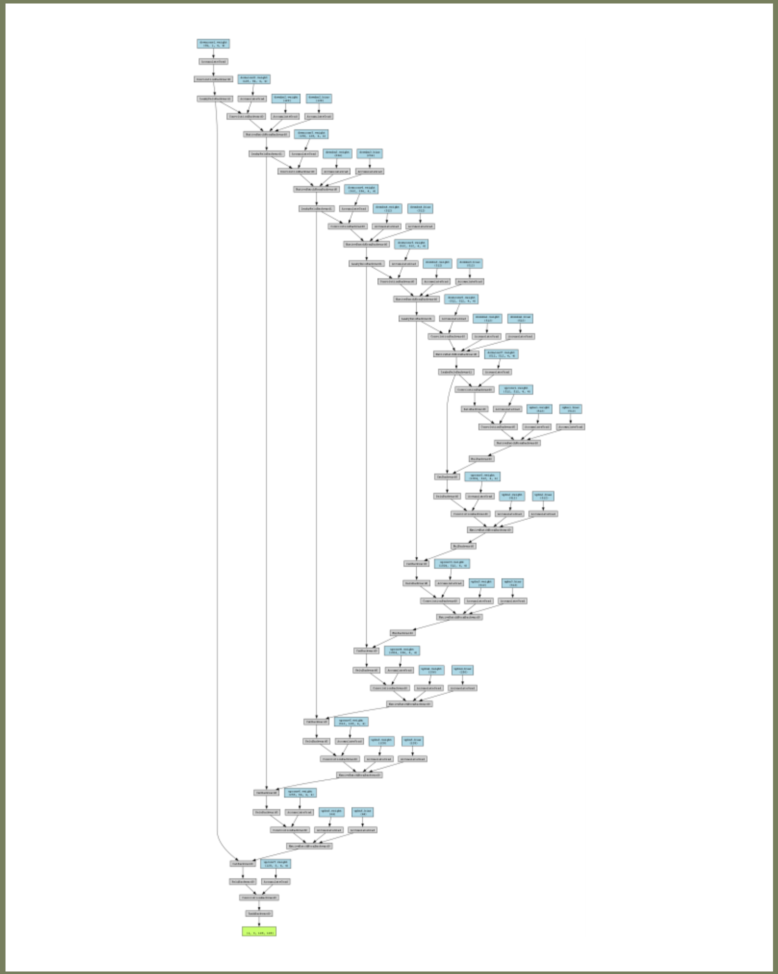
We used a fully convolutional neural network, called **Patch Discriminator**

DISCRIMINATOR

- uses a series of **Convolutional Layer** that, exploiting the fact hat convolution has a **local receptive field**, aims to classify the underneath patch of the image, returning a **14x14 patches matrix**.
- The loss is then computed comparing the output of the discriminator with a **matrix of labels**, in which all elements are :
  - 1. associated to real labels;
  - or 0. associated to fake images;



U-NET

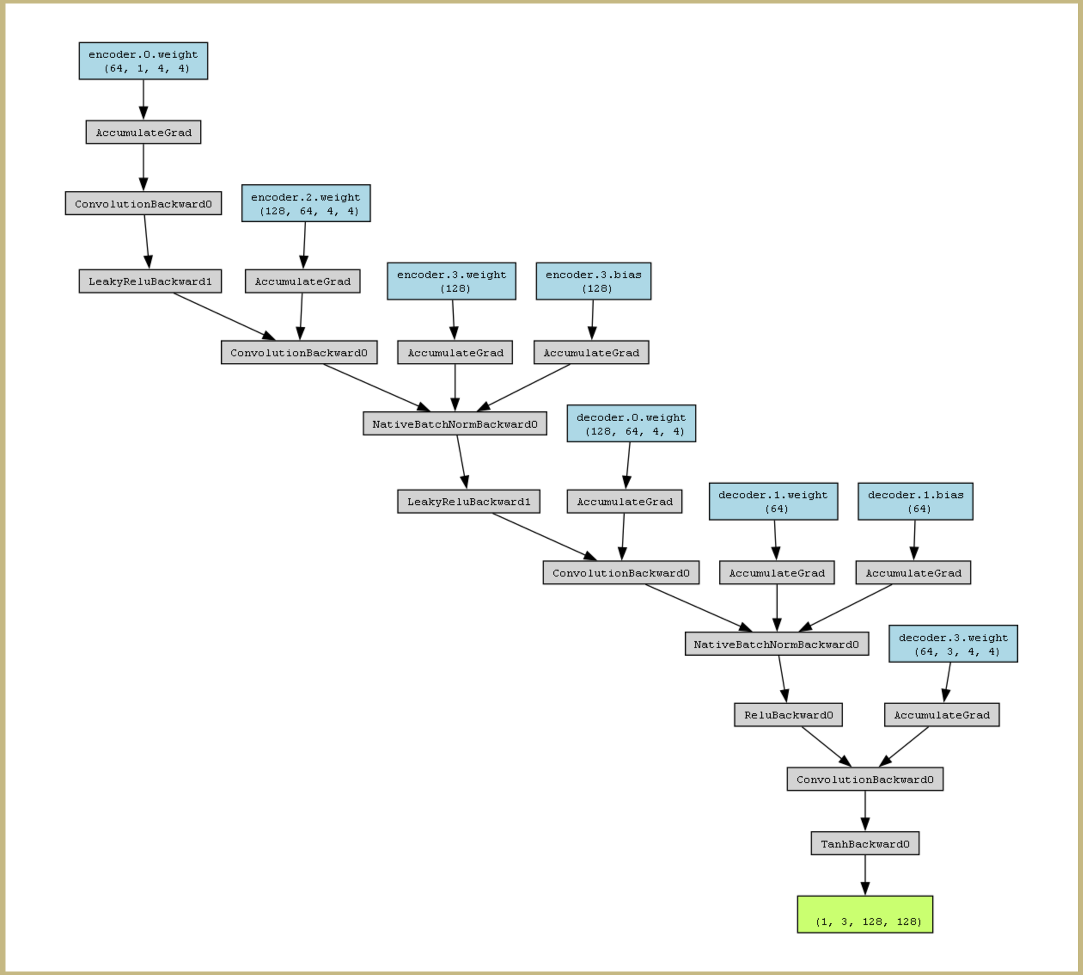


- **Encoder = 7 down-sampling layers;**
- **Decoder = 7 up-sampling layers;**
- Each layers consist of:
  - Linear Convolutional Layer,
  - Batch Normalization,
  - LeakyRelu (encoder) or Relu (decoder) element-wise activation layer.
- Skip connections
- Dropout Layers in the up-sampling
- Last Convolutional Layer + Tanh activation.

AUTOENCODER

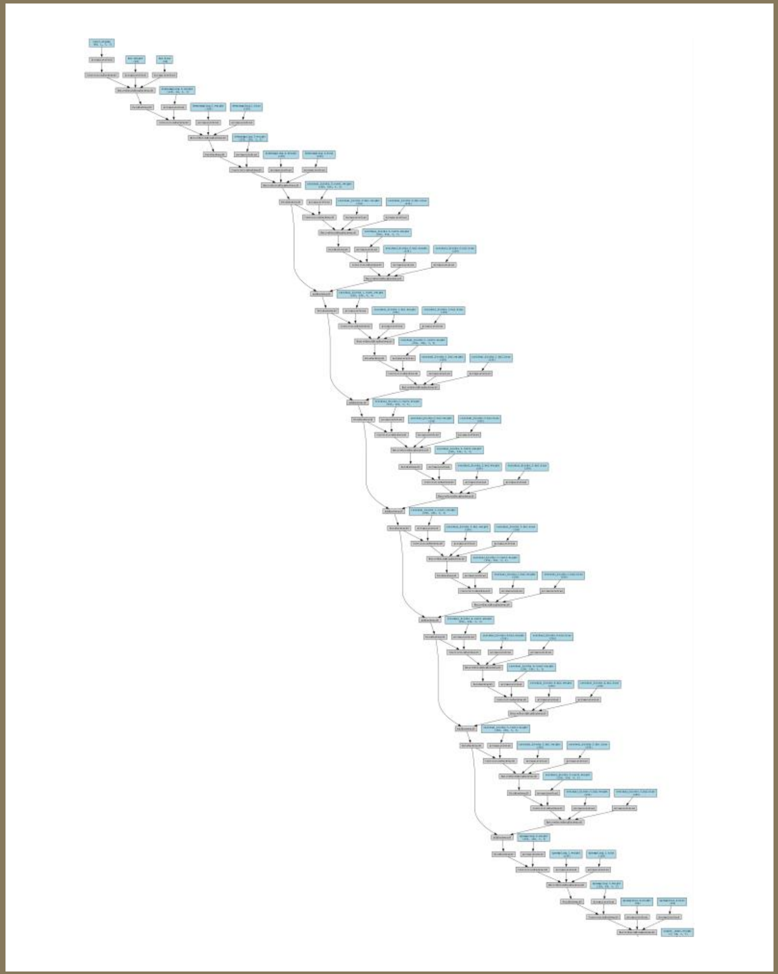
- Very similar to our principal model;
- Main differences:
  - **shallower architecture** (lower numbers of layers both in the down-sampling and up-sampling);
  - use of a **Max Pooling layer**, to effectively decrease spatial dimension;
  - **No Skip Connections;**

We wanted to see if a less deep network with the same encoder-decoder structure, but without skip connections, we would have achieved comparable results.



RESNET

- **Stressing even more the concept of Residual Connection**, in order to reduce the number of parameters.
- It consists of:
  - Initial 7x7 Convolutional Layer with Batch Normalization,
  - Down-sampling module;
  - **Residual Blocks**, to capture and propagate essential image details;
  - Up-sampling module;
  - Final Convolution and Tanh activation.



CONFIGURATIONS

(*) : a cGAN, with the indicated architecture as generator and, as a discriminator, a Patch CNN

U-Net (*) on : - **General** dataset, - with **10k and 20k** images - for **15, 30 and 45 epochs**	GENERAL RESULTS ON U-NET : comparison between training on **different amount of data** & **different epoch number**
U-Net on: - **Specific** Dataset - with **10k** images - for **15, 30 and 45 epochs**	COMPARISON with SPECIFIC DATASET: (w.r.t. different epoch number)
U-Net on: - **Specific** Dataset - with **10k LAB** images - for **45 epochs**	COMPARISON with LAB COLOR SPACE
Autoencoder on: - **Specific** Dataset - with **10k** images - for **45 epochs** **ResNet** on : - **Specific** Dataset - with **10k** images - for **45 epochs**	COMPARISON between different ARCHITECTURES

EVALUATION AND RESULTS

- We analysed the plots of the **training losses** of both generator and discriminator.
- In the **evaluation,** we computed the **same losses used during the training on the test set**.
- But, as the majority of "generative" tasks, is very complex to find adequate metrics for the evaluation, in fact:
  - MAE (as other regression based metrics) assume a single possible correct output for each pixel, so it's not an effective metric;
  - Density Based (as BCE) instead tends to give more importance to the structure of the image, instead of the colorization;

As a consequence of those considerations we relied on the **VISUAL EVALUATION** of the results on a **subset of the test-set.**

U
N
E
T

Trained on
10k images

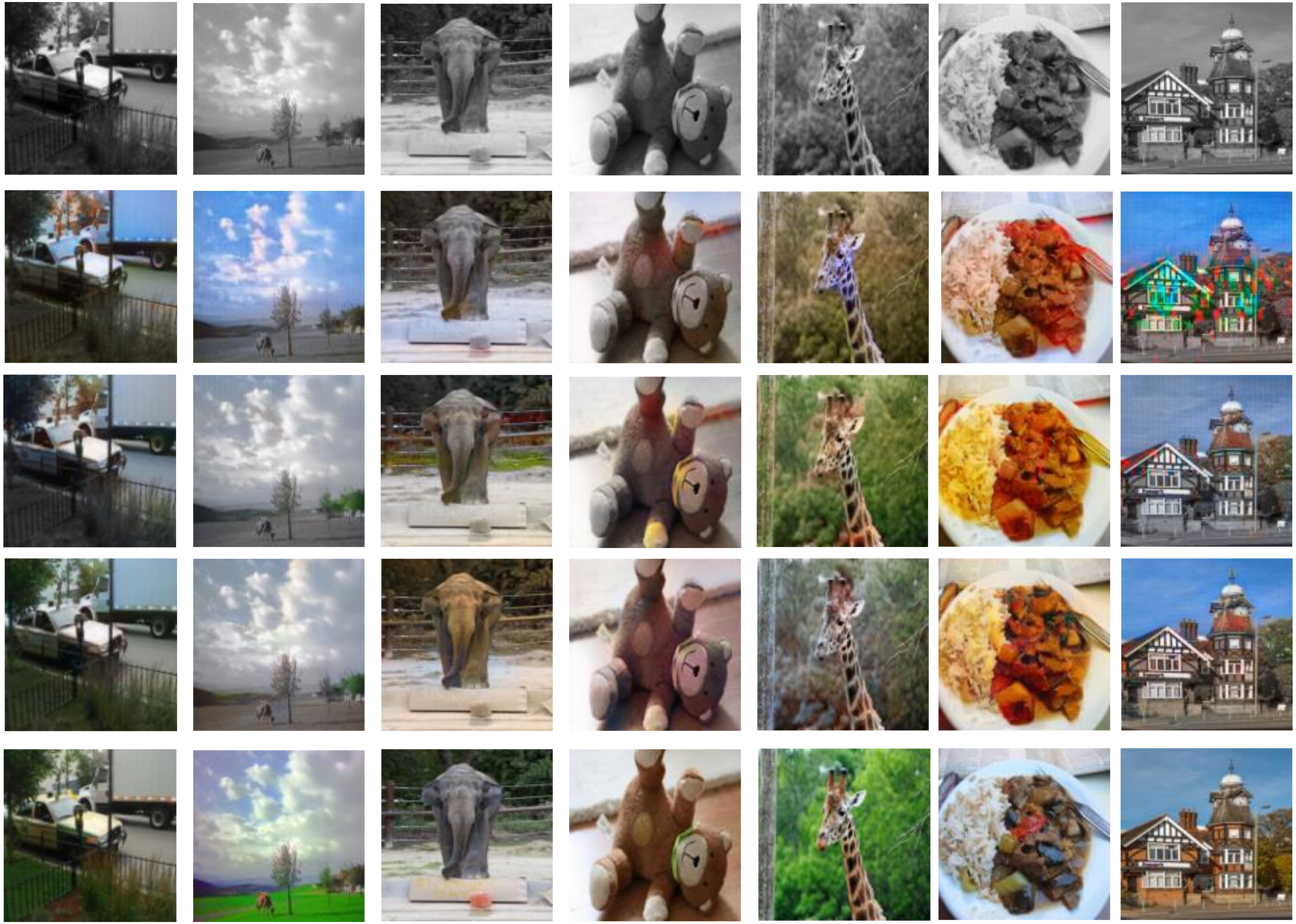
B&W input

15 epochs
training

30 epochs
training

45 epochs
training

Original
image



U
N
E
T

Trained on 20k images

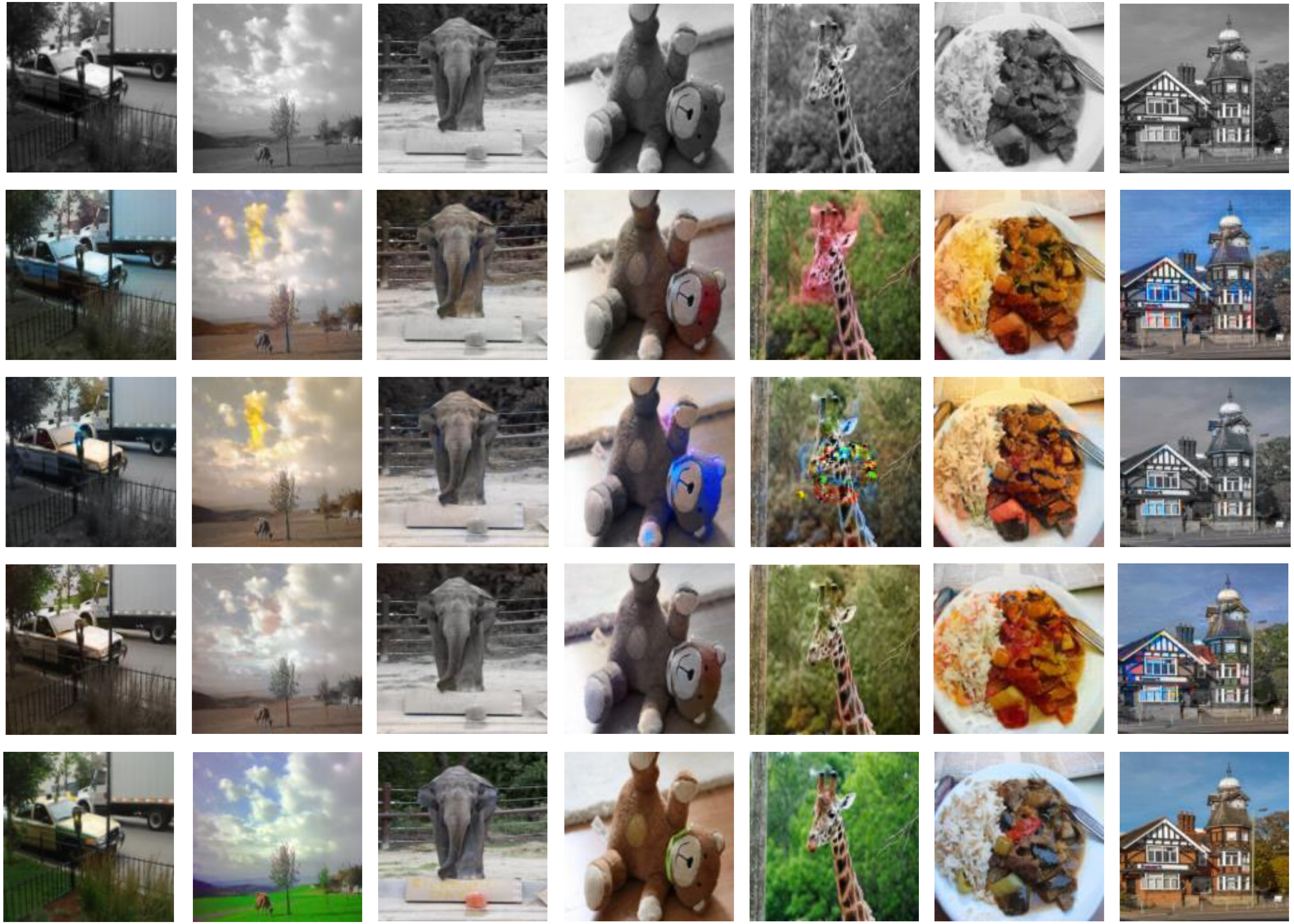
B&W input

15 epochs training

30 epochs training

45 epochs training

Original image



UNET

Trained on 10k images Car Dataset

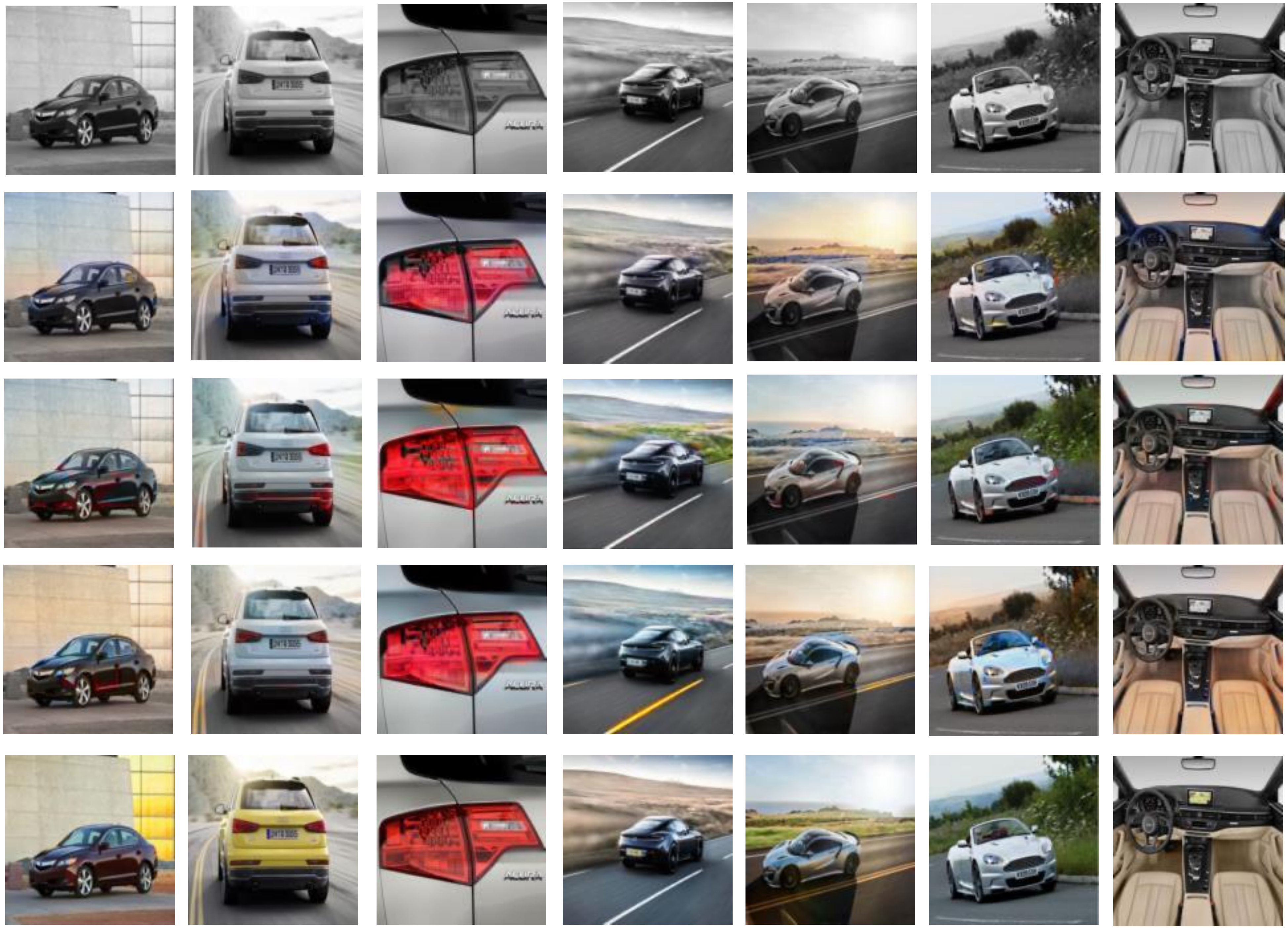
B&W input

15 epochs training

30 epochs training

45 epochs training

Original image



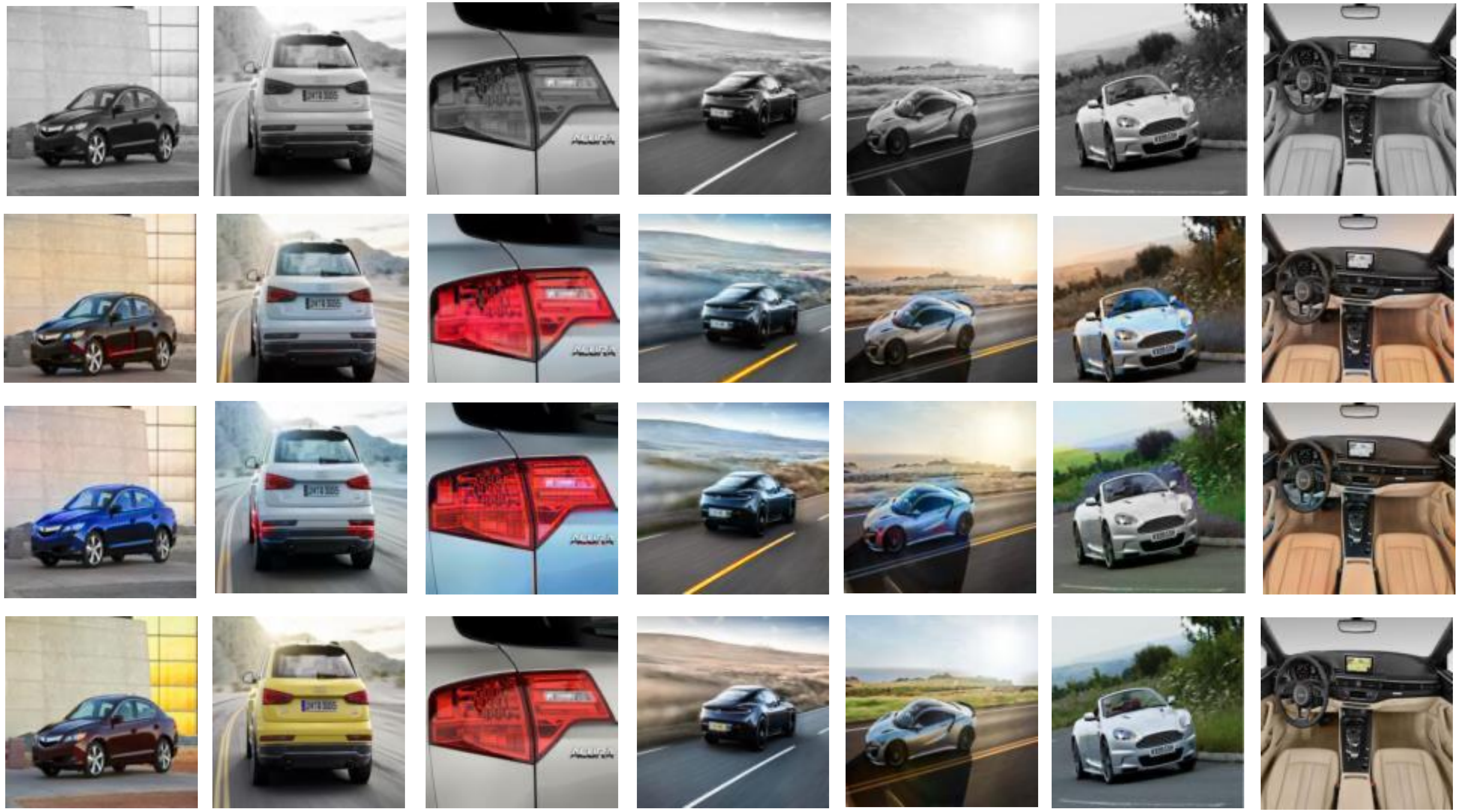
LAB vs RGB

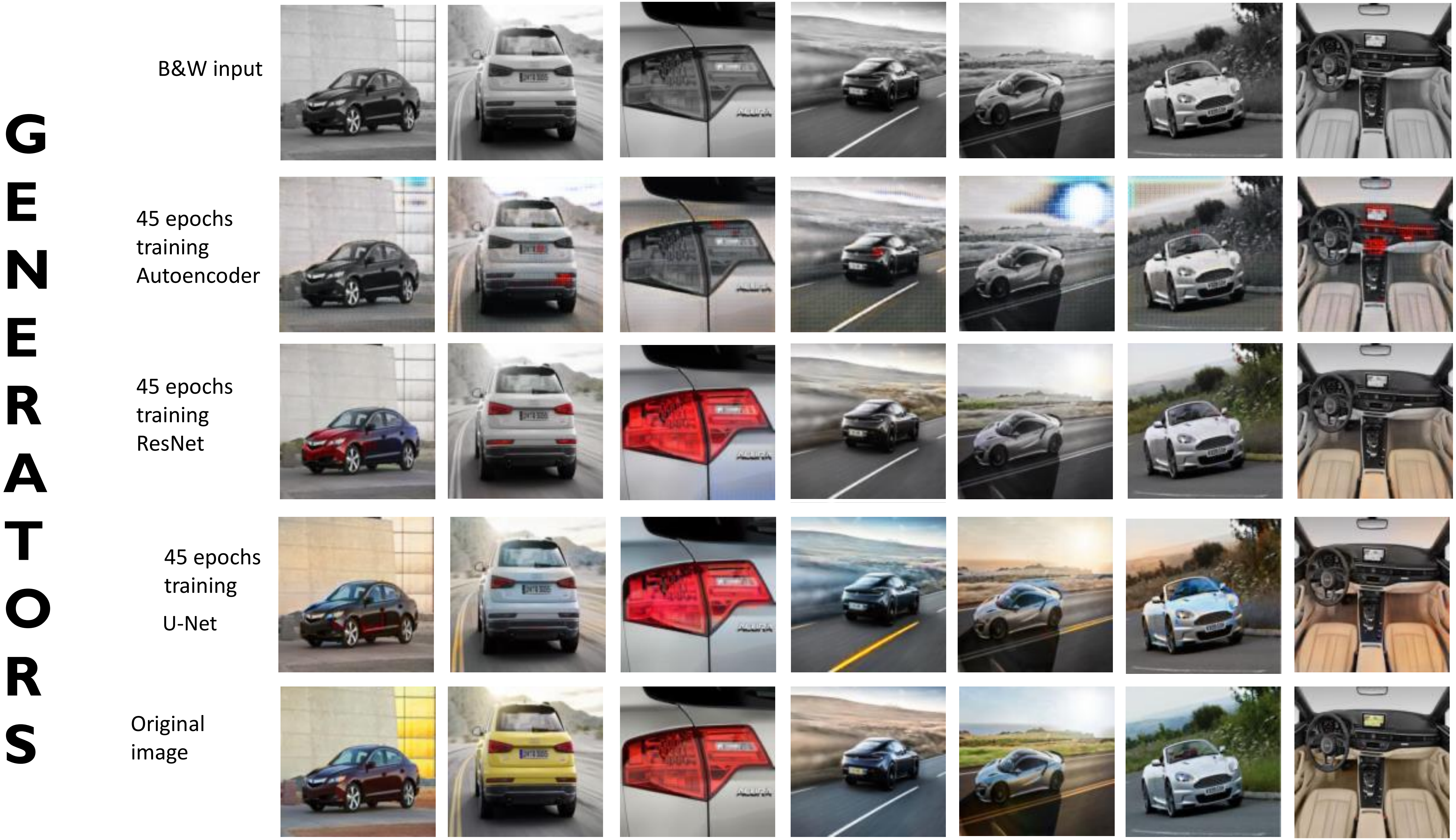
B&W input

45 epochs training Unet in RGB color space

45 epochs training Unet in Lab color space

Original image





GENERATORS
B&W input
45 epochs training Autoencoder
45 epochs training ResNet
45 epochs training U-Net
Original image

FINAL REMARKS