Ostbayerische Technische Hochschule Amberg-Weiden
Fakultät Elektrotechnik, Medien und Informatik

Master Künstliche Intelligenz

Documentation

Deep Vision project
Image Colorization

by

Ziegler, Andreas
during summer semester 2022

Contents

Chapter 1

Introduction

The goal of this work is to explore image colorization with Neural Networks. For this task multiple dimensions have to be added to the input image with color information. For humans this task is for the most part quite simple due to the understanding of the semantics of the image. So humans can conclude, that for example the sky is mostly blue, and the forest is most likely green.
For automatic colorization there are a variety of ways of doing this, but the general idea is to take a grayscale image and map it to color.
There is a paper[5] by Richard Zhang, Phillip Isola and Alexei A. Efros that explores this problem and offers a solution that is similar to what will be implemented here.

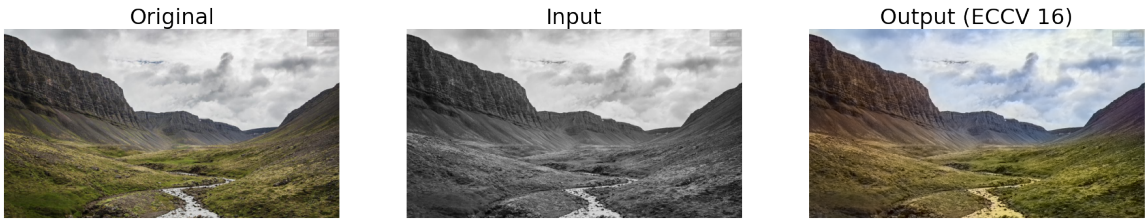

Figure 1.1: Idea of the colorization problem

For this approach a convolutional neural network is used. The model is trained on the landscape dataset [3] dataset. For training purposes the images are converted to grayscale and then feed into the model. The output is compared with the original image. The used models are a variation of vgg16 and alexnet. Furthermore, multiple loss functions are used and compared to each other.
At the end the results are compared to the original paper to see if this work reaches similar results.

Chapter 2

Materials

For this task we can use any Dataset that consists of colored pictures. In the original paper the ImageNet Dataset has been used but due to its size it wasn't possible to use it. Instead, a Dataset with Landscape pictures has been used.

2.1 Dataset

The Landscape Pictures Dataset [3] is used. It contains pictures of different landscapes form the website Flickr [2]:

- landscapes (900 pictures)
- landscapes mountain (900 pictures)
- landscapes desert (100 pictures)
- landscapes sea (500 pictures)
- landscapes beach (500 pictures)
- landscapes island (500 pictures)
- landscapes Japan (900 pictures)

Figure 2.1: Dataset Sample

2.2 Test-Train-Split

To split the Dataset into a train and test set, a multiprocess train test splitter is implemented to speed up the copying of the dataset form the input directory to the data folder. While the dataset is split, the progress is shown in the console.
During the process the input images are checked if they meet the criteria for the colorization task. If not, they aren't copied to the data directory. This is to get rid of label data in the dataset or grayscale images which can't be used for the training.

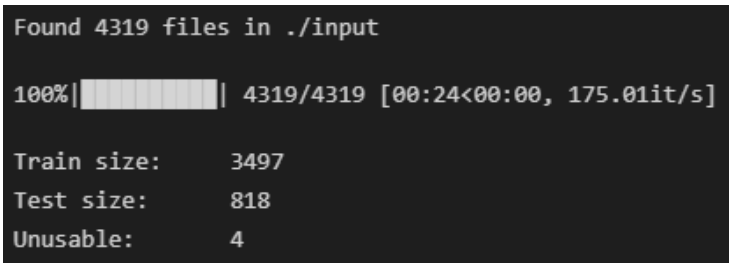

Figure 2.2: Train test split

2.3 Color space

In the RGB color space 3 variables for every color channel have to be calculated to get color and the grayscale data is only indirect used in them. To reduce the complexity of the problem the LAB color space is used. It uses the grayscale image as lightness values and then only the a and b values need to be added by the colorization. The distribution of colors can be seen in Figure 2.3 for L = 50.

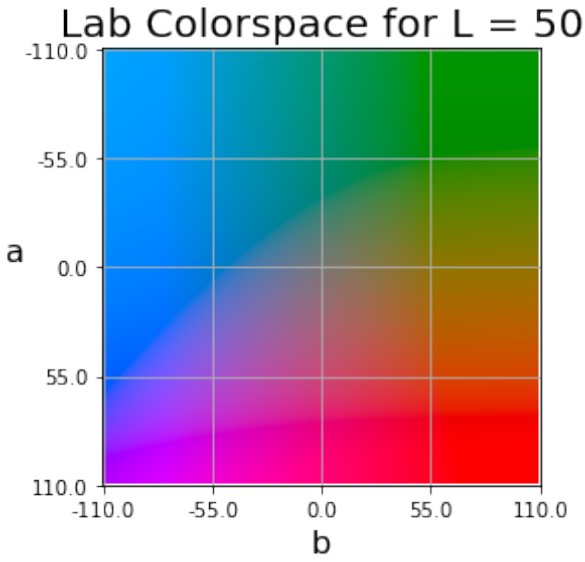

Figure 2.3: LAB color Space

2.4 Color distribution

The distribution of the colors in the LAB color space has been analyzed and is shown in Figure 2.4. It can be seen, that it is not uniform. Colors closer to the center are much more likely to be used and the colors further away from the center are rarely or never used. Compared to the ImageNet Dataset which has been used and analyzed in the original paper, the Landscape Dataset is missing some colors close to the edges of the color space, but the same shape of the distribution can be seen.

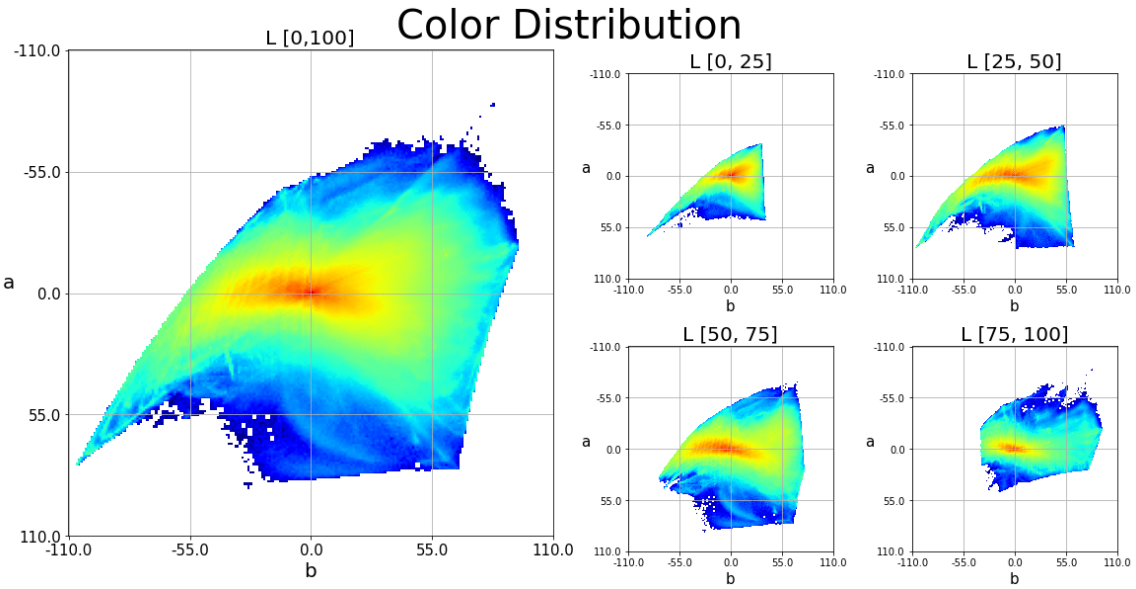

Figure 2.4: Color distribution

When the distribution is plotted only for a specific lightness range, as shown in 2.4, we see that the distribution in dependent on the lightness value. This concludes, that conclusions about color can be made from the lightness value.

Chapter 3

Methods

For colorization 2 methods have been used and tested with 2 different models.

3.1 Colorization as regression

The first approach is to view colorization as a regression problem. Therefore, we try to fit a model which can map the grayscale values to color values.
For this regression the L1 loss and L2 loss are compared. The L1 loss calculates the distance to the correct prediction and the L2 loss calculates the distance to the correct prediction squared.

3.2 Colorization as classification

The second approach is to view colorization as a classification problem. For this to work the color ab space has to be divided into classes, by dividing the 2d colorspace in squares with a certain width.
For this classification the cross entropy loss, once with and one without weights, is used. It is calculated as the negative logarithm of the probability of the correct label. The cross entropy loss increases as the predicted probability of the correct label decreases. To calculate the weights the color distribution 2.2 is used to calculate a normal distribution for all classes. The weight distribution is shown in 3.1.

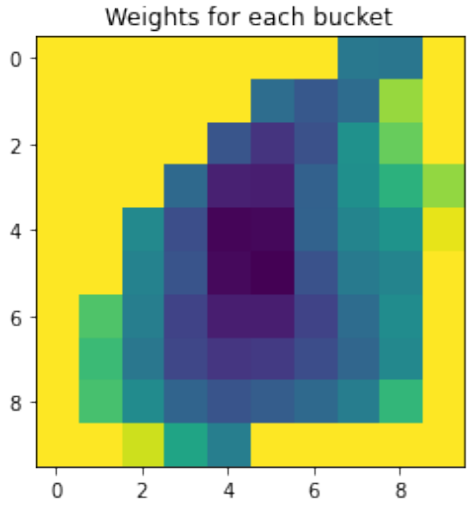

Figure 3.1: Weight distribution

3.3 Model

The model is based on a vgg16, from which all fully connected layers and MaxPool layers are removed. Then the output is replaced with more convolutional layers to build an autoencoder. Furthermore, the "bottleneck" is made bigger to contain more information.

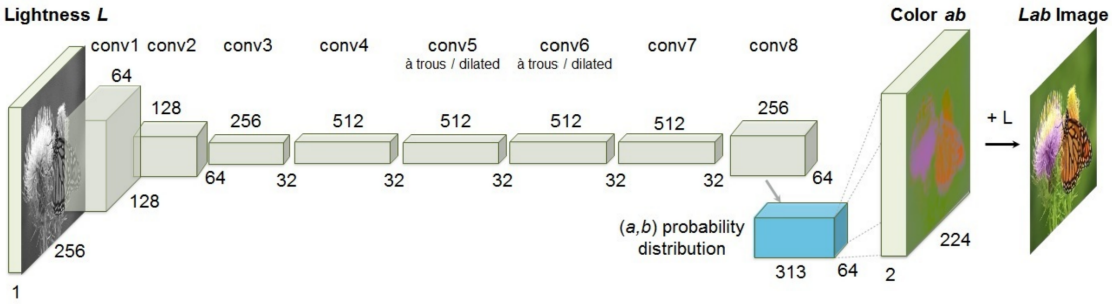

Figure 3.2: ECCV16 architecture

Every block resembles 2 or 3 convolutional layers with ReLU activation function and an BatchNorm layer. All changes in resolution are an effect of the convolutional layers, which requires an up sampling layer at the output of the model.
For the classification approach the model has been altered to not output two values for a and b but to output probabilities for each class, that corresponds to one square in the ab color space.
To compare a second model, the same modifications have been made to an alexnet but due to its different input layers it requires a higher up sampling rate at the end.

3.4 Rating

The rating of colorization problems is not easy. We don't have a good objective way to say if this output is good or bad, because art and pictures are highly subjective and a picture can seem convincing in many different color tones.
As a compromise the simple L1 loss to the ground truth has been used to compare some metrics for the performance.

3.5 Optimizations

For performance and hardware limitation reasons some optimizations have been used.

- First the Dataset has been cashed in memory. This is done by using the Dataset.cache method implemented in torchdatasets [4].
- To speed up the training and to reduce memory usage cuda automatic mixed precision is used [1].
- Unused variables have been deleted from the GPU e.g. the training model outputs during validation, to allow higher batch size.
- All image augmentations are done on the GPU, to speed up training times.
- Multiprocess datasplitter and dataloader are used to speed up the training.

Which these optimizations the limited hardware resources have been used better to train these models, without reducing the accuracy of the training.

Chapter 4

Results and Discussion

The best results have been achieved with the colorization as a regression problem. The training of the classification model from scratch has not been possible to an acceptable result.

4.1 Loss

The losses for all trainings are shown in 4.1. In the next sections relevant losses will be compared.

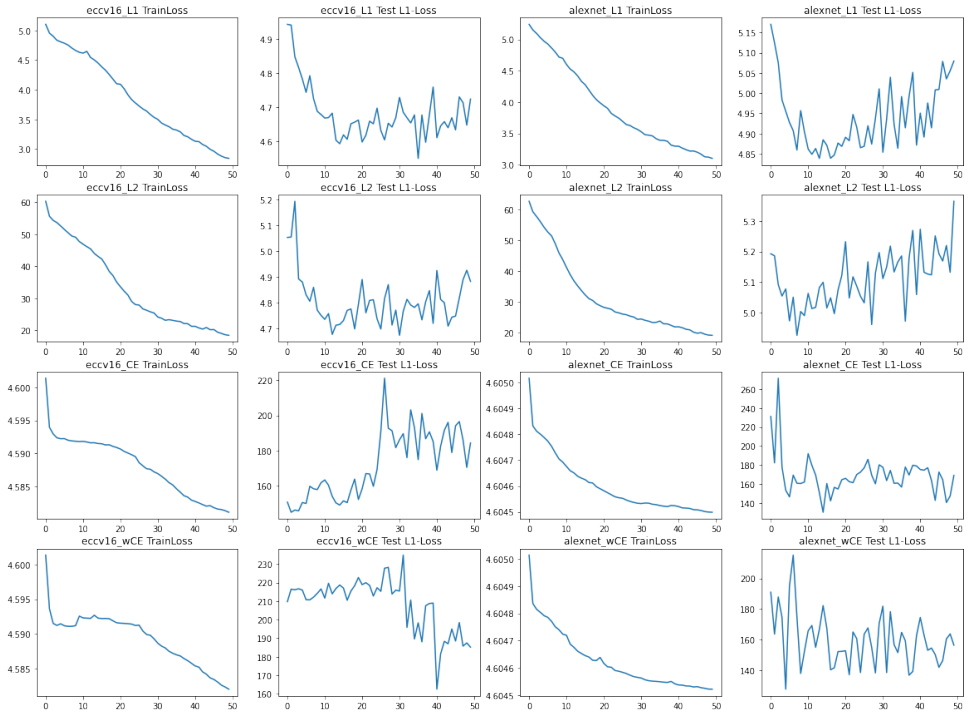

Figure 4.1: Losses

As discussed in 3.4 the Loss is just a very rough estimate of the quality of the colorization. So not much information can be gained from this. But the 2 methods are compared to see if we have some abnormalities.

4.1.1 L1/L2 loss

In the loss graph we can't see much which gives us useful information about the model. More can be found out by comparing pictures for every epoch in Section 4.3.1.

Figure 4.2: L1/L2 Losses

4.1.2 CE/wCE loss

It stands out, that the loss is magnitudes bigger than for the L1/L2 loss 4.2. The reason will be discussed later when looking at the outputs of the model 4.3.2.

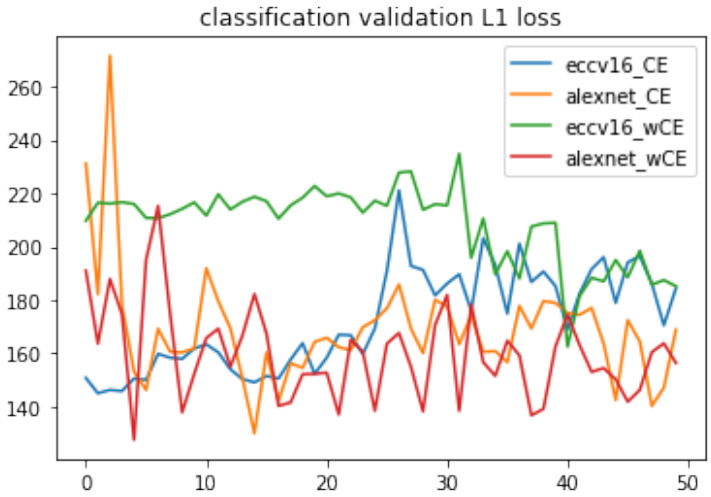

Figure 4.3: CE/wCE Losses

4.2 Model

There has been no big difference between the two models. The eccv16 model proposed by the original paper has slightly better results as seen in picture 4.2 and as seen in picture 4.4 the alexnet model outputs look not quite as good but are acceptable as well.

4.3 Improvements over epochs

In the following pictures the model output every 10 epochs is shown.

4.3.1 Regression

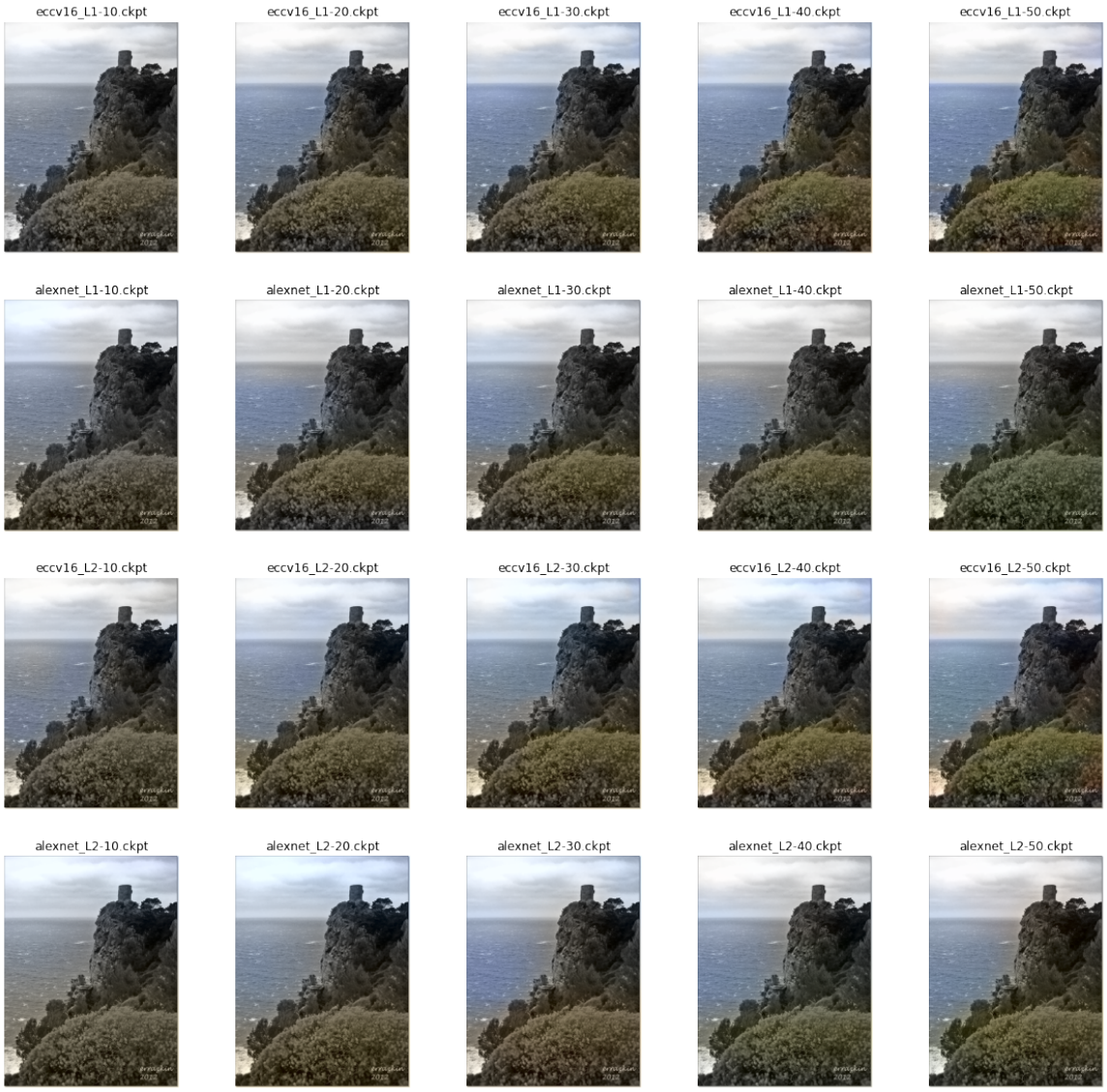

Figure 4.4: Regression improvements

Here can be seen, that the model produces acceptable colorization already after some epochs. These are not perfect and desaturated colors are produced.

4.3.2 Classification

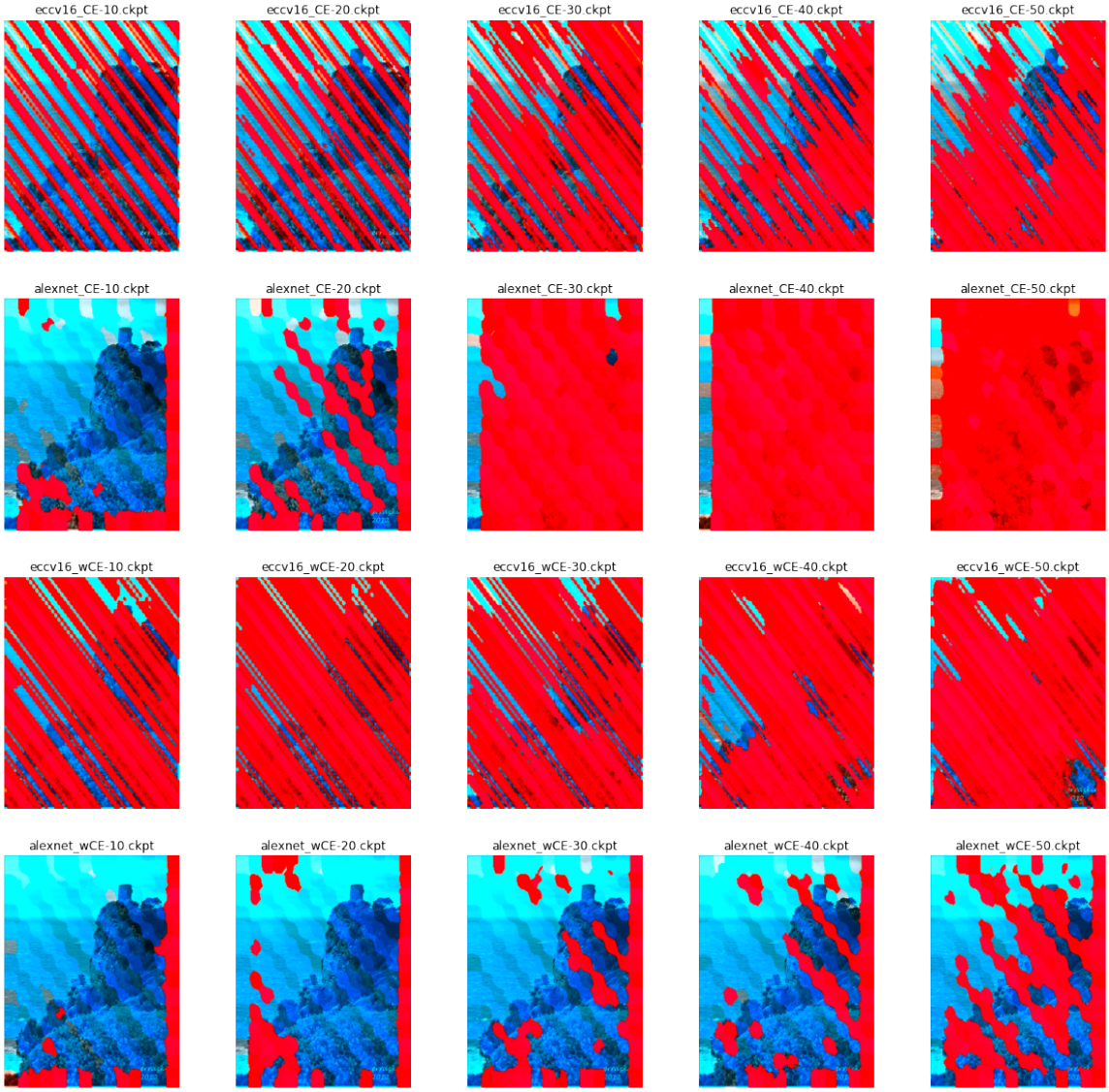

Figure 4.5: Classification improvements

The training of the classification model is not possible to an acceptable result. Which can be explained by multiple reasons. First the original paper uses the complete ImageNet Dataset which has far more pictures than the used dataset. Second the bucket size for classification has had to be reduced due to GPU memory limitations. Third the training was only run for 50 epochs which is not enough to train a complex classification model for this many classes from scratch.

Chapter 5

Conclusion

This work has shown, that a basic colorization is quite easy to implement with a neural network as a regression problem. For this only a small dataset and few epochs are needed. But this approach doesn't produce a big color range.
The colorization as a classification problem is more difficult to implement. For this a large dataset and many epochs are needed. But this approach would produce a large color range in some cases.

As seen in 5.1 the regression Model outputs quite convincing results, which are in some cases even better than the original paper. But in general they are quite desaturated.

To continue this work the conversion from classes to ab space can be improved further, which will improve the colorization for the second method.
Due to the restricted timeframe of this work a small dataset and few epochs are used. To improve colorization the training can be rerun with more pictures over more Epochs.
Another approach for this problem can be the use of transformers, which see more use in the last 2 years for vision tasks, which can be analyzed to continue this work.

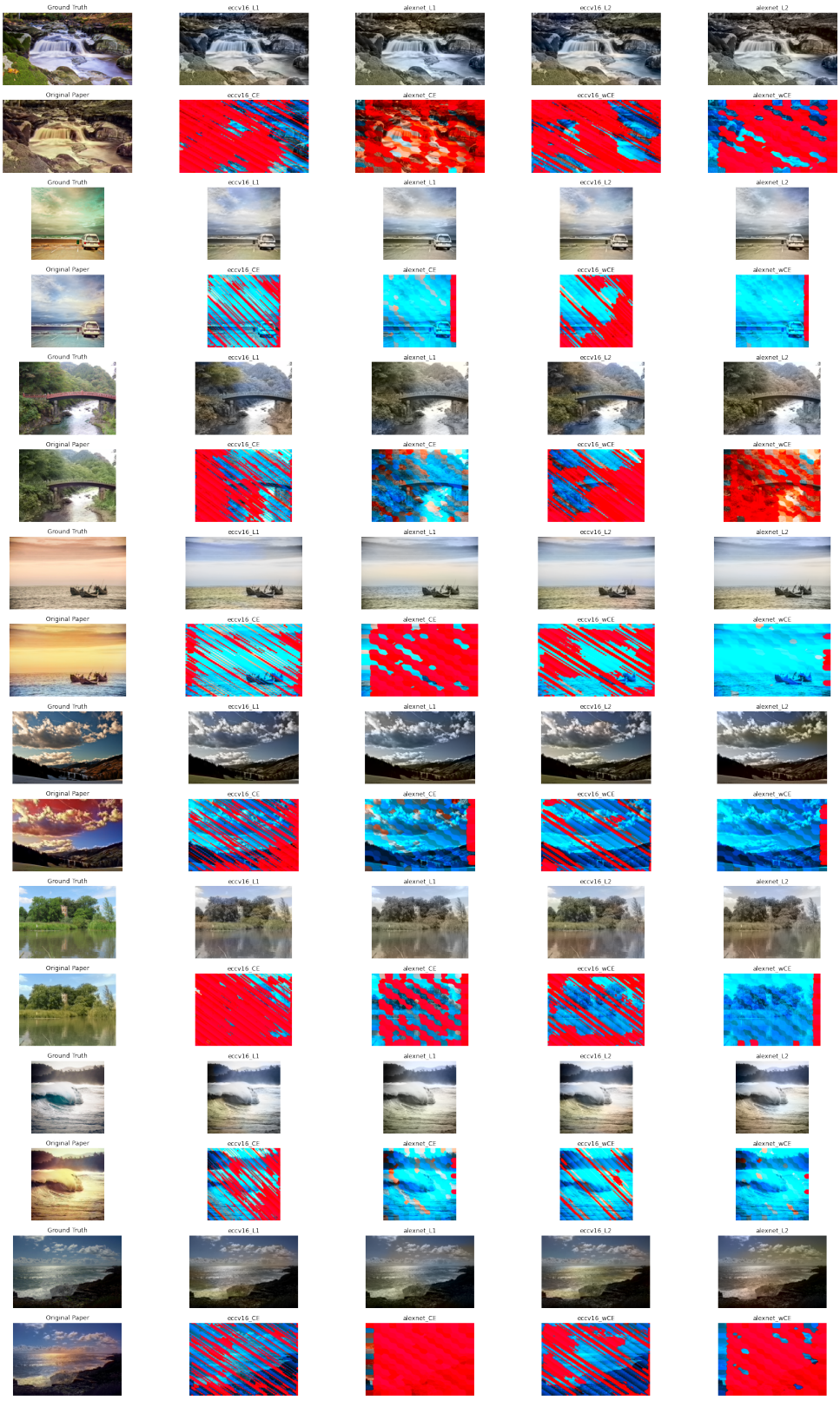

Figure 5.1: Pictures

References

[1] *DO MORE WITH MIXED PRECISION TRAINING*. https://developer.nvidia.com/automatic-mixed-precision

[2] *Flickr*. https://www.flickr.com/

[3] *Landscape Dataset*. https://www.kaggle.com/datasets/arnaud58/landscape-pictures

[4] *Torch Datasets*. https://pytorch.org/docs/stable/torchvision/datasets.html

[5] Zhang, Richard ; Isola, Phillip ; Efros, Alexei A.: Colorful Image Colorization. In: *ECCV*, 2016