Investigating how Image Classification and Colorization Tasks Affect Each Other

Lotta Piefke, Berit Reise, Ruth Verdugo

Department of Cognitive Science, University of Osnabrück
Implementing Artificial Neural Networks with Tensorflow
Winter Semester 2022-23

1 Introduction **1**
2 Related Literature **3**
3 Method **5**
3.1 Tasks 5
3.2 Datasets 7
3.3 Architecture 8
3.4 Implementation 11
4 Results/Analysis **11**
4.1 Basic Architecture 11
4.2 Analysis of Failure 15
4.3 More Complex Architecture with Adam and Dropout 16
4.4 Residual Connections 19
5 Conclusion **21**
6 Limitations **21**
7 References **22**

1 Introduction

Image classification is a rudimentary principle within the field of computer vision. The majority of datasets used for image classification commonly contain color images. These datasets are used to train deep Convolutional Neural Networks (CNN; LeCun et al., 2010) (Gowda & Yuan, 2019). Since datasets often consist of a limited amount of color images, optimizing colorization in a way that is both autonomous and accurate, can significantly increase information within that dataset which can later be used for training a model. According to a study by Baldassarre, Morín, and Rodés-Guirao (2017) applying a colorization step on grayscale images can improve CNNs such as Inception, Residual Network (ResNet), or Visual Geometry Group (VGG) results on image classification or object detection tasks.

Improving colorization and therefore classification through colorization can have a significant impact on a variety of domains. These applications are relevant for re-mastering of historical images and improvement of surveillance feeds (Baldassarre et al., 2017). Furthermore, they are promising computer-aided solutions within the medical field, for example in skin-cancer diagnosis (Goyal et al., 2020). These technologies can also be used for the creation of artistic images. For example, a system developed by Leon Gatys which takes multiple characteristics of an image's style to create a new image (Ajit et al., 2020).

In this project, we propose to train a convolutional neural network on colorization and classification. We will first test our model on the Natural-Color dataset (NCD; created and provided by Anwar et al. (2020)) to verify the network is built correctly and learning. Then we will test if our model can learn from a more complex dataset by using the Stanford Dogs dataset (Khosla et al., 2011) which contains images of dogs. Furthermore, we will investigate how the colorization task changes the classification of the dogs and vice versa. We hypothesize that colorization could be improved from high-level concepts like body parts, background, etc. which also help distinguish and therefore more accurately categorize dog breeds.

For the purpose of our project we have chosen a convolutional neural network architecture deployed by researchers Iizuka, Simo-Serra, and Ishikawa in 2016. We chose this architecture since convolutional neural networks (CNNs) are used to replicate visual sensory processes similar to humans or other living animals. Using various functionalities, CNNs are used in the field of object detection, object recognition, or style reconstruction (Ajit et al., 2020). These functionalities can be used for classification tasks containing images (Iizuka et al., 2016). Important processing abilities of these networks include the following: Optical flow, super-resolution, contour detection, semantic segmentation, and multi-scale fusion. Typically, multi-scale fusion is when images are fused by scaling and concatenating them to matching resolutions. Iizuka and colleagues used multi-scale fusion to fuse both global and local features into one feature vector. A network that has the latter and can handle two tasks (e.g. colorization and classification) should both be able to handle images of any resolution and depth estimation (Iizuka et al., 2016).

Additionally to the architecture, when working with a colorization task, color space should be considered. This is essential because the chosen color space has an impact on accuracy. In the context of deep learning, color is represented via a specific color space. When working with gray-scale images, “color spaces help us to reproduce analog and digital representations of color” (Gowda & Yuan, 2019). Some popular color spaces include but are not limited to RGB, HSV, and LAB. In this project, we utilized the LAB color space. “L” represents lightness, “A” represents the green-red color component and “B” stands for blue-yellow (Gowda & Yuan, 2019). The model gets the “L” information as input, and has to predict the “A” and “B” values. Compared to RGB, this has the advantage that the model has to predict only two channels instead of three. This makes the model smaller and the learning faster.

Since we will be working with varying images, there are many aspects that can affect the accuracy of our model. Some include changes in illumination, variations in viewpoints, and

occlusions in the images. Diving further into past research can help us determine what factors to take into consideration when developing our model.

2 Related Literature

In a review of CNN architectures conducted by researchers Ajit et al., (2020) multiple advantageous features of the models were discussed. A CNN is typically built by concatenating multiple convolutional layers, the most important part of the CNN. A convolutional layer takes a 2D image and produces an activation map for it. This is done by convolving the pixel matrix belonging to the image. The matrix then acts as a sort of feature detector and the activation map then saves the features that are important for the particular image that is currently processed. Backpropagation is used to optimize the weights in the network by minimizing the error function (Ajit et al., 2020).

As previously mentioned, CNNs are often used for object detection. Joseph Redmond and colleagues (2015) advanced the object detection field by combining both the localization features and detection abilities through the creation of his unified real time object detection model called You Only Look Once (YOLO). The Single Shot Detector (SSD) model further advanced the field by classifying the objects that were detected faster than its predecessor the R-CNN via a single forward pass (Ajit et al., 2020). On top of real time object detection, researchers worked to develop networks that could recognize digits. Among one of the most prominent models is the LeNet developed in 1998 based on the MNIST dataset (Ajit et al., 2020). The MNIST dataset contains 70,000 images of handwritten digits, which models have to learn to classify (Mohapatra et al., 2015).

The ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) brought many more architectures such as AlexNet, Visual Geometric Group (VGG) Net, GoogLeNet, and Microsoft

ResNet (Ajit et al., 2020). According to Ajit et al., (2020) these networks were recognized for the following features:

- Including dropout layers to avoid overfitting (AlexNet)
- Using blocks to:
    - Decrease the need for computational power through reducing the number of parameters to be computed.
    - Allow ReLu activation twice in each block (VGGNet)
- Use of global average pooling to take the average of every feature map individually. (GoogleLeNet)
- Using skip connection to solve the vanishing gradient problem in a deep network. (Microsoft ResNet)
- Using bottlenecks to reduce the number of parameters. (Microsoft ResNet & GoogLeNet)
- Use of Batch Normalization after each convolution. (Microsoft ResNet)

In more recent years, the dilated convolution model has become more popular in the field of image segmentation. This type of model addresses the problem of high computation costs seen in a classical CNN by replacing the convolution kernels of the traditional CNN with convolution kernels with “holes”. This assists the expansion of space to be analyzed by the model yet maintains the number of parameters at a lower number which in turn affects computation costs (Lei et al., 2019).

There are other factors to take into consideration besides architecture when a colorization task is involved. In a study conducted by researchers Shin et al. in 2002 they investigated skin pixel classification with various color spaces. Results were mixed; an analysis on skin detection using RGB color space gave the best results. Yet, when conducting an analysis on skin pixel classification using a Bayesian model using multiple color spaces, LAB

color space gave the most accurate results. In order to further investigate the impact of color space on classification accuracy, researchers Gowda and Yuan (2019) conducted a study comparing multiple color spaces. They preprocessed the datasets CIFAR-100, SVHN, and Imagenet using multiple color spaces to compare their accuracy in classifying. They found that there is no “best” color space but rather that different models and classes have certain color spaces that are more suitable to them. Therefore making use of multiple color spaces can have a positive impact on classification accuracy (Gowda & Yuan, 2019).

3 Method

3.1 Tasks

Iizuka and colleagues (2016) came up with an architecture that is able to train a model in an end-to-end fashion to colorize and classify grayscale images. Since their model could reach great results for a large-scale scene database, we questioned whether such a model will also be able to show high performance on a different, arguably harder, dataset. To test this, we recreated the architecture for the colorization model following Iizuka et al. (2016; more on this in 3.3). Firstly, we trained and tested our architecture on an easier dataset (Natural-Color dataset; see 3.2) to verify the architecture was correctly implemented and can in fact learn to colorize images. Then, we tested whether the model, trained on a much harder dataset, could still provide decent colorization. For this, we worked with the Stanford Dog Dataset (Khosla et al., 2011) which will be described in more detail in 3.2. Additionally, newer approaches in machine learning focus on multi-task learning of models. This is a promising approach which often leads to improved performance of models (Zhang, & Yang, 2022). In this fashion, Iizuka et al. (2016) also used a classification component in their architecture to exploit information given by class labels to improve learning of global priors and thus also improve colorization. Trailing their

approach, we also augment the model's learning task to include classification. Both image datasets included labels for their depicted entities such that we also trained models on the categorization task. Furthermore, it is tested how the two tasks together, colorization and classification, behave. In other words, when trained on both tasks, how does the model perform and does information from the classification task improve colorization? The idea behind this is that global features as learned for classification could benefit colorization. A possible reason for this could be that the model is less likely to overfit when sharing these features across the network.

Colorization is a task where given a grayscale image, the color of each image pixel is predicted to get the overall colorization of the image. This color can be represented by a chosen color space (e.g., RGB, CIE Lab, YUV, and others). The model gets grayscale images and tries to predict the original. This is quite challenging as many objects can have multiple accurate colors in real life, i.e. a t-shirt can be blue, yellow, green, etc. and every color is a correct possibility for a t-shirt. For 'good' (i.e., natural) colorization, it is important that individual objects in the image are detected. The borders of individual objects can be seen as very coarse, global information of the image. Also, differentiating things like day/night-time and indoors/outdoors are considered important global information. Other features might also be of relevance. For instance, mid-level features and low-level features like the textures of an object can help to correctly colorize it (Iizuka, et al., 2016). Furthermore, one could exploit the global priors of pre-trained models to provide the trained model with even more diverse global information than the model would solely get from the dataset it is trained on. Having access to such a huge variety of global feature information can enhance the model's performance (e.g., see Baldassarre et al., 2017). However, this was not done here, as we were interested in the model's performance without a huge amount of prior global information.

On the other hand, classification is a task where given a certain input, here an image, the model predicts a categorical output, i.e. which class the input belongs to. The classes are

usually a set of category labels and membership is unambiguous. The model is provided with examples of images and the class they belong to during training and tries to learn a function that classifies images into their correct category. It is important that the model learns an appropriate function such that it can then generalize well to unseen images. However, there are also many deep learning approaches with vague or overlapping category memberships, and unsupervised learning approaches. Here, we focus only on supervised learning and clear-cut classes.

Considering both colorization and classification, it becomes clear that they have some overlapping characteristics. For both tasks it is important for the model to notice objects in the image and certain features of these objects (e.g. texture, shape) to correctly color or classify them. Thus, it is reasonable to assume that a model that trains on both of those tasks will benefit from the information they utilize individually as it could be relevant for the other task respectively. To sum up, we train multiple models; we train models on the colorization task and the classification task individually and a model that trains on the two tasks combined. Further, we train those three models on the Natural-Color dataset and the Stanford Dog Dataset respectively.

3.2 Datasets

The Natural-Color dataset (NCD) was created by Anwar et al. (2020) and contains images of fruits and vegetables. Some examples can be seen in Figure 1. There are 20 classes and 721 images in total. There are approximately 30 images per class but the exact number varies slightly. The images depict the fruit/vegetable with a white background and nothing else but the single object is visible in the picture. That makes the dataset perfect as a sort of baseline for testing a model, as the images are very simple and contain no distractions. For training, the images were split into training and validation sets, containing 649 images and 72 images respectively.

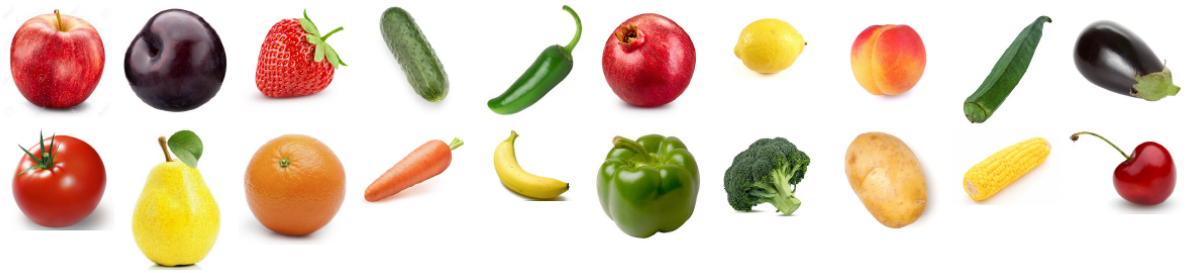


Figure 1. Images from the Natural-Color Dataset. (Anwar et al., 2020, Fig. 9, p.14)

The Stanford Dog dataset (SDD; Khosla et al., 2011) contains images of dogs in a multitude of situations, e.g. with their owners, while running around outside, wearing costumes, with other dogs or objects in the same picture. Some example images can be seen in Figure 2. There are 120 classes, i.e. dog breeds, and 20 580 images in total. There are approximately 150 images per class but the exact number varies slightly. For training, the images were split into training and validation sets, containing 18 522 images and 2 058 images respectively.

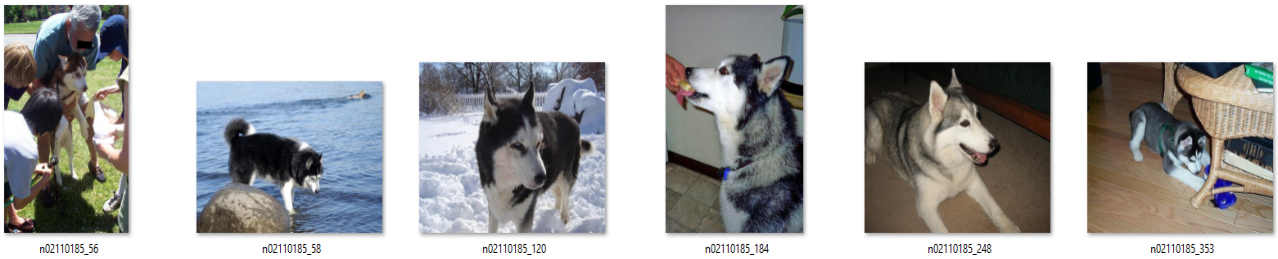


Figure 2. Images from the Stanford Dogs Dataset (Khosla et al., 2011). Examples from the class of Siberian Husky.

3.3 Architecture

Our architecture closely follows Iizuka and colleagues' model which can be seen in Figure 3. The model is made of six important parts: the low-level feature network, the mid-level feature network, the global feature network, the fusion layer, and lastly the classification and the colorization network. When training the model only on the colorization task, the global features

network and thus the fusion layer are not utilized. On the other hand, when training only the classification task, the mid-level features and the fusion layer are not used. Only when training on both tasks, all components are brought together. Generally, we used ReLU as the activation function for all convolutional layers, except for the last one for the final output of the network.

The low-level feature network has six layers and gradually reduces the size of the feature maps; the size of output channels goes from 64 to 128 to 256 to 512 . Each layer has 3 x 3 convolutional kernels and instead of using max-pooling to reduce the size of the feature maps, the strides of the convolutional layers are increased (with a stride of 2) and padded (1 x 1). Both the classification branch and the colorization branch use this low-level feature network, meaning that the weights are shared between the two when training together. The obtained low-level features are then fed into the mid-level feature network and the global feature network respectively.

Mid-level and global features networks are processing further the 512-channel features they got from the prior network part. The mid-level features network has two convolutional layers and reduces the output to 256 channels containing the mid-level features. The global feature network on the other hand has four convolutional layers and then three fully-connected layers. The output here is a 256-dimensional vector.

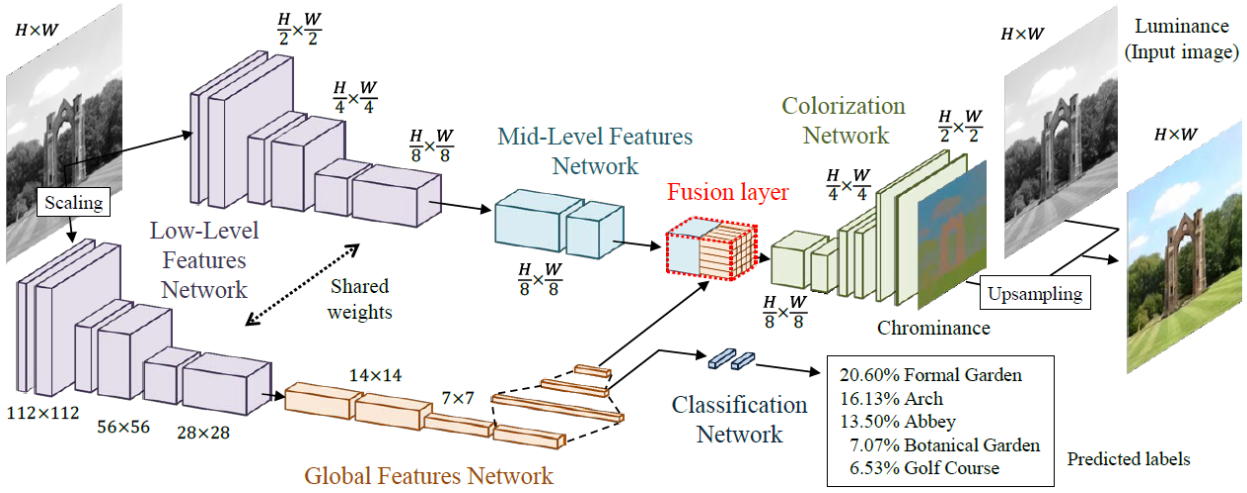


Figure 3. Architecture of Iizuka et al. (2016) that we replicated in our model. (Iizuka et al., 2016, Figure 2, p. 4)

After that, the fusion layer is of importance to combine the information provided by the global features network and the mid-level features network. This is a crucial part since the mid-level features are providing more local information about the image and the global features network more broad knowledge about the image as a whole (Iizuka et al., 2016).

After the features are brought together they are processed by multiple convolutional layers and upsampling layers in the colorization network. Upsampling is done by using the nearest neighbors approach, resulting in the output being twice as wide and twice as tall as before. This is done until the output is half of the size as the original input. The convolutional and upsampling layers alternate. The final output layer of the colorization network is a convolutional layer with a Sigmoid function. The final output is the chrominance, i.e. the color information, of the input grayscale image. This corresponds to the A and B components of the LAB colorspace. These components are normalized during preprocessing such that they lie in between [0,1] and after the layer with the sigmoid function we can now easily compare the A and B components of the target image with the components predicted by the model. For this, we calculate the mean squared error (MSE) after the target image's components were scaled to the model output. This loss is then backpropagated through the model and updates the parameters during training.

In contrast, the classification network exhibits two fully-connected layers where the first is a hidden layer with 256 outputs and the second is an output layer with the number of classes in the dataset as outputs. For the Natural-Color dataset there are 20 and for the Stanford Dog dataset there are 120 classes. The classification network uses cross-entropy loss. Importantly, this loss of the classification network only affects the networks directly connected to it and not the colorization network parts, i.e. the colorization network and the mid-level features network.

Considering the overall model when training on both tasks, the global loss then becomes a joint loss of the MSE loss for the colorization task and the cross-entropy loss of the

classification task. To ensure that the loss of both networks are on the same scale and that the model focuses on minimizing both losses equally, the classification loss gets divided by 200.

3.4 Implementation

Our implementation was done in python and uses the deep learning framework tensorflow**.** Firstly, the raw images were preprocessed. Again, we mainly followed the steps of Iizuka et al. (2016). The pictures of the Natural-Color Dataset were scaled to 128 x 128 pixels, and the Stanford Dog Dataset images were scaled to 256 x 256 pixels. Then, random flips and rotations were applied to the images. These steps make the model more robust and enhance the model's ability to generalize well later. We used a batch size of 64 and included batch normalization layers throughout the model. In order to stay close to Iizuka and colleagues' implementation, we used the ADADELTA optimizer (Zeiler, 2012) in the first experiments. In later experiments we chose the state-of-the-art optimizer ADAM (Kingma & Ba, 2014). The complete code can be found under https://github.com/lolotta/IANNwTF-Final-Project .

4 Results/Analysis

4.1 Basic Architecture

After training on the Natural-Color dataset for 50 epochs, we saw that the categorization training loss went down both for the classification model and the combined model. But for the classification model, the test loss is much bigger than the training loss (Fig 4). This shows that the classification model is not learning to classify unseen images, but rather memorizes the rather small training dataset. This supports our hypothesis that a combined model that learns two tasks is forced to generalize more and overfits less.

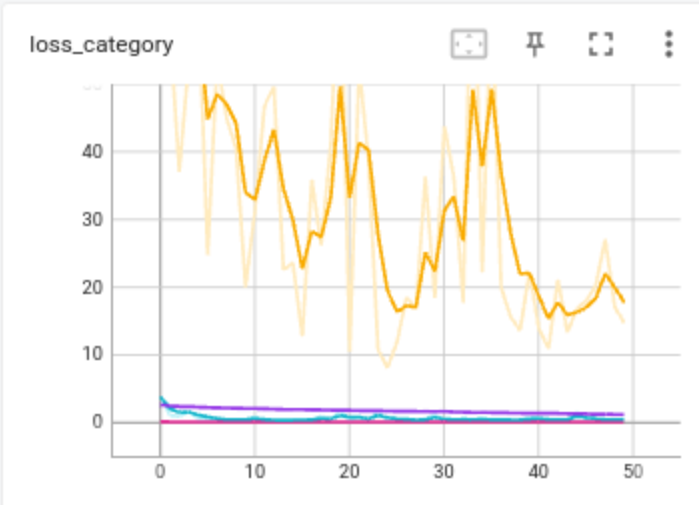


Figure 4. Categorization loss. Orange and blue denote the test and train loss of the classification network respectively. Pink and purple show the test and train categorization loss of the combined model.

But when we had a look at the accuracy, we saw that both models overfitted the data (Fig 5).

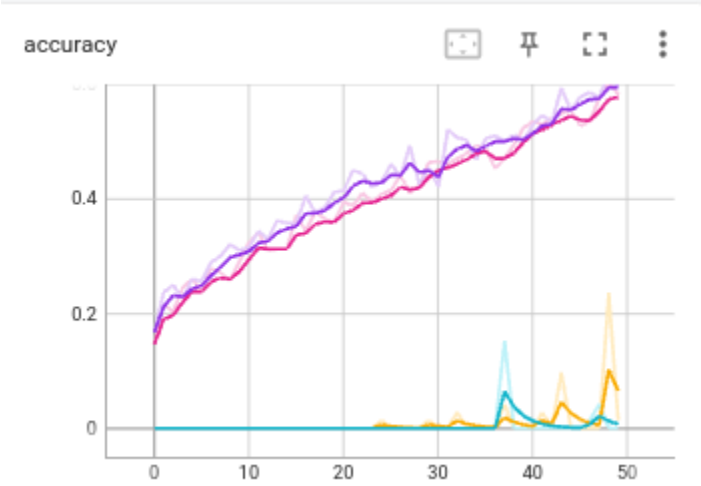


Figure 5. Categorization accuracy. Orange and blue denote the test and train accuracy of the classification network respectively. Pink and purple show the test and train loss of the combined model.

The performance on the colorization task is hard to measure in numbers, but a subjective analysis showed that the models were able to learn to colorize the images, although not perfectly. The combined model showed good saturation of the colors although it sometimes got colors wrong (Fig 6A). The colorization model showed noise in the predicted images and had less vibrant colors (Fig 6B).

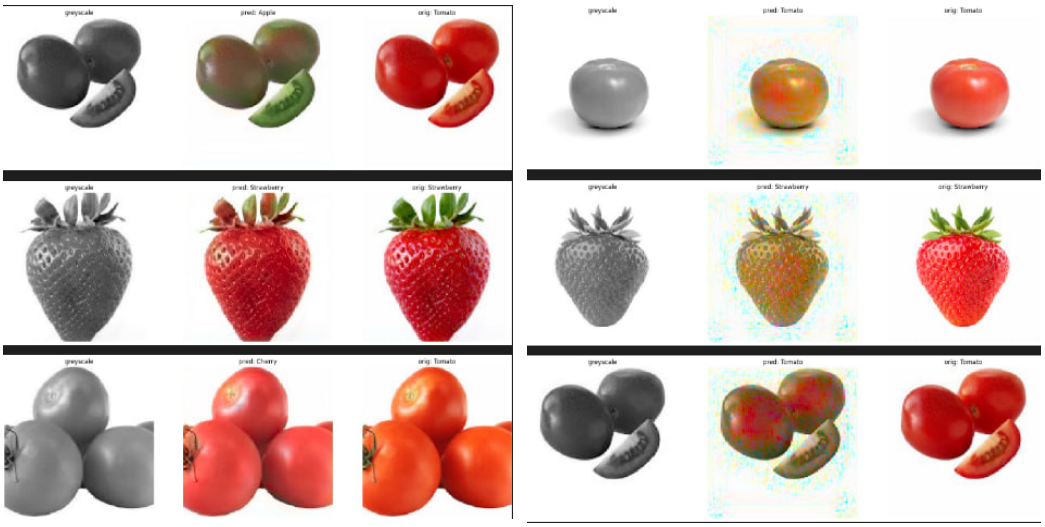


Figure 6. Colorization results for Natural-Color dataset from the standard architecture. The greyscale images are inputs to the model. In the middle are the predictions of the model, and on the right are the original images. 6A shows the predictions of the combined model. 6B shows the predictions of the colorization model.

After confirming that the model was implemented correctly, we applied the architecture to the Stanford Dogs Dataset. We hypothesized that the classification task in the Natural-Color Dataset was too easy and the dataset too small. The bigger and more complex dataset with images of dogs could be more prone against overfitting.

The model was trained on the Stanford Dogs Dataset for 100 epochs. The categorization accuracy on the test dataset for both the combined and classification model were 0.

To evaluate the colorization, we had a look at the predicted images (Fig 7, 8). All predicted images used for evaluation are from the test set. At first glance, the predictions look reasonable, just a bit unsaturated. But we noticed that both models just put a somewhat beige filter on top of the images. When looking closer at the images, visual noise can be seen quite clearly.

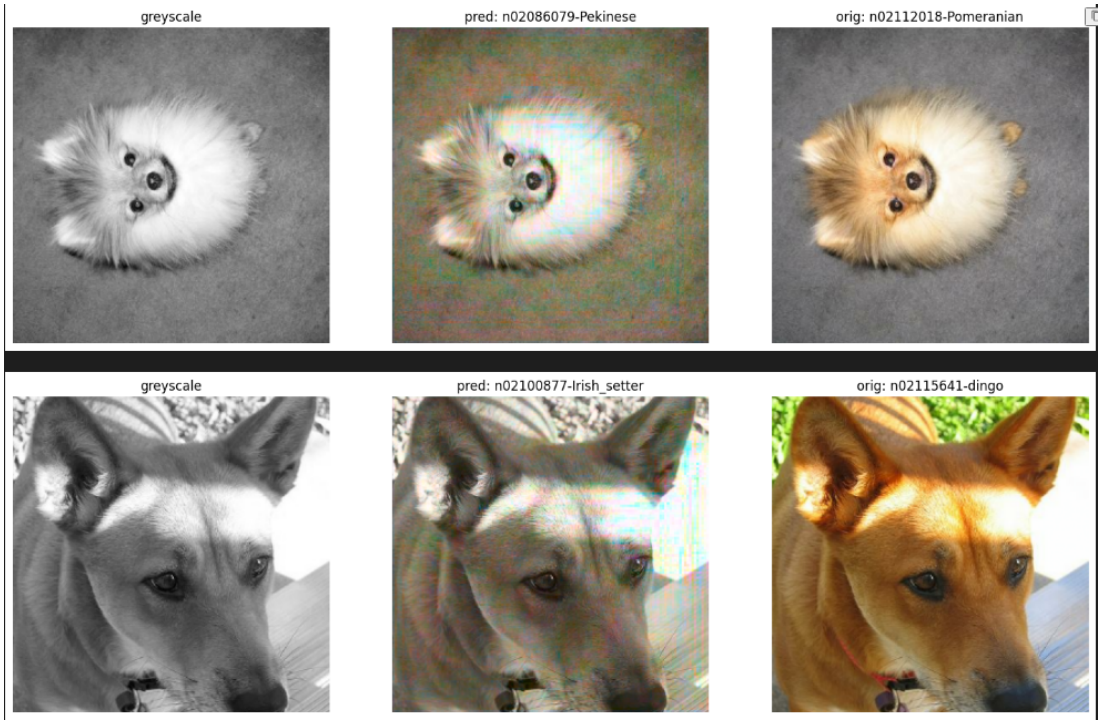


Figure 7. Colorization results for Stanford Dog dataset with the basic architecture. Predictions of the combined model are shown. The greyscale images are inputs to the model. In the middle are the predictions of the model, and on the right are the original images.

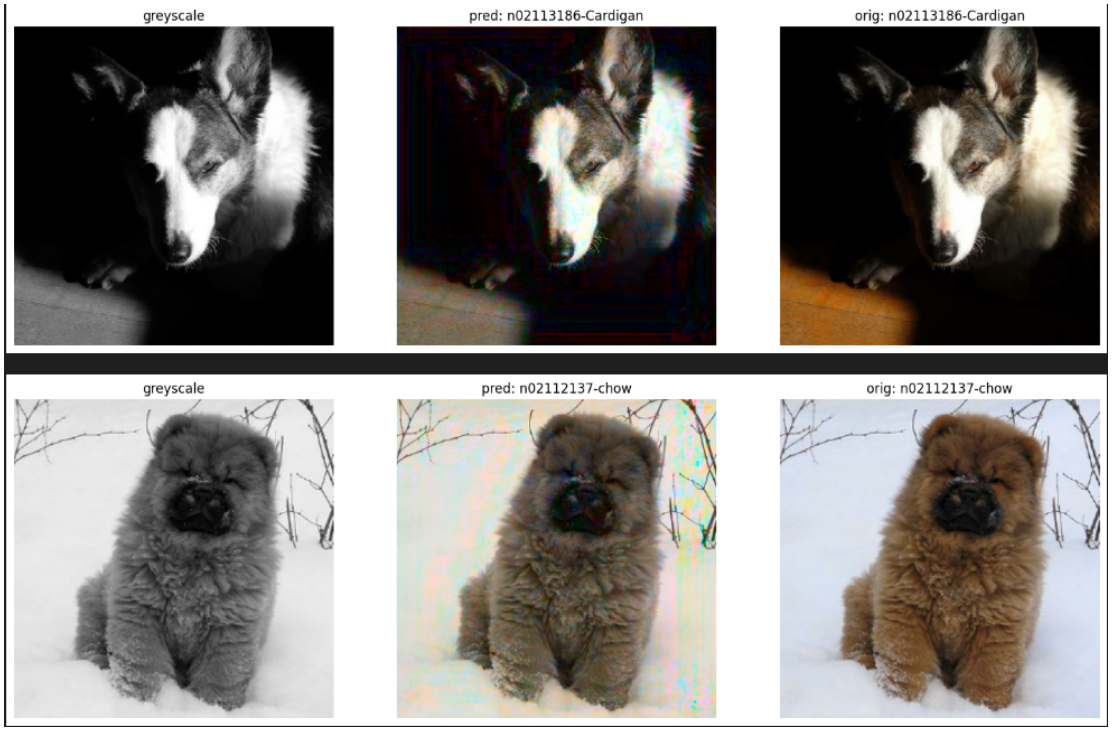


Figure 8. Colorization results for Stanford Dog dataset with the basic architecture. Predictions of the combined model are shown. The greyscale images are inputs to the model. In the middle are the predictions of the model, and on the right are the original images.

4.2 Analysis of Failure

To investigate potential reasons for the rather poor performance for the Stanford Dog dataset, we analyzed the image datasets more closely. Considering that the model learned some sort of beige-filter for the images, we hypothesized that the dataset might have a skewed color distribution that exhibits a prevalence of brown-grayish colors as you would expect for dogs since their fur is mostly along that color scheme. Color analysis revealed that indeed the most common colors of the dataset are brown, gray, and green (Fig 9). We hypothesize that if the most common colors of the dataset were more evenly distributed across the color spectrum, the model might learn more saturated and diverse colorization and not just a beige filter.

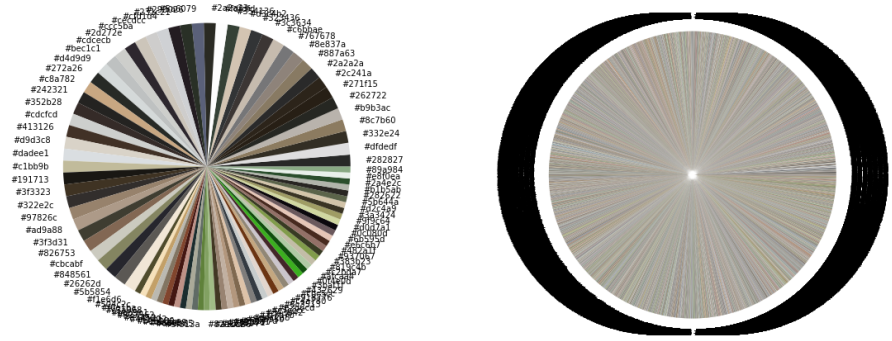


Figure 9. Color Analysis of Stanford Dog dataset. On the left hand side is a pie chart of the 100 most common colors of the dataset (i.e. mostly brown, gray and some green tones). On the right hand side is a pie chart of all colors. The overall impression is clearly gray-brown-ish.

However, there are also other factors that can influence the model and what it learns during training. A qualitative analysis by looking through the images revealed that there are multiple factors that can hinder a model to learn appropriate colorization and classification. Among those factors are: Often the images contain only parts of a dog, or have multiple other objects like humans in them. There is also a huge inter-class similarity and intra-class

dissimilarity, which means that some breeds look alike while other dogs of the same breed look very unalike. The dataset includes 120 dog breeds - the task to classify a dog correctly would be very hard for a human, so it might not be as surprising that the model struggles to properly learn this during only 100 epochs.

When analyzing our model on classification performance, we noticed that it would predict “Bernese Mountain Dog” for every input. Further analysis showed that the training set included more images of Bernese Mountain Dogs than images of other breeds. In the test set no Bernese Mountain Dog images were included. The model learned to predict Bernese Mountain Dog for every image. This explains why the accuracy on the test set was 0.

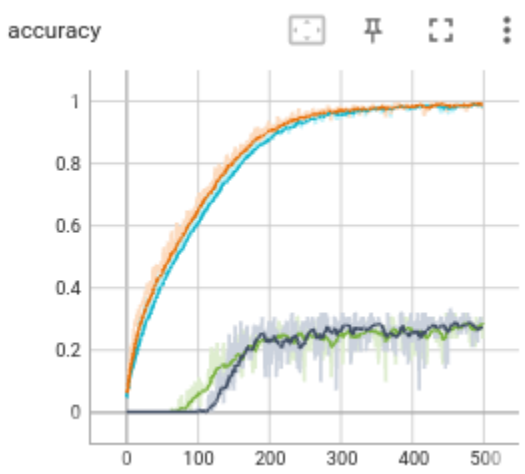
4.3 More Complex Architecture with Adam and Dropout

To counteract the struggles the model had with the Stanford Dogs Dataset, we decided to include other state-of-the-art model components. Since the original model was published in 2016, there are new techniques that can be used. First, to counteract large overfitting, we added Dropout layers in the network. Second, we decided to change the optimizer to ADAM (Kingma & Ba, 2014). This modified architecture will be referred to as *new architecture* below.

To check whether this architecture shows improvements, we let the model train for 500 epochs on the Colorful Dataset. It can be seen that the model overfits on the categorization task again (Figure 10A and 10B).

While also performing better on the training dataset than on the test set, it can be seen (Fig 11) that the test loss for the colorization task decreases for both networks. The loss of the combined model is slightly lower than the loss of the colorization model.

A



B

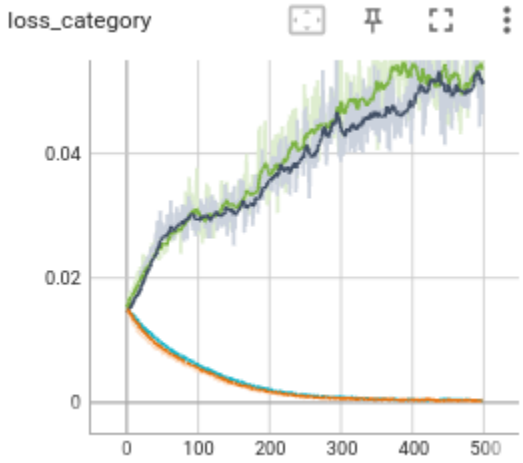


Figure 10. Black and blue denote the performance of the classification model on the test and train dataset respectively. Green and orange show the test and train performance of the combined model. In 10A, the accuracy can be seen. In 10B, the loss of the categorization task is shown.

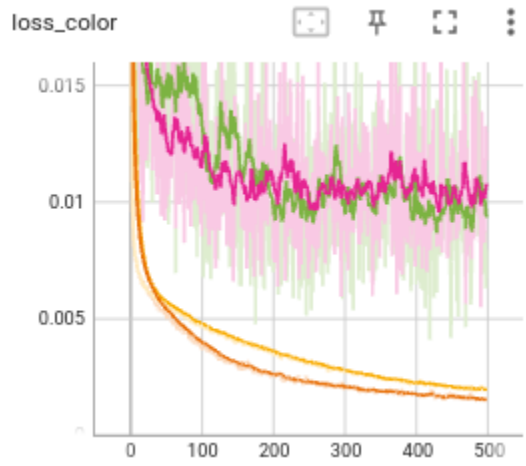


Figure 11. Pink and yellow denote the colorization loss on the test and train dataset respectively. Green and orange show the test and train colorization loss of the combined model.

It is hard to conclude whether the new architecture learned the colorization task properly based on the loss metric, therefore, we had a look at some color predictions again. In Figure 12, we can see that both models are able to colorize the fruits. Compared to the basic architecture, the new colorization model shows no visual noise and has more vibrant colors. But the combined model is better at recognizing fruits as a whole and coloring them in the same color. This might

be because of the better global information that the combined model gets from the classification task. We would say that the colorization task was solved pretty well, especially in the combined model. While the models for classification were not performing well, we hypothesized again that the more diverse and complex Stanford Dogs Dataset was better suited for the classification task.

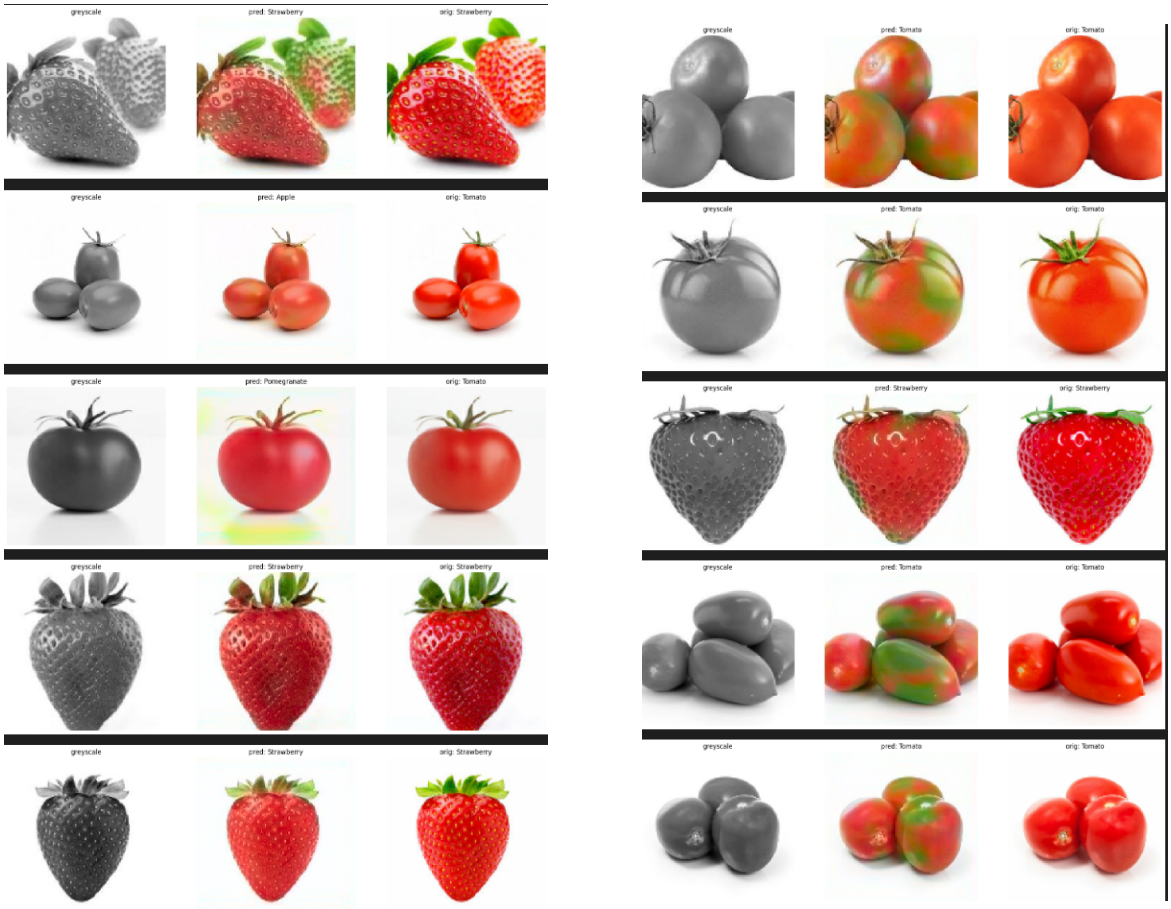


Figure 12. Colorization results for Natural-Color dataset with the new architecture. The greyscale images are inputs to the model. In the middle are the predictions of the model, and on the right are the original images. 12A shows the predictions of the combined model. 12B shows the predictions of the colorization model.

The new model containing Dropout layers and using ADAM optimizer was then trained on the Stanford Dogs Dataset for 100 epochs. The accuracy showed that both models were incapable of learning the correct dog labels. On the test runs, the combined model performed very close to a chance performance of 0.0083 accuracy. The classification model performed

even worse at about 0.0005. On the colorization task, the loss decreased for both the colorization and the combined model (Fig 9). Compared to the models with basic architecture, the new models showed no beige filter (Fig 13). The predicted images from the colorization model, however, showed areas with pink noise. The combined model showed no visual noise and often colored the images realistically.

Figure 13. Colorization results for Stanford Dog Dataset with the new architecture. The greyscale images are inputs to the model. In the middle are the predictions of the model, and on the right are the original images. In A, Predictions of the combined model are shown. In B, predictions of the colorization model are shown.

4.4 Residual Connections

The new architecture showed huge improvements on the colorization task. In order to increase classification performance, we decided to implement a version of the models that included residual connections. Residual connections are a practice of connecting the output of an earlier convolutional layer to the input of a later convolutional layer where the layers in between were skipped by that output. The prior output is then simply summed with the later

input. This brings the information of an earlier layer into later layers of the network and avoids that this information gets lost. These residual connections are known to improve results since they impede overfitting (He et al., 2015).

The code for the ResNet models can also be found in the github repository.

Unfortunately, results for the Natural-Color dataset already showed that the residual connections did not improve model performance. The categorization loss and accuracy (Fig 14) in the combined model showed the same overfitting behavior as the models without residual connections (Fig 10). Additionally, the classification and colorization models with residual connections did not show improvements compared to the models without residual connections either.

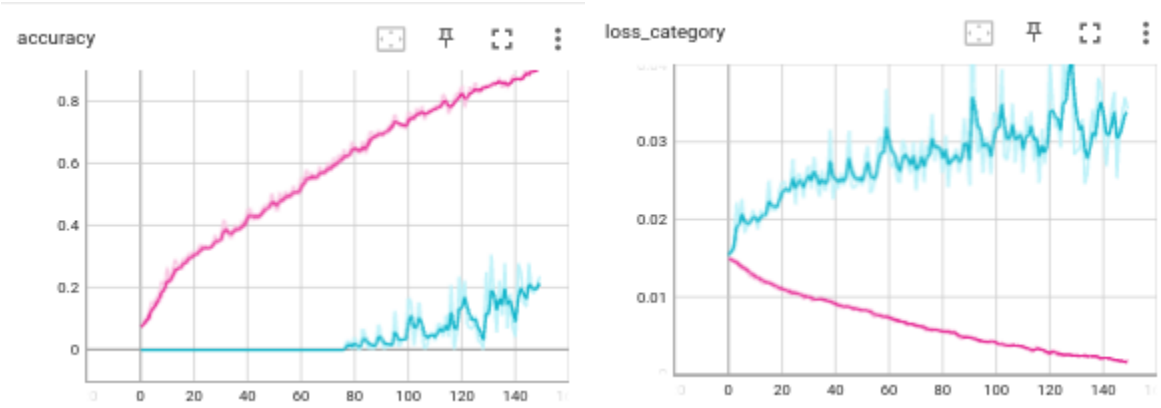


Figure 14. Pink shows performance of the combined model with residual connections on the training set, blue shows performance on the test set. In 14A, the accuracy can be seen. In 14B, the loss of the categorization task is shown.

Because of limited computing resources and the ResNet not looking promising on the Natural-Color dataset, this model was not trained on the dogs dataset.

5 Conclusion

In this paper we presented a CNN based architecture for three models; a colorization model, a classification model, and a combination of the two. We initially tested the colorization and classification models on the Natural-Color Dataset. Our colorization model was successful but results were overfitting in the categorization model. Overfitting could be due to the small size of the dataset. Furthermore, we saw an improvement of colorization in the combined model. Perhaps this is due to classification aiding in identifying an object as a whole and therefore developing a uniform color during colorization. Our next step was to apply our models using the Stanford Dog Dataset. Unfortunately, both colorization and classification models were unsuccessful.

Since we based our project on an architecture created by Iizuka et al. in 2016, we applied some more recent techniques such as using the ADAM optimizer, dropout layers, and residual connections. The new architecture significantly improved results in the colorization task. Yet adding residual connections proved ineffective in increasing classification performance when applied to the Natural-Color Dataset. Owing to these results, we did not implement it on the Stanford Dog Dataset due to limited computational resources.

Our results did not show a significant influence of colorization on classification. Furthermore, the losses and accuracies of the combined model in comparison to the single models were relatively similar.

6 Limitations

There were multiple limitations impacting the scope of our project including limited time, computational costs, and the ambiguity of the colorization problem. Computational costs and limited time came hand in hand. Training often took many hours which forced us to train our models throughout the entire day or leave running overnight. As our assignment included the

requirement for one to two weeks of work, we used this as a guideline. Additionally, the colorization task is considered to be ambiguous (Iizuka et al., 2016). In the case of dogs, when using the Stanford Dog Dataset, colors vary within the breed itself and therefore multiple colors can be considered realistic. This leads to the model often using the most dominant colors it has learned which in this case seems to be beige.

7 References

Ajit, A., Acharya K., & Samanta, A. (2020). A Review of Convolutional Neural Networks. *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, pp. 1-5, https://doi.org/10.1109/ic-ETITE47903.2020.049

Anwar, S., Tahir, M., Li, C., Mian, A., Khan, F. S., & Muzaffar, A. W. (2020). Image colorization: A survey and dataset. *arXiv preprint arXiv:2008.10774*. https://github.com/saeed-anwar/ColorSurvey

Baldassarre, F., Morín, D. G., & Rodés-Guirao, L. (2017). Deep koalarization: Image colorization using cnns and inception-resnet-v2. *arXiv preprint arXiv:1712.03400*.

Bhatt D., Patel C., Talsania H., Patel J., Vaghela R., Pandya S., Modi K., Ghayvat H. (2021). CNN Variants for Computer Vision: History, Architecture, Application, Challenges and Future Scope. *Electronics 10*(20):2470. https://doi.org/10.3390/electronics10202470

Gowda, S. N., & Yuan, C. (2019). ColorNet: Investigating the importance of color spaces for image classification. In *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part IV 14* (pp. 581-596). Springer International Publishing.

Goyal, M., Knackstedt, T., Yan, S., & Hassanpour, S. (2020). Artificial intelligence-based image classification methods for diagnosis of skin cancer: Challenges and opportunities. *Computers in Biology and Medicine*, *127*, 104065.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770-778. https://doi.org/10.48550/arXiv.1512.03385

Huang, S., Jin, X., Jiang, Q., & Liu, L. (2022). Deep learning for image colorization: Current and future prospects. *Engineering Applications of Artificial Intelligence*, *114*, 105006.

Iizuka, S., Simo-Serra, E., & Ishikawa, H. (2016). Let there be color! Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. *ACM Transactions on Graphics (ToG)*, *35*(4), 1-11. https://github.com/satoshiiizuka/siggraph2016_colorization

Khosla, A., Jayadevaprakash, N., Yao, B. & Fei-Fei, L. (2011). Novel dataset for Fine-Grained Image Categorization. *First Workshop on Fine-Grained Visual Categorization (FGVC), IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Accessible at http://vision.stanford.edu/aditya86/ImageNetDogs/

Kingma, D.P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. *CoRR, abs/1412.6980*. https://doi.org/10.48550/arXiv.1412.6980

LeCun, Y., Kavukcuoglu, K., and Cl´ement Farabet, C. (2010). Convolutional networks and applications in vision. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pages 253–256. https://doi.org/10.1109/ISCAS.2010.5537907

Lei, X., Pan, H., & Huang, X. (2019). A dilated CNN model for image classification. *IEEE Access*, *7*, 124087-124095.

Mohapatra, R. K., Majhi, B., & Jena, S. K. (2015, December). Classification performance analysis of mnist dataset utilizing a multi-resolution technique. In *2015 International Conference on Computing, Communication and Security (ICCCS)* (pp. 1-5). IEEE.

Redmon, J., Divvala, S.K., Girshick, R.B., & Farhadi, A. (2015). You Only Look Once: Unified, Real-Time Object Detection. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 779-788. https://doi.org/10.48550/arXiv.1506.02640

Shin, M.C., Chang, K.I. and Tsap, L.V. (2002). Does colorspace transformation make any difference on skin detection?. In *Applications of Computer Vision,* 2002.(WACV 2002). Proceedings. Sixth IEEE Workshop on (pp. 275-279). IEEE.

Zeiler, M. D. (2012). ADADELTA: an adaptive learning rate method. *CoRR abs/1212.5701.* https://doi.org/10.48550/arXiv.1212.5701

Zhang, Y., & Yang, Q. (2022). A Survey on Multi-Task Learning. *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 12, pp. 5586-5609. doi: 10.1109/TKDE.2021.3070203.