Full Color Deep Networks

Nishad Tahur

Delft University of Technology

TU Delft

Full Color Deep Networks

by

Nishad Tahur

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday June 28, 2022 at 10:30 AM.

Student number:	4681517	
Project duration:	November 8, 2021 – July 1, 2022	
Thesis committee:	Dr. J.C. van Gemert,	TU Delft, thesis advisor
	Dr. N. Tömen,	TU Delft, daily supervisor
	Dr. T. Höllt,	TU Delft

TU Delft

Preface

This report contains the findings of my research on Full Color Deep Networks to obtain the degree of Master of Science at the Delft University of Technology. The main content of this report is the scientific paper presented in Chapter 1. The scientific paper presents a proposed architecture to incorporate a color dimension for neural networks to work with throughout all of its layers. The chapters following the scientific paper contain additional background information, details and experiments. The research has been conducted under the Pattern Recognition and Bioinformatics group, in particular the Computer Vision Lab.

Firstly I would like to thank Attila Lengyel for presenting me the opportunity to research this topic. I am also grateful for his guidance throughout the research, and even though it was cut a little short, it has helped me considerable amounts. I would also like to thank Dr. N. Tömen who has also supervised me throughout the whole research and helped me stay focused when I drifted off. I also appreciate the time spent on reviewing my work and providing useful feedback on how to write scientifically. I express my gratitude towards Dr. J.C. van Gemert, my thesis advisor, who helped me keep a critical mind towards my findings. I also thank Dr. T. Höllt for agreeing to be part of my thesis committee and taking time out of his busy schedule to review my report.

Finally, I would like to thank my friends and family for keeping me motivated, being understanding while I was busy with my research and helping me unwind after I reached my milestones.

Nishad Tahur
Delft, June 2022

Contents

1

Scientific Paper

Full Color Deep Networks

Nishad Tahur
Delft University of Technology
Delft, The Netherlands
i.h.n.tahur@student.tudelft.nl

Abstract

Color information has been shown to provide useful information during image classification. Yet current popular deep convolutional neural networks use 2-dimensional convolutional layers. The first 2-dimensional convolutional layer in the network combines the color channels of the input images, which produces feature maps per channel with only spatial dimensions, height and width, getting rid of the color dimension. In this work we introduce Full Color Deep networks which use 3-dimensional convolutions to retain the color dimension beyond the first layer. The 3D kernels convolve over the color and spatial dimensions. The network can extract features from all three dimensions in all layers which are subsequently used by the classifier. We show that the Full Color Deep networks perform at least as well as the current CNNs but outperform them in learning color information and using that information in other downstream tasks.

1. Introduction

Color information can be important depending on the image classification task at hand [1]. Typical deep convolutional neural networks (CNNs), however, use 2-dimensional layers such as convolutions and pooling. After this point we refer to such standard CNNs as 2D networks. The 2-dimensional layers collapse over the color dimension by additively superimposing the feature maps in the first layer. The 2D feature maps per channel, in the height and width dimensions, are subsequently used as input for the rest of the layers. This brings forth the question if color information is lost beyond the first convolutional layer. Several works found that color information does play an important role in image classification throughout the whole network and that some filters even have their own role for specific colored input [1, 8]. We believe, however, that color information can play a bigger role in a classification task and that the neural network can learn useful color features by retaining the color dimension. To this end, we propose Full Color Deep networks (FCDNets) which use 3-dimensional convolutional layers. The 3D convolutions enable the network to focus on extracting features in the color dimension of the input image. Figure 1 illustrates the main difference between 2D and FCD networks and a 3D convolution is visualized in Figure 2.

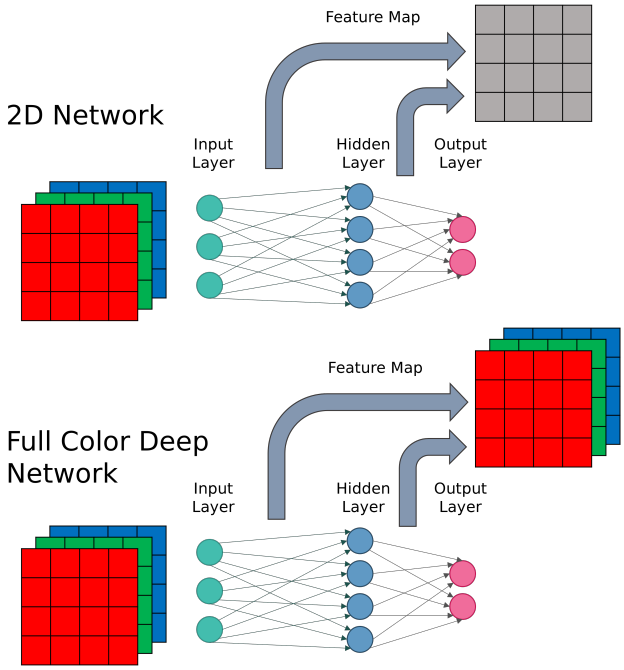

Figure 1. The main difference between standard CNNs (top), which use 2-dimensional convolutions, and our proposed Full Color Deep networks (bottom), which use 3-dimensional convolutions. The Full Color Deep networks maintain feature maps with 3 separate color channels in the third color dimension.

3D CNNs with 3D convolutions are commonly used in other domains such as medical imaging [28], which has a third spatial dimension, or in video recognition [13, 21], which has a third dimension in time. To our knowledge, however, our proposed method is the first to apply 3D convolutions over the color dimension. We have the following

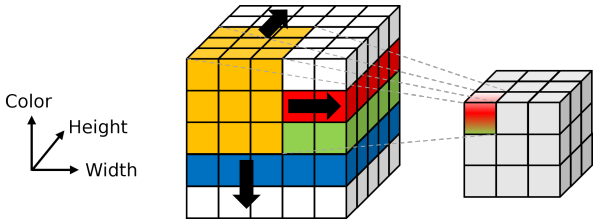

Figure 2. Cube representing an RGB image with zero padding in the color dimension (left). Cube representing the intermediate output of one step in the convolution (right). The yellow cube is the kernel which slides over the color and spatial dimensions.

main contributions. First, we investigate whether an FCD network is viable for 2D image classification. A part of the parameters in the network is dedicated to extracting features in the color dimension, it is thus not trivial to investigate whether its classification performance is on par with the performance of 2D networks. We empirically show that FCD networks are indeed viable for image classification. We also systematically vary the color bias per class in datasets and show how that influences the performance of 2D and FCD networks. For that purpose, we create several toy datasets from the colored versions of MNIST [20] and Fashion-MNIST [31]. The created datasets have a different amount of class specific color information, allowing us to capture the behaviour of the networks with respect to color information. Finally we compare both type of networks and show when an FCD network is a better alternative than the 2D network. We find that for regular tasks on the MNIST and Fashion-MNIST datasets, both types of networks perform comparably. We show, however, that FCD networks are better at learning color information even when it is not necessary for the prediction task. The FCD networks can afterwards apply the learnt color information in other downstream tasks to achieve better performances.

2. Related Work

This section contains related work on current research of how color information is processed in neural networks and in which fields CNNs with 3D convolutional layers are deployed.

2.1. Color Encoding in CNNs

Color has been found to play an important role in image classification [1, 8, 23, 24]. Buhrmester et al. [1] found that color information becomes increasingly important if there are many classes. Engilberge et al. [8] showed results that networks contain class invariant neurons which are sensitive to color. The neurons in the earlier layers are found to be more sensitive than those is later layers. Rafegas et al. [23, 24] demonstrated similar behaviour and presented the insight that even though neurons are less color selective in later layers, the distribution of color selective and non-color selective neurons stays relatively the same throughout all layers. We find color selective neurons being present throughout all layers and 2D networks removing the color dimension after the first layer counter-productive. This motivated us to investigate the influence of retaining the color dimension beyond the first layer of a neural network.

Buhrmester et al. [1] created a formula to measure the color sensitivity of a neuron. Rafegas et al. [23, 24] used a measure which they called the color selectivity index. Both measures compute the sensitivity of a neuron to color by comparing the activation values of a colored image and its grayscale counterpart. This seems to be a useful measure and we will be using the color selectivity index to analyze the performances of the 2D and FCD networks.

2.2. 3D CNNs and Features in the Third Dimension

Currently CNNs with 3D convolutions are deployed in several fields of deep learning. CNNs are deployed in medical visualization to detect abnormalities [17]. Success has been found using 3D CNNs for finding brain hemorrhages in volumetric data for promising results [29]. Kayalibay et al. [15] and Kamnitsas et al. [14] have also demonstrated that 3D CNNs can perform well on a scarce amount of data and even outperform state of the art results on challenging data. Chen et al. [3] transformed the DenseNet [12] to use 3D kernels on a brain tumor segmentation task and achieved state of the art results. Using 3D kernels, they were able to extract features in all three spatial dimensions. In our work, we transform VGGNet-based [27] networks to use 3D kernels in the convolutional layers.

3D CNNs are commonly used in video recognition tasks [32]. Tran et al. [30] and Ji et al. [13] proposed novel 3D CNNs and showed the results on human action recognition datasets. They showed the benefits of using 3D CNNs over 2D CNNs, which included being able to better learn temporal dimension features. 3D CNNs are even able to deliver impressive results on facial emotion recognition in videos [10] which have less obvious characteristics when compared to human action. Just like in image classification, transfer learning is researched for video recognition [2, 26]. We also aim to demonstrate the effects of training on a certain dataset and evaluating the trained networks on other datasets.

3. Method

In this section we start by giving a brief explanation on how images are processed by a deep neural network in 2D versus 3D. After that we introduce the FCDNet architectures. We explain how we can convert a 2D network to an FCD network or vice versa and this is followed up by the exact chosen architectures with their parameter values used

in the experiments. Finally we describe how we created the controlled setting of our toy colored MNIST and Fashion-MNIST datasets.

3.1. Batch Processing 2D vs 3D

Most, if not all, current image classification neural networks make use of 2D convolutional layers [11, 12, 19, 27]. An image classifier processes images in 4-dimensional batches which correspond to the batch size, number of feature maps, height and width, of the feature maps. We investigate the influence of keeping a separate color dimension because we hypothesize it will have a positive influence on the eventual performance. Thus in the FCD networks we work with 3-dimensional images, one color dimension and two spatial dimensions. The batches have five dimensions, batch size, number of feature maps, color dimension of the feature maps and the spatial dimensions, height and width, of the feature maps.

3.2. From a 2D Network to an FCD Network

All of the networks used in this paper are based on the VGGNet [27]. We use convolutional layers with 3x3 kernels, followed by a ReLU activation function and the final layer is a linear layer for classification. We make use of convolutional, ReLU, max pool (MP), adaptive average pool (AAP) and linear layers [22]. It is relatively straightforward to transform a network from 2D to FCD, but there are several things to keep in mind. The kernel, padding and stride of the convolutional layer will be 3-dimensional. The MP layer computes the maximum in three dimensions and the AAP layer will create features in the three dimensions. We make a fair comparison between the 2D and FCD networks, which means that only changing the layer type is not sufficient. Only changing the layer type gives the FCD network more parameters to work with, making the comparison unfair. We define a fair comparison as comparing the performance of the 2D and FCD networks having roughly the same number of parameters throughout all of the layers.

The number of parameters in a network is decided by the number of parameters in the feature extraction and classification parts of the network. Each convolutional layer has a number of input channels, output channels and a kernel size. The AAP layer in our network is the bridge between the feature extraction and classification parts. The linear layer in the classification part has a number of input features and output features. The number of parameters can be computed as follows:

$$P_F = \sum_{i}^{N} (C_{in_i} * C_{out_i} * \prod_{j}^{k} (K_{i_j}) + C_{out_i}), \quad (1)$$

$$P_C = \sum_{l}^{M} (F_{in_l} * F_{out_l} + F_{out_l}), \quad (2)$$

where P_F is the number of parameters in the feature extraction part of the network. N is the total number of convolutional layers i. C_{in_i} is the number of input channels and C_{out_i} is the number of output channels, K_i is the convolutional kernel with k dimensions and K_{i_j} is the size of the kernel in dimension j. P_C is the number of parameters in the classification part of the network. M is the number of linear layers l. F_{in_l} denotes the number of input features and F_{out_l} the number of output features. Both equations have an additional term at the end that denotes the bias terms per layer. The number of input features of the first linear layer is equal to the number of output channels of the final convolutional layer multiplied by the number of features the AAP layer extracts. The output of the final linear layer is equal to the number of classes of the dataset. The total number of parameters of a network is equal to $P_F + P_C$.

The FCD network has more parameters than a 2D network if a one-to-one conversion is applied on the number of input and output channels per convolutional layer. A scaling in the number of channels per convolutional layer is applied to keep the number of parameters roughly the same, according to the procedure of Cohen and Welling [5]. This results in the 2D networks having more input and output channels per convolutional layer compared to their FCD network counterparts. The 2D networks thus learn more 2D filters while the FCD networks learn less 3D filters but with more parameters. We expect that the 2D networks focus more on spatial features while FCD networks split their focus on spatial and color dimension features.

3.3. The FCD Architectures

Many different types of 3D convolutions exist. Several examples are convolutions with cube-shaped kernels which do a same convolution with zero padding [16] in the color dimension. The convolutional kernels can also be customized to combine the color channels at different stages of the network or to do different types of combinations. Each has its own value for certain aspects of the task at hand. We experiment with seven different FCDNet architectures. All of the architectures have three convolutional layers. Each convolutional layer does a same convolution in the spatial dimensions. Each convolutional layer is followed by a ReLU and MP layer. The MP operation is performed per color channel with a kernel size and stride of 2x2 in the spatial dimensions. The final MP layer is followed by an AAP layer, which extracts one feature per color channel, and then a linear layer. The key characteristics for the convolutions can be found in Figure 3. Architectures 6 and 7 are left out of the figure because they are combinations of the Architectures 5 and 1 and Architectures 4 and 1 respectively.

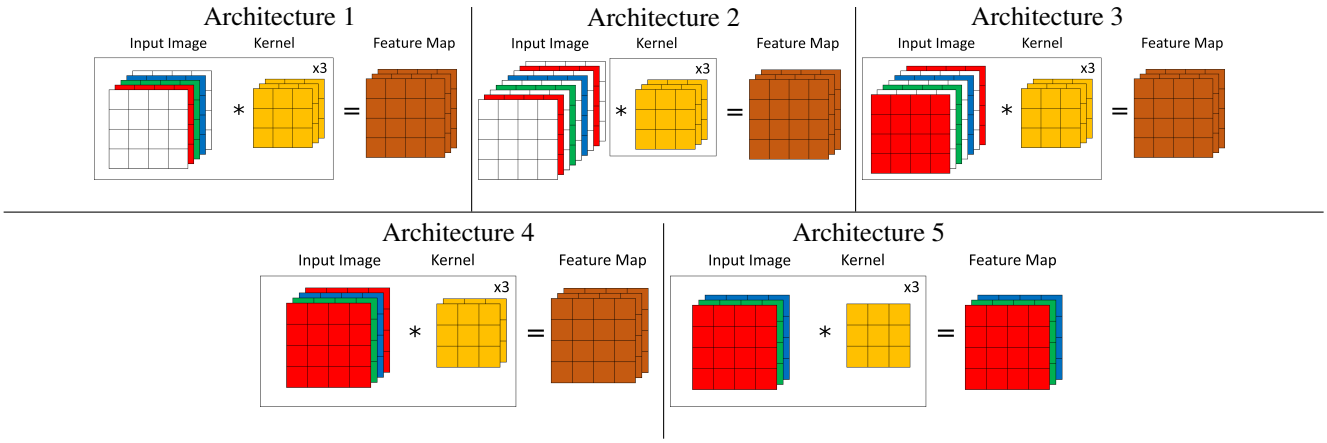

Figure 3. The key characteristics for five out of seven Full Color Deep network architectures. We sketch the type of convolution per architecture. "x3" means for all three convolutional layers. After each iteration of convolutions, a ReLU and max pool layer are applied (not shown). Each 4 by 4 matrix represents a channel. The rectangle around Architectures 1, 3, 4 and 5 represents that for each convolutional layer, the input has to be padded before the convolution. The input is only padded once for Architecture 2. The convolutions in Architecture 2 use a stride of 2 in the color dimension. White channels represent zero padding. The red, green and blue channels represent respectively the first, second and third channel of the input to the convolutional layer.

Table 1. The seven different architectures used in the experiments. C_{out} is the number of output channels and K is the kernel size. The number of input channels for conv1 is 1 and for conv2 and conv3 it is the number of output channels in the layer prior. AAP stands for adaptive average pool. Each convolutional layer is followed by a ReLU and then a max pool layer with kernel size 1x2x2. The output of the linear layer is dependent on the dataset, in case of MNIST and Fashion-MNIST it is 10.

FCD Architecture	1	2	3	4	5	6	7
conv1 (C_{out}, K)	16, 3x3x3	16, 3x3x3	16, 3x3x3	16, 2x3x3	16, 1x3x3	16, 1x3x3	16, 2x3x3
conv2 (C_{out}, K)	32, 3x3x3	32, 3x3x3	32, 3x3x3	32, 2x3x3	32, 1x3x3	32, 3x3x3	32, 3x3x3
conv3 (C_{out}, K)	64, 3x3x3	64, 3x3x3	64, 3x3x3	64, 2x3x3	64, 1x3x3	64, 3x3x3	64, 3x3x3
AAP (output)	(3, 1, 1)	(3, 1, 1)	(3, 1, 1)	(3, 1, 1)	(1, 1, 1)	(3, 1, 1)	(3, 1, 1)
linear (input)	192	192	192	192	64	192	192

The first architecture uses cube-shaped kernels which convolve over the color and spatial dimensions. A same convolution with zero padding is applied in the color dimension. The cube-shaped kernels allow the network to learn color features in combinations of color channels.

The second and third architectures make use of cube-shaped kernels and two padding schemes. The second architecture pads the input once, as shown in Figure 3, such that the resulting feature map after three convolutions has three channels in the color dimension. The third architecture pads the input images in each layer and the convolutions have a stride of two in the color dimension. These padding schemes enable the network to learn one or two channel color features in the first layer of the network. Learning features for the combination of all color channels is delayed till further in the network. A deep neural network is known to learn low level features in the earlier layers of the network and high level features in the deeper layers of the network [33]. We believe color information is a low level feature. Thus we give the network more opportunities to learn features by allowing the networks focus on single or double color channels.

The fourth architecture has 2x3x3 shaped convolutional kernels and the fifth architecture has 1x3x3 convolutional kernels. We do a circular padding in the color dimension for the fourth architecture and do not apply any padding in the color dimension for the fifth architecture. These architectures are also chosen with the reason in mind to delay the combination of the color channels. The fourth architecture has convolutions over all combinations of two out of three color channels in the first layer. The fifth architecture does not combine the color channels, thus the AAP layer extracts only one feature over all spatial and color channels, combining the features in the separate color channels.

The sixth and seventh architectures are variants of the fourth and fifth architectures. The architectures have a convolutional kernel size of 1x3x3 and 2x3x3 in the first layer respectively and 3x3x3 kernels for the following layers. This enables the networks to focus either on a color specific or a mix of two color channels specific features in the first

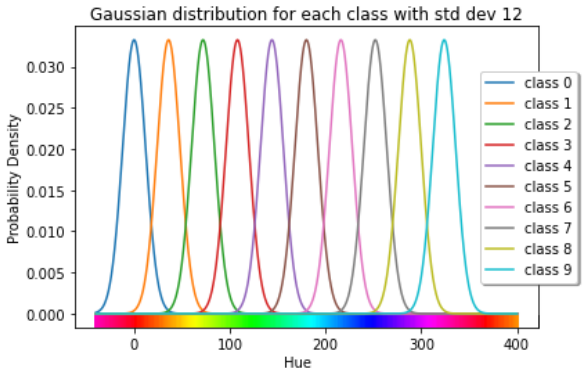

Figure 4. Gaussian distributions for each of the classes in the (Fashion-)MNIST dataset with standard deviation 12. The eventual hue values are calculated modulo 360.

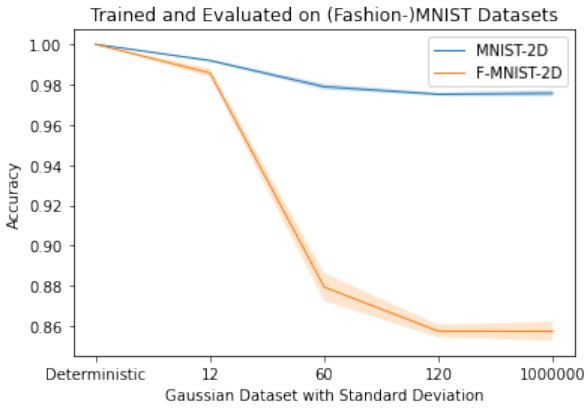

Figure 5. 2D networks trained and evaluated on the colored (Fashion-)MNIST datasets with the corresponding standard deviation shown on the x-axis. Five models with different random seeds are trained per dataset and the mean and standard deviation values are plotted.

layer and combine the color channels in earlier subsequent layers than the other architectures. A same convolution is applied in the second and third layer for both architectures and three features are extracted in the AAP layer.

3.4. Final Configurations

The final configurations of the FCDNet architectures are shown in Table 1. We considered these networks in the comparison of the best FCDNet architecture. One important difference between the 2D and FCD networks is that the AAP layer of the 2D network retains only one collapsed feature in the spatial dimensions while the FCD network retains three features, one per color channel. The importance is the fact that we explicitly give the FCD network color features to work with. Moreover, we use the cross-entropy loss function and the Stochastic Gradient Descent optimizer.

3.5. Colored MNIST and Fashion-MNIST

Variations of the MNIST and Fashion-MNIST datasets are used to create a controlled environment and observe the behaviour of the neural networks. The datasets are transformed to contain color information per class. Five specific datasets are created. The color of the digit is dependent on the HSV colorspace. The saturation and value values are always 1 in a range of 0 to 1 and the hue values vary in the range of 0 to 360. One type of dataset is a Deterministic dataset. 360 is divided in steps of $\frac{360}{10} = 36$ and one class is assigned one value. Class 0 is assigned hue value 0 for example. HSV values 0/1/1 are converted to RGB values, which correspond to the color red and the image is transformed correspondingly.

The four other datasets follow a particular strategy. One class is assigned a hue value. Class 5 for example is assigned hue value 180 which corresponds to the color cyan. Instead of every encountered instance of class 5 being cyan, a Gaussian distribution is drawn for class 5 with mean value 180 and a standard deviation value differing for the four datasets. The four standard deviation values are 12, 60, 120 and 1,000,000. Whenever an image with label 5 is encountered, a random sample is picked from its Gaussian distribution, which will represent the hue value of the image. An instance of the distributions of the dataset with standard deviation 12 is shown in Figure 4. The reasoning behind the standard deviation values is as follows. Standard deviation 12 indicates a low deviation from the mean value of the class so in more than 80% of the cases, the picked sample is closer to its own mean value than other mean values. In standard deviation 60 more than 80% of the values are within the right half of the hue circle and in standard deviation 120 more than 80% of the samples will deviate at most 180 from the mean value. Finally we have standard deviation 1,000,000 which represents a Uniform dataset meaning that any digit can be assigned any hue value between 0 and 360. The datasets in this work are referred to as the Deterministic, Gaussian 12, Gaussian 60, Gaussian 120 and Uniform datasets. Even though the digits can be assigned different colors, the number of labels in these datasets is equal to 10, corresponding to the digits 0 up to and including 9. By training both 2D and FCD networks on the datasets with different standard deviation values, we observe what the impact of color is on the networks. We introduce additional datasets in the experiments. Examples for all of the datasets are included in the appendix.

4. Experiments

In this section we present the experiments we have performed along with our hypotheses and results. The accuracy plots per dataset have a mean line and a standard deviation area, because five networks are trained and evaluated for

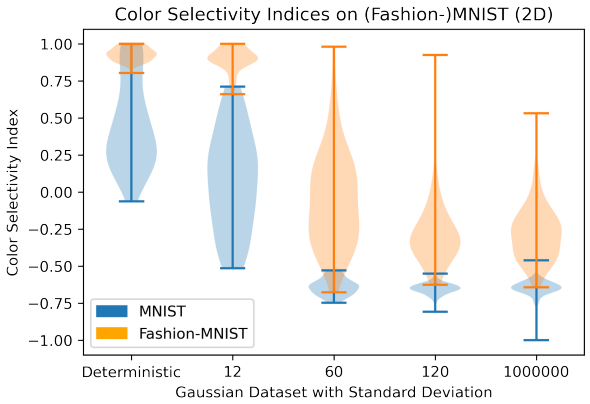

Figure 6. Color selectivity indices of 2D Architecture 1 in Table 2 trained on the five colored (Fashion-)MNIST datasets.

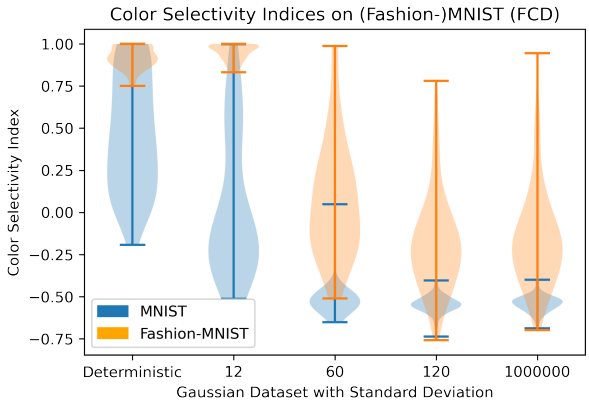

Figure 7. Color selectivity indices of FCD Architecture 4 in Table 1 trained on the five colored (Fashion-)MNIST datasets.

Table 2. Hyperparameters of the two different 2D architectures used in the experiments. C_{out} is the number of output channels and K is the kernel size. The number of input channels for conv1 is 3 and for conv2 and conv3 it is the number of output channels in the layer prior. AAP stands for adaptive average pool. Each convolutional layer is followed by a ReLU and then a max pool layer with kernel size 2x2. The output of the linear layer is dependent on the dataset, in case of MNIST and Fashion-MNIST it is 10.

2D Architecture	1	2
conv1 (C_{out}, K)	16, 3x3	23, 3x3
conv2 (C_{out}, K)	32, 3x3	45, 3x3
conv3 (C_{out}, K)	64, 3x3	91, 3x3
AAP (output)	(1, 1)	(1, 1)
linear (input)	64	64

each dataset with a different fixed starting seed for more reliable representations. The accuracy is computed as the number of correctly predicted samples divided by the total number of samples.

4.1. How Does Color Influence Performance?

Firstly, we investigate how color currently affects the performance of neural networks. We expect the CNNs to perform well whenever color bias per class is high but worse when the color bias per class is low. We hypothesize that if color bias is high, the neural network has a clear feature which makes the classification task easier. We use the five colored MNIST datasets. Since MNIST is a fairly easy dataset for CNNs, we use a small network and ∼16.67% of the data. A simple 2D network was used for this purpose (Table 2 Architecture 1).

Much color bias per class is present in the Deterministic and Gaussian 12 datasets. In the other three datasets, the color bias per class decreases as the standard deviation increases until there is no correlation between the color and the class, i.e. there is no color bias in the Uniform dataset. As shown in Figure 5, the accuracy is lower for the Uniform dataset than for the Deterministic dataset. We also believe that the networks trained on the Uniform dataset, have to classify based on only the spatial features, since color does not add valuable information, which is a harder task. We believe that color information is lost for those networks since it was not a useful feature. This information could be useful, however, for other downstream tasks. We introduce FCD networks to retain color information. The colored versions of the Fashion-MNIST dataset are used as control datasets and we observe the same behaviour in Figure 5.

The color selectivity index [24] assigns a value to a neuron in a convolutional layer. A value close to 1, at least higher than 0.25, means the neuron is color selective, i.e. color in the input image highly activates that neuron. A value less than 0.1, all the way down to -infinity, means color in the input image does not activate that neuron. The color selectivity index is a useful measure that can help us understand the behaviour of the networks. A violin plot has a range indicating the minimum and maximum values with a shaded area which represents the distribution of the values. We have 91 color selectivity indices in the final convolutional layer of the 2D network and 64 color selectivity indices in the final convolutional layer of the FCD network. The violin plot of the color selectivity indices in Figure 6 supports our belief that the neural network only uses spatial features for Uniform datasets, almost none of the neurons are color selective. For the networks trained on the Deterministic dataset, however, there are more color selective neurons.

4.2. Best Full Color Deep Network Architecture

There are many different ways to combine the color channels in an FCDNet, which is why in this experiment we evaluate the seven FCDNet architectures in Table 1. We in-

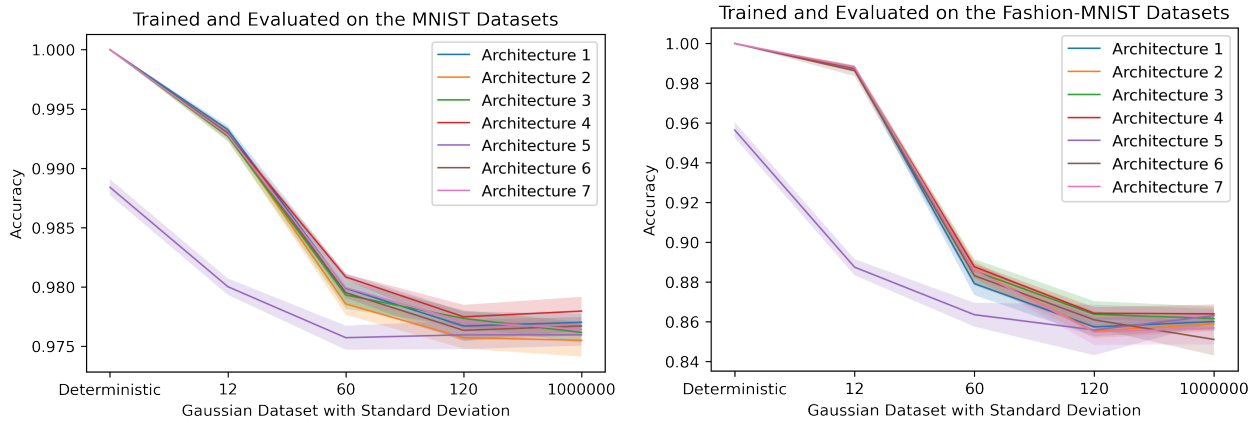

Figure 8. Accuracies of the seven FCDNet architectures on the five MNIST datasets (top) and Fashion-MNIST datasets (bottom). Five networks are trained for each dataset and the mean and standard deviation values are plotted.

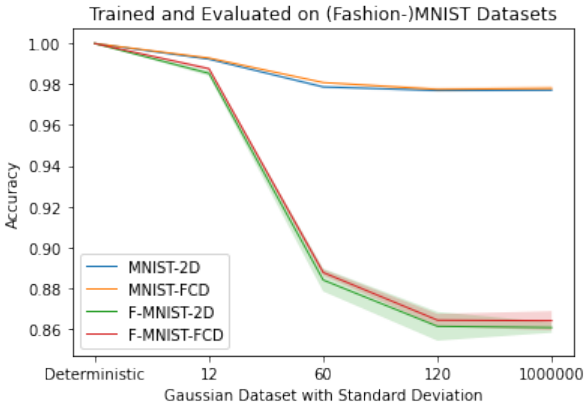

Figure 9. Accuracies of the optimal 2D and FCDNet architectures on the MNIST and Fashion-MNIST datasets of size 10,000.

vestigate which one attains the highest accuracy. We use the five datasets of MNIST and Fashion-MNIST. We use only 10,000 out of 60,000 images for all of the datasets. We observe that all of the accuracies for all architectures are fairly similar with the exception of Architecture 5 (Figure 8). We believe there is a deviation because the architecture is set up differently compared to the others. With this architecture the network is only able to learn spatial features in the RGB channels separately and combine them at the end. This causes the network to perform poorly when color bias per class is prevalent but similar to the others when the opposite is the case. Even though the rest of the networks achieve a similar accuracy, Architecture 4 is performing better. The architecture using 2x3x3 convolutions with circular padding thus performs best on at least the MNIST and Fashion-MNIST datasets.

4.3. Comparison 2D vs FCD on Colored MNIST and Fashion-MNIST

In this experiment we investigate whether the best FCDNet architecture in the experiment of section 4.2 can perform at least as well as its 2D counterpart. The colored MNIST datasets are fairly simple so we also evaluate the networks on the colored Fashion-MNIST datasets. We believe that, for at least the Deterministic datasets and the Gaussian datasets with low standard deviation, the 2D and FCD networks will perform comparably. FCD network can be outperformed on the Uniform dataset because that task is performed only utilising spatial features and 2D networks have more focus on spatial features. We use Architecture 4 in Table 1 and its fair counterpart Architecture 2 in Table 2. The accuracies of the 2D and FCD networks are similar on all of the datasets (Figure 9). A trend similar to the experiment in section 4.1 is observed, the accuracies are high when the standard deviation is low, and the accuracies are low if the standard deviation is high. The FCD networks thus perform at least as well, if not better, than 2D networks on at least the MNIST and Fashion-MNIST datasets. We performed an optimal comparison between the two types of networks, in which we compare the networks when they have the highest number of filters per convolutional layer. The results are very similar however, so we have included the results in Appendix Section 5.

4.4. Out of Distribution Datasets

We found that FCD networks can perform comparably to 2D networks on image classification. With three following experiments we give some more intuition on how color information is processed and we also show that FCD networks are better at retaining color information than 2D networks. We define the retention of color information as the networks

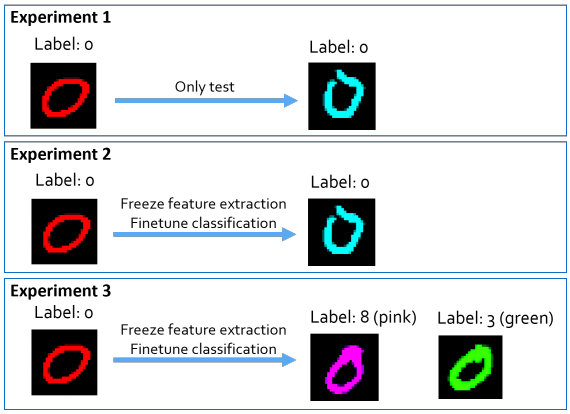

Figure 10. A visual description of the extra experiments performed on out of distribution datasets. In experiment 1 the pretrained networks are immediately evaluated on the Deterministic dataset with hueshift 180. In experiment 2 the weights in the classification layers of the pretrained networks are finetuned on the Deterministic Hueshift 180 dataset and then evaluated. In experiment 3, the weights in the classification layers of the pretrained networks are finetuned on a 10 color uniform dataset in which the label is the color and afterwards evaluated.

learning color features such that deeper layers of the network are able to use those features to make predictions. Two new datasets are introduced. The first dataset is a Deterministic dataset as described before. The only differences are the hue values per class, those are shifted by 180 modulo 360, e.g. class 7 was originally assigned hue value 252, in the new dataset it is assigned (252 + 180) % 360 = 72. This way we obtain a Deterministic dataset in which the digits are assigned a color in the opposite of the hue circle. The second dataset has 10 colors around the hue circle in steps of 36 just like the Deterministic dataset. A sample in the dataset has a uniform chance to be assigned any of the 10 colors. The labels in this dataset correspond to the color of the samples. A visual description of the experiments is shown in Figure 10.

The first of the three experiments uses the pretrained networks in the experiment of section 4.3 and evaluates those on the first dataset introduced in this experiment. The classification labels for both the initial training and downstream tasks are the same. The only difference in the downstream task is the color of the digit. The accuracies on the Deterministic are zero (Figure 11). As shown in Figure 12, the 2D and FCD networks only learned color information in the initial training task. This results in the networks to classify each class as five classes later, the classes that are assigned the color in the original Deterministic datasets. The same behaviour applies on the Gaussian 12 datasets. The accuracies are higher for the networks initially trained on Gaussian 60 but the accuracies are not as high as in Figure 9, $\sim$90% compared to $\sim$98% on MNIST and $\sim$60% compared to $\sim$89% on Fashion-MNIST, note the difference in the y-axes. Color information was thus important, but not as important as for the Deterministic and Gaussian 12 datasets. Finally, the networks perform comparable to the networks in Figure 9 when pretrained on the Gaussian 120 and Uniform datasets, $\sim$97.4% and $\sim$97.8% compared to $\sim$97.5% and $\sim$97.6% on MNIST and $\sim$84.4% and $\sim$86.1% compared to $\sim$85.7% and $\sim$85.7% on Fashion-MNIST. This tells us that those pretrained networks mostly used spatial features to classify the images. The color selectivity indices, as shown in Figure 7, are similar to the color selectivity indices in Figure 6.

The purpose of the second experiment is to investigate at what aspect the FCD network excels compared to the 2D network. We freeze the weights in the layers up to and including the adaptive average pool layer of the pretrained networks and finetune the weights in the classification layer on the first proposed dataset. This way we can assess how much color information was learned by the feature weights of the pretrained networks and whether that information is useful for other downstream tasks. As shown in Figure 11, the accuracies are similar for the 2D and FCD networks for the first two datasets of MNIST and Fashion-MNIST. As the standard deviation of the dataset increases, however, the FCD networks achieve higher accuracies than the 2D networks. In our work a 2D network has more filters per layer and has 2D feature maps per channel. Thus it only learns spatial information for datasets with high standard deviation and has more focus on it than its FCD counterpart. The accuracies of the FCD networks, for both the MNIST and Fashion-MNIST datasets, being higher than their 2D counterparts for higher standard deviation datasets thus tells us that the 3-dimensional kernels are able to learn color features even when their initial task has no use for this information. This observation is even more prevalent in the following experiment.

The final of the three experiments was performed with the purpose to isolate the color features learned by the networks. Here we do the same as the second experiment but on the second dataset introduced in this experiment. The classification task in this experiment is different from the pretrain tasks. The labels in this dataset are equal to the colors in the images. We train only the weights in the classification layer for both the 2D and FCD networks and thus we can investigate what features the pretrained networks learned in their initial tasks with regards to color information. The accuracies of the FCD networks on the downstream task are close to 1.0 while the accuracies are lower for the 2D networks. Both 2D and FCD networks had the same input and objective in the pretrain task. The results indicate that the FCD networks learned meaningful features for the new dataset while the 2D networks did not. Experiment 3 in Figure 11 is thus a prime example that the FCD

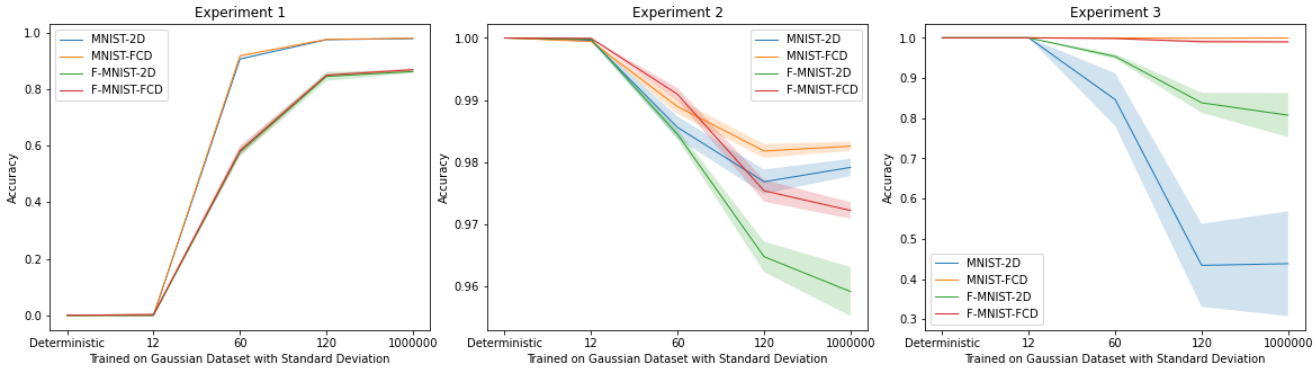

Figure 11. Accuracies of networks pretrained on the 5 MNIST and Fashion-MNIST datasets and evaluated on out of distribution datasets.

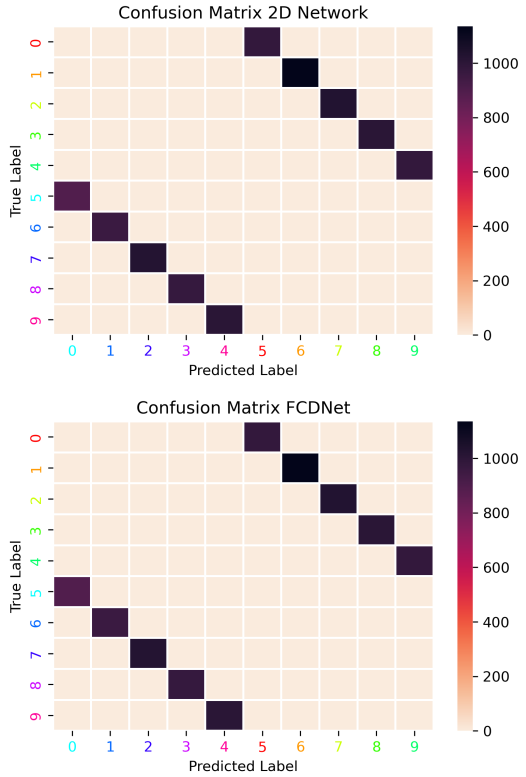

Figure 12. Confusion matrix of the 2D (top) and FCD networks (bottom) pretrained on the Deterministic MNIST dataset and evaluated on the Deterministic dataset with hueshift 180. The color of the label represents the color the class was assigned.

networks retained color information better than the 2D networks.

4.5. Vehicle Color Recognition

We investigate how our networks perform on a real world problem, the color recognition of vehicles on a highway. We use the vehicle color recognition dataset for this purpose [4]. The classification task of this dataset is to predict the color of the vehicle. In the previous experiments we have been using toy datasets and thus it is not sure if the networks can be applied in real life situations. We use the optimal architectures we found before, FCDNet Architecture 4 in Table 1, and its 2D counterpart, Architecture 2 in Table 2. Since the classification task is recognizing the color of vehicles, we believe that such small networks are still able to do the job. We believe they might even perform better than bigger networks since the receptive fields are so small that they are only able to focus on smaller features such as the color. We trained the networks until the accuracies saturated. As shown in Figure 13, the FCD network achieves higher accuracies than the 2D network throughout the whole learning process. Even though the number of parameters throughout all of the layers are relatively the same, the FCD networks are better at learning the color features in the images.

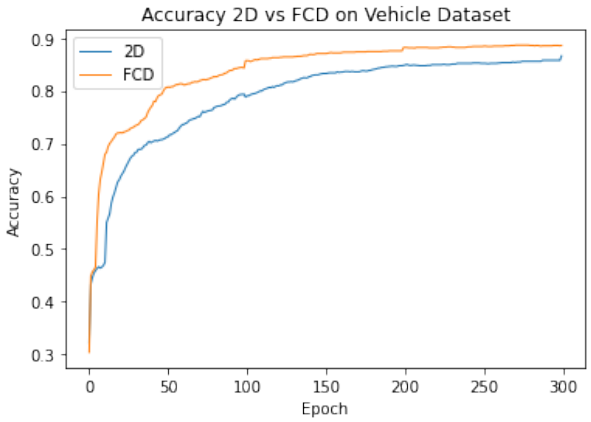

Figure 13. Accuracies of 2D Architecture 2 in Table 2 and FCD Architecture 4 in Table 1 on the vehicle color recognition dataset.

Our hypothesis on why the FCDNets are outperforming the 2D networks is that the vehicle color recognition dataset

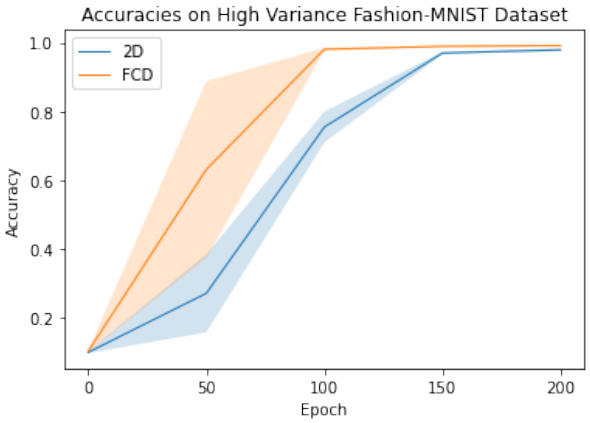

Figure 14. Accuracies of the 2D and FCD networks on a high variance Fashion-MNIST dataset.

has too much variance per class. The 2D network puts too much focus on all of the features in the input images that it gets distracted from the main feature, the color of the vehicles. We believe the FCD network also gets distracted, however, since they have a dedicated color dimension to extract features from, they are able to more easily identify the useful feature, which is the vehicle color. We modified the Fashion-MNIST dataset to recreate the setting. To recreate the variance, we only included eight samples per class and added background noise to each image. As shown in Figure 14, the FCDNets achieve higher accuracies than the 2D networks. This shows as well that the FCD networks are less susceptible to noise than the 2D networks.

5. Discussion

This section contains the limitations we observed during our experiments, the main findings of our work and directions for future work regarding this research.

5.1. Limitations

There are some limitations on using FCD networks over 2D networks. First, the time it takes to train is longer. We found from all of the experiments that it takes approximately 50% longer to train the FCD networks. Next we trained a VGG-M network as in the work of Rafegas et al. [24] on the vehicle color recognition dataset. This gave two results. One of the results was that the optimal architecture found for the colored MNIST and Fashion-MNIST datasets was not the optimal architecture for the vehicle color recognition dataset. Architecture 1 in Table 1 had a better performance. The second find was that Architecture 1 performed comparably with its 2D counterpart and that Architecture 4 performed worse than its 2D counterpart (Appendix Section 5). The FCD networks might thus not necessarily be better than 2D networks when the networks are bigger. Additionally, our analysis is limited by the architecture variants and datasets we experimented with in this work. Other architectures using 3D convolutions might perform better than the architectures presented here.

5.2. Conclusion

We introduce Full Color Deep networks (FCDNets), convolutional neural networks which use 3D convolutional layers such that the color dimension can be retained beyond the first convolutional layer. We also introduce colored versions of the MNIST and Fashion-MNIST datasets in which we systematically vary the color bias for the classes. We compare different FCDNet architectures to each other and we find that FCDNet Architecture 4 in Table 1 performs the best in terms of accuracy for the colored versions of the MNIST and Fashion-MNIST datasets. The FCDNets perform comparably, if not better, than their 2D counterparts when trained and evaluated on the MNIST and Fashion-MNIST datasets. The FCDNets manage to attain much better performance in retaining color information. We define the retention of color information as the networks learning color features such that deeper layers of the network are able to use those features to make predictions.

Using the colored MNIST and Fashion-MNIST datasets, we show that FCDNets preserve color information even when it is not explicitly required and do not lose their image classification performance compared to the 2D networks. We also show that FCDNets do not only work in toy settings but also on a real world problem where they outperform their 2D counterparts.

5.3. Future Work

In our work we showed that small FCD networks perform better than 2D networks on the vehicle color recognition dataset. We did not have the time to extensively experiment with bigger architectures. It might thus be interesting to investigate if current big networks such as the VGG-M network have FCD counterparts which can perform comparably or better on the vehicle color recognition dataset. A point of interest is whether the optimal FCD network architecture is task dependent so whether each task has its own optimal architecture. Other architectures besides the ones presented in this research might also be viable options. The performance of 2D and FCD networks can also be compared on a more natural dataset such as ImageNet [6] or CIFAR10 [18] to explore the benefits, in terms of training time and accuracy, of using FCD networks in those settings. Furthermore, using different color spaces to represent images can give different results for the same task [7, 9, 25]. It might be interesting to research the performance of an FCDNet depending on the color space of the input dataset in case the RGB colorspace is not the best colorspace for FCDNets.

References

[1] Vanessa Buhrmester, David Münch, Dimitri Bulatov, and Michael Arens. Evaluating the impact of color information in deep neural networks. In *Iberian conference on pattern recognition and image analysis*, pages 302–316. Springer, 2019. 2, 3

[2] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 3

[3] Lele Chen, Yue Wu, Adora M DSouza, Anas Z Abidin, Axel Wismüller, and Chenliang Xu. Mri tumor segmentation with densely connected 3d cnn. In *Medical Imaging 2018: Image Processing*, volume 10574, page 105741F. International Society for Optics and Photonics, 2018. 3

[4] Pan Chen, Xiang Bai, and Wenyu Liu. Vehicle color recognition on urban road by feature context. *IEEE Transactions on Intelligent Transportation Systems*, 15(5):2340–2346, 2014. 10

[5] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999. PMLR, 2016. 4

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 11

[7] Javier Diaz-Cely, Carlos Arce-Lopera, Juan Cardona Mena, and Lina Quintero. The effect of color channel representations on the transferability of convolutional neural networks. In *Science and information Conference*, pages 27–38. Springer, 2019. 11

[8] Martin Engilberge, Edo Collins, and Sabine Süsstrunk. Color representation in deep neural networks. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 2786–2790. IEEE, 2017. 2, 3

[9] Shreyank N Gowda and Chun Yuan. Colornet: Investigating the importance of color spaces for image classification. In *Asian Conference on Computer Vision*, pages 581–596. Springer, 2018. 11

[10] Jad Haddad, Olivier Lézoray, and Philippe Hamel. 3d-cnn for facial emotion recognition in videos. In *International Symposium on Visual Computing*, pages 298–309. Springer, 2020. 3

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[12] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 3, 4

[13] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2012. 2, 3

[14] Konstantinos Kamnitsas, Christian Ledig, Virginia FJ Newcombe, Joanna P Simpson, Andrew D Kane, David K Menon, Daniel Rueckert, and Ben Glocker. Efficient multi-scale 3d cnn with fully connected crf for accurate brain lesion segmentation. *Medical image analysis*, 36:61–78, 2017. 3

[15] Baris Kayalibay, Grady Jensen, and Patrick van der Smagt. Cnn-based segmentation of medical imaging data. *arXiv preprint arXiv:1701.03056*, 2017. 3

[16] Osman Semih Kayhan and Jan C van Gemert. On translation invariance in cnns: Convolutional layers can exploit absolute spatial location. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14274–14285, 2020. 4

[17] Justin Ker, Satya P Singh, Yeqi Bai, Jai Rao, Tchoyoson Lim, and Lipo Wang. Image thresholding improves 3-dimensional convolutional neural network diagnosis of different acute brain hemorrhages on computed tomography scans. *Sensors*, 19(9):2167, 2019. 3

[18] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 11

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012. 4

[20] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010. 3

[21] Pavlo Molchanov, Shalini Gupta, Kihwan Kim, and Jan Kautz. Hand gesture recognition with 3d convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1–7, 2015. 2

[22] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 4

[23] Ivet Rafegas and Maria Vanrell. Color representation in cnns: parallelisms with biological vision. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 2697–2705, 2017. 3

[24] Ivet Rafegas and Maria Vanrell. Color encoding in biologically-inspired convolutional neural networks. *Vision research*, 151:7–17, 2018. 3, 7, 11

[25] Rajan Sachin, V Sowmya, D Govind, and KP Soman. Dependency of various color and intensity planes on cnn based image classification. In *International symposium on signal processing and intelligent recognition systems*, pages 167–177. Springer, 2017. 11

[26] Anton Saveliev, Mikhail Uzdiaev, and Malov Dmitrii. Aggressive action recognition using 3d cnn architectures. In *2019 12th International Conference on Developments in eSystems Engineering (DeSE)*, pages 890–895. IEEE, 2019. 3

[27] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3, 4

[28] Satya P Singh, Lipo Wang, Sukrit Gupta, Haveesh Goli, Parasuraman Padmanabhan, and Balázs Gulyás. 3d deep learning on medical images: a review. *Sensors*, 20(18):5097, 2020. 2

[29] Satya P Singh, Lipo Wang, Sukrit Gupta, Balazs Gulyas, and Parasuraman Padmanabhan. Shallow 3d cnn for detecting acute brain hemorrhage from medical imaging sensors. *IEEE Sensors Journal*, 21(13):14290–14299, 2020. 3

[30] Du Tran, Jamie Ray, Zheng Shou, Shih-Fu Chang, and Manohar Paluri. Convnet architecture search for spatiotemporal feature learning. *arXiv preprint arXiv:1708.05038*, 2017. 3

[31] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 3

[32] Guangle Yao, Tao Lei, and Jiandan Zhong. A review of convolutional-neural-network-based action recognition. *Pattern Recognition Letters*, 118:14–22, 2019. 3

[33] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014. 5

2 Introduction

Convolutional neural networks (CNNs) have gained a lot of popularity over the recent years showing prominent performance on computer vision tasks such as image classification, segmentation, object recognition and much more [1,2,11]. In image classification in particular, AlexNet [7] was one of the first introduced deep neural networks that showed significant results, even when compared to the competition at that time. VGGNet [12] was introduced under the premise that the networks could be deeper with small convolutional kernels and achieve even better results. Besides these, PyTorch [9] has an extensive list of image classifiers and their performances on their website. One common element between the networks is that they use 2-dimensional layers, 2-dimensional convolutional layers in particular. 2D convolutional layers process RGB input images and produce 2D feature maps, in the height and width dimensions, per channel. This removes the color dimension of the input images.

This work investigates the behaviour of CNNs when the color dimension is retained. We introduce Full Color Deep (FCD) networks that are designed to handle 3D feature maps, in the color and spatial dimensions, throughout all layers of the network. We also investigate the behaviour of CNNs with regards to color bias per class by evaluating the 2D and FCD networks on datasets which have a varying color bias per class.

2.1. Motivation

CNNs have had significant breakthroughs [8]. CNNs using 3-dimensional convolutional layers have been applied to several fields already which include video action recognition and medical imaging for example [6, 13]. We are the first, to our knowledge, to apply CNNs with 3D convolutional layers on images which utilize the color dimension as the third dimension. We believe that color information is important in the image classification task. Image classifiers using 3D convolutions to extract features in the color dimension might perform better than regular 2D networks when color bias is present in the dataset such as the Vehicle Color Recognition dataset [4]. This motivates us to explore beyond the regular 2D networks and introduce FCD networks to investigate whether better architectures exist for downstream tasks in which color information is important.

2.2. Research Questions

In this research we investigate the influence of retaining a color dimension beyond the first layer of the convolutional neural network. The main research question accompanying this research is:

How does the retention of a color dimension of the input images influence the performance of a neural network?

This can be deconstructed in the following three sub-questions:

How does color information influence the performance of the neural network?

How do we process the retained color dimension with a neural network?

How much color information is used when retaining the color dimension?

The first sub-question is to understand how color information is currently processed by neural networks. That means how and when color information is used during classification. The second sub-question is

to investigate what type of convolutions we will use to include a color dimension which is related to the type of architecture. The third sub-question is to investigate whether the retention of the color dimension enables the neural networks to learn more color information than the regular 2D networks.

2.3. Outline

This report aims to provide additional information to the work presented in the scientific paper. Chapter 3 provides prior knowledge on the convolutional neural networks. Chapter 4 explains the methods in more detail. Finally, chapter 5 provides extra explanations on experiments and also contains additional experiments that were performed but did not add new insights to the findings in the scientific paper.

3

Prior Knowledge on Convolutional Neural Networks

This chapter provides prior knowledge on the convolutional neural networks (CNNs) discussed in the scientific paper. A CNN consists of several layers, each containing nodes, also known as neurons. The nodes are interconnected and the whole network represents a human brain. The CNN has an input layer, output layer and hidden layers in between. The hidden layers in this work are convolutional, ReLU and pooling layers. In image classification, the CNN processes input images to obtain an output which corresponds to the object in each image. Since the 2D networks and Full Color Deep (FCD) networks mentioned in the scientific paper are fairly similar, we start by explaining the 2D networks.

3.1. 2D Convolutional Neural Networks

In the scientific paper we briefly discussed how the data is processed in a CNN. The well known image classifiers available on the website of PyTorch [9] are 2D networks.

3.1.1. Dataset Processing

A dataset is a collection of images with labels which describe the object in the image. We worked with the MNIST and Fashion-MNIST datasets. Both datasets contain 60,000 train images and 10,000 test images. The train images are used to train the network and are split into two sets, train and validation images. The validation set is a control dataset used to monitor the learning process during training and it is in our work 20% of the training data. During one training loop, also known as an epoch, all train images are used as input for the network. The train images are split into batches of, in our case, 32 images and then processed. Each batch is processed as 4-dimensional data. The first parameter is the number of images in a batch. The second parameter is the number of input channels. The third and fourth parameters are the spatial height and width of the images. The network produces an output tensor of 32 predicted labels for each input batch. The predicted labels are compared to the actual labels, also known as the ground truth labels, and adjustments are made to the network to improve the predictions.

3.1.2. Training a Network

The input for a network is a tensor of images and labels, the output is a tensor of predictions. A network produces an output after a batch is processed. We use Architecture 2 in Table 2 as our 2D network. The batch is processed through convolutional, ReLU, max pool (MP), adaptive average pool (AAP) and linear layers. We first explain each component separately and describe how they work together afterwards.

Convolutional Layer

A convolutional layer has input channels and output channels. The input for the first convolutional layer is a batch of 2D images with three color channels, RGB. The number of input channels is thus equal to three. The convolutional layers have 3x3 kernels, also known as filters, which traverse the input

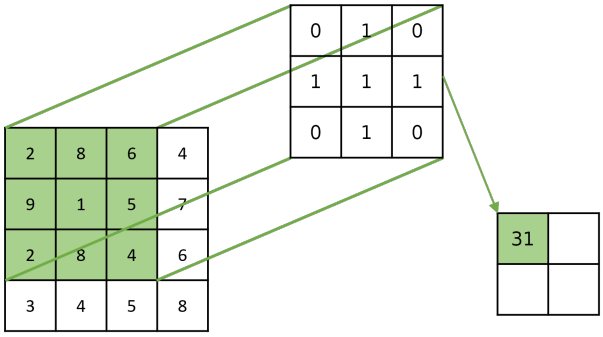

Figure 3.1: A convolution operation of a 3x3 kernel on a 4x4 input image producing a 2x2 feature map.

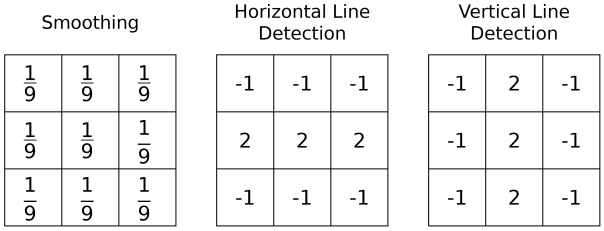

Figure 3.2: Examples of averaging and edge detection filters.

images to produce so called feature maps. The kernels, in a convolutional layer, slide over each value and produce an output value. Figure 3.1 shows an example of one step of one kernel. The number of input channels in the input layer is 3 and the number of output channels is 23. Thus there are $3 * 23$ 3x3 kernels. The output is a 2D feature map per channel where each is a combination of the three input channels. They are called feature maps because each filter highlights the features in the input it has to detect. Filters have different functions. Figure 3.2 shows averaging and edge detection filters. The job of a convolutional layer is to detect features in input images making use of its filters. The problem with this setup is that the CNNs collapse the color dimension in the first convolutional layer. The outputs are 2D feature maps, in the height and width dimension, per channel. CNNs have achieved astonishing results, however, we believe that the networks can perform better if it learns explicit features in the color dimension. 2D feature maps per channel limits this process.

Activation Function

The activation function is an important part of the network and in our case, each convolutional layer is followed by a ReLU activation function. The activation function in a network decides for every neuron in a layer by how much its output should be weighed as input for the next layer. There are several activation functions such as the sigmoid, tanh or a linear function. We chose the ReLU activation function because it is currently regarded as the better non-linear function when compared to sigmoid and tanh. Furthermore, a non-linear activation function is important because it allows the network to do more complex tasks by introducing non-linearity in the values.

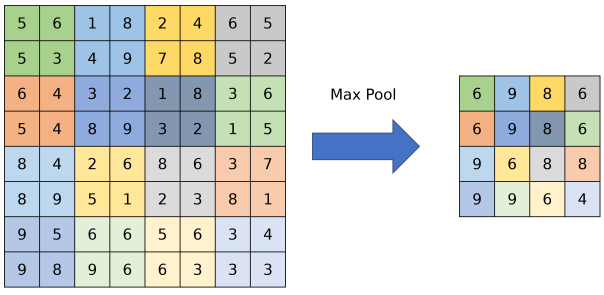

Figure 3.3: A max pool operation with stride and kernel size of 2x2.

Pooling Layer

A pooling layer in CNNs is used to downsample a feature map and that way summarize its features. Pooling layers decrease the number of computations a network has to do since the feature maps are downsampled. Pooling layers also make the network translation invariant. Translation invariance means that the position of features in an image does not matter. The introduction of translation invariance is important because the convolutional layers make the network encode location specific features. Our networks use MP layers and an AAP layer. MP layers downsample the feature maps with a scale of two in the height and width with a 2x2 kernel and stride (Figure 3.3). The job of the MP layer is to retain the maximum value in the 2x2 patch, thus retaining the features in the whole feature map that are the most eminent.

We use an AAP layer between the linear and final MP layer. For an AAP layer, instead of specifying a kernel size and a stride, we specify the number of features, and the kernel size and stride are automatically initialized. Our networks retain one spatial feature per feature map of the final MP layer. The AAP layer computes the average feature per channel and passes the features on to the classification part of the network.

Linear Layer

The final layer of our network is the linear layer, also known as a fully connected layer. We have one linear layer which is the classification part of our network. All layers prior, are combined the feature extraction part. This layer takes as input the flattened features produced by the AAP layer. Flattening is a simple process in which a multi-dimensional vector of values is transformed to a 1-dimensional vector of values. The linear layer has as input the number of features. The output is equal to the number of classes in the dataset. The linear layer is able to learn connections between the features extracted and the output. It is used to predict how probable a certain output is based on the input features.

Loss Function and Optimizer

A loss function is used in the training process to compute the difference between the ground truth labels and the predicted labels. The loss function assigns a value to the difference, also called the loss. The loss function also calculates the gradients for each of the weights. The gradients for weights indicate the change to the network weights to make the predicted labels more similar to the ground truth labels. Once the gradients are calculated, the optimizer decides on how drastically to change the weights. A learning rate is initialized to this end. A low learning rate indicates a small step towards the right direction, a high learning rate indicates a big step towards the right direction.

Difference 2D and 3D Convolutions

A 2D convolution in convolutional layers uses a 2D kernel to convolve over the input image. The output is computed as

$$O[m,n] = \sum_{i=-\lfloor k/2 \rfloor}^{\lfloor k/2 \rfloor} \sum_{j=-\lfloor k/2 \rfloor}^{\lfloor k/2 \rfloor} K[i,j]I[m+i,n+j], \tag{3.1}$$

where $O[m,n]$ is the coordinate (m,n) in output O, K is a $k \times k$ kernel and I is the input. As shown in Figure 3.4, a 2D convolution of an RGB input still produces a 2D output. A 3D convolution in convolutional layers uses a 3D kernel to convolve over the input image. The output is computed as

$$O[p,m,n] = \sum_{l=-\lfloor c/2 \rfloor}^{\lfloor c/2 \rfloor} \sum_{i=-\lfloor k/2 \rfloor}^{\lfloor k/2 \rfloor} \sum_{j=-\lfloor k/2 \rfloor}^{\lfloor k/2 \rfloor} K[l,i,j]I[p+l,m+i,n+j], \tag{3.2}$$

where $O[p,m,n]$ is the coordinate (p,m,n) in output O, K is a $c \times k \times k$ kernel, where c is the number of channels in the color dimension, and I is the input. As shown in Figure 3.5, a 3D convolution over an RGB input produces a 3D output, with a first dimension in color followed by the dimensions in height and width.

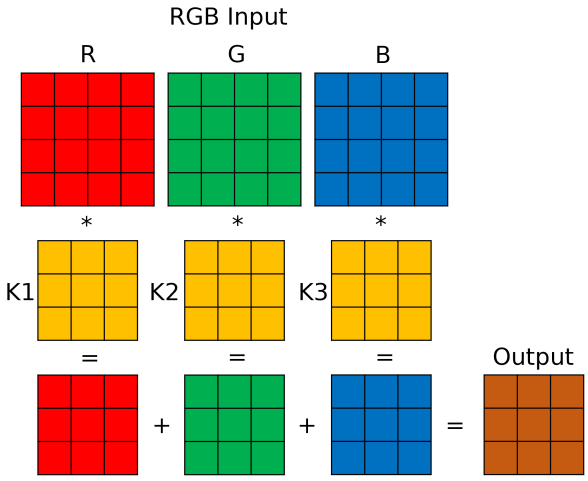

Figure 3.4: A 2D convolution over an RGB image. Three kernels, one per color channel. The outputs per convolutional kernel are additively superimposed to produce a 2D feature map.

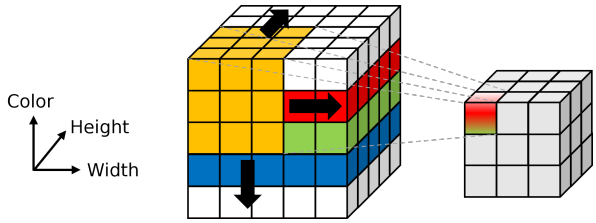

Figure 3.5: A 3D convolution over an RGB cube representing an input (left) and one yellow cube-shaped kernel. The output is a 3D feature map (right) in which one intermediate step is displayed.

Pipeline

Now we present an overview of how these components work together to train a model. A model is trained for several epochs. In one epoch, the training data and validation data are processed. In case of 60,000 images, 48,000 are used for training and 12,000 for validating. The training data is processed by the network in batches of 32. Kernel weights are learned after a batch is used as input. The kernels change the input to detect features and produce feature maps per channel. The ReLU activation function decides how much the feature maps weigh as input for the next convolutional layer and the MP layer emphasizes the most prominent features and adds translation invariance. The convolutional operations, ReLU activation function and MP operations are executed three times. The kernels of the early convolutional layers learn to detect low level features such as edges and corners while the later ones learn to detect high level features [14]. The final convolutional layer produces feature maps depending on the number of output channels, 91 in our case. The AAP layer extracts one global feature per feature map and passes those as input to the linear layer which uses the features to predict an output. The output, predicted labels, are compared to the ground truth labels using the loss function, and the optimizer updates the weights in the network accordingly. This process is repeated for every epoch.

The validation data follows a similar process, however, it does not update the weights. The purpose of the validation data is an extra check to verify that the network is learning general weights and not weights that are specific to the training data. The validation data thus contains input images that are not in the training data. Finally, the network, after being trained for several epochs, is evaluated on the test dataset which contains samples never seen before by the network. The evaluation metric is usually the accuracy which is the number of correctly predicted labels divided by the total number of labels multiplied by a hundred.

3.2. Full Color Deep Networks

The FCD networks used in this work are fairly similar to the 2D CNNs explained above, however, there are several differences. The purpose of the FCD networks is to retain a color dimension so instead of the

batch size being 4-dimensional, it is 5-dimensional. An extra dimension is in the third place representing the color dimension. The convolutional layers are 3D instead of 2D. 3D convolutional layers learn 3D kernel weights to detect features over 3D input images, one color dimension and two spatial dimensions. This way we enable the network to learn features in the color dimension. The 3D convolutional layers produce 3D feature maps per channel. The ReLU stays the same across both type of networks. The MP layer is converted to a 3D MP layer because it has to do pooling operations over 3D feature maps per channel. The pooling operation uses a 1x2x2 kernel size which means it does a MP operation per channel. The 3D feature maps per channel thus stay 3-dimensional. The AAP layer passes three features, one per color channel, to the linear layer per output channel of the final convolutional layer. Only one feature was passed in the 2D networks. This way we are explicitly passing color features to the linear layer for the classification task. Finally, besides the number of inputs differing, the linear layer has no changes when compared to the 2D networks.

4

Supplementary Methods

This chapter contains additional information on the methods presented in the scientific paper. We first give a more detailed explanation on how we derived the number of input and output channels per convolutional layer. Then we elaborate on the color selectivity index and we conclude with examples of all synthetic datasets we used in the scientific paper.

4.1. Comparison

In the scientific paper we mention a fair comparison between the 2D and FCD networks. We define a fair comparison as comparing the two networks when the number of parameters across the networks are roughly the same. We aim to keep the distribution of the parameters across the layers similar as well. Equation 1 and 2 in the paper are used to compute the number of parameters in the feature extraction and classification parts of the network. The authors of [5] presented a method to keep the number of parameters roughly the same between 2D network and networks using 3+-dimensional convolutions. We use 3D convolutions. The method is to scale the number of input channels and output channels of the convolutional layers in the 2D network with a factor of $\sqrt{x}$, where x is the 3D kernel size divided by the 2D kernel size, in our case $x = 2$. Scaling with a factor of $\sqrt{2}$ works because if we write out P_F for a 2D network, we get $\sum_i^3 C_{in_i}^{2D} * C_{out_i}^{2D} * 3 * 3$. We can disregard the bias terms because they add up a little to the total number of parameters. For the FCD network, P_F is equal to $\sum_i^3 C_{in_i}^{FCD} * C_{out_i}^{FCD} * 2 * 3 * 3$. Now if we substitute the C_{in}'s and C_{out}'s from the equation of FCD into the equation of 2D with the appropriate factor $\sqrt{2}$, we get

$$\sum_i^3 C_{in_i}^{2D} * C_{out_i}^{2D} * 3 * 3 = \sum_i^3 C_{in_i}^{FCD} * \sqrt{2} * C_{out_i}^{FCD} * \sqrt{2} * 3 * 3 = \sum_i^3 C_{in_i}^{FCD} * C_{out_i}^{FCD} * 2 * 3 * 3$$

This seems to be a perfect solution, however, there are exceptions. We can not control C_{in_1} which is the number of input channels of the first layer, in our case $C_{in_1}^{2D} = 3$ and $C_{in_1}^{FCD} = 1$. The number of parameters in the 2D case is equal to $3 * C_{out_1}^{FCD} * \sqrt{2} * 3 * 3 + C_{out_1}^{FCD}$ while in the FCD case it is equal to $1 * C_{out_1}^{FCD} * 2 * 3 * 3 + C_{out_1}^{FCD}$. The 2D networks have approximately twice as many parameters in the first convolutional layer. We can not do much about this if we wish to keep the transition from 2D networks to FCD networks, or vice versa, methodical. Another reason is the number of bias terms. The number of bias terms is dependent on the number of output channels per layer and that differs between the two types. Finally, the number of parameters is also influenced by the classification part of the network. In the 2D network $P_C = 1 * C_{out_3}^{2D} * F_{out} + F_{out}$ while for the FCD network $P_C = 3 * C_{out_3}^{FCD} * F_{out} + F_{out}$. This means the FCD network does have approximately twice as many parameters in the classification part of the network. Overall the number of parameters in both networks differ in small amounts. In the fair comparison experiment in section 4.3, the 2D network has 47,870 parameters while the FCD network has 48,410 parameters.

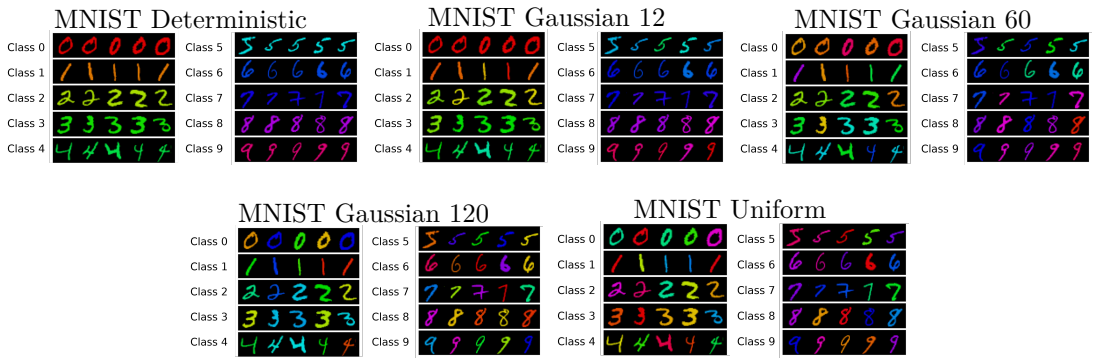

Figure 4.1: Samples of the Deterministic, Gaussian 12, Gaussian 60, Gaussian 120 and Uniform MNIST datasets.

4.2. Color Selectivity Index

The color selectivity index [10] is a metric to measure how color selective a certain neuron is in the network. Color selective is defined as how much the neuron takes color into consideration for the image classification task. The authors used an opponent-like colorspace rather than an RGB colorspace. They additionally normalized the values of each axis in the range [-1, 1]. The new values of the images are computed with the following three equations.

$$O_1 = (R + G + B - 1.5)/1.5 \tag{4.1}$$

$$O_2 = (R - G) \tag{4.2}$$

$$O_3 = (R + G - 2B)/2 \tag{4.3}$$

The R, G and B correspond to the values in the red, green and blue channels respectively. Whenever an image is used as input to the network, we are able to see which neurons are activated, by which image patches and the value of activation. The color selectivity index is determined by computing two sums. The first sum is over the n image patches that have the highest activation values. The second sum is again over those top n image patches but in grayscale. n can be defined by the user, however, the authors of the paper and we use $n = 100$. The color selectivity index is computed by

$$\alpha(n^{L,i}) = 1 - \frac{\sum_{j=1}^{N} w'_{j,i,L}}{\sum_{j=1}^{N} w_{j,i,L}}, \tag{4.4}$$

where $\alpha(n^{L,i})$ represents the color selectivity index of the i^{th} neuron n in layer L, $w_{j,i,L}$ represents the activation value of neuron i in layer L on the j^{th} ranked image patch and $w'_{j,i,L}$ represents the activation value of its grayscale counterpart. The value of the color selectivity index is between a range of -inf and 1 with 1, at least higher than 0.25, being very color selective and every value below 0.1 being non-color selective. We use this metric in our work to verify the behaviour of several experiments, e.g. the experiments in sections 4.1 and 4.4 of the scientific paper.

4.3. Dataset Examples

This section contains examples of all datasets from all classes used in the scientific paper. The datasets in Figures 4.1 and 4.2 are used in the experiments in section 4.1, 4.2 and 4.3 of the scientific paper. The datasets in Figures 4.3 and 4.4 are used in the experiment in section 4.4 of the scientific paper. The dataset in Figure 4.5 is used in the experiment in section 4.5 of the scientific paper.

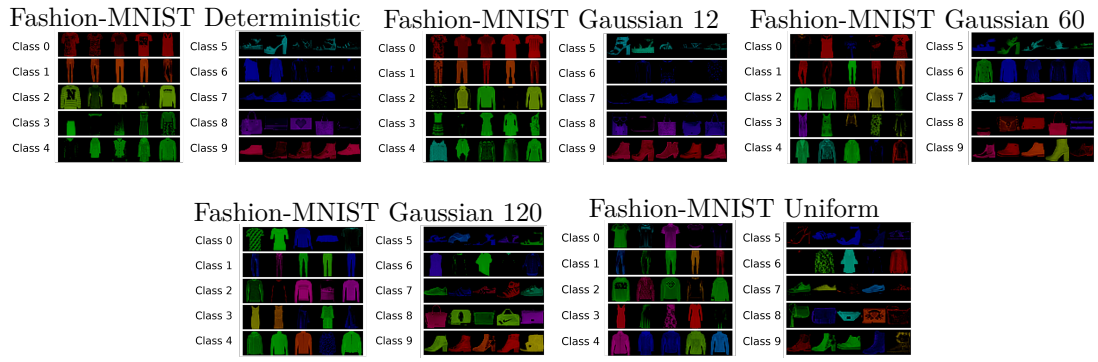

Figure 4.2: Samples of the Deterministic, Gaussian 12, Gaussian 60, Gaussian 120 and Uniform Fashion-MNIST datasets.

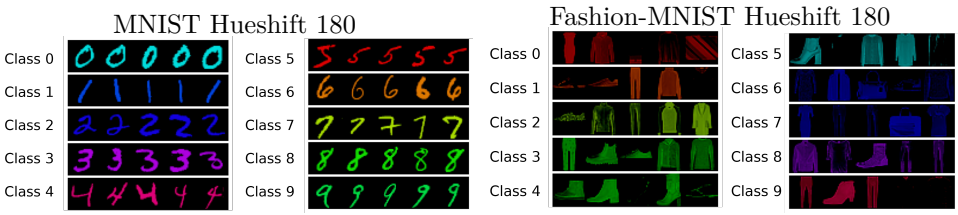

Figure 4.3: Samples of the Deterministic datasets with hueshift 180 of MNIST and Fashion-MNIST.

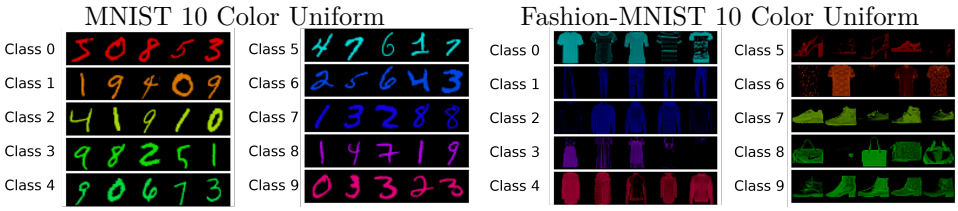

Figure 4.4: Samples of the 10 color uniform datasets of MNIST and Fashion-MNIST.

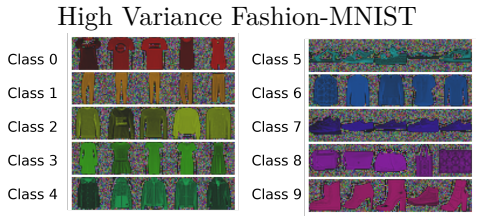

Figure 4.5: Samples of the high variance Fashion-MNIST dataset.

5

Additional Experiments

This chapter contains experiments that were performed during the research but that did not yield results that could be added to the scientific paper. We first explain the optimal comparison which is the comparison of the two types of networks when both have the most number of neurons per convolutional layer. Then we provide additional details on the experiments in section 4.4 of the scientific paper. We conclude with additional experiments on the vehicle color recognition dataset.

5.1. Optimal Comparison

We do an optimal comparison besides the fair comparison in the paper. We define the optimal comparison as the comparison between the 2D and FCD networks when they are at their best performance, in our case, the most number of neurons per layer. This experiment is to investigate whether either type of network performs better in classification whenever more resources are allocated to the network. We compare the performances using the accuracies. We systematically increase the number of input and output channels in the convolutional layers of the 2D and FCD networks. We keep increasing the number of channels per layer until the accuracy saturates, so when the accuracy does not increase anymore with an increase in the number of channels.

Our optimal 2D network is Architecture 2 in Table 2 of the scientific paper. We tried several configurations. The first configuration is the network itself. The second configuration is each input and output channel per convolutional layer multiplied by four. In the third configuration each is multiplied by five and it is multiplied by six in the fourth configuration. The optimal FCD architecture is Architecture 4 in Table 1 of the scientific paper. The first configuration is the network itself. The second configuration is the number of input and output channels per convolutional layer multiplied by four and in the third configuration it is multiplied by five. The results are shown in Figure 5.1.

The third configuration is the best 2D network configuration for both the MNIST and Fashion-MNIST datasets. We did not try any other configurations because the accuracies already saturated and for the Fashion-MNIST dataset, the fourth configuration performed worse than the third configuration on higher standard deviation datasets. The third configuration is the best FCDNet configuration because the performance is already saturated for the MNIST dataset but for the Fashion-MNIST dataset, it is better than the second configuration. The plot in Figure 5.2 shows that the behaviour is similar to the fair comparison in Figure 9 in the scientific paper. Thus an increase in the number of neurons per layer does not give an advantage to either network type on the MNIST or Fashion-MNIST datasets.

5.2. Out of Distribution Dataset Experiments

In section 4.4 of the scientific paper we introduce two type of experiments. We immediately evaluate the network on an out of distribution dataset in the first experiment. In the second and third experiments we train the classification weights on out of distribution datasets before evaluation. A dataset is called out of distribution when the network training on it has not seen the samples before. The purpose of the first experiment is to investigate the behaviour of how color information is processed in the neural networks. The neural networks learn features in the feature extraction part and use those features to classify the input to the classes of the dataset. We find in experiment 1 of Figure 11 in the scientific

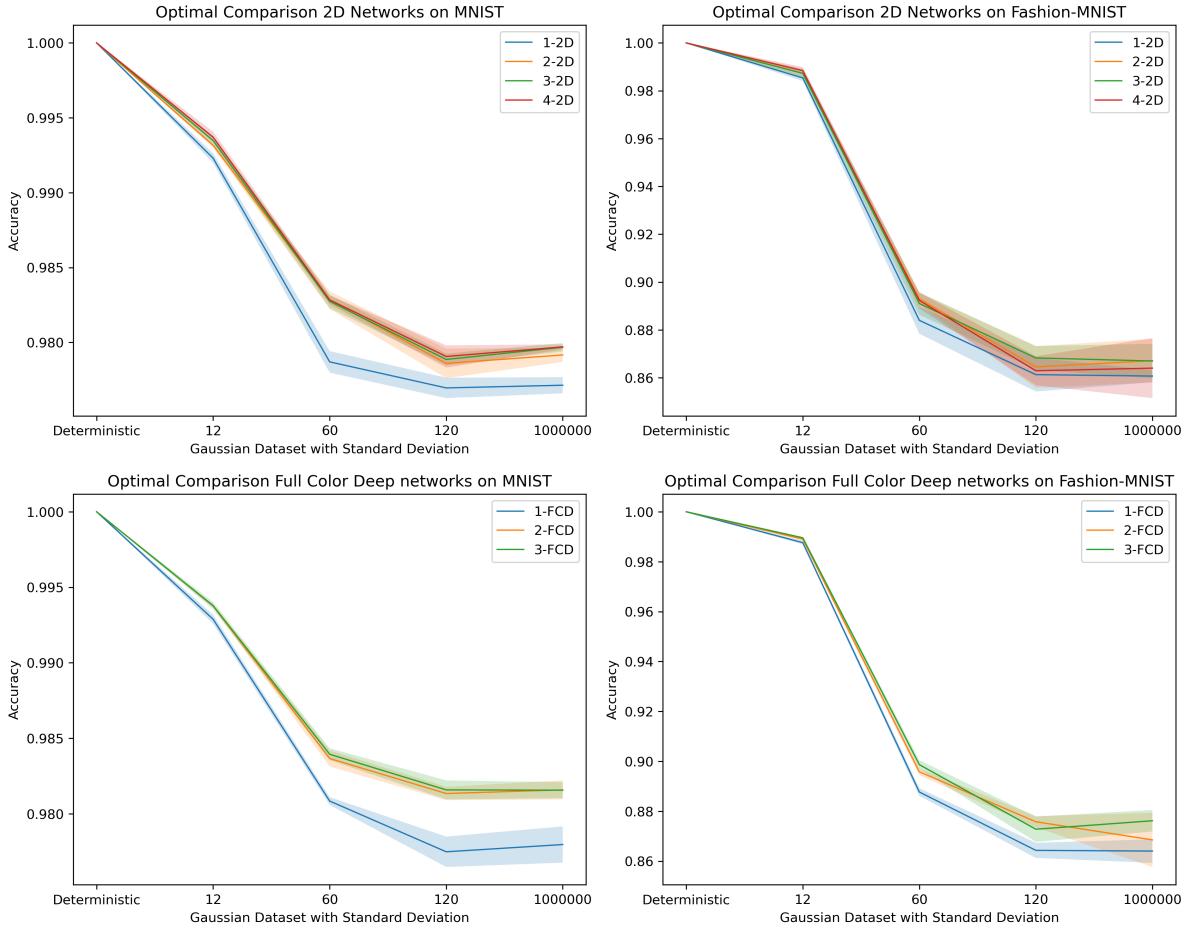

Figure 5.1: Accuracies of the optimal 2D and Full Color Deep networks on the MNIST and Fashion-MNIST datasets.

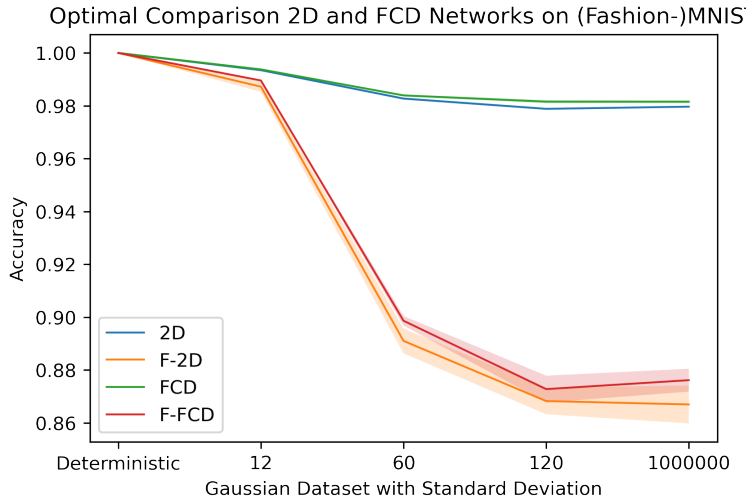

Figure 5.2: Performance comparison of the optimal 2D and Full Color Deep networks on the MNIST and Fasion-MNIST datasets.

paper that both 2D and FCD networks have an accuracy of 0, or close to 0, with the networks pretrained on the Deterministic dataset. The only difference in the pretrain task is the color of the digit in the image. This tells us that the networks only learned color features in the feature extraction part and made connection on the color features and the output classes in the classification part.

The FCD networks, pretained on the uniform dataset, perform better than the 2D networks in experiment 2 of Figure 11. The classification weights are trained in this experiment meaning that the networks can learn proper connections between the features extracted and the output class. The results tell us that the FCD networks extract more color features in the pretrain task than the 2D networks for the linear layer. The FCD networks perform comparably to 2D networks in the pretrain task. Spatial features are the only useful features in this task. The higher accuracies of the FCD networks in experiment 2 of Figure 11 in the scientific paper thus indicate that the FCD networks learned features in the color dimension making them achieve higher accuracies than the 2D networks.

In the third experiment the test dataset has samples that can be any of 10 distinct colors and the label of the image is equal to the color it has. This is different from the other dataset in which the label is the digit. Color is thus the only feature that has a connection to its label. The accuracies of the 2D networks in experiment 3 of Figure 11 for the datasets with high standard deviation are low, which tells us that the networks have learned barely any color features. Meanwhile the FCD networks reach an accuracy of, close to, 1. The classification task itself is fairly easy as it is comparable to training and learning on a deterministic dataset which we find in other plots to reach an accuracy of 1 for both 2D and FCD networks. When the network has to use pretrained weights for a dataset in which the color bias per class is low, however, we find that the FCD networks remarkably outperform the 2D networks.

5.3. Vehicle Color Recognition

We performed an additional experiment, besides the experiments in the scientific paper, on the vehicle color recognition dataset [4]. We use the VGG-M network as Rafegas et al. [10]. They initially used the network because it was similar to the network in another work [3] in which the authors showed that such deep networks start rivalling the performances of primates in representational tasks. Due to this reason and Rafegas et al. showing that the network has color selective neurons throughout all of the layers, we decided to also use this network for our experiments. We translated the network to the optimal FCD network we found in the experiment in section 4.2 of the scientific paper by dividing the number of output channels for each convolutional layers by a factor of $\sqrt{2}$. As shown in Figure 5.4, 2D VGG-M network outperforms the FCD variant. Figure 5.4 shows the accuracy of the 2D VGG-M network versus the accuracy of its FCD counterpart according to Architecture 1 in the scientific paper. The architectures for all networks are shown in Figure 5.3. We find that Architecture 1 is actually outperforming Architecture 4 in the FCD case. This gives us the insight that the optimal architecture is task dependent. Furthermore, Architecture 1 performs comparably to the 2D VGG-M network which is not what we observe in the results when smaller networks are used. In that case the FCD network outperforms the 2D network. These findings tell us that more research has to be done to figure out when to use what architecture and when the FCD networks will actually outperform the 2D networks.

VGG-M Net

- Network:
  - Conv2D: 96 output channels, 7x7 kernel
  - ReLU
  - MaxPool2D (KS = 2, stride = 2)
  - Conv2D: 256 output channels, 5x5 kernel
  - ReLU
  - MaxPool2D (KS = 2, stride = 2)
  - Conv2D: 512 output channels, 3x3 kernel
  - Conv2D: 512 output channels, 3x3 kernel
  - Conv2D: 512 output channels, 3x3 kernel
  - MaxPool2D (KS = 2, stride = 2)
  - AdaptiveAvgPool2D (1, 1)
  - Linear: 512 – 4096
  - Linear: 4096 – 4096
  - Linear: 4096 – 8

FCDNet Architecture 1

- Network:
  - Conv3D: 55 output channels, 3x7x7 kernel
  - ReLU
  - MaxPool3D (KS = (1, 2, 2), stride = (1, 2, 2))
  - Conv3D: 148 output channels, 3x5x5 kernel
  - ReLU
  - MaxPool3D (KS = (1, 2, 2), stride = (1, 2, 2))
  - Conv3D: 296 output channels, 3x3x3 kernel
  - Conv3D: 296 output channels, 3x3x3 kernel
  - Conv3D: 296 output channels, 3x3x3 kernel
  - MaxPool3D (KS = (1, 2, 2), stride = (1, 2, 2))
  - AdaptiveAvgPool3D (3, 1, 1)
  - Linear: 888 – 4096
  - Linear: 4096 – 4096
  - Linear: 4096 – 8

FCDNet Architecture 4

- Network:
  - Conv3D: 68 output channels, 2x7x7 kernel
  - ReLU
  - MaxPool3D (KS = (1, 2, 2), stride = (1, 2, 2))
  - Conv3D: 181 output channels, 2x5x5 kernel
  - ReLU
  - MaxPool3D (KS = (1, 2, 2), stride = (1, 2, 2))
  - Conv3D: 362 output channels, 2x3x3 kernel
  - Conv3D: 362 output channels, 2x3x3 kernel
  - Conv3D: 362 output channels, 2x3x3 kernel
  - MaxPool3D (KS = (1, 2, 2), stride = (1, 2, 2))
  - AdaptiveAvgPool3D (3, 1, 1)
  - Linear: 1086 – 4096
  - Linear: 4096 – 4096
  - Linear: 4096 – 8

Figure 5.3: VGG-M-Net with its parameters and the Full Color Deep network counterparts according to Architectures 1 and 4.

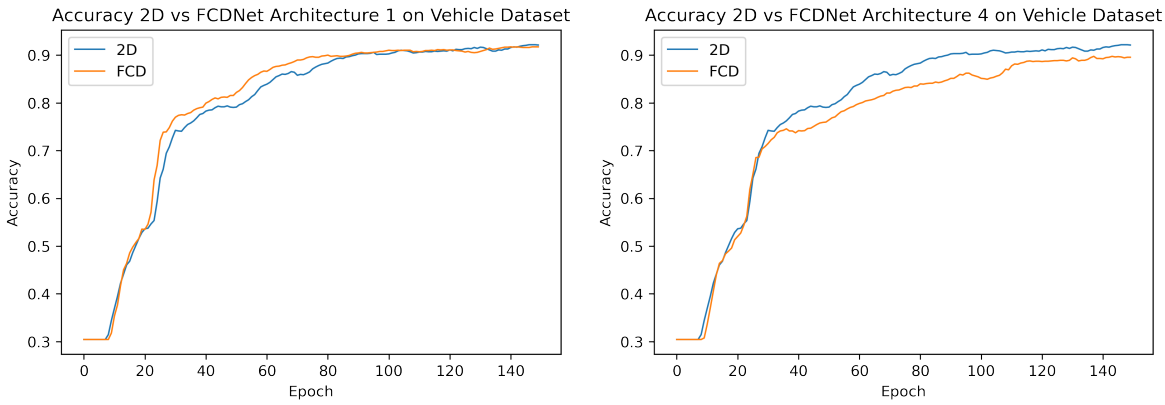

Figure 5.4: Accuracies of the VGG-M network versus the first and fourth proposed Full Color Deep network architectures in the scientific paper.

Bibliography

[1] Pulkit Agrawal, Ross Girshick, and Jitendra Malik. Analyzing the performance of multilayer neural networks for object recognition. In *European conference on computer vision*, pages 329–344. Springer, 2014.

[2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[3] Charles F Cadieu, Ha Hong, Daniel LK Yamins, Nicolas Pinto, Diego Ardila, Ethan A Solomon, Najib J Majaj, and James J DiCarlo. Deep neural networks rival the representation of primate it cortex for core visual object recognition. *PLoS computational biology*, 10(12):e1003963, 2014.

[4] Pan Chen, Xiang Bai, and Wenyu Liu. Vehicle color recognition on urban road by feature context. *IEEE Transactions on Intelligent Transportation Systems*, 15(5):2340–2346, 2014.

[5] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999. PMLR, 2016.

[6] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2012.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[8] Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[9] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[10] Ivet Rafegas and Maria Vanrell. Color encoding in biologically-inspired convolutional neural networks. *Vision research*, 151:7–17, 2018.

[11] Waseem Rawat and Zenghui Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation*, 29(9):2352–2449, 2017.

[12] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[13] Satya P Singh, Lipo Wang, Sukrit Gupta, Haveesh Goli, Parasuraman Padmanabhan, and Balázs Gulyás. 3d deep learning on medical images: a review. *Sensors*, 20(18):5097, 2020.

[14] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.