Colorful Image Colorization with Tensorflow

Sophia Schulze-Weddige Malin Spaniol
Maren Born
Implementing Artificial Neural Networks with Tensorflow
Universität Osnabrück

April 16, 2020

1 Introduction/Motivation

Based on the paper *Colorful Image Colorization* (Zhang et al., 2016) this project aims to reimplement a similar artificial neuronal net that transforms grayscale images into colorful pictures.
This involves first creating a dataset based on pictures that are converted into the CIELAB colorspace (Lab), such that the first channel "L" can be considered as input as it is grayscale whereas the "a" and "b" channel are the target labels to be predicted. Thus, the problem can be handled as classification task. In the second step, the aim was to closely rebuilt the layers of the original model (which used "caffe" (Jia et al., 2014)) using tensorflow 2.0 (richtige version?).
Other project have trained convolutional neural networks (CNNs) on the color prediction problem before (e.g. Cheng et al. (2015), Dahl (2016)). The training data is easily available which enables training on large datasets. Problem about previous approaches is that they try to predict the ground truth rather than a possible truth. A conservative loss function tries to minimize Euclidean error between estimate and ground truth. As objects can have various plausible colors, these predictions are multimodal. Thus, the approach of Zhang et al. (2016) innovates a loss function that predicts plausible colors for pixels, rather than the original color (Zhang et al., 2016).

we want to built a network that can colorize images
- this project aims to produce colorful images, given a greyscale picture.
- transforming greyscale into plausible colors is an easy task for humans
- We see a greyscale picture showing a woman playing volleyball at the beach. As we can recog-

nize the scene and the form and relate to it. The sand is yellow, the sea is blue and the ball is white.
- But coloring it in life would be a much more difficult task. As we also need to consider different textures, shades and so on. Seeing and imagining things does not make people a proper painter.
- Surface structure and the semantics of the scene are necessary to validly color images.
- this project aims does not aim to generate the true color for pictures but at least a good and prediction.
- aus dem paper: model enough of the statistical dependencies between semantics and the textures of greyscale images and their color versions in order to produce visually compelling results.

2 Important background knowledge (including reference to most relevant publications)

- was ist colorization
- what is CNN for image colorization?
In image classification, usually convolutional neural networks (CNNs) are used, which are inspired by the visual cortex of the brain. The idea is that highly specialized components or, in the case of CNNs, filters learn a very specific task, which is similar to the receptive fields of neurons in the visual cortex (QUELLE https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1359523/). These components can then be combined to high-level features, which can then in turn be combined to objects that can be used for classification. In CNNs this concept is implemented by several successive convolutional layers in which one or more filters are slid over the input generating so-called feature maps. Each unit in one feature map looks for the same feature but in different locations of the input. In recent years, CNNs became so good that they outperform humans in many classification tasks (QUELLE http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/ , QUELLE https://arxiv.org/abs/1409.0575). Although machine learning exhibits very promising results and a lot of research and literature is available on the topic, many branches of industry still rely on traditional computer vision techniques in their implementation of image classification.

CNN apply trainable filtering kernels to extract features from an input image and project these features onto feature maps (?) (?) (?). Based on these maps, the label is predicted by a standard fully-connected layer. Though using a CNN on the raw data images seemed promising, the results were only slightly better than chance level for the training accuracy.
- wie ist der status quo: – related work

- was sind die schwierigkeiten von den colorization sachen
- was hat Zhang gemacht - was hat er besser gemacht

```
#packages needed
import numpy as np
import tensorflow as tf
import cv2
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Activation,
    BatchNormalization
  for loop
'ich bin ein string'
if
return
MyClass,__init__
```

- We started looking at this paper:
https://arxiv.org/abs/1603.08511 ,
- at their best solution

3 The model and the experiment (MAIN PART). This part should feature code.

- hier nur erklären was wir machen
- Image colorization in general
- the model of the original paper
- theoretical basis:

3.1 Dataset

- loading large amount of data
https://machinelearningmastery.com/how-to-load-large-datasets-from-directories-for-deep-
das ist (Brownlee, 2019)

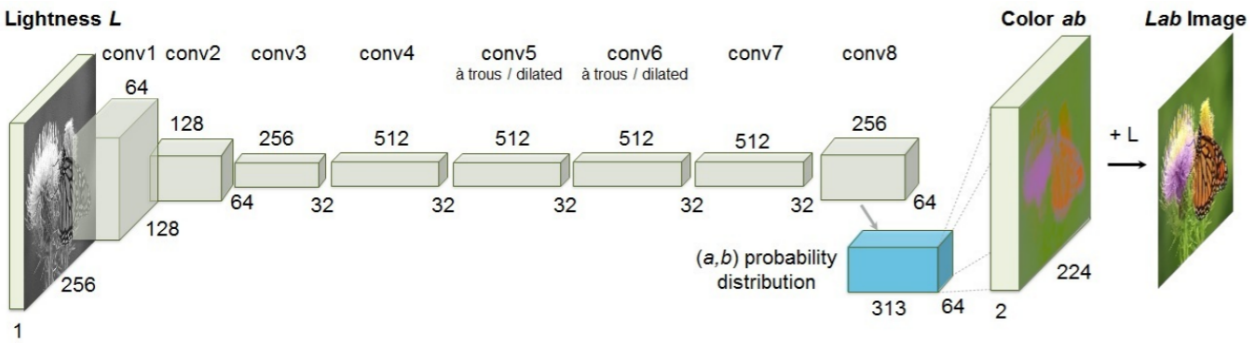

Figure 1: The network architecture of Zhang et al. (2016).

3.2 Preprocessing

- image preprocessing documentation
https://keras.io/preprocessing/image/#imagedatagenerator-class
- preprocessing via ImageDataGenerator() from keras.prepeocessing.image
- takes in traindata, validation data and test data
- featurewise-center and featurewise std
- classmode: none -> is for predictions

3.3 Modelstructure

- besser hier layer und loss-function

3.4 Layer

- built the same layers aas Zhang et al, but no probability distribution
-networkstructure and design

3.5 Loss-Function

- the original loss-function
- we tried what loss function?

4 Visualization and discussion of your results.

oder lieber bilder als screenshot einfügen?

```python
import numpy as np
import tensorflow as tf
#from skimage import color
import cv2

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Activation, BatchNormalization
#from keras.layers import Dense
```

Figure 2: Hier kann man dann auch noch etwas dazu schreiben

4.1 Training

4.2 Testing

4.3 Conclusion and Future Work

- vergleich ziehen zu Zhang

5 Literaturverzeichnis

Brownlee, J. (2019). How to load large datasets from directories for deep learning in keras. https://machinelearningmastery.com/how-to-load-large-datasets-from-directories-for-deep-learning-with-keras/.

Cheng, Z., Yang, Q., and Sheng, B. (2015). Deep colorization. *2015 IEEE International Conference on Computer Vision (ICCV)*.

Dahl, R. (2016). Automatic colorization. https://tinyclouds.org/colorize/.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*.

Zhang, R., Isola, P., and Efros, A. A. (2016). Colorful image colorization. *CoRR*, abs/1603.08511.