# Image Colorization: A Convolutional Network Approach

Nitesh Pradhan[1], Saransh Gupta[1], Gaurav[1]

[1] Manipal University Jaipur, Jaipur, India
nitesh.pradhan@jaipur.manipal.edu

Abstract. In recent days, due to some application of image colorization as automatic coloring of old pictures, correctly identification of thief from sketch image etc., image colorization has become hottest area in research domain. Machine learning plays an essential role in this concern. This paper illustrates deep convolutional neural network approach with VGG16 pre-trained classifier that takes gray scale image as input image and gives its equivalent colored image as output image. In this research, deep convolutional neural network is categorized into encoder, fusion and decoder part and VGG16 pre-trained model is used for extracting high level features from an image. The proposed network is trained till 2000 epochs. The final colored image is compared with ground truth image.

**Keywords:** Image colorization; LAB color space; convolutional neural network; deep learning; Classifier.

.

## 1 Introduction

Computer vision and deep learning is a powerful domain with many applications such as image generation from the text description, image to image translation, face aging, face generation, image colorization from gray image or black-and-white image etc. Normally image colorization problem is defined in term of International commission on Illumination (CIE) LAB color space. As red, green, blue color space, CIE color space also a 3-channel (a, b and L) space where a and b used to encode the color information in green-red component and blue-yellow component respectively and L channel used to encode the intensity information of an image.

On the contrary to Cyan, Magenta, Yellow, Key (CMYK) and Red, Green, Blue (RGB) models of shading, Lab color model aims to inexact human vision. It mainly focusses to bring about perceptual consistency. The L component helps in the coordination of the human impression of delicacy such that it doesn't produce the Helmholtz– Kohlrausch results into record. Hence by altering A and B channels, optimal and precise shading balance can be obtained. Generally, people use Photoshop for image colorization which is long and tedious task because a face alone needs up to 20 layers of blue, green and pink shades to get it just right. Image colorization has many

practical applications such as colorizing old movies or photographs, color recovery, artist assistance and color image encoding.

## 2 State of Art

In 2016, [Zhang et. al. (2016)] designed a fully automatic algorithm, that produced exceptionally realistic results. At training time, they used classification and class-rebalancing to incorporate greater diversity of colors in the result. When model is given the lightness channel L as input, for an image it calculates the corresponding a* and b* color channels. At test time entire network is a feed-forward pass in a CNN. The model is trained on more than a million color images. Evaluated the results using a 'Colorization Turing Test," where the people were asked to select between a generated and ground truth color image. Method successfully managed to befool people on 32% of the trials.

The algorithm designed by [Iizuka et al. (2016)] which had four main components: a low-level features network, a mid-level features network, a global features network and a colorization network. Set of Low-level features extracted from the image are used to compute the sets of global image features and mid-level image features. "Fusion layer" is used to combine or fuse the mid-level and the global features. These fused features are used as an input to the colorization network that finally outputs the colored image. The entire network learns in an end-to-end fashion. CIE L*a*b* color-space is used for the training images. No such pre-processing or post-processing is required. Main features of the proposed architecture are:

- It jointly learns global and local features for an image using an end to end approach.
- Classification labels are exploited to increase proposed network performance.
- A style transfer technique used.

Trained the model on the Places scene dataset, which consists of 2,448,872 training images and 20,500 validation images distributed among 205 subjects corresponding to the various types of scenes such as volcano, conference center etc. Evaluated the model through a user study and found that the output of the model is considered "natural" 92.6% of the time.

In 2017, [Zhang et al. (2017)] again, presented a deep learning algorithm for user guided image colorization. Given the grayscale version and user inputs, the designed network predicted the color of an image. The proposed model uses two networks first the Local Hints Network which uses sparse user points and second the Global Hints Network which uses global statistics. The model has been trained on nearly a million images, with simulated user inputs. A single feed-forward pass is used to perform the colorization, making it considerably fast and enabling real-time use.

In the same year 2017, later on [Baldassarre et al. (2017)] used the CIE L*a*b* color space for the images, where L, the luminance contains all the main features of

the image and a*b*called Chromaticity contains all the color information of the image. Used a pre trained Inception ResnetV2 for high level feature extraction to obtain an intuition about the contents of the image Decoder Encoder convolution network used for coloring the image. Network when provided with the luminance component of an image as the input, the model outputs a*b* components which are combined with the input to obtain the final colored image. Size of the training dataset is kept small, which restricts the model to small variety of images. Training results were presented by assessing the public acceptance evaluation of the images generated by the network through a user study.

In the last few years, Convolution Neural Network has shown tremendous advancement in the object detection and classification. Experiments and research show that CNN's have almost reduced the error rate for object detection to half [Krizhevsky et al. (2012)].

The ImageNet dataset is a large collection of images designed for research in visual object recognition. It consists of more than 14 million hand –annotated images which indicate what objects are present in the picture and in at least one million of the images, bounding boxes are also provided. ImageNet dataset is distributed among more than 20,000 classes.

Since 2010, the ImageNet organizes an annual contest called the ImageNet Large Scale Visual Recognition Challenge (ILSVRC).Algorithms for object detection and image classification at large scale are evaluated in The ImageNet Challenge (ILSVRC) [Russakovsky et al. (2015)]. A specially "trimmed" list of one thousand non-overlapping classes is used for the challenge. The ILSVRC follows the footsteps of the PASCAL VOC challenge [Everingham et al. (2010)] established in 2005, which had a smaller dataset of about 20,000 spread across twenty object classes. VGGNet [Simonyan et al. (2014)] was invented by Visual Geometry Group (VGG) from University of Oxford.

VGG 16 emerged as the winner of ILSVRC 2014 in the classification task. VGG-16 obtains 8.8% error rate on the ImageNet Dataset. VGG16 had been used for malicious software classification based on deep neural network bottleneck feature [Rezende et al. (2018)]. Apart from VGG16, ResNet-50 [He Kaiming et al. (2015), Szegedy et al. (2016)] used transfer learning concept for malicious software classification in deep neural network era [Rezende et al. (2017)]. VGG16 based fully convolutional structure has been used to classify the weld defect image [Liu Bin et al. (2018)] which achieves a high accuracy with a relative small dataset. Testing dataset had 3000 images. Achieved a test accuracy of 97.6% and train accuracy of 100 % on two main defects.

A common observation after reviewing all these models is that the models which take 'global features' under consideration outperforms the one who does not. This is because global features provide information about what is present in the image and helps in mapping the detected objects with their corresponding probabilities. This further helps the colorization network to learn what kind of colors are possible for what specific kind of object/image. Most of the reviewed papers have used a classifier network to extract global features and a separate colorization network to color the

image on the basis of high-level features (global features) [Iizuka et al. (2016)], Baldassarre et al. (2017)].

Highly inspired by such an architecture authors decided to design a model which uses a Classification network to determine object type and category under consideration and a Colorization network which estimates the output colors on the basis of high-level and mid-level features. But instead of training a model from scratch to extract global features, authors decided to use a pre trained network for that, which would reduce the training time.

# 3 Methodology

To perform the Image colorization on gray scale image hyper parameters, pre-processing and network architecture are explained in further sub-sections.

## 3.1 Hyper Parameter

Rectified liner unit (ReLu) has been used as the activation function for all the layers of the encoder as well as decoder network except the last layer. tanh has been used in the last layer to map the predicted values in the same interval as of the real values. This is done for the easy comparison between the predicted values and the real values. Since the real values (values for A and B channels) lies in the range of [-1, 1], tanh is used since it outputs the values in the same range for any input given to it.
Over the estimated and target output, the ideal model parameter is identified by minimizing the objective function. To quantify the loss of the model, Mean Square Error is used between estimated pixel values and its real values. Mean Square Error for an input image X is defined by equation (1).

$$C(\mathrm{X},\theta) = \frac{1}{2\mathrm{HW}} \sum_{\mathrm{k}\in(\mathrm{a,b})} \sum_{\mathrm{i}=1}^{\mathrm{H}} \sum_{\mathrm{j}=1}^{\mathrm{W}} \left(\mathrm{X}_{\mathrm{k_{ij}}} - \widetilde{\mathrm{X}}_{\mathrm{k_{ij}}}\right)^2 \quad (1)$$

All model parameter represented by θ. $X_{k_{ij}}$ and $\tilde{X}_{k_{ij}}$ indicate the pixel estimation of the ij :th pixel estimation of the k: th segment of the objective and reproduced picture, separately. During training, Adam Optimizer is used to back-propagate the loss to update the model parameters with an initial learning rate η = 0.001.

## 3.2 Data Pre-processing

All the training images have been pre-processed before feeding them to the convolution network. Pre-processing is required at two ends namely colorization end and classification end.

- **Colorization End**

All the images present in the training dataset are in RGB format having pixel values in a range of [0, 255]. As the first step of pre-processing the pixel values are brought in a range of [0, 1] by dividing them by 255. It helps in reducing the convergence time of the network loss.
Secondly, the images are converted in CIELAB (also known as CIE L*a*b* or sometimes abbreviated as simply "Lab") color space. It is due to the following reasons:
L stands for Lightness or Luminance channel which is the grayscale version of the image.
a and b represent the Chrominance where a is for red-green and b is for blue-yellow.
Converting it to Lab reduces the problem to the determination of only two extra color channels i.e. a & b whereas in RGB it would have been 3 channels as red, green and blue.
The L layer of the image which will be our input (as it is nothing but the grayscale values) can also be used in our final step of merging the L, a and b layers.
In the Lab color space format, an image of size H X W is passed as input. To generate the complete color image $\widetilde{\boldsymbol{X}} \in \boldsymbol{R}^{\boldsymbol{H}*\boldsymbol{W}*\boldsymbol{3}}$, luminance and a & b component required. So authors are passing a luminance component to the model by which model will predict a & b components. The relationship between luminance and a & b are defined in equation (2). Figure 1 shows the L*a*b color space with respective to the RGB image.

$$F : X_L \rightarrow \left(\tilde{X}_a, \tilde{X}_b\right) \tag{2}$$


**Fig. 1.** RGB to LAB format

- **Classification End**

Since VGG 16 takes 224 X 224 images with 3 channels as input. All the images are resized to 224 * 224 and converted to grayscale (after converting to grayscale, the image is then converted to RGB to get three channels in grayscale) before passing them to the classification end as shown in figure 2.


**Fig. 2.** RGB to grayscale conversion

- **Image Augmentation**

It is required to avoid over fitting. Since our training dataset is small with only 10000 images, the model can learn the details and noise in our training dataset to a large extent and cannot generalize the features when working on the test dataset. Over fitting adversely affects the performance of the network on the test set. Image Augmentations techniques are methods of artificially increasing the variations of images in our dataset by using horizontal/vertical flips, rotations, and variations in the brightness of images, horizontal/vertical shifts, etc. It creates random batches of training data.

### 3.3 Network Architecture

Convolutional Neural Network (ConvNet/CNN) [O'Shea et al. (2015)] is a type of Deep Learning algorithm which takes an image as input, extracts features from it and assigns weights and biases to various aspects/objects in the image and is able to differentiate one from the other. ConvNets have the ability to automatically extract the features/characteristics from the images whereas on the contrary classical classification algorithms require manually designed filters for the same. Hence convolution neural network manages everything in an end to end fashion. Figure 3 represents the used network architecture for image colorization. The network architecture comprises of two different networks, the classification network, and the colorization network.

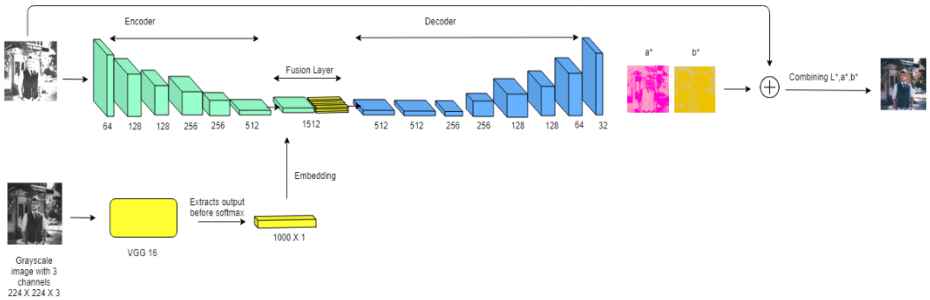


**Fig. 3.** Network Architecture

- **Classification Network**

From the various pre-trained classification networks like Xception [Chollet et al. (2017)], VGG16, VGG19, ResNet50, InceptionV3, InceptionResNetV2, Mobile Net [G. Howard et al. (2017)], author choose VGG 16, which is the winner of ILSRVC 2014. The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes.
VGG 16 is an improvement over AlexNet [Krizhevsky et al. (2012)] by replacing the large filters with size 11 X 11 and 5 X 5 in first and second layers respectively by multiple smaller filters of size 3 X 3 one after the other. VGG 16 takes 224 X 224 X 3 images as input. All the images are resized to 224 X 224 2 which is already discussed under the pre-processing section.
The output (embedding) of the last classification layer just before the softmax function is extracted from VGG 16. It is 1000 X 1 dimensional list or column vector. It represents the probabilistic measure of what all objects/aspects are present in the image. This high level of information helps to predict more realistic colors for the objects in the images. For example, if the network knows its sky then it would eventually color it blue rather than coloring it with any arbitrary color.
The output from the classification network is merged with the output from the encoder part of the colorization network which will be discussed under the colorization network. Since VGG 16 is pre-trained on 1000 classes and ImageNet dataset, transfer learning can be used to train it on smaller datasets with lesser number of classes. Generally, it is helpful when working with a specific type of datasets like cats and dogs or the natural scenes datasets etc.

- **Colorization Network**

Colorization network is logically divided into three main components. The encoder to extract the mid-level features whereas fusion layers to merge the mid-level features with high-level features obtained from the classification network and decoder to use these features to predict the values for the color channels.
**Encoder:** The Encoder takes H × W (256 X 256 tensor with 1 channel) gray-scale images as input. The gray scale image is, in fact, the luminance component (L*) in Lab color space. It uses six convolutional layers with 3 × 3 kernels each. Every convolution layer uses a stride size of 2 consequently halving the dimension of their output and hence reducing the number of computations required. Padding is used to ensure that the images are exactly halved. A batch normalization layer is placed symmetrically between the convolution layers right after the third convolution layer as shown in Table 1.
The first three convolution layers having a smaller number of filters are used to detect low-level features. The next three convolution layer having a higher number of filters are used to detect mid-level and high-level features using the low level features obtained from the previous three layers. It outputs H/64 X W/64 X 512 feature maps representing a combination of low-level and mid-level features.

**Table 1.** Layer structure for Encoder

| Layer | Kernel Size | Stride Size | Output Image Size |
|---|---|---|---|
| Conv2D | 64 X (3 X 3) | 2 X 2 | H/2 X W/2 |
| Conv2D | 128 X (3 X 3) | 2 X 2 | H/4 X W/4 |
| Conv2D | 128 X (3 X 3) | 2 X 2 | H/8 X W/8 |
| Batch Norm | - | - | H/8 X W/8 |
| Conv2D | 256 X (3 X 3) | 2 X 2 | H/16 X W/16 |
| Conv2D | 512 X (3 X 3) | 2 X 2 | H/32 X W/32 |
| Conv2D | 512 X (3 X 3) | 2 X 2 | H/64 X W/64 |

**Fusion Layer:** The fusion layer takes the feature maps vector from the Classification network, replicates it H/64*W/64 times and combines it to the feature maps or to the vector obtained as output from the encoder part of the Colorization network. Along the depth axis, the feature vector or map obtained which has dimensions of H/64 X W/64 X 1512. This helps to obtain a single feature map or vector containing both the mid-level and high-level features obtained from Classification network and the Encoder respectively. This also ensures even distribution of information throughout the image. After this a convolution layer with 512 kernels or filters of size 1X1, with stride size 1 is applied on the feature vector obtained after fusion, to obtain a feature volume of dimensions H/64 X W/64 X 512. Table 2 shows the layer structure of the fusion layer.

**Table 2.** Layer Structure of Fusion Layer

| Layer | Kernel Size | Stride Size | Output Image Size |
|---|---|---|---|
| Conv2D | 512 X (1 X 1) | 1 X 1 | H/64 X W/64 |

**Decoder:** The decoder takes H/64 × W/64 × 512 feature volume as input and applies De Convolution using Transpose Convolution Layer. It uses eight Transpose Convolution layers with 3 X 3 kernel size each and a stride size of 2 for all the layers except the first and the last layer as shown in Table 3. For every layer with a stride size of 2, the size of the feature volume is doubled to match the original image dimensions. The number of kernels is gradually decreased from 512 to 2 because finally, out target is to predict two channels a* and b*.

**Table 3**. Layer Structure of Decoder

| Layer | Kernel Size | Stride Size | Output Image Size |
|---|---|---|---|
| Conv2DTrans | 512 X (3 X 3) | 1 X 1 | H/64 X W/64 |
| Conv2DTrans | 256 X (3 X 3) | 2 X 2 | H/32 X W/32 |
| Conv2DTrans | 256 X (3 X 3) | 2 X 2 | H/16 X W/16 |
| Conv2DTrans | 128 X (3 X 3) | 2 X 2 | H/8 X W/8 |
| Batch Norm | - | - | H/8 X W/8 |
| Conv2DTrans | 128 X (3 X 3) | 2 X 2 | H/4 X W/4 |
| Conv2DTrans | 64 X (3 X 3) | 2 X 2 | H/2 X W/2 |
| Conv2DTrans | 32 X (3 X 3) | 2 X 2 | H X W |
| Conv2DTrans | 2 X (1 X 1) | 1 X 1 | H X W |

# 4 Experimental Results and Discussion

## 4.1 About Dataset

The results obtained from the deep learning networks depend almost equally on the network architecture as well as on the dataset. Choice of the dataset has a significant role in determining the parameters like training accuracy, training loss, validation accuracy, and validation loss. Moreover, the size of the dataset can also give a hint about whether a model is over fitted or under fitted.

Most of the previous automatic colorization models [Larsson et al. (2016)] have used the easily and extensively available ImageNet dataset. ImageNet's huge size and free availability make it a good choice for our work. Authors used the publicly available dataset on FloydHub which us available at www.floydhub.com/emilwallner/datasets/colornet. The images in the dataset are collected from Unsplash, a platform which provides free high-quality images clicked by professional photographers. The dataset has a great diversity of images ranging from natural scenes to human faces, animals to gadgets. It approximately contains 9500 training images and 500 validation images all having a uniform size of 256 X 256.

## 4.2 System Configuration

The model is trained on the Google Colab, which is a free online training platform. Google Colab uses Tesla K80 GPU and provides 12 GB RAM and 12 hours of continuous use. Due to RAM limitations the batch size is kept small.

## 4.3 Result and Discussion

Colorization of grayscale images is considered as a problem with no 'correct' answer. For example, a red chair is indistinguishable from a blue chair when photographed in black-and-white. Here, authors present a method using deep learning tech-

niques to colorize a grayscale image and produce a colored image with possible colors. If the output is able to fool human eyes then it is considered as a valid output.

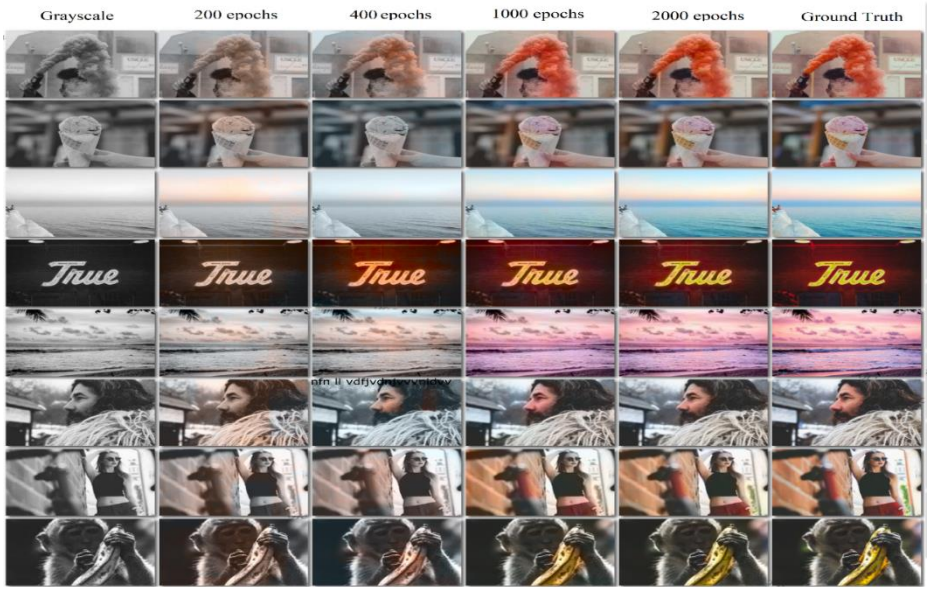


**Fig. 4.** Obtained result compared with ground truth image

The results were exceptionally good on a certain set of images, where the network generated almost the ground truth. The estimated color improved gradually as the number of epochs was increased up to a limit of 2000 epochs.

But due to the small size of the training dataset, it was limited to coloring only the main objects which were easily detected by the classifier network. Hence, all the objects in the image weren't colored. Our model is trained for 2000 epochs with a batch size of 32 as shown in figure 4.

## 5 Conclusion and Future Work

Deep learning plays an important role in image colorization task. In this paper, a novel approach introduced for image colorization which first classifies the image and then performs colorization on it. The proposed method applied to several images of different category. It is observed that introduced network perform better if the image belongs to the natural scenes category such as sky, sea, etc. The reason is, in our dataset maximum images are from natural scenes category. So for unseen images, network highly depends on the dataset. To overcome this issue, the proposed network should be trained with a large dataset which contains images of all categories.

This work can be further extended to be applied on videos like old black and white movies and feed obtained from the CCTV cameras. To get an idea of how the images are perceived by the observer or how compelling the colors look to a human observer, a public survey can be conducted asking the people to label the images colored by our network as fake or real. This can help in calculating the accuracy of the model. The accuracy of the model can be calculated by how many times the colored images are able to fool the human eye.

## References


1. Zhang Richard, Isola, Phillip & Efros, Alexei. (2016). Colorful Image Colorization. 9907. 649-666. 10.1007/978-3-319-46487-9_40.
2. Iizuka, Satoshi & Simo-Serra, Edgar & Ishikawa, Hiroshi. (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. 35. 1-11. 10.1145/2897824.2925974.
3. Zhang Richard & Zhu, Jun-Yan & Isola, Phillip & Geng, Xinyang & Lin, Angela & Yu, Tianhe & Efros, Alexei. (2017). Real-Time User-Guided Image Colorization with Learned Deep Priors. ACM Transactions on Graphics. 36. 10.1145/3072959.3073703.
4. Baldassarre, Federico & Gonzalez Morin, Diego & Rodés-Guirao, Lucas. (2017). Deep Koalarization: Image Colorization using CNNs and Inception-ResNet-v2.
5. Krizhevsky, Alex & Sutskever, Ilya & E. Hinton, Geoffrey. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Neural Information Processing Systems. 25. 10.1145/3065386.
6. Russakovsky, Olga & Deng, J & Su, Hao & Krause, J & Satheesh, Sanjeev & Ma, S & Huang, Z & Karpathy, A & Khosla, A & Bernstein, M. (2015). ImageNet Large Scale Visual Recognition Challenge. Int. J. Comput. Vis.. 115. 1-42.
7. Everingham, Mark & Van Gool, Luc & K. I. Williams, Christopher & Winn, John & Zisserman, Andrew. (2010). The Pascal Visual Object Classes (VOC) challenge. International Journal of Computer Vision. 88. 303-338. 10.1007/s11263-009-0275-4.
8. Simonyan, Karen & Zisserman, Andrew. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv 1409.1556.
9. Rezende, Edmar & Ruppert, Guilherme & Carvalho, Tiago & Theophilo, Antonio & Ramos, Fabio & De Geus, Paulo. (2018). Malicious Software Classification Using VGG16 Deep Neural Network's Bottleneck Features. 10.1007/978-3-319-77028-4_9.
10. He, Kaiming & Zhang, Xiangyu & Ren, Shaoqing & Sun, Jian. (2015). Deep Residual Learning for Image Recognition. 7.
11. Rezende, Edmar & Ruppert, Guilherme & Carvalho, Tiago & Ramos, Fabio & De Geus, Paulo. (2017). Malicious Software Classification Using Transfer Learning of ResNet-50 Deep Neural Network. 10.1109/ICMLA.2017.00-19.
12. Szegedy, Christian & Ioffe, Sergey & Vanhoucke, Vincent. (2016). Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. AAAI Conference on Artificial Intelligence.
13. Liu, Bin & Zhang, Xiaoyun & Gao, Zhiyong & Chen, Li. (2018). Weld Defect Images Classification with VGG16-Based Neural Network. 10.1007/978-981-10-8108-8_20.
14. O'Shea, Keiron & Nash, Ryan. (2015). An Introduction to Convolutional Neural Networks. ArXiv e-prints.

15. Chollet, Francois. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. 1800-1807. 10.1109/CVPR.2017.195.
16. G. Howard, Andrew & Zhu, Menglong & Chen, Bo & Kalenichenko, Dmitry & Wang, Weijun & Weyand, Tobias & Andreetto, Marco & Adam, Hartwig. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.
17. Larsson, Gustav & Maire, Michael & Shakhnarovich, Gregory. (2016). Learning Representations for Automatic Colorization. 9908. 577-593. 10.1007/978-3-319-46493-0_35.