Understanding the Effects of Orthogonal Convolution in Transfer Learning for Medical Image Analysis

by

Tsz Chan

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved April 2023 by the
Graduate Supervisory Committee:

Baoxin Li, Chair
Jianming Liang
Yezhou Yang

ARIZONA STATE UNIVERSITY

May 2023

ABSTRACT

Insufficient training data poses significant challenges to training a deep convolutional neural network (CNN) to solve a target task. One common solution to this problem is to use transfer learning with pre-trained networks to apply knowledge learned from one domain with sufficient data to a new domain with limited data and avoid training a deep network from scratch. However, for such methods to work in a transfer learning setting, learned features from the source domain need to be generalizable to the target domain, which is not guaranteed since the feature space and distributions of the source and target data may be different. This thesis aims to explore and understand the use of orthogonal convolutional neural networks to improve learning of diverse, generic features that are transferable to a novel task.

In this thesis, orthogonal regularization is used to pre-train deep CNNs to investigate if and how orthogonal convolution may improve feature extraction in transfer learning. Experiments using two limited medical image datasets in this thesis suggests that orthogonal regularization improves generality and reduces redundancy of learned features more effectively in certain deep networks for transfer learning. The results on feature selection and classification demonstrate the improvement in transferred features helps select more expressive features that improves generalization performance. To understand the effectiveness of orthogonal regularization on different architectures, this work studies the effects of residual learning on orthogonal convolution. Specifically, this work examines the presence of residual connections and its effects on feature similarities and show residual learning blocks help orthogonal convolution better preserve feature diversity across convolutional layers of a network and alleviate the increase in feature similarities caused by depth, demonstrating the importance of residual learning in making orthogonal convolution more effective.

ACKNOWLEDGMENTS

I would like to thank my advisor and chair Dr. Baoxin Li for his knowledge, feedback, and patience through the completion of this project. This thesis would not be possible without his invaluable guidance which has pushed me to achieve academic rigor in my research. I would also like to express my deepest gratitude to my thesis committee, Dr. Jianming Liang and Dr. Yezhou Yang, for their willingness to work with me. Their insightful suggestions and experience helped shape this research project.

Finally, I would like to thank my family for their unwavering support to help me reach my goals and all my friends, especially Sajid Anwar, Nina Winemiller, Kelly Huynh, Joe Dong, Michael Bi, and Mark Bi, for their continuous advice and encouragement. I am incredibly grateful for all the joy and wonderful memories they have brought me that have served as great motivation for my work.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Chapter 1

INTRODUCTION

Use of deep convolutional neural networks (CNNs) to solve a target task in a small data regime remains a challenging problem despite numerous recent advancements in the field of deep learning related to architecture designs and methods to improve training and learning. In a supervised learning setting, large labeled datasets are often required to sufficiently train a deep network to learn the underlying features before the model is capable of solving a task [23]. For a task such as natural image classification, the problem is made less difficult given the abundance of labeled data that is openly available for training e.g. ImageNet [7]. However, for more complex image recognition tasks such as medical image analysis, the problem is more challenging given the difficulty in acquiring enough data to sufficiently train a deep CNN to accurately classify medical images.

1.1 Traditional Approaches to Medical Image Classification

Medical image diagnosis is cost intensive. Images generated from computerized tomography (CT) scans and magnetic resonance imaging (MRI) are expensive to produce and require adequate resources such as time and expertise to examine and interpret the results. Moreover, this process is susceptible to human error. To make diagnoses more cost effective and improve general patient care by reducing possible medical errors, diagnostic tasks can be automated by training a model to learn specific features pertaining to the imaging data and make data-driven predictions, which can assist with efficiently producing a more accurate diagnosis.

Traditional machine learning approaches focused on using handcrafted features

to classify medical images. Classifiers built using methods such as random forest (RF) and support vector machines (SVM) operated on features extracted from a feature extraction step through methods such as principle component analysis (PCA) [6, 20] or scale-invariant feature transform (SIFT) [5] before classification can be done. These classifiers were particularly well-suited for medical image analysis because they perform well with limited imaging data. While traditional methods have been proven to do well in a small data setting, they make use of handcrafted features, which may require prior domain knowledge to generate or may not be robust enough to capture and make use of inherent information embedded in the data. Furthermore, deep CNNs have been shown to outperform traditional machine learning methods when working with larger datasets, making deep networks a more attractive option as they remove the need for feature engineering and offer greater representational power than traditional methods provided there is sufficient training data.

1.2 Current Deep Learning Landscape in Medical Image Classification

Deep CNNs have been studied extensively in a wide range of applications such as image recognition and natural language processing. Various works in literature have studied different methods such as weight initialization schemes [12, 14, 16], data augmentation [41], skip connections in network architectures [15], and regularization techniques [18, 31, 44] to improve learning, convergence, and overall performance of deep CNNs. However, despite these improvements, the lack of available training data remains a central problem that limits the applicability of deep CNNs in a very small data regime.

The challenging nature of recognizing more subtle details in medical images makes medical image analysis a difficult task. This is further complicated by the lack of available labeled imaging data, making deep CNNs less than effective in a limited

data setting as training a network from scratch capable of solving the target task becomes even more challenging. To address this problem, transfer learning has been widely proposed as a method to make deep CNNs usable in a small data regime [28, 34, 50]. In transfer learning, the features captured by the model learned from one data source is transferred and applied to a new data domain to avoid duplicating effort of training a deep CNN from scratch. However, for this to work, the learned features need to be transferable across the source and target domain, which is not guaranteed as the feature space between the two domains may be different.

More recently, an efficient method for orthogonal convolution was developed and proposed as a way to improve learning of features in deep CNNs [49]. The authors of this research demonstrate orthogonality can be imposed on filters of convolutional layers in a network as a form of regularization loss in the cost function, which can enhance feature diversity and improve generalization performance of a deep CNN. The results of their experiments show OCNN achieves performance gains in several different tasks where the source and target data are in the same domain, but it is unclear how the improvements translate when the domains are different. Thus far, there has not been any research in investigating the impact of orthogonal regularization on deep CNNs in a transfer learning context. Therefore, it is important to take a step towards understanding the impact of orthogonal convolutional in transfer learning for problems in a small data regime such as medical image classification.

1.3 Summary of Contributions

In this thesis, we examine the application of orthogonal regularization in deep CNNs under a limited data setting. We present using orthogonal regularization loss to enhance feature extraction in deep CNNs and investigate the effects on feature selection and classification from a transfer learning perspective using limited med-

ical image datasets. We also analyze the effectiveness of orthogonal regularization on different deep CNN architectures and the impact residual connections have on the benefits provided by orthogonality convolution. This thesis makes the following contributions:

- We demonstrate orthogonal regularization loss can be used in the context of transfer learning to improve learning of generic features in deep CNNs, which can enhance extraction of diverse, expressive features that are transferable across domains for vision related tasks such as medical image classification and improve generalization performance.

- We provide an example approach using orthogonal convolutional neural networks as feature extractors for transfer learning that achieved better classification performance on two medical image datasets than previous literature.

- We show the effectiveness of orthogonal regularization loss is dependent on residual learning and demonstrate residual connections help alleviate the negative effects of depth on orthogonal regularization.

Chapter 2

LITERATURE REVIEW

Deep CNNs have achieved great success in solving image classification problems and have become one of the most widely used tools in computer vision research. Their prevalent use has led to numerous advancements in deep learning in recent years. In this chapter, we review several recent methods in deep learning and highlight their contributions and impact on image classification.

In the following sections of this chapter, we first discuss the changes in CNN architectures through the years (Section 2.1). Next, we describe different data augmentation techniques (Section 2.2) and regularization methods (Section 2.3) used to improve generalization performance. In the last section, we discuss the problems of insufficient data and different data distributions and the use of transfer learning (Section 2.4) to address these problems.

2.1 CNN Architectures

Since the introduction of AlexNet in 2012, which set a new record for classification in the ImageNet Large-Scale Visual Recognition Challenge 2012 (ILSVRC 2012) and showed the feasibility of training a deep CNN with many layers to accurately classify images [23], various architectures have been proposed and achieved state of the art classification performance on existing and new benchmark datasets.

Inception Network

One example of an architecture that has achieved state of the art classification performance is GoogLeNet or Inception network proposed by Google in 2014. Motivated

by the straightforward idea that increasing the network size through depth (i.e. the number of layers) or width (i.e. the number of units in a layer) would yield higher performance, the Inception network introduces an Inception module used to make a large CNN more trainable by making the model more computationally efficient and less susceptible to overfitting due to the increased number of parameters from additional depth or width [46]. The Inception module (Figure 2.1) works by performing multiple convolutions and a pooling operation on the input. Specifically, the input is convolved with a 1×1, 3×3, and 5×5 filter and pooled separately to produce 4 different outputs which are concatenated. Since using 5×5 filters is expensive because there are more parameters, 1×1 convolutions are convolved with the input before applying 3×3 and 5×5 convolutions to reduce the dimensions of the output feature maps. By repeatedly stacking these modules on top of each other, this design effectively allows an increase in the depth of the network to improve discriminative power without compromising computational complexity from increasing the size of outputs in each module. This architecture along with similar findings from the Visual Geometry Group (VGG) network [43] around the same time demonstrated the benefits of deeper networks on classification performance.

Residual Neural Network

While research has shown an increase in the size of networks tend to improve performance, deeper CNNs were found to be more difficult to train and harder to approximate a more optimal solution of the weights in a feasible amount of time [14]. In particular, the accuracy degrades with more layers added, leading to higher training and test error. Furthermore, the difficulty of optimizing a system varies and depends on the problem and not all models are similarly easy to train. Originally proposed in 2016, the residual network (ResNet) presents the concept of a residual learning

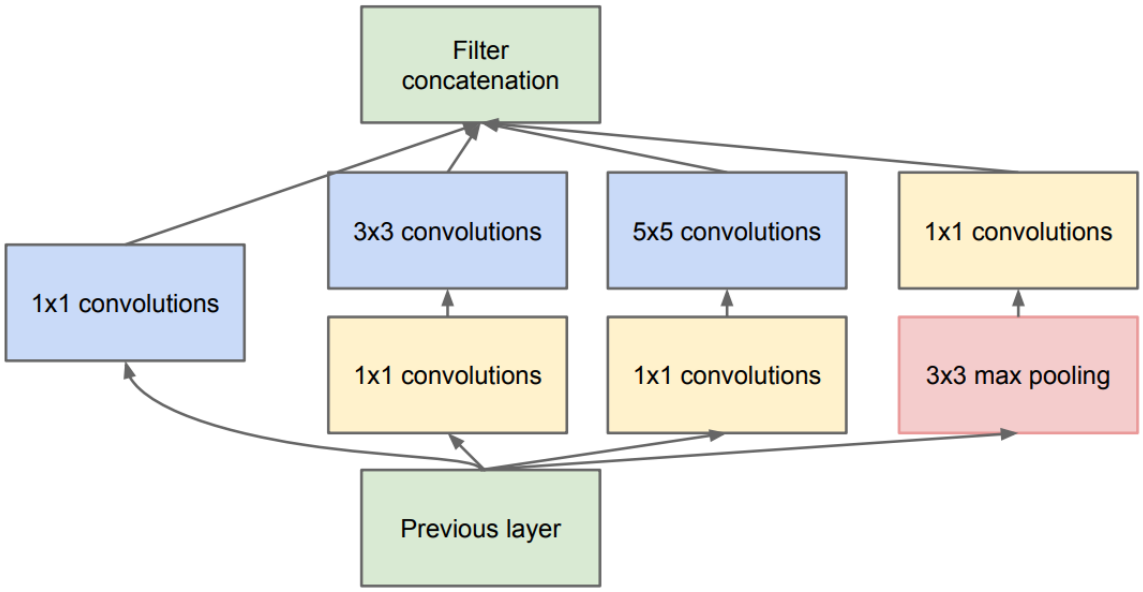


Figure 2.1: An Inception module depicting the different operations used to construct an Inception network [46].

block (Figure 2.2) to address this degradation problem in increasingly deeper CNNs. A residual learning block consists of stacking multiple layers with a connection from an input of a layer to an output of a subsequent layer. In each connection, the input of an early layer is passed directly to a later layer without any modifications and summed with the output of a later layer. This type of skip connection forces the layers to approximate a residual function between the input and output, i.e.

$$H(x) = F(x) + x \tag{2.1}$$

where $H(x)$ is the desired mapping and $F(x)$ is the approximated function, instead of

$$H(x) = F(x) \tag{2.2}$$

when a skip connection is not present. Since the desired mapping is based on a previous input, it is hypothesized that it would be easier to approximate a function with reference to a previous input than to approximate a completely new function.

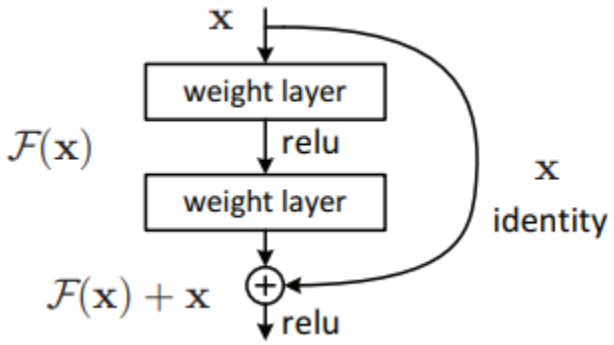


Figure 2.2: A residual learning block that serves as the building blocks of a ResNet [14].

In a way, the residual connection could be understood as improving information flow from early layers to subsequent layers of a network to provide better learning and make it easier to find a more optimal solution of the weights. This proposed design reported a boost in classification performance on the ImageNet dataset compared to plain deep networks without residual connections [14], and has led researchers to new studies and adopt the use of residual learning in existing and new architectures such as in a hybrid version of Inception known as InceptionResNet [45].

Densely Connected Convolutional Network

Inspired by the use of skip connections to address the issue of information loss by the time data has propagated to the end layers of a network, the Dense Convolutional Network (DenseNet) proposes a more extreme approach of using skip connections to connect all layers with each other to maximize information flow through a network [17]. Each dense block (Figure 2.3) is made up of multiple layers with each layer forwarding its output to all subsequent layers in the block. Unlike ResNet which uses summation for its skip connection, a dense block uses concatenation to concatenate all feature maps from preceding layers to form one input for processing. This design encourages feature reuse throughout the network as different feature maps computed

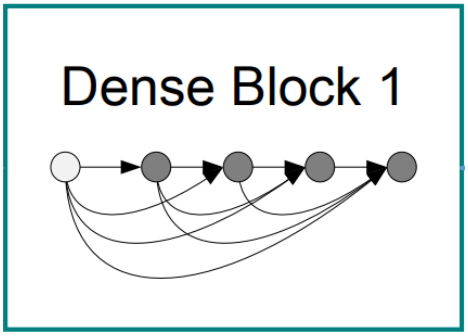


Figure 2.3: A dense block with skip connections feeding the outputs of early layers to subsequent layers [17].

in preceding layers are forwarded and reused in the inputs of subsequent layers, which increases variation of inputs at every layer and drives the model to learn a more compact representation of the data. DenseNet offers an alternative approach to increasing depth or width to improve a network's discriminative power without sacrificing model complexity and efficiency from adding more layers or units to a layer by encouraging feature reuse to reduce redundancy in features and improving compactness of the model.

Improvements in architectures have proved valuable to increasing performance as demonstrated across different classification benchmark datasets such as ImageNet [7], Canadian Institute For Advanced Research 10 (CIFAR-10) and its 100 class variant (CIFAR-100) [22], and Street View House Numbers (SVHN) [30]. However, the improvement in performance from architectural changes have largely benefitted tasks that dealt with large datasets and not those with limited data. In cases where there is little labeled data available such as medical image classification, training a deep CNN from scratch to solve the target task remains a challenge despite innovation in architecture design and the use of state of the art neural networks due to deep CNNs having many parameters across different layers in a network. With additional depth and width to increase representational power, the models have more parameters and become even more complex. When there is a lack of sufficient training data, these

models may be overparameterized, i.e. the model has more parameters than necessary to represent the training data. This may lead to overfitting where the model learns to only predict labels from training data and is incapable of generalizing to unseen samples. Architectural changes such as dense blocks and replacing fully connected (FC) layers with a global average pooling layer [27] have been proposed as alternatives to avoid further increasing model complexity to alleviate overparameterization; however, deep CNNs remain difficult to train from scratch and suffer from overfitting when dealing with limited training data. In the next sections, we discuss different techniques used to address the problem of limited data and overfitting.

2.2 Data Augmentation

The performance of deep learning models are heavily reliant on the training dataset. Without enough data to sufficiently train a deep network, the model may suffer from overfitting and may not generalize well to new data. One way of dealing with the problem of limited data is to use data augmentation techniques to artificially increase the amount of training data available. This solution offers two main benefits: it expands the size of the dataset available for training and it enhances the quality of the dataset by increasing variation in the samples of the data.

Simple Image Transformations

The most basic form of data augmentation is to use simple transformations to synthesize additional samples of data that can be used for training. Some of these transformations include flipping, rotation, cropping, translation, and noise injection [41]. Flipping can be done by reflecting an image over its x-axis or y-axis, rotation can be done by turning the image between 1 to 359 degrees, and cropping can be done by selecting a part of the image to keep and removing the unwanted areas. Similar to

cropping, translation can be done by shifting the image in any direction so only the desired part of the image is kept in frame. Noise injection can be done by injecting an image with values randomly sampled from a distribution. Some examples include Gaussian, salt, pepper, and salt and pepper noise. Besides these simple geometric transformations, more complex image processing operations such as applying image filters to smooth or blur images can be used to further increase the size of the original dataset for training. All of these augmentation techniques can be combined to produce an augmented training set that is several times larger than the original dataset and improve variance in the data which helps with training.

Cutout, Mixup, and CutMix

More recently, researchers have come up with several new augmentation strategies to address the problem of limited data and further improve generalization and model performance. Cutout [8] and Mixup [52] are two proposed techniques that have been shown to yield performance gain in several benchmark datasets. In Cutout, a random pixel is selected as the centerpoint of a region to place a mask of zero values over the image. This method augments the training set with partially occluded versions of the original images and encourages the network to learn to rely more on surrounding features that are less important when making predictions and become less reliant on prominent features, which may not always be present. In Mixup, two samples are randomly selected from the dataset and their inputs and labels are linearly interpolated to synthesize a new sample. This type of augmentation has been shown to be a form of regularization that helps with generalization [53]. The creation of Cutout and Mixup have also inspired CutMix [51] which combines the two methods by replacing the removed patch of an image in Cutout with a region from another image with their labels mixed (Figure 2.4). These new data augmentation strategies

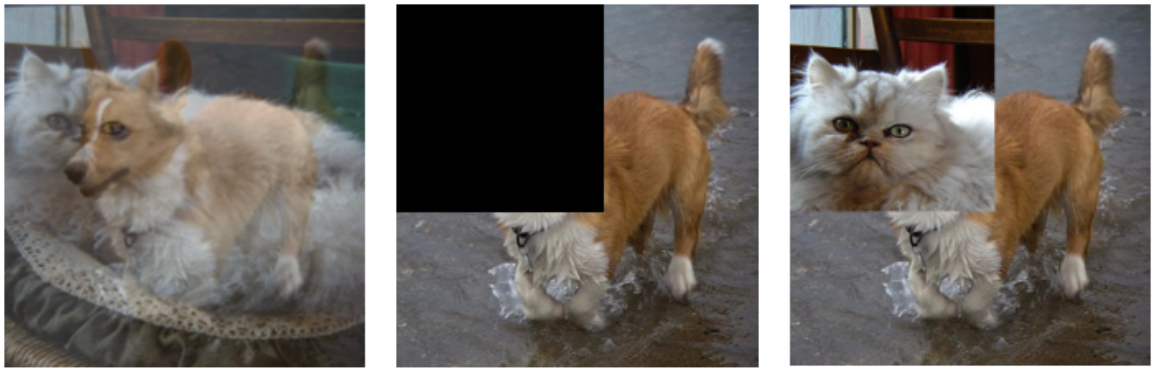


Figure 2.4: From left to right: examples of images transformed using Mixup, Cutout, and CutMix [51].

can be used to inflate the original dataset and yield new performance improvements that simpler augmentation methods may not provide.

Generative Modeling

Generative modeling using generative adversarial networks (GANs) [13] is another approach that has been used for data augmentation [33, 36]. Unlike discriminative models which model the conditional distribution $P(y|x)$ and can be used to predict a target Y given some observation X, generative models model the joint distribution $P(x, y)$ which can be used to generate a sample (x, y) from the approximated distribution. As a type of deep generative model, GANs are constructed from neural networks (a generator and a discriminator). The generator receives a random input to generate a sample of plausible data and the discriminator distinguishes the generated data from a sample of real data. The two networks are repeatedly pitted against each other until training stops, at which point the generator may be able to produce a sample with characteristics resembling the real data that can fool the discriminator. With a sufficiently trained GAN, the original dataset can be augmented with new samples generated from the approximated distribution as if the artifical data came from the actual distribution that produced the original data. Using a GAN to gen-

erate new data seemingly from the original data's distribution, new samples can be added to the original dataset, making it possible to train a deep network to solve a target task that previously only had limited data.

Although data augmentation provides a way to synthesize new samples that can be used with the original dataset to help train deep networks, whether or not the augmentation techniques are applicable to the problem and can help improve classification performance depends on the original dataset and the task. For instance, simple geometric transformations such as flipping, rotation, cropping, translation may improve performance of a model that is trying to classify a car or an airplane by synthesizing additonal images in different orientations and positions, which may help avoid any possible orientation and positional bias in the dataset. More complex augmentation methods such as Cutout, Mixup, and Cutmix may be useful in further improving performance if the dataset contains images of occluded cars or planes. However, applying these augmentation strategies to a dataset of letters and digits for a letter and digit recognition task would alter the labels of some images post-transformation and result in negatively impacting the model's ability to accurately predict the correct digit or letter (e.g. rotating or flipping the digit 6 would change it to a 9 or occluding the letter "R" would change it to the letter "P", resulting in the model learning to misclassify the two digits and letters). Furthermore, training a GAN to approximate the real data distribution so that the model can be used to generate new samples for data augmentation is itself a difficult task as training the deep GAN requires enough quality data to begin with [19]. Modeling the real data distribution of medical images compared to natural images is much more difficult due to the intricacies of the data. While it is possible the generated samples from a GAN is realistic enough to fool the discriminator, the approximated distribution may remain different enough than the real distribution such that the deep network trained on the augmented dataset

would not generalize well and yield inaccurate results. Despite data augmentation being a standard solution to help overcome the problem of limited data and help address overfitting, the benefits are domain and task dependent and improvement in performance is not guaranteed, requiring the need for possible alternative methods.

2.3 Regularization Methods

Regularization is one way of addressing overfitting to improve generalization performance of deep CNNs. The idea behind regularization is that finding a set of optimal weights for a model through approximation in an unbounded search space is hard. By constraining the parameter space through the use of some constraints, it would be less difficult for the weights to converge to a more optimal solution in a constrained parameter space. In the previous section, data augmentation is discussed as a way of addressing overfitting by directly inflating the dataset. This can be seen as doing regularization at the data level. Alternatively, regularization can be applied to loss functions in an explicit form and to network architectures in an implicit form.

L1 and L2 Regularization

Traditionally, regularization has been applied to loss functions to improve generalizability of models [31]. A regularization term $R(w)$ is added to a task loss, e.g. cross entropy loss, to form a regularized loss function that is optimized:

$$J(w) = -\sum_{i=1}^{M}\sum_{j=0}^{K} I\{y_i = k\} \cdot log(\hat{y_i}) + \lambda R(w), \tag{2.3}$$

where M is the number of samples, K is the number of classes, I is the function evaluating to 0 or 1, y_i is the ground truth label of a sample, $\hat{y_i}$ is the corresponding prediction of a sample, λ is the regularization hyperparameter that controls the severity of the penalty of the regularization loss, and w is the weights. Two commonly

used regularization losses to help address overfitting are L1 and L2 regularization loss. The L1 regularization loss, also called L1-norm, is defined as

$$L_1(w) = \lambda \sum_{i=0}^{N} |w_i| \tag{2.4}$$

and the L2 regularization loss, also known as L2-norm or weight decay, is defined as

$$L_2(w) = \lambda \sum_{i=0}^{N} w_i^2. \tag{2.5}$$

In L1 regularization, the weights are encouraged to be sparse, i.e. more zero values, as they are optimized because larger weights would yield a higher loss which counteracts the goal of minizing the loss function. Since L1 encourages sparse weights, the resulting model has reduced complexity as there are fewer weights contributing to the final prediction, making the model less likely to be overparameterized. In L2 regularization, the weights are encouraged to be small but not necessarily zero because larger weights are penalized more heavily as they yield a larger loss compared to smaller weights. This results in the model being more sensitive to outliers in the data as misclassification would yield a bigger loss due to the loss term being squared, but in turn, the model has greater flexibility to accommodate for unseen data when making a new prediction. The sparsity from L1 regularization and smaller weights from L2 regularization due to misclassification encourages a less complex model with higher flexiblity and provide a boost to generalization performance.

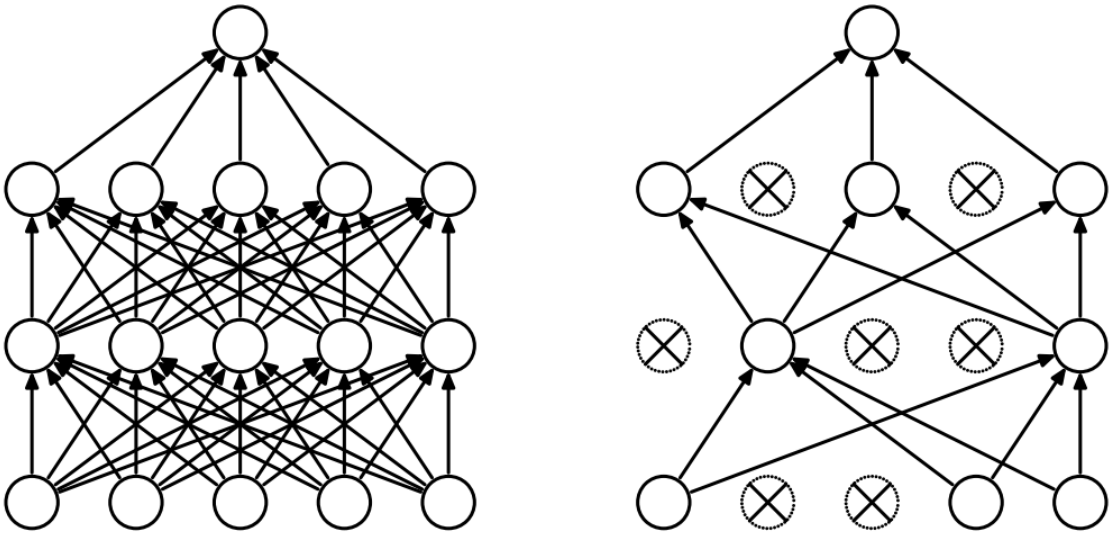
Dropout

Ensemble methods like random forest have been shown to be more accurate and outperform their single model counterparts like a single decision tree [3]. However, applying this idea to neural networks is difficult because training several deep networks is computationally expensive if there are many layers, when the dataset is

large, and training requires many epochs. Dropout [44] is a regularization method that builds upon the idea that combining many models can help improve performance but avoids the actual need to train multiple neural networks. As a form of implicit regularization, dropout is done at the architecture level without requiring change to the loss function by randomly deactivating hidden units in a layer and using only the activated units in the layer to calculate the ouputs and doing backpropagation for weight updates during training (Figure 2.5). The deactivated units are chosen by a hyperparameter p representing some probability of retaining an activated unit. Repeatedly done in training with each step yielding a network with different deactivated and activated units, dropout simulates training many different neural networks and improves variation in the learning of the model by exposing units to different combinations of inputs, which reduces possible correlation among neurons in the network. This averaging effect induced by dropout has been shown to improve overall performance and made it one of the most widely used regularization method for addressing overfitting in training deep neural networks [13, 15, 23].

Batch Normalization

Batch normalization is a method that has become an integral part of model architectures to accelerate training and improve overall model performance. While the method is designed to facilitate training of deep networks and not specifically to address the problem of overfitting, batch normalization has been found to be useful in improving generalization performance, and in some cases make other regularization methods such as dropout obsolete [18]. Figure 2.6 depicts the algorithm of batch normalization. Applied as a layer, batch normalization normalizes a batch of inputs received from its preceeding layer. This normalization procedure has been thought of having the effect of alleviating gradients from exploding or vanishing, which enable



(a) Standard Neural Net (b) After applying dropout.

Figure 2.5: A neural network (a) without dropout and (b) with dropout. Inactive units are crossed out and do not contribute to the final prediction [44].

the use of a higher learning rate to accelerate training by taking larger gradient steps and help the model converge quicker to the optimal solution. Since normalization is done on batches selected randomly, the variance and mean value computed on a batch changes at each step during training. This can be seen as having a stochastic effect similar to stochastic gradient descent (SGD) and can be understood as a form of implicit regularization that helps improve generalization [29]. Although previous research has found batch normalization to be an effective regularization method, its effects are not completely well understood and remain under discussion [2, 29].

Orthogonality

Orthogonality has been extensively explored as a form of regularization in deep CNNs. Existing research have advocated methods such as orthogonal weight initialization [39] and applying orthogonal regularization as constraints during training [1] to improve performance and analyzed their effects on deep networks. In a recent work,

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Figure 2.6: The batch normalization algorithm applied to a mini-batch of inputs during training [18].

the authors of orthogonal convolutional neural networks (OCNNs) [49] developed a simpler, efficient way to enforce orthogonality in convolution layers of a deep CNN by minimizing the total loss

$$L = L_{task} + \lambda L_{orth} \tag{2.6}$$

, where L_{task} is the task loss such as cross entropy, λ is the orthogonal regularization penalty, and

$$L_{orth} = ||Z - I_{r0}||^2 \tag{2.7}$$

is the orthogonal regularization loss. The authors proved that orthogonal regularization loss can be directly computed by calculating the self-convolution

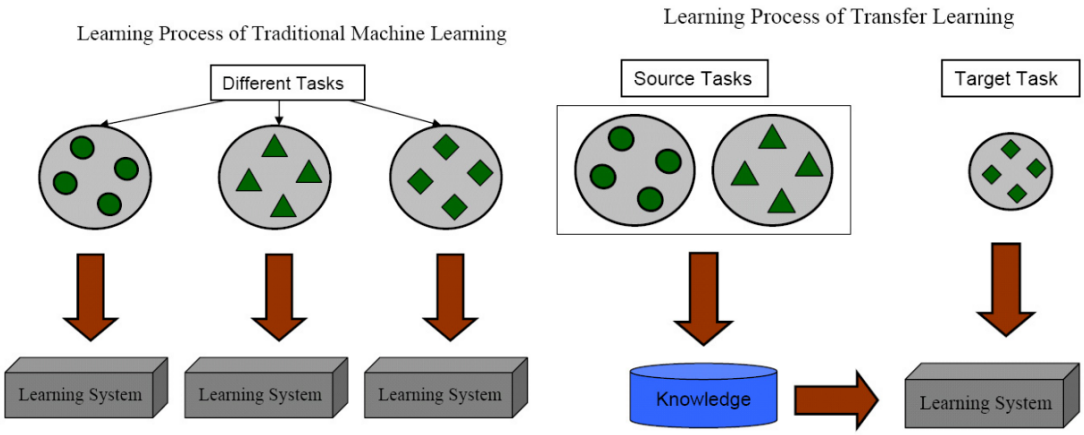
$$Z = Conv(K, K, padding = P, stride = S) \tag{2.8}$$

, where $K \in R^{M\times C\times k\times k}$ is the kernel at a convolutional layer and P and S are the padding and stride used in the layer, and $I_{r0} \in R^{M\times M\times(2P/S+1)\times(2P/S+1)}$ is a simple tensor, where all of its entries are zeros except for the center entry is an identity matrix

[49]. Their experiments and results found different depth variations of ResNets regularized with orthogonal regularization loss all outperform their baseline counterparts without orthogonal regularization and other versions of the model with orthogonality. The ease of implementation and the increase in performance provided by orthogonal regularization loss make orthogonal convolution a promising regularization method for improving generalization performance of a deep CNN.

2.4 Transfer Learning

In an ideal scenario, there is a large amount of data for training and the training data is assumed to share the same features and come from the same data distribution as the target data that will be tested by the model. However, in many real-world applications, training data is scarce and the domain of the training and test data is often different. For medical image analysis, annotating data is resource intensive and susceptible to human error, making data collection difficult. Furthermore, the training and test domain may be different because data are collected from various patients. Under these circumstances, training a deep CNN from scratch to solve the target task in a limited data setting may yield unsatisfactory results. To address this problem, transfer learning has been proposed as a way to improve model performance by transferring knowledge learned from one domain and applying it to a new domain [32]. Figure 2.7 depicts a comparison between learning in traditional machine learning and in transfer learning. In a traditional machine learning process, the learning system is trained from scratch using only the data for the target task, whereas in transfer learning, the learning system is trained using data for the target task and the distilled knowledge learned previously from a separate task.



(a) Traditional Machine Learning (b) Transfer Learning

Figure 2.7: A comparison between the learning process in traditional machine learning and in transfer learning [32].

Feature Extraction

The approach to transfer learning that has been used to make deep CNNs usable in a low data regime with the domain of the training and test data possibly being different is to use pre-trained networks. CNNs have been known to learn low level features like edges, corners, and colors in early layers and learn higher level features that are more task specific through the middle and end layers of a network. These features or knowledge are essentially captured by the weights or learned filters across different layers after training. If a deep network that was previously trained on some prior dataset such as ImageNet is able to achieve acceptable performance, the weights of the network would capture a diverse set of features that may be useful in other recognition tasks and help avoid duplicating the effort needed to learn low level features from scratch.

One common way to use pre-trained networks for transfer learning is to use them as feature extractors [9, 34]. In feature extraction (Figure 2.8), since the output layer of a network is task specific, the output layer is removed while the rest of

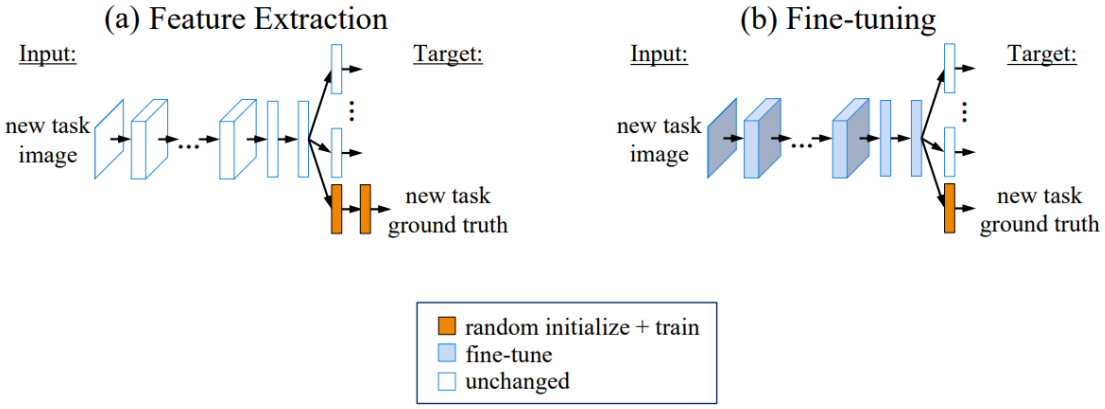


Figure 2.8: Illustration of the difference between (a) feature extraction and (b) finetuning for transfer learning. Parameters except for the output layer are frozen in feature extraction, whereas parameters are permitted to change, with some possibily being frozen, in finetuning [26].

the parameters of the network are left unmodified. A separate classifier is needed to output the final predictions for the new task, which may be a new output layer that is appended to the end of the pre-trained network to derive new predictions or a separate classifier such as a SVM that accepts extracted features as input and is trained on the extracted features to make the final prediction.

Finetuning

Unlike feature extraction where the parameters are not modified, finetuning uses the weights of a pre-trained network for one task as the starting point for training and modifies the parameters of the network. In finetuning (Figure 2.8), all parameters of the existing network can be tuned by retraining the network on the new data using a small learning rate to avoid drastically distorting the parameters. Alternatively, the weights of some layers of the network can be frozen while the last few layers are retrained. This tuning process allows the weights to adapt to the new training data and produce a network capable of solving the new target task. Finetuning has

been used in various medical image analysis tasks to achieve higher performance than training a deep CNN from scratch [40, 48].

Chapter 3

TRANSFER LEARNING WITH ORTHOGONAL CONVOLUTION

Analyzing medical images is a resource intensive process. Accurately interpreting CT scans and MRI images requires extensive domain knowledge, not to mention the process is susceptible to human error. To address these issues, machine learning algorithms have become a popular tool for research to automate the process of medical image analysis to reduce costs and possible medical errors [6, 10]. Traditional machine learning approaches used complex feature engineering to create handcrafted features from images, which required domain specific knowledge, before a classifier is trained on the extracted features to make a prediction. More recently, deep CNNs have become one of the most widely used tool to build end-to-end models for medical image analysis as they are able to learn features automatically from the data without complicated feature engineering to create handcrafted features. Furthermore, deep CNNs have been shown to provide comparable representational and discriminative power to tradtitional non-neural based models, and in some cases better, as demonstrated by their performance on different medical image analysis tasks such as detecting lung nodules [24] and diagnosing diabetic retinopathy [25]. While the performance of deep CNNs have been impressive, training a deep CNN for a task such as medical image classification requires a lot of labeled data, which is difficult to obtain due to costs associated with collecting and labeling medical images. As a way to address this problem, transfer learning has been used as a method to transfer knowledge learned from one domain to another domain to overcome challenges of training a deep CNN without sufficient training data in the original domain.

One common approach to transfer learning for CNNs is to use pre-trained net-

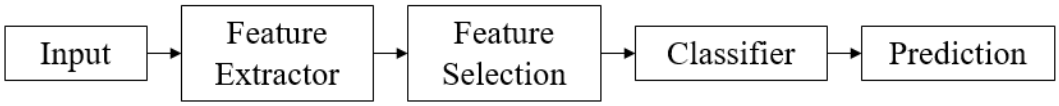


Figure 3.1: The process for making a prediction using transfer learning with feature selection: a feature extractor extracts the relevant features from an input, a feature selection process reduces the dimension of the extracted feature vector, and a trained classifier receives the selected features and makes the final prediction.

works to extract relevant features from the dataset and train a separate classifier such as a SVM, random forest, or another CNN on the extracted features to make the final prediction. In particular, deep networks pre-trained on ImageNet have been often used as feature extractors as their learned features have been demonstrated to be generic and can be repurposed to solve new tasks [9]. Although pre-trained networks can extract relevant features for solving novel tasks, the usefulness of individual features in the entire feature vector varies. For instance, a ResNet50 extracts a 1×2048 feature vector; however, there is no guarantee that all 2048 features in the feature vector are useful for the classifier to learn to make an accurate prediction. In fact, the presence of many irrelevant and redundant features in the feature vector may hinder performance and make solving the learning problem more difficult due to the high dimensionality of the extracted data. One way to address high dimensionality is to use feature selection to reduce the number of features in the feature vector used for prediction by selecting a subset of features that are relevant and discarding the irrelevant ones. This method of using pre-trained networks to extract features, feature selection to reduce extracted feature dimensionality, and a separate classifier to make predictions on the extracted data have been used in literature to detect lymph node metastasis in prostate cancer patients [38] and bladder cancer tissue [37]. Figure 3.1 shows an overview of the process for transfer learning with feature selection.

In a recent work on orthogonality, the authors of orthogonal convolutional neural networks demonstrated imposing orthogonality on convolutional filters through the use of orthogonal regularization loss helps deep networks learn more diverse and expressive features and thereby improve overall network performance [49]. Based on this finding, we investigate the impact of orthogonal convolutional neural networks on feature extraction for transfer learning and the effects of orthogonal regularization on subsequent feature selection and classification in a limited data context. In the end, we analyze and try to understand how orthogonal regularization loss improves feature learning from a transfer learning perspective.

3.1 Related Work

Using pre-trained networks as feature extractors for transfer learning has been predominantly used to transfer learned features from one domain to a new domain and make deep CNNs usable for medical image analysis tasks with limited labeled data. This feature extraction step has been combined with feature selection to improve overall performance. Recently, the introduction of OCNN demonstrated an improvement in learned features when imposing orthogonality on convolutional filters compared to its non-orthogonalized counterparts. In this section, we review the research relevant to our investigation on orthogonality for transfer learning in a limited data setting.

3.1.1 Hybrid Approach for Prostate and Bladder Cancer Detection

A three step process of feature extraction, feature selection, and classification is used to detect lymph node metastasis in MRI images of prostate cancer patients [38]. The proposed method used ResNet18 pre-trained on ImageNet to perform feature extraction, a custom feature selection algorithm to reduce the number of features, and a decision tree classifier to distinguish between normal and metastatic lymph nodes in

the MRI images. Using this hybrid approach, the authors found using a pre-trained CNN for feature extraction outperformed traditional methods such as gray level co-occurrence matrix (GLCM) and Gabor filters used for texture analysis. Furthermore, the authors found using feature selection to reduce the number of features before classification improved overall classification performance for all three types of feature extraction method.

In a similar work, this three step hybrid approach is used to detect normal and bladder cancer tissue in CT images of bladder cancer patients [37]. The authors selected five different deep CNNs pre-trained on ImageNet and finetuned the models using CT scans from the bladder cancer dataset, and used a custom feature selection algorithm to reduce the number of features before classification. In their work, they examined five different types of classifiers – k-nearest neighbor (KNN), naive Bayes, SVM, discriminant analysis, and decision tree – to evaluate the performance of their hybrid approach. The results of their experiment showed feasibility of using deep CNN as feature extractors for medical image analysis and the effectiveness of a hybrid approach.

3.1.2 Orthogonal Convolutional Neural Networks

Orthogonality is a desirable property in deep CNNs. Previous work has shown orthogonality improves speed and stability of convergence and enhances network performance [1]. More recently, the authors of orthogonal convolutional neural networks proposed a new method for imposing orthongonality by regularizing the convolutional layers with orthogonal regularization loss [49] to improve generalization performance of deep CNNs. The authors implemented an orthogonal convolution version of three different ResNets with varying depths and compared their performance to baseline versions without orthogonal regularization and networks with a different implementa-

tion of orthogonality. The results of their experiments showed orthogonally regularized networks outperform other versions on classification benchmark datasets such as CIFAR-10 and ImageNet among other tasks across all three variations of the architecture. In their analysis, they found learned filters of an OCNN are less correlated, suggesting OCNN learns more diverse and expressive features that helps improve generalization performance [49]. However, it is unclear if the improvements to learned features by orthogonal convolution translates in a transfer learning setting.

3.2 OCNN for Transfer Learning

In this section, we introduce the datasets used in our investigation and use pretrained networks to extract features from the data and analyze the extracted features. We present our methodology for investigating the impact of OCNN for transfer learning and present our experiments and results.

3.2.1 Datasets

In our investigation, we use the prostate cancer dataset and bladder cancer dataset previously used in the research for automatically detecting lymph node metastasis in prostate MRI images [38] and bladder cancer tissue in CT scans [37]. The prostate cancer dataset contains 126 64 × 64 region of interests (ROI) from different MRI images, with 84 of them depicting a ROI where no cancer tissue is present (normal or control class) and the other 42 depicting a ROI where cancer tissue is present (lesion class). The bladder cancer dataset contains 200 64 × 64 ROIs from different CT scans with 100 images depicting a normal region and the other 100 containing a cancerous region. Figure 3.2 and Figure 3.3 show examples of normal and lesion images from the prostate and bladder cancer datasets.

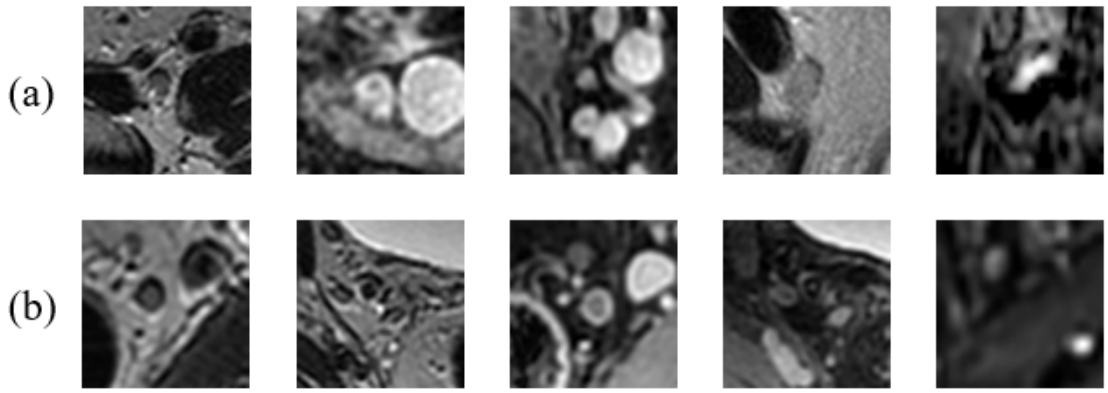


Figure 3.2: Examples of MRI images from the prostate cancer dataset depicting ROIs of (a) normal prostate regions and (b) cancerous prostate regions.

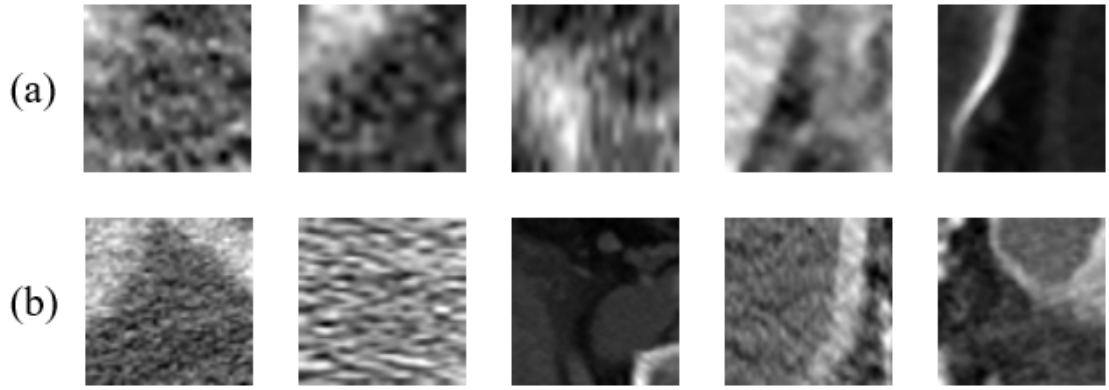


Figure 3.3: Examples of CT scans from the bladder cancer dataset depicting ROIs of (a) normal bladder regions and (b) cancerous bladder regions.

3.2.2 Preliminary Feature Analysis

We analyze the features extracted from the prostate and bladder cancer datasets using pre-trained networks without orthogonal convolution to gain some initial understanding of the distribution of the data. Four different types of architectures – DenseNet121, InceptionV3 [47], InceptionResNetV2, and ResNet50V2 – pre-trained on ImageNet are selected from Tensorflow to use as feature extractors. Since the networks are pre-trained on ImageNet which expect the input size to be $224 \times 224 \times 3$, the images from the prostate cancer and bladder cancer datasets are resized before

Model	Feature Vector Size
DenseNet121	1×1024
InceptionV3	1×2048
InceptionResNetV2	1×1536
ResNet50V2	1×2048

Table 3.1: Size of the extracted feature vector of each of the four selected models.

their features are extracted. Table 3.1 shows the output feature vector size of the four architectures.

We examine the mean and standard deviation of all extracted features between the normal and lesion classes in both datasets. We use kernel density estimation [42] to approximate, visualize, and compare the distributions of the two classes. Figure 3.4 and Figure 3.5 show the distributions of the mean and standard deviation of all extracted features belonging to the normal and lesion classes in the prostate and bladder cancer datasets for each selected model. For the prostate cancer dataset, the distributions of the mean and standard deviation of features between the two classes largely overlaps and shares a similar location of the peak across all the selected models, making it difficult to discern any differences between the distributions. However, the height of the peak differs between the two classes, and in some instances such as Figure 3.4 (a), (c), and (d) there appears to be a small second peak towards the right end of the distribution for the lesion class unlike the normal class. Similar findings are observed in the distributions of the lesion and normal class in the bladder cancer dataset. There is a large overlap in both class distributions; however, the height of the peak is different between both class distributions and the location of the peak as shown in Figure 3.5 (a) and (c) are different. Furthermore, a smaller second peak can be observed in Figure 3.5 (b) and (d).

In addition to examining the mean and standard deviation of all features, we

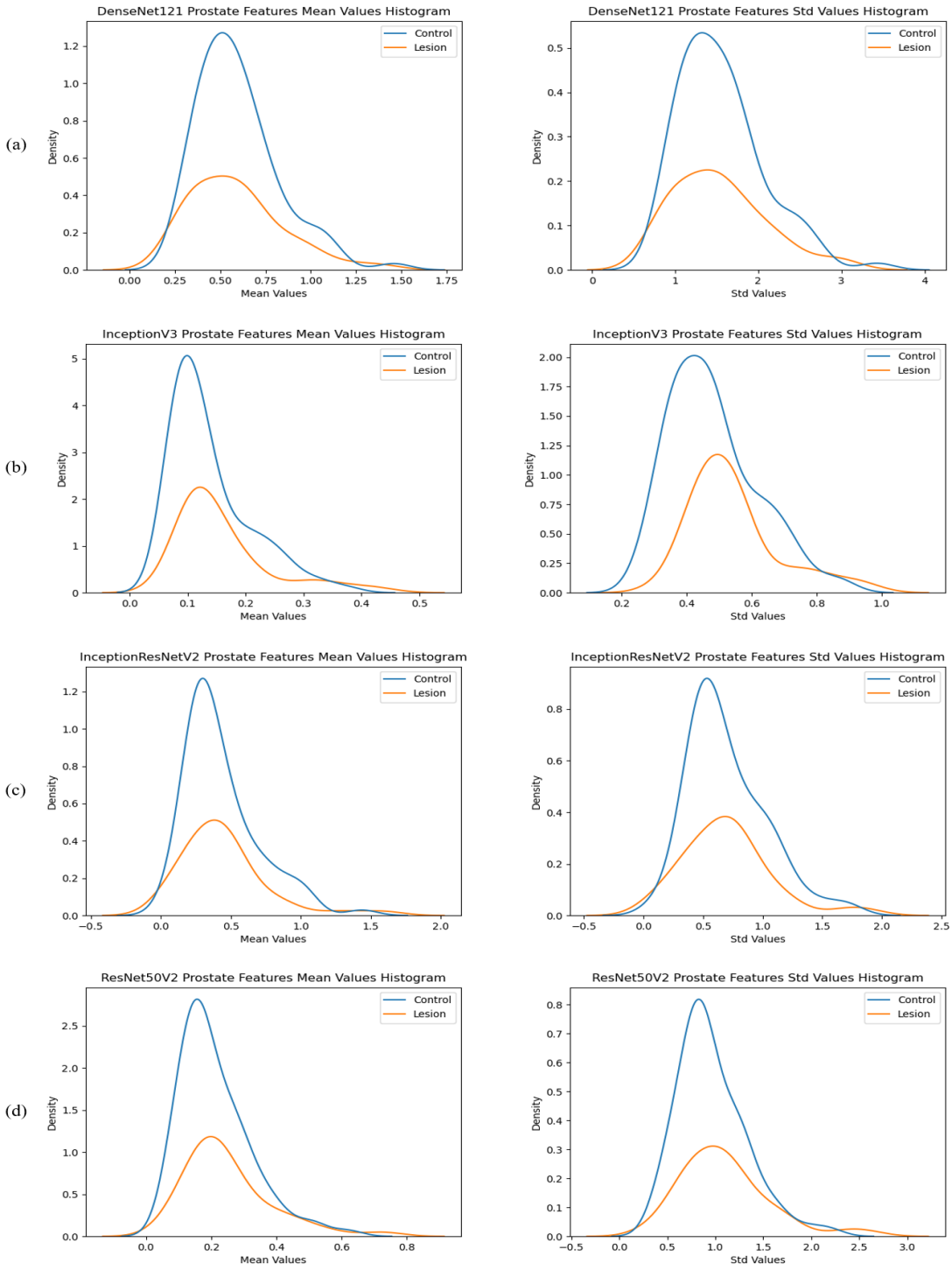


Figure 3.4: Kernel density estimated distributions of the mean and standard deviation of extracted features belonging to the normal and lesion classes in the prostate cancer dataset. The features are extracted from the two datasets using (a) DenseNet121, (b) InceptionV3, (c) InceptionResNetV2, and (d) ResNet50V2.

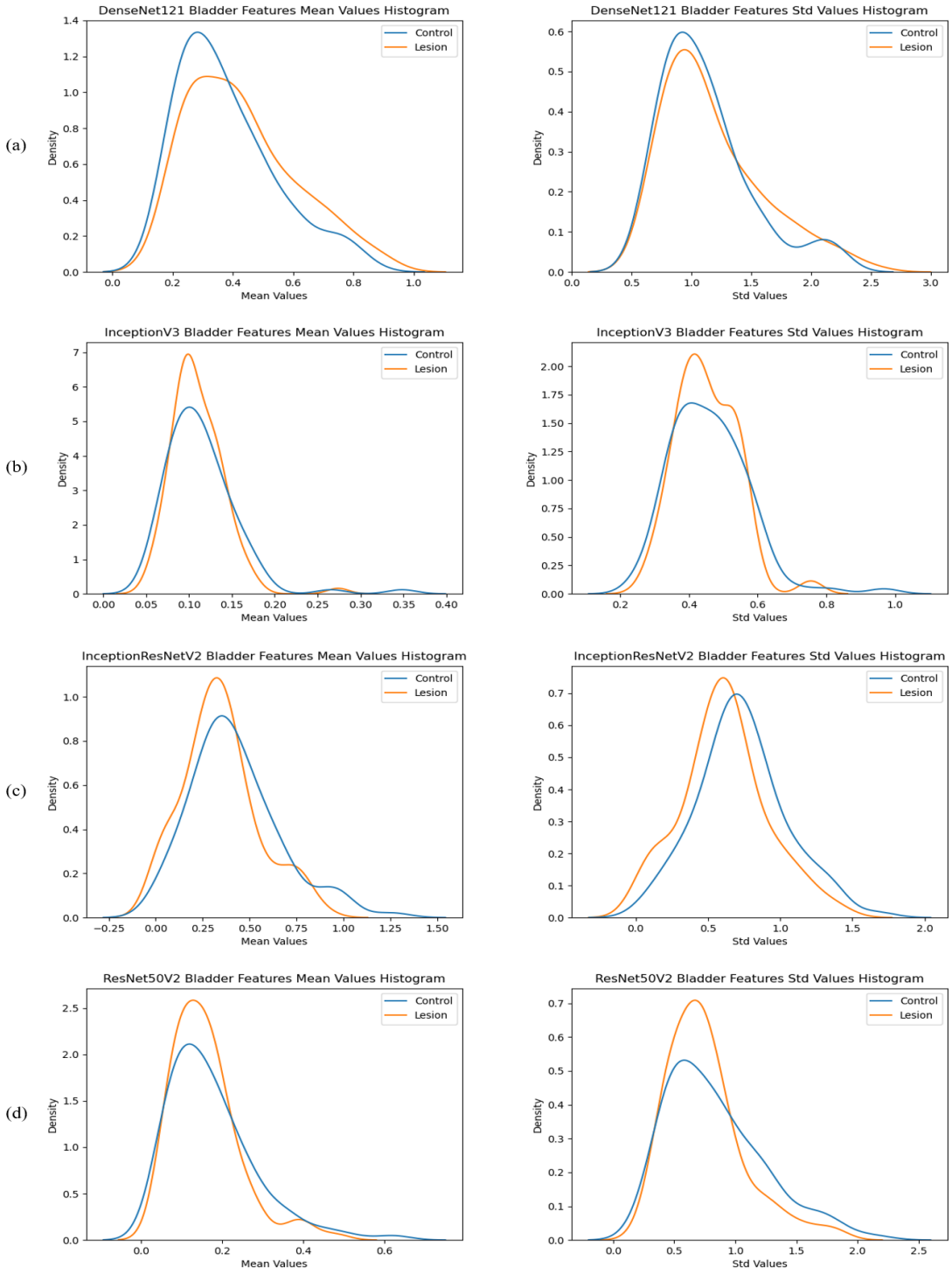


Figure 3.5: Kernel density estimated distributions of the mean and standard deviation of extracted features belonging to the normal and lesion class in the bladder cancer dataset. The features are extracted from the two datasets using (a) DenseNet121, (b) InceptionV3, (c) InceptionResNetV2, and (d) ResNet50V2.

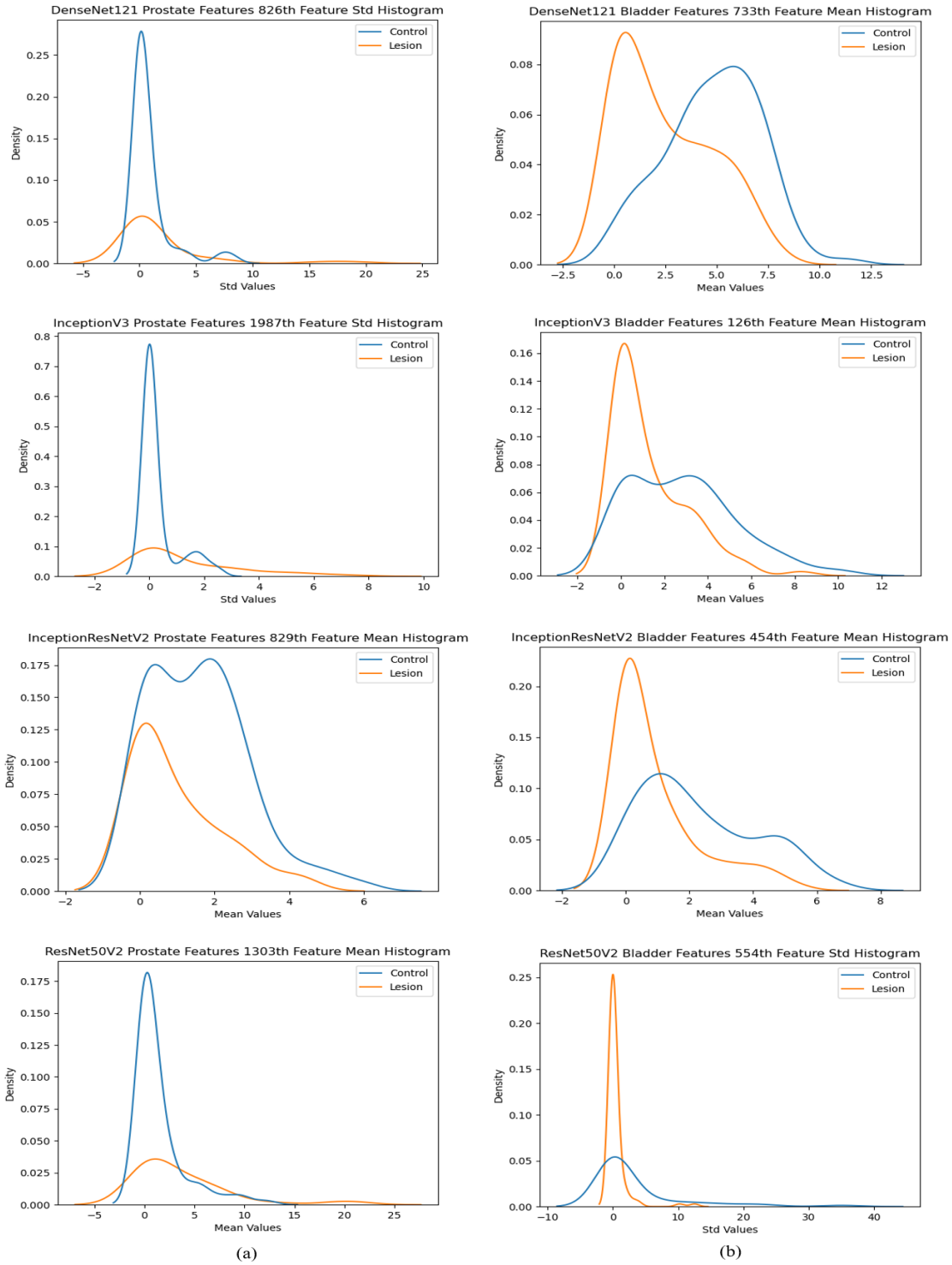


Figure 3.6: Kernel density estimated distributions of the mean and standard deviation of individual features belonging to the normal and lesion classes in (a) the prostate cancer dataset and (b) the bladder cancer dataset for DenseNet121, InceptionV3, InceptionResNetV2, and ResNet50V2. Individual features are selected by computing the mean and standard deviation per feature in each class and finding the top five features with the largest difference in value between the two classes.

also examine the mean and standard deviation of individual features that have the largest difference in value between the two classes to see if the two distributions are distinguishable. For each class in the dataset, the mean and standard deviation is computed for each feature in the extracted feature vector, and the top five features that have the largest difference in value between the normal and lesion classes are selected for visualization. Figure 3.6 shows the distributions of individual features between the two classes in each dataset. We observe more noticeable differences in the distributions of the two classes for individual features such as different location of peaks, number of peaks, and concentration of density for both datasets.

3.2.3 Feature Extraction with OCNN for Feature Selection and Classification

In our investigation, we use DenseNet121, InceptionV3, InceptionResNetV2, and ResNet50V2 as feature extractors to investigate the effects of orthogonal convolution on transfer learning for medical image classification. We use a version of the pre-trained models with orthogonal regularization loss and a version without to extract various features from the datasets, use feature selection to reduce the dimensions of the extracted feature vectors, and train a separate classifier to classify normal and lesion images.

Feature Extraction

For feature extraction, we implement orthogonal convolution and calculate the orthogonal regularization loss at every convolution layer in a network. The models are optimized using cross entropy loss and orthogonal regularization loss. Versions of the models without orthogonal regularization are used as baseline feature extractors for comparison. Since training a deep CNN on ImageNet is resource intensive and takes a long time, we opt to use CIFAR-10 for pre-training our models instead of

using ImageNet to ensure the setup and comparison of the results are equivalent. The CIFAR-10 dataset is divided into a training set of 45000 images, validation set of 5000 images, and a test set of 10000 images to verify the model is sufficiently trained. To train the feature extractors, data augmentation with translation, rotation, and horizontal flip is used to augment the original training set. The model is trained using a batch size of 256, an initial learning rate of 0.001 decayed by a factor of 0.1 at epochs 80, 100, and 120, and with Adam [21] as the optimizer over 200 epochs. An orthogonal regularization penalty of $\lambda = 0.01$ without weight decay or dropout is used in the training protocol. We employ early stopping to prevent over training and select the model with the lowest validation loss as the final model to use for feature extraction.

Feature Selection

Since the extracted features have high dimensions and not all extracted features may be relevant for prediction, feature selection is used to reduce the number of irrelevant extracted features and decrease dimensionality before doing classification to boost performance. The extracted features from the deep CNNs are normalized by dividing each feature with the feature standard deviation. Then, a linear kernel SVM with L1 regularization is used for feature selection to select a subset of features. L1-norm is used for regularization because it encourages weights to be 0, which can be seen as performing some form of feature selection [54]. The SVM is optimized with squared hinge loss and L1 regularization defined as

$$\min_{w} \frac{1}{2} w + C \sum_{i=1}^{M} \max(0, 1 - y_i w^T x_i)^2 \tag{3.1}$$

[11]. We conduct a hyperparameter search over the interval [0.02, 0.10] with steps of 0.005 for the parameter C of the SVM to find the optimal L1 regularized L2-SVM for

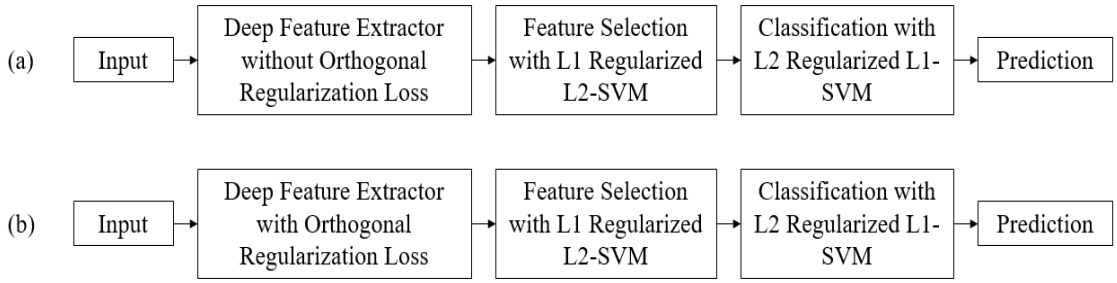


Figure 3.7: Overview of the method for feature extraction, feature selection, and classification (a) without orthogonal regularization loss and (b) with orthogonal regularization loss.

reducing the number of extracted features before doing classification.

Classification

For classification, a separate SVM from the feature selection step is used to classify images as normal or lesion. We use a linear kernel SVM and train the model with hinge loss and L2 regularization defined as

$$\min_{w} \frac{1}{2} w^T w + C \sum_{i=1}^{M} \max(0, 1 - y_i w^T x_i) \tag{3.2}$$

[4] to penalize missclassifications more heavily and handle possible outliers in the data. Similar to the feature selection step, a hyperparameter search over the interval [0.02, 0.10] with steps of 0.005 is done to find the parameter C and the optimal L2 regularized L1-SVM for classifying normal and lesion images. We measure and evaluate the performances of the classifiers using accuracy and F1-score. The performance measurements are collected using cross validation with each test set maintaining the ratio of samples between classes. The same test set is held out from training of the L1 regularized L2-SVM in the feature selection step and the classifier to not introduce any bias in the classification step. Figure 3.7 shows an overview of the feature extraction, feature selection, and classification process.

Architecture	Baseline Accuracy (%)	OCNN Accuracy (%)
DenseNet121	85.36	86.01
InceptionV3	89.25	89.31
InceptionResNetV2	90.45	90.76
ResNet50V2	82.09	83.55

Table 3.2: Test accuracy on CIFAR-10 of the four selected models trained with and without orthogonal regularization loss.

3.2.4 Experimental Results

To evaluate the effects of orthogonal regularization loss on transfer learning, we conduct experiments to evaluate the classification performance of orthogonal regularized networks compared to their baseline versions. We pre-train all networks on CIFAR-10 and use them as feature extractors to extract features from the prostate cancer dataset and the bladder cancer dataset and benchmark their classification performance under two settings: (1) without feature selection and (2) with feature selection. The performance is measured using accuracy and F1-score and the results of the baseline without orthogonal regularization loss and the version of the networks with orthogonal regularization loss are compared for settings (1) and (2). In addition, classification results of the networks pre-trained on ImageNet without orthongonal regularization are provided for comparison. All reported accuracies and F1-scores are averages obtained using 10-fold cross validation on the same test set with each fold containing a different test set. Table 3.2 shows the test accuracy on CIFAR-10 by the trained networks.

Feature Extractor	Accuracy(%)	F1-Score
Baseline DenseNet121 (C=0.02)	71.35	0.5055
OCNN DenseNet121 (C=0.02)	68.91	0.4739
ImageNet DenseNet121 (C=0.02)	76.22	0.6176
Baseline InceptionV3 (C=0.035)	62.76	0.4136
OCNN InceptionV3 (C=0.035)	59.04	0.4155
ImageNet InceptionV3 (C=0.02)	64.10	0.3668
Baseline InceptionResNetV2 (C=0.02)	60.19	0.3994
OCNN InceptionResNetV2 (C=0.02)	67.24	0.4805
ImageNet InceptionResNetV2 (C=0.07)	73.91	0.5709
Baseline ResNet50V2 (C=0.02)	61.35	0.3516
OCNN ResNet50V2 (C=0.02)	66.73	0.4750
ImageNet ResNet50V2 (C=0.02)	69.49	0.4344

Table 3.3: Prostate cancer data classification results of L2 regularized L1-SVM corresponding to different feature extractors. Results are obtained using 10-fold cross validation without feature selection. C is the parameter used for the SVM classifier.

Prostate Cancer Dataset

Table 3.3 shows the classification results on the prostate cancer dataset without feature selection for setting (1) in our experiment. Under this setting, the baseline outperforms OCNN in both metrics for DenseNet121 and InceptionV3 while the OCNN outperforms the baseline for InceptionResNetV2 and ResNet50V2. Across all architectures, the network pre-trained on ImageNet outperforms both the baseline and the OCNN. Table 3.4 shows the classification results on the prostate cancer dataset with feature selection for setting (2). For this setting, the OCNN outperforms the baseline in both metrics for DenseNet121, InceptionResNetV2, and ResNet50V2 except InceptionV3. Out of all architectures, OCNN ResNet50V2 is the only architecture

Feature Extractor	# of Features Retained	Accuracy(%)	F1-Score
Baseline DenseNet121 (C_{FS}=0.04, C_{class}=0.02)	33	88.78	0.8111
OCNN DenseNet121 (C_{FS}=0.045, C_{class}=0.065)	33	91.22	0.8631
ImageNet DenseNet121 (C_{FS}=0.095, C_{class}=0.02)	59	96.09	0.9385
Baseline InceptionV3 (C_{FS}=0.025, C_{class}=0.045)	15	78.59	0.5571
OCNN InceptionV3 (C_{FS}=0.025, C_{class}=0.02)	12	73.08	0.3438
ImageNet InceptionV3 (C_{FS}=0.065, C_{class}=0.02)	81	95.26	0.9159
Baseline InceptionResNetV2 (C_{FS}=0.04, C_{class}=0.035)	37	85.64	0.7629
OCNN InceptionResNetV2 (C_{FS}=0.045, C_{class}=0.025)	43	91.41	0.8356
ImageNet InceptionResNetV2 (C_{FS}=0.1, C_{class}=0.095)	63	96.86	0.9524
Baseline ResNet50V2 (C_{FS}=0.035, C_{class}=0.07)	21	78.53	0.5693
OCNN ResNet50V2 (C_{FS}=0.065, C_{class}=0.03)	59	94.42	0.9088
ImageNet ResNet50V2 (C_{FS}=0.06, C_{class}=0.04)	71	94.42	0.9000

Table 3.4: Prostate cancer data classification results of L2 regularized L1-SVM with feature selection using L1 regularized L2-SVM. Results are collected using 10-fold cross validation. C_{FS} denotes the parameter used in the SVM for feature selection and C_{class} denotes the parameter used in the SVM for classification.

Classifier	Accuracy (%)	F1-Score
ResNet-18 (original)	57.14	0.6625
ResNet-18 (selected)	76.19	0.8171
GLCM (original)	57.94	0.7024
GLCM (selected)	61.90	0.7273
Gabor (original)	59.52	0.7000
Gabor (selected)	65.08	0.7484

Table 3.5: Previous classification results on the prostate cancer dataset in literature using a custom feature selection method [38].

that achieves similar accuracy and F1-score to its version pre-trained on ImageNet. Looking at the number of features retained by feature selection, the number of selected features for OCNN is equal to or greater than the number of the baseline for DenseNet121, InceptionResNetV2, and ResNet50V2 except InceptionV3. In all three architectures where the number of features for OCNN are equal to or greater than the baseline, the OCNN version reports higher accuracy and F1-score. For all models, the number of selected features is higher for the network pre-trained on ImageNet than the baseline and OCNN version. Our example approach using orthogonal convolution achieves higher performance than previously recorded results in literature [38] which are outlined in Table 3.5.

Bladder Cancer Dataset

Table 3.6 shows the classification results on the bladder cancer dataset without feature selection for setting (1). Under this setting, the OCNN outperforms the baseline only for InceptionResNetV2 and ResNet50V. OCNN InceptionResNetV2 is the only architecture to outperform its version pre-trained on ImageNet. Table 3.7 shows the classification results on the bladder cancer dataset with feature selection for setting

Feature Extractor	Accuracy(%)	F1-Score
Baseline DenseNet121 (C=0.02)	67.00	0.6695
OCNN DenseNet121 (C=0.03)	62.00	0.6126
ImageNet DenseNet121 (C=0.02)	67.50	0.6663
Baseline InceptionV3 (C=0.03)	58.50	0.5916
OCNN InceptionV3 (C=0.02)	58.00	0.5639
ImageNet InceptionV3 (C=0.02)	61.50	0.5972
Baseline InceptionResNetV2 (C=0.025)	63.00	0.6215
OCNN InceptionResNetV2 (C=0.02)	74.50	0.7346
ImageNet InceptionResNetV2 (C=0.035)	68.50	0.6953
Baseline ResNet50V2 (C=0.06)	59.50	0.5830
OCNN ResNet50V2 (C=0.02)	64.00	0.6223
ImageNet ResNet50V2 (C=0.02)	65.50	0.6360

Table 3.6: Bladder cancer data classification results of L2 regularized L1-SVM corresponding to different feature extractors. Results are obtained using 10-fold cross validation without feature selection. C is the parameter used for the SVM classifier.

Feature Extractor	# of Features Retained	Accuracy(%)	F1-Score
Baseline DenseNet121 (C_{FS}=0.065, C_{class}=0.075)	75	85.00	0.8507
OCNN DenseNet121 (C_{FS}=0.03, C_{class}=0.02)	30	82.00	0.8201
ImageNet DenseNet121 (C_{FS}=0.025, C_{class}=0.055)	34	84.00	0.8352
Baseline InceptionV3 (C_{FS}=0.02, C_{class}=0.025)	12	70.50	0.7200
OCNN InceptionV3 (C_{FS}=0.02, C_{class}=0.05)	17	71.00	0.7107
ImageNet InceptionV3 (C_{FS}=0.02, C_{class}=0.02)	28	81.50	0.7963
Baseline InceptionResNetV2 (C_{FS}=0.035, C_{class}=0.025)	36	79.50	0.7994
OCNN InceptionResNetV2 (C_{FS}=0.095, C_{class}=0.055)	102	93.50	0.9329
ImageNet InceptionResNetV2 (C_{FS}=0.025, C_{class}=0.02)	36	84.00	0.8364
Baseline ResNet50V2 (C_{FS}=0.035, C_{class}=0.02)	42	73.00	0.7078
OCNN ResNet50V2 (C_{FS}=0.045, C_{class}=0.02)	77	87.00	0.8685
ImageNet ResNet50V2 (C_{FS}=0.02, C_{class}=0.025)	36	85.00	0.8517

Table 3.7: Bladder cancer data classification results of L2 regularized L1-SVM with feature selection using L1 regularized L2-SVM. Results are collected using 10-fold cross validation. C_{FS} denotes the parameter used in the SVM for feature selection and C_{class} denotes the parameter used in the SVM for classification.

(2). Similar to setting (1), the OCNN version outperforms the baseline for InceptionResNet50V2 and ResNet50V2. Moreover, both the OCNN InceptionResNetV2 and ResNet50V2 are the only architectures to outperform its ImageNet version for setting (2). Out of all the architectures, DenseNet121 is the only network to have its baseline outperform its OCNN and ImageNet version. Looking at the number of features retained, the models with a higher number of features selected have a higher accuracy and F1-score. Comparing our results to previous literature outlined in Table 3.8, our example approach using orthogonal convolution with InceptionResNetV2 is able to achieve higher performance than previously recorded results in literature.

Our findings support the claim that orthogonality convolutional neural networks learns more diverse and expressive features. The numerical results of our experiments show the generality that exist in learned features of deep CNNs, which hold some form of implicit knowledge that may be better captured with orthogonality and transferred across domains. With enhanced diversity in transferred features, subsequent feature selection to remove irrelevant and redundant features is made more difficult, i.e. it is harder to select features for removal which leads to more retained features. Despite the increase in the number of features selected which would diminish performance if the features were irrelevant and redundant, the results showed a boost in classification performance for the end classifier, demonstrating an improvement in diversity and transferability of features with orthogonal regularization which helps in improving generalization performance. In our experiments, we also attempted to use one SVM for both feature selection and classification. However, a lower performance is observed so we maintain the use of two separate SVMs for feature selection and classification in our process.

Feature Extractor	Classifier	Accuracy (%)	F1-Score
AlexNet	KNN	70.75 ± 0.90	0.7654 ± 0.0083
	NB	80.53 ± 0.29	0.8476 ± 0.0024
	SVM	78.35 ± 0.58	0.8335 ± 0.0043
	LDA	79.87 ± 0.37	0.8486 ± 0.0026
	DT	73.73 ± 1.13	0.7877 ± 0.0089
GoogLeNet	KNN	70.38 ± 0.68	0.7664 ± 0.0056
	NB	76.42 ± 0.49	0.8168 ± 0.0039
	SVM	76.61 ± 0.53	0.8221 ± 0.0043
	LDA	78.99 ± 0.36	0.8439 ± 0.0027
	DT	75.62 ± 1.36	0.8001 ± 0.0116
InceptionV3	KNN	73.54 ± 0.63	0.7856 ± 0.0053
	NB	79.88 ± 0.31	0.8445 ± 0.0025
	SVM	77.40 ± 0.42	0.8290 ± 0.0031
	LDA	79.35 ± 0.12	0.8422 ± 0.0009
	DT	72.95 ± 1.27	0.7749 ± 0.0110
ResNet50	KNN	79.56 ± 0.71	0.8317 ± 0.0059
	NB	82.48 ± 0.39	0.8654 ± 0.0030
	SVM	77.71 ± 0.51	0.8317 ± 0.0040
	LDA	78.62 ± 0.28	0.8395 ± 0.0022
	DT	80.79 ± 1.11	0.8416 ± 0.0089
XceptionNet	KNN	78.37 ± 0.49	0.8312 ± 0.0038
	NB	83.95 ± 0.35	0.8701 ± 0.0030
	SVM	83.22 ± 0.47	0.8742 ± 0.0033
	LDA	86.07 ± 0.38	0.8939 ± 0.0026
	DT	81.45 ± 0.99	0.8466 ± 0.0086

Table 3.8: Previous classification results on the bladder cancer dataset in literature using deep CNNs as feature extractors and a custom feature selection method [37].

3.3 Analysis of OCNN for Transfer Learning

The results of the experiments on the prostate and bladder cancer datasets show orthogonal regularization is more effective in increasing performance when applied to InceptionResNetV2 and ResNet50V2 compared to DenseNet121 and InceptionV3. To understand why orthogonal regularization is more effective in improving performance for those networks in a transfer learning setting, we investigate transferred feature similarities by measuring correlation between feature maps extracted by learned filters at different layers of a network. In addition to analyzing transferred feature similarities, we provide a brief analysis of the time complexity to understand how orthogonal convolution may affect training and inference time in the context of transfer learning.

3.3.1 Related Work

One method for depicting similarities between variables is to use a correlation matrix. The authors of orthogonal convolutional neural networks used this method to investigate filter similarities of OCNN. Specifically, they used guided backpropagation to generate guided backpropagation patterns at different layers, computed the correlation matrix over the patterns, and plotted a histogram of the off-diagonal elements [49]. In our analysis to understand why orthogonal regularization is more effective for improving transferrable features in some deep CNNs, we use a similar idea but instead of using guided backpropagation patterns, we directly use the extracted feature maps from the prostate and bladder cancer datasets produced at different layers. The extracted feature maps are used because they are produced by convolving filters in a convolutional layer with the input, and if the learned filters at a convolutional layer of a CNN are similar, the generated feature maps would also share more similarities.

3.3.2 Transferred Feature Analysis

In this analysis, we investigate transferred feature similarities through the use of extracted feature maps generated by different layers of OCNN DenseNet121, InceptionV3, InceptionResNetV2, and ResNet50V2 and their baseline versions on the prostate and bladder cancer datasets. A set of feature maps produced at a layer is a 4-D tensor $F \in R^{M \times W \times H \times C}$, where M is the number of output channels or feature maps, W is the width of the input, H is the height of the input, and C is the number of input channels. Each feature map is flatten to transform the feature maps to a 2-D tensor $F_{flatten} \in R^{M \times WHC}$. The similarities between each feature map can be computed by calculating the correlation matrix $corr(F_{flatten}) \in R^{M \times M}$ for $F_{flatten}$, where the positive upper or lower triangluar elements in the correlation matrix $corr(F_{flatten})$ represent pairwise correlation between each feature maps. Similar as before, kernel density estimation is used to estimate the distribution and visualize the similarities between feature maps of OCNN and baseline networks for all architectures and datasets.

We estimate the distributions of the feature similarities at different layers of a network and generate a snapshot of the change in feature similarities through the layers. Figure 3.8 shows the snapshots of the distributions at different layers of DenseNet121 and InceptionV3 and Figure 3.9 shows the snapshots of InceptionResNetV2 and ResNet50V2 on the prostate cancer dataset. In Figure 3.8, the distributions show both the OCNN and baseline curves of DenseNet121 and InceptionV3 are aligned to the left in the early layers but begin to widen and shift right as the view progresses through the middle layers and towards the end, indicating an increase in feature redundancy for both OCNN and baseline. In particular, the distribution at layer conv5_block16_2_conv of DenseNet121 shows the OCNN curve shifting fur-

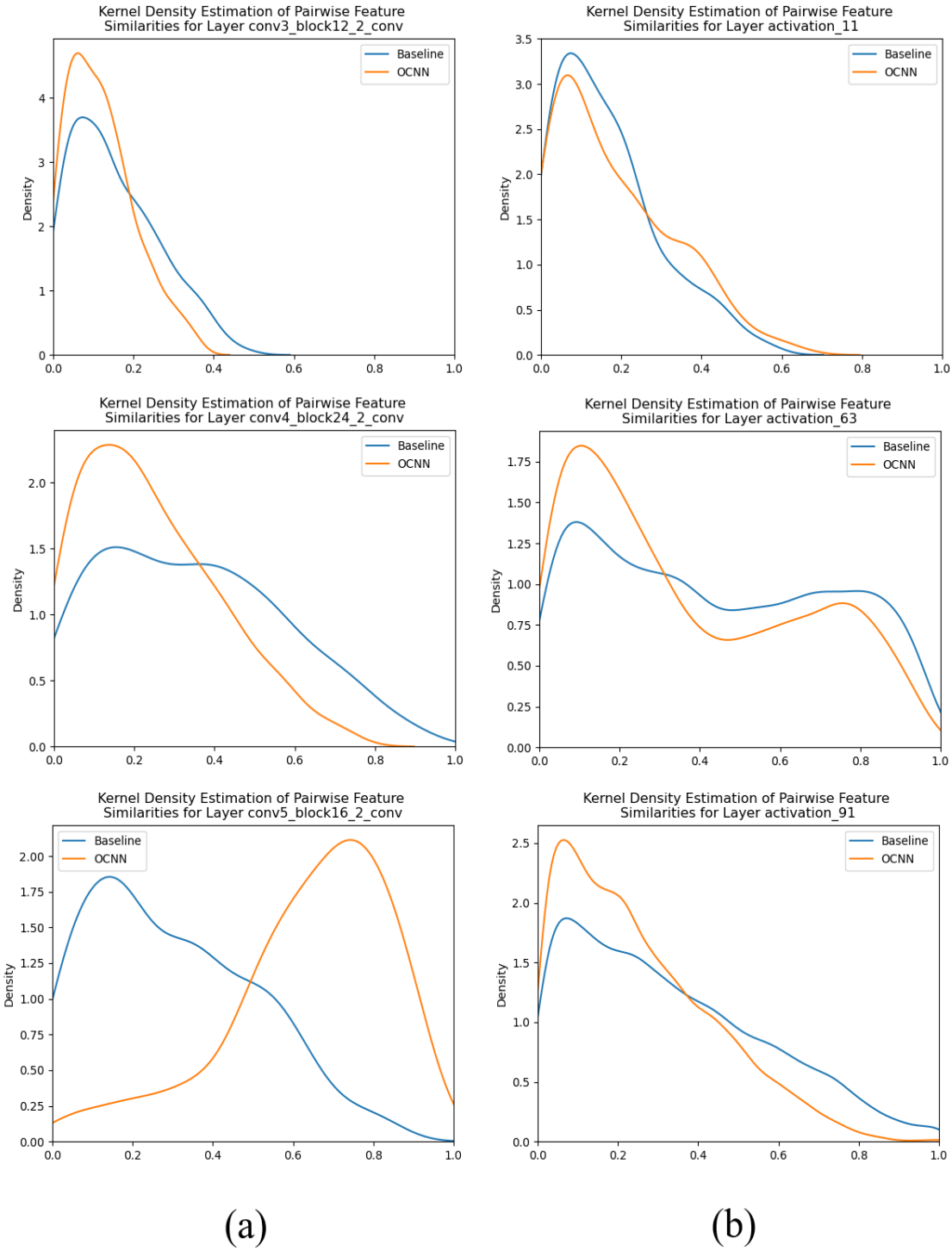


Figure 3.8: Snapshots of the distributions of pairwise feature similarities at different layers of (a) DenseNet121 and (b) InceptionV3 on the prostate cancer dataset depicting an increase in feature similarities for OCNN and baseline at some layers.

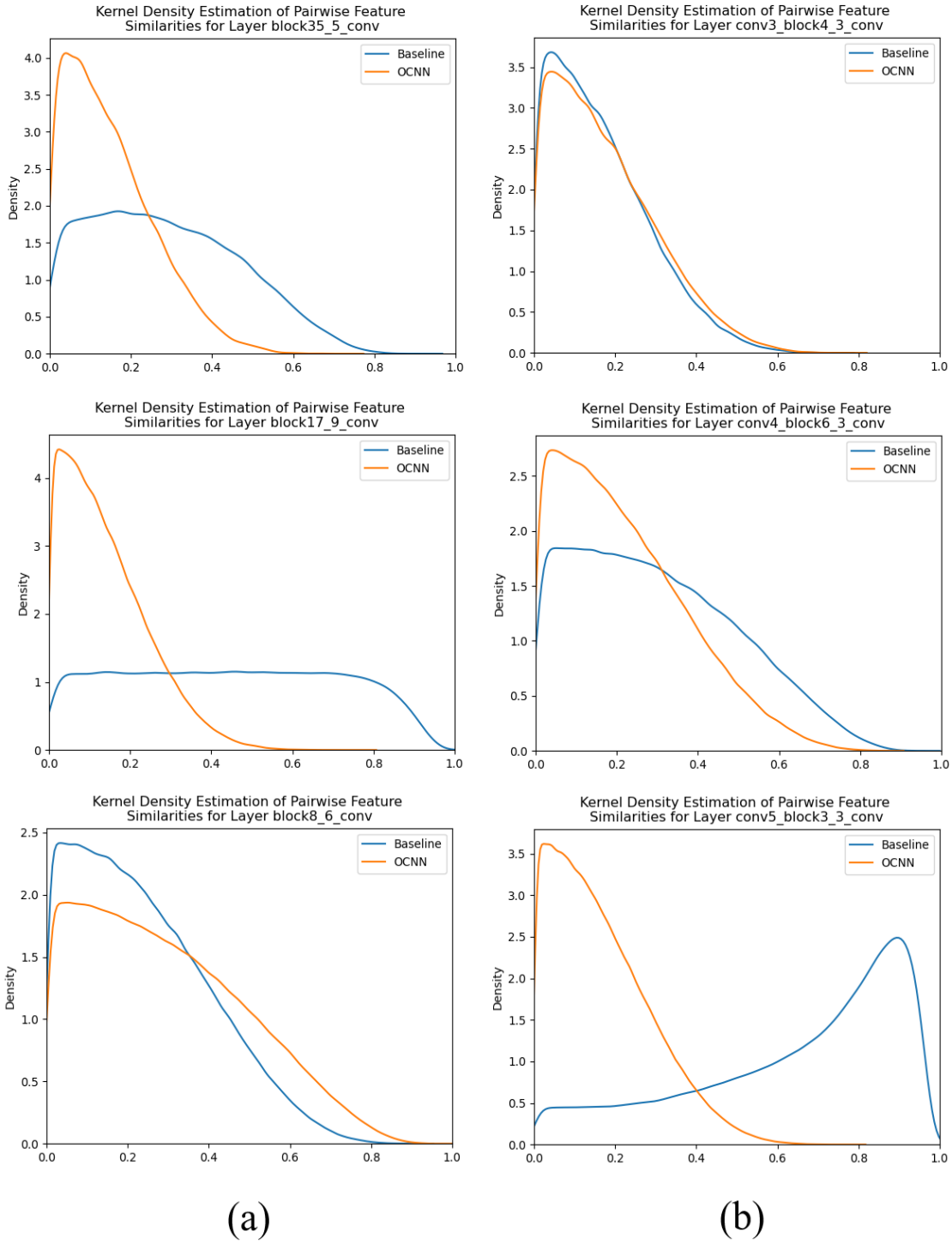


Figure 3.9: Snapshots of the distributions of pairwise feature similarities at different layers of (a) InceptionResNetV2 and (b) ResNet50V2 on the prostate cancer dataset depicting the feature maps of OCNN remaining less correlated without a significant increase in similarities compared to baseline.

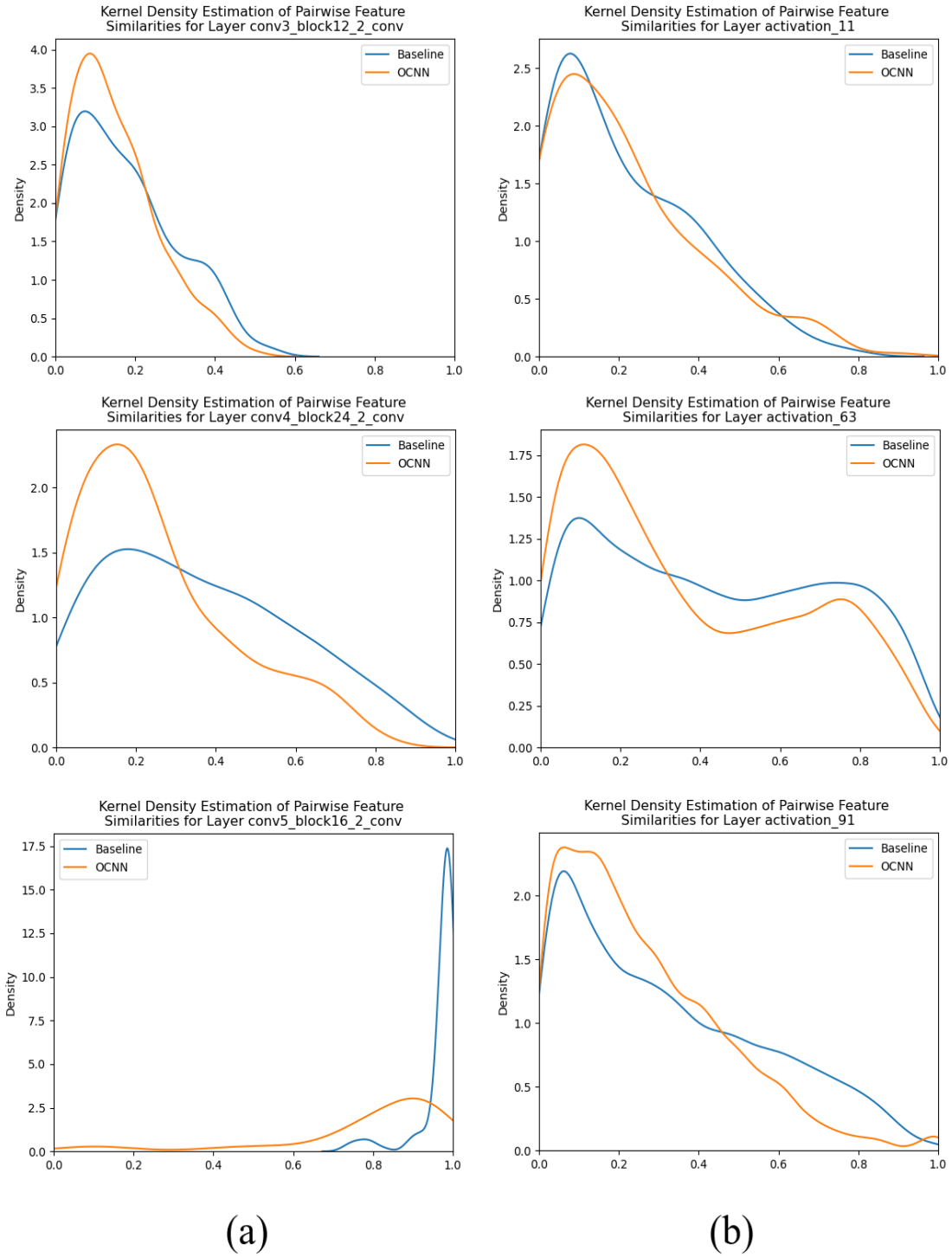


Figure 3.10: Snapshots of the distributions of pairwise feature similarities at different layers of (a) DenseNet121 and (b) InceptionV3 on the bladder cancer dataset depicting an increase in feature similarities for OCNN and baseline at some layers.

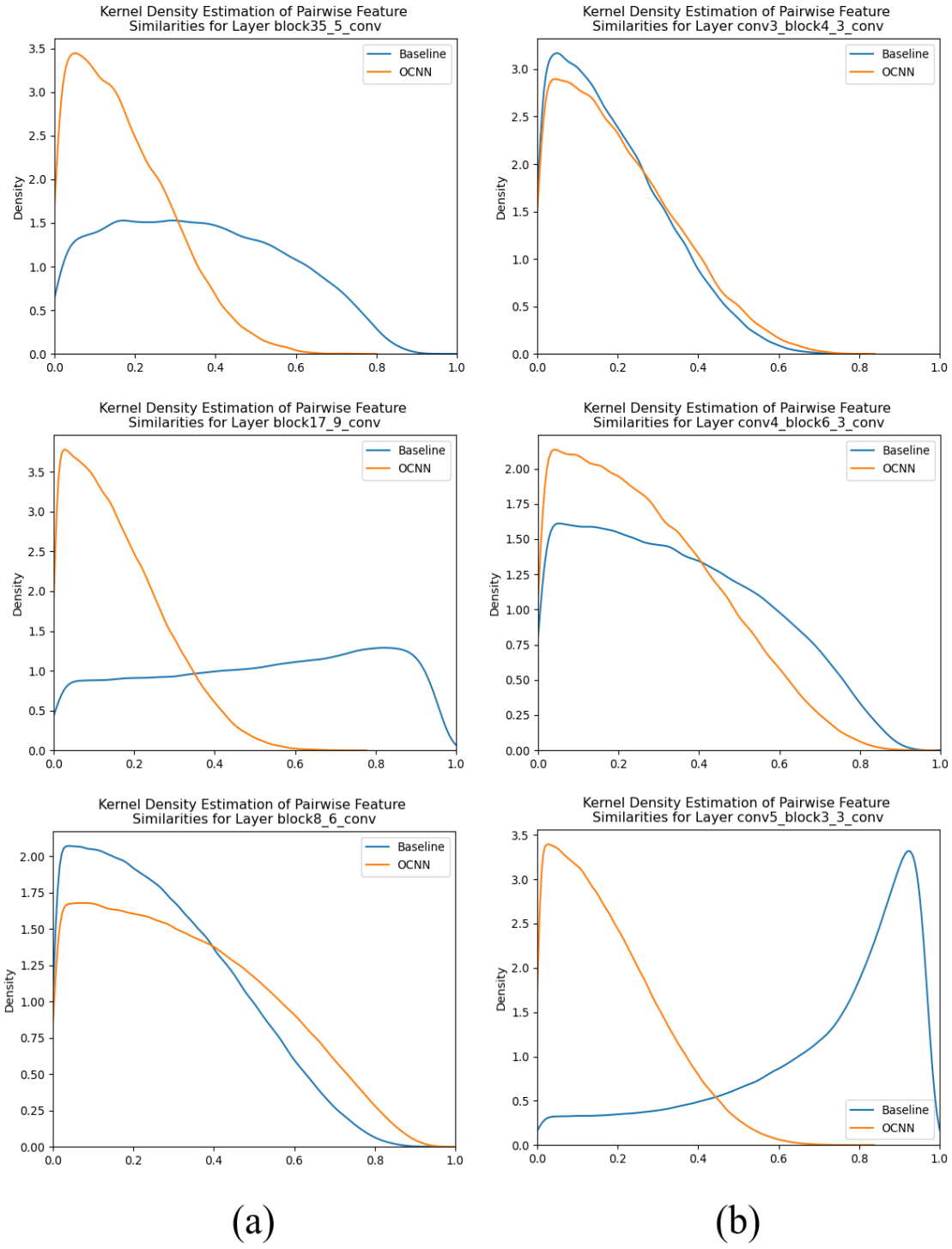


Figure 3.11: Snapshots of the distributions of pairwise feature similarities at different layers of (a) InceptionResNetV2 and (b) ResNet50V2 on the bladder cancer dataset depicting the feature maps of OCNN remaining less correlated without a significant increase in similarities compared to baseline.

ther right than the baseline curve, demonstrating higher feature similarities in the OCNN compared to baseline. In Figure 3.9, the snapshots show the OCNN curves of InceptionResNetV2 and ResNet50V2 remain left aligned throughout the layers of the network while the baseline curves begin to widen and shift right through the middle layers and towards the end, illustrating fewer redundant features for OCNN compared to the baseline model. At layer block17_9_conv and layer block8_6 of InceptionResNetV2, the distributions show the baseline curve shifting left and becoming narrower, which would suggest an improvement in similarities between feature maps across the two layers. However, feature similarities is only a measure of redundancy within the layer. Since features computed at a layer is dependent on the features from the previous layer, i.e. the features are only propagated forward, the decrease in feature similarities going forward does not imply an improvement in the transferred features.

Figure 3.10 shows the feature similarity distributions of DenseNet121 and InceptionV3 and Figure 3.11 shows the distributions of InceptionResNetV2 and ResNet50V2 on the bladder cancer dataset. Similar to Figure 3.8 and Figure 3.9, we observe the OCNN and the baseline curves widen and shift right for DenseNet121 and InceptionV3 in Figure 3.10 and the OCNN curves for InceptionResNetV2 and ResNet50V2 remain left aligned while the baseline curves widen and shift right in Figure 3.11. The curves in these snapshots across both datasets illustrate OCNN transferred features remain diverse in early layers of DenseNet121 and InceptionV3 but become increasingly similar as the features progress through the layers of the network, whereas the OCNN transferred features remain less redundant for InceptionResNetV2 and ResNet50V2. These observations in feature similarities align with previous classification results on the prostate cancer and bladder cancer datasets. Since features in the early layers of a network are known to be general while the features in the end layers are more task

specific, the results of this analysis along with the classification results in previous experiments show orthogonal regularization helps extract diverse, generic features more effectively in InceptionResNetV2 and ResNet50V2 compared to DenseNet121 and InceptionV3.

3.3.3 Time Complexity

Although no experiments were conducted to investigate the training time of the networks with and without orthogonal regularization, we provide a brief analysis of the time complexity of OCNN based on the operations involved in computing the regularization loss to understand how orthogonal convolution may affect training time. Since orthogonal convolution only applies to convolutions and not other operations in a network, the training time of an OCNN would increase based on the number of convolutions in a network. Furthermore, the orthogonal regularization loss is computed using self-convolution of the kernels in a convolutional layer; therefore, training time would only increase based on the configuration of convolutions in a layer such as the size of the kernels, stride, padding, and number of kernels irrespective of the input size. Lastly, inference time is unaffected as orthogonal convolution is implemented as a loss term in the loss function which is only applicable during training and not test time. For transfer learning, this means pre-training with orthogonal regularization does not incur much additional overhead relative to the original cost required for training and the time needed for feature extraction remains unchanged.

Chapter 4

ANALYSIS OF RESIDUAL LEARNING ON ORTHOGONAL CONVOLUTION

In the last chapter, we demonstrated orthogonal regularization loss helps deep CNNs extract more diverse, generic features that are transferable across domains using the prostate cancer and bladder cancer datasets, showing orthogonal regularization can be an effective tool in improving generalization performance for deep CNNs. In particular, it was observed orthogonal regularization is more effective in reducing feature redudancy in transferred features for InceptionResNetV2 and ResNet50V2 compared to DenseNet121 and InceptionV3. In this chapter, we analyze the impact skip connections and depth have on orthogonal regularization to provide justification on how those networks are able to be better at improving generality and preserving feature diversity.

4.1 Experiments

Among the four different architectures selected in the previous experiments in demonstrating the effectiveness of orthogonal regularization loss on transfer learning, three of the four selected models are similar in that their architecture includes the use of skip connections. Specifically, DenseNet121 used concatenation for its dense skip connections to encourage feature reuse by propagating features from preceeding layers through to subsequent layers while InceptionResNet50V2 and ResNet50V2 used summation for its residual skip connections. In the previous experiments, it was observed orthogonal regularization appears to be more effective in residual neural networks compared to densely connected neural networks and neural networks without residual connections. This fundamental difference in architecture could be a factor in

how orthogonal regularization helps improve generality and preserve feature diversity more effectively at different layers of a network.

To investigate how residual connnections may help maintain lower feature similarities in orthogonal regularization, we evaluate and compare four networks with and without residual connections and orthogonal regularization loss on CIFAR-10 in terms of classification performance and feature similarities at different convolutional layers. For the network with residual connections, we use the same ResNet50V2 model used in the previous experiments. For the plain network without residual connections, we use the same ResNet50V2 model but with its residual connections removed. Table 4.1 outlines the convolutional layers of the architecture.

4.1.1 Dataset

In this experiment, CIFAR-10 [22] is used to train all four networks. The CIFAR-10 dataset contains 32×32 natural colored images belonging to 10 different classes. The training set contains 50000 training images with 6000 images per class and the test set contains 10000 images with 1000 images per class. We split the training set into a training and validation set and allocate 10% of training data as validation data to validate and monitor the training of the two networks. A similar data augmentation strategy with translation, rotation, and horizontal flip as previous experiments is used to inflate and vary the training set to facilitate training the deep networks.

4.1.2 Training Protocol

The residual networks with and without orthogonal regularization and the plain networks with and without orthogonal regularization are trained using a similar training protocol but slightly different hyperparameters. All networks are trained using a batch size of 256, using Adam as the optimizer, and with early stopping. For the

Layer Name	Filter
conv1	$7 \times 7, 64$
conv2_x	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3_x	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 256 \\ 1 \times 1, 512 \end{bmatrix} \times 4$
conv4_x	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$
conv5_x	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$

Table 4.1: Convolutional layers and filter sizes of the ResNet50V2 and Plain50 models. The group of filters in each bracket corresponds to a block. Other layers such as pooling, batch normalization, and softmax used in the architecture are omitted in this table.

residual networks with and without orthogonal regularization, we train both networks over 200 epochs using an initial learning rate of 0.001 that decays by a factor of 0.1 at epochs 80, 100, and 120. An orthogonal regularization penalty of 0.01 is used for the residual network with orthogonal regularization. For the plain network without orthogonal regularization, we use an initial learning rate of 0.0005 and train the network over 1000 epochs without decaying the learning rate. For the plain network with orthogonal regularization, we use an initial learning rate of 0.003, orthogonal regularization penalty of 0.01, and train the network over 1800 epochs.

Model	Accuracy (%)
Baseline ResNet50V2	81.73
OCNN ResNet50V2	84.81
Baseline Plain50	64.29
OCNN Plain50	80.30

Table 4.2: Test accuracy on CIFAR-10 of ResNet50V2 and Plain50 with and without orthogonal regularization.

4.2 Results

Figure 4.2 shows the accuracy on CIFAR-10 test set by the trained ResNet50V2 and Plain50 with and without orthogonal regularization. Without residual connections, the plain network obtained the worst test accuracy on CIFAR-10 out of the four models in the experiment followed by the plain network with orthogonal regularization, ResNet50V2 without orthogonal regularization, and ResNet50V2 with orthogonal regularization. For the plain network, using orthogonal regularization provides a increase in test accuracy of 16%. These results align with expectations as residual connections are known to help with training stability and facilitate optimization which addresses accuracy degradation issues due to depth while orthogonal regularization enhances feature diversity and expressiveness, both of which help improve network performance.

Figure 4.1 shows the distributions of the feature similarities at the convolutional layers of (a) block 1 and (b) block 2 in the residual and plain networks. In Figure 4.1 (a), the curves of the convolutional layers in the first block before the first residual connection show ResNet50V2 and Plain50 with orthogonal regularization are similarly left aligned and narrower compared to their counterparts without orthogonal regularization. In Figure 4.1 (b), the distributions of the convolutional layers in the

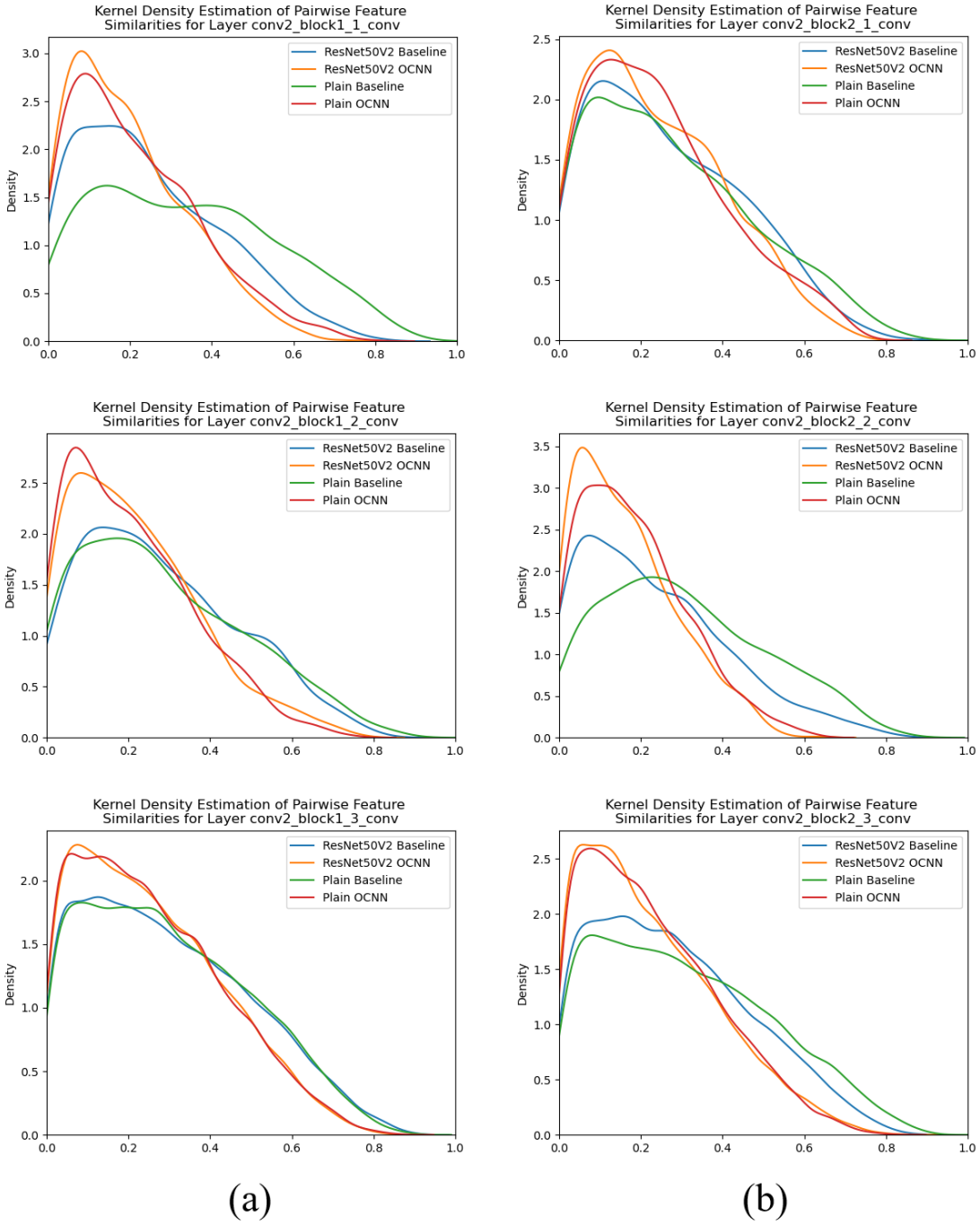


Figure 4.1: Kernel density estimated distributions of feature similarities at convolution layers of (a) block 1 and (b) block 2 of ResNet50V2 and Plain50 with and without orthogonal regularization.

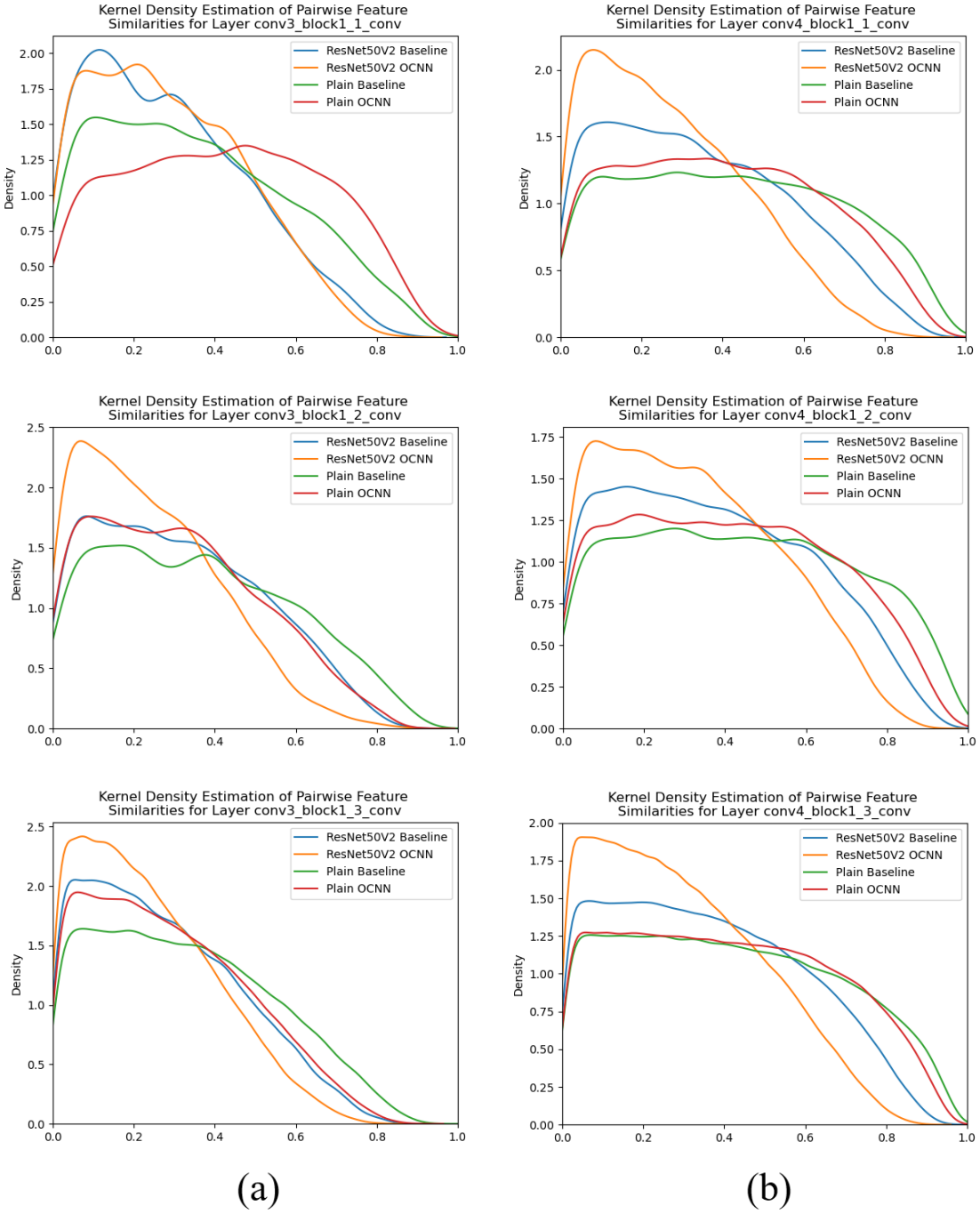


Figure 4.2: Kernel density estimated distributions of feature similarities at convolution layers of (a) block 4 and (b) block 8 of ResNet50V2 and Plain50 with and without orthogonal regularization.

second block after the first residual connection depict the curves of the models with orthogonal regularization remaining left aligned and narrower compared to the curves of the models without orthogonal regularization. These results demonstrate the features in both the residual and plain networks are similarly diverse in the early layers of the network and residual connections do not provide any significant improvements to feature similarities.

Figure 4.2 shows the distributions of the feature similarities at the convolutional layers of (a) block 4 and (b) block 8 in the residual and plain networks. In Figure 4.2 (a), the curves of Plain50 OCNN at block 4 flatten and shift right while the curves of ResNet50V2 OCNN remain narrower compared to the curves of the other models, showing features of Plain50 OCNN become more similarities at block 4 compared to ResNet50V2. Figure 4.2 (b) illustrates similar findings. The curves of ResNet50V2 OCNN remain the most left aligned and narrow followed ResNet50V2 Baseline, Plain50 OCNN, and Plain50 Baseline. These results demonstrate features of ResNet50V2 OCNN remain more diverse compared to all other models with increasing depth while the features of Plain50 OCNN become more similar and worse than ResNet50V2 Baseline, which the feature similarities at earlier layers showed the contrary. Our findings show residual connections help alleviate negative effects of depth on orthogonal regularization and the choice of architecture; in particular, the use of residual connections can make orthogonal regularization more effective.

Chapter 5

CONCLUSIONS

Real-world supervised learning problems such as medical image classification are often under less than ideal conditions where there is a lack of labeled data for training and the distributions of the source and target data domain may be different. These ill conditions pose significant challenges for deep convolutional neural networks to work as they may become overparameterized due to insufficient training data and is incapable of generalizing to unseen samples in the target domain. One solution to this problem is to use transfer learning to apply learned knowledge from one domain to a new domain and avoid the effort required to train a network from scratch, which relies on features from the source data being transferable to the target data.

This thesis explores the use of orthogonal convolutional neural networks as feature extractors to understand if and how orthogonal convolution may improve the transferability of features for transfer learning. The results of this thesis suggest orthogonal regularization can enhance the extraction of more diverse, expressive, and generic features that are transferable across domains and improve generalization performance of the target task. To understand the potential effectiveness of orthogonal convolution on certain architectures, we demonstrate orthogonal regularization is more effective in residual based neural networks and show residual connections appear to help orthogonal convolution better preserve feature diversity across layers in a network and alleviate the increase in feature similarities caused by depth. The results of this work demonstrates the use of residual learning is an important factor in effectively utilizing orthogonal regularization for deep convolutional neural networks.

5.1 Future Directions

This thesis only considers the application of orthogonal regularization in pre-training deep networks for feature extraction and it remains an open question how orthogonal convolution affects network performance when finetuning parameters of a network for transfer learning. Moreover, it remains unclear how other types of skip connections besides residual connections affect diversity and transferability of features enhanced by orthogonal regularization. The following are several future research directions that may provide a more comprehensive understanding of orthogonal regularization for transfer learning.

Effects of Orthogonal Convolution on Finetuning

An alternative approach to using pre-trained networks as feature extractors for transfer learning is to finetune parameters of a pre-trained network. In finetuning, parameters at the end layers of a network, which correspond to more task specific features, can be finetuned with a small learning rate to adjust to the target task while all other parameters are frozen. Since only features at the end layers are modified and the benefits provided by orthogonal regularization have been observed to be lessen at higher depth, an interesting research direction would be to study the effects of orthogonal convolution on feature diversity, transferability, and generalization performance in finetuning and determine if there are any improvements compared to feature extraction.

Effects of Dense Connections on Orthogonal Convolution

Besides residual connections, dense connections using concatentation is another example of skip connections that are commonly used in network architectures. Unlike

residual connections which have been hypothesized to help improve information and gradient flow, dense connections help encourage feature reuse and reduce learning of redundant feature maps [17]. It is unknown whether dense connections can help better preserve feature diversity through increasing depth like residual connections for orthogonal convolution.

Applications of Orthogonal Convolution Beyond Image Classification

Applications of orthogonal convolution can be extended to other architectures such as autoencoders and other tasks beyond image classification such as image segmentation. For instance, autoencoders are a type of networks that aim to encode an input into a latent representation and decode the encoded data to reconstruct the input. One future research direction would be to study whether orthogonal convolution can improve autoencoders in encoding a more efficient feature representation that allows for more accurate reconstruction of the input for tasks such as image denoising. Another research direction would be to study the effects of orthogonal convolution on encoder-decoder architectures such as U-Net [35] used for image segmentation.

REFERENCES

[1] Bansal, N., X. Chen and Z. Wang, "Can we gain more from orthogonality regularizations in training deep cnns?", URL https://arxiv.org/abs/1810.09102 (2018).

[2] Bjorck, J., C. Gomes, B. Selman and K. Q. Weinberger, "Understanding batch normalization", URL https://arxiv.org/abs/1806.02375 (2018).

[3] Breiman, L., "Random forests", Machine learning **45**, 1, 5–32 (2001).

[4] Chang, C.-C. and C.-J. Lin, "LIBSVM: A library for support vector machines", ACM Transactions on Intelligent Systems and Technology **2**, 27:1–27:27, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm (2011).

[5] Cheung, W. and G. Hamarneh, "N-sift: N-dimensional scale invariant feature transform for matching medical images", in "2007 4th IEEE International Symposium on Biomedical Imaging: From Nano to Macro", pp. 720–723 (2007).

[6] Criminisi, A. and J. Shotton, *Decision Forests for Computer Vision and Medical Image Analysis* (Springer Publishing Company, Incorporated, 2013).

[7] Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database", in "2009 IEEE Conference on Computer Vision and Pattern Recognition", pp. 248–255 (2009).

[8] DeVries, T. and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout", URL https://arxiv.org/abs/1708.04552 (2017).

[9] Donahue, J., Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition", URL https://arxiv.org/abs/1310.1531 (2013).

[10] El-Naqa, I., Y. Yang, M. Wernick, N. Galatsanos and R. Nishikawa, "A support vector machine approach for detection of microcalcifications", IEEE Transactions on Medical Imaging **21**, 12, 1552–1563 (2002).

[11] Fan, R.-E., K.-W. Chang, C.-J. Hsieh, X.-R. Wang and C.-J. Lin, "Liblinear: A library for large linear classification", the Journal of machine Learning research **9**, 1871–1874 (2008).

[12] Glorot, X. and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks", in "Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics", pp. 249–256 (JMLR Workshop and Conference Proceedings, 2010), URL https://proceedings.mlr.press/v9/glorot10a.html.

[13] Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, "Generative adversarial nets", in "Advances in Neural Information Processing Systems", edited by Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence and K. Weinberger, vol. 27 (Curran Associates, Inc., 2014).

[14] He, K., X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition", URL http://arxiv.org/abs/1512.03385, arXiv:1512.03385 [cs] (2015).

[15] He, K., X. Zhang, S. Ren and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification", URL http://arxiv.org/abs/1502.01852, arXiv:1502.01852 [cs] (2015).

[16] Hu, W., L. Xiao and J. Pennington, "Provable Benefit of Orthogonal Initialization in Optimizing Deep Linear Networks", URL http://arxiv.org/abs/2001.05992, arXiv:2001.05992 [cs, math, stat] (2020).

[17] Huang, G., Z. Liu, L. Van Der Maaten and K. Q. Weinberger, "Densely Connected Convolutional Networks", in "2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)", pp. 2261–2269 (2017), iSSN: 1063-6919.

[18] Ioffe, S. and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", URL http://arxiv.org/abs/1502.03167, arXiv:1502.03167 [cs] (2015).

[19] Karras, T., M. Aittala, J. Hellsten, S. Laine, J. Lehtinen and T. Aila, "Training generative adversarial networks with limited data", URL https://arxiv.org/abs/2006.06676 (2020).

[20] Khedher, L., J. Ramírez, J. Górriz, A. Brahim and F. Segovia, "Early diagnosis of alzheimer's disease based on partial least squares, principal component analysis and support vector machine using segmented mri images", Neurocomputing **151**, 139–150, URL https://www.sciencedirect.com/science/article/pii/S0925231214013137 (2015).

[21] Kingma, D. P. and J. Ba, "Adam: A method for stochastic optimization", URL https://arxiv.org/abs/1412.6980 (2014).

[22] Krizhevsky, A. and G. Hinton, "Learning multiple layers of features from tiny images", Tech. Rep. 0, University of Toronto, Toronto, Ontario (2009).

[23] Krizhevsky, A., I. Sutskever and G. E. Hinton, "ImageNet classification with deep convolutional neural networks", in "Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1", NIPS'12, pp. 1097–1105 (Curran Associates Inc., Red Hook, NY, USA, 2012).

[24] Kumar, D., A. Wong and D. A. Clausi, "Lung nodule classification using deep features in ct images", in "2015 12th Conference on Computer and Robot Vision", pp. 133–138 (2015).

[25] Lam, C., D. Yi, M. Guo and T. Lindsey, "Automated detection of diabetic retinopathy using deep learning", AMIA summits on translational science proceedings **2018**, 147 (2018).

[26] Li, Z. and D. Hoiem, "Learning without forgetting", URL `https://arxiv.org/abs/1606.09282` (2016).

[27] Lin, M., Q. Chen and S. Yan, "Network in network", URL `https://arxiv.org/abs/1312.4400` (2013).

[28] Litjens, G., T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. W. M. van der Laak, B. van Ginneken and C. I. Sánchez, "A Survey on Deep Learning in Medical Image Analysis", Medical Image Analysis **42**, 60–88, URL `http://arxiv.org/abs/1702.05747`, arXiv:1702.05747 [cs] (2017).

[29] Luo, P., X. Wang, W. Shao and Z. Peng, "Towards understanding regularization in batch normalization", URL `https://arxiv.org/abs/1809.00846` (2018).

[30] Netzer, Y., T. Wang, A. Coates, A. Bissacco, B. Wu and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning", in "NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011", (2011).

[31] Ng, A. Y., "Feature selection, L1 vs. L2 regularization, and rotational invariance", in "Proceedings of the twenty-first international conference on Machine learning", ICML '04, p. 78 (Association for Computing Machinery, New York, NY, USA, 2004), URL `https://doi.org/10.1145/1015330.1015435`.

[32] Pan, S. J. and Q. Yang, "A survey on transfer learning", IEEE Transactions on knowledge and data engineering **22**, 10, 1345–1359 (2010).

[33] Perez, L. and J. Wang, "The effectiveness of data augmentation in image classification using deep learning", URL `https://arxiv.org/abs/1712.04621` (2017).

[34] Razavian, A. S., H. Azizpour, J. Sullivan and S. Carlsson, "CNN Features Off-the-Shelf: An Astounding Baseline for Recognition", in "2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops", pp. 512–519 (IEEE, Columbus, OH, USA, 2014), URL `https://ieeexplore.ieee.org/document/6910029`.

[35] Ronneberger, O., P. Fischer and T. Brox, "U-net: Convolutional networks for biomedical image segmentation", (2015).

[36] Sandfort, V., K. Yan, P. J. Pickhardt and R. M. Summers, "Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks", Scientific Reports **9**, 1, 16884, URL `https://www.nature.com/articles/s41598-019-52737-x` (2019).

[37] Sarkar, S., K. Min, W. Ikram, R. W. Tatton, I. B. Riaz, T. Wu, A. C. Silva, A. H. Bryce, C. Moore, T. H. Ho, G. Sonpavde, H. M. Abdul-Muhsin, D. B. Blumenthal and P. Singh, "Performing Automatic Identification and Staging of

Urothelial Carcinoma in Bladder Cancer Patients Using a Hybrid Deep-Machine Learning Approach", Unpublished Manuscript (2022).

[38] Sarkar, S., T. Wu, M. Harwood and A. C. Silva, "A Hybrid Artificial Intelligence Framework for Identifying Lymph Node Metastasis of Prostate Cancer Patients", Unpublished Manuscript (2022).

[39] Saxe, A. M., J. L. McClelland and S. Ganguli, "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks", URL `https://arxiv.org/abs/1312.6120` (2013).

[40] Shin, H.-C., H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura and R. M. Summers, "Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning", IEEE transactions on medical imaging **35**, 5, 1285–1298 (2016).

[41] Shorten, C. and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning", Journal of Big Data **6**, 1, 60 (2019).

[42] Silverman, B. W., *Density estimation for statistics and data analysis* (Routledge, 2018).

[43] Simonyan, K. and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", URL `https://arxiv.org/abs/1409.1556` (2014).

[44] Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting", Journal of Machine Learning Research **15**, 56, 1929–1958, URL `http://jmlr.org/papers/v15/srivastava14a.html` (2014).

[45] Szegedy, C., S. Ioffe, V. Vanhoucke and A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning", URL `http://arxiv.org/abs/1602.07261`, arXiv:1602.07261 [cs] (2016).

[46] Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, "Going Deeper with Convolutions", URL `http://arxiv.org/abs/1409.4842`, arXiv:1409.4842 [cs] (2014).

[47] Szegedy, C., V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the inception architecture for computer vision", URL `https://arxiv.org/abs/1512.00567` (2015).

[48] Tajbakhsh, N., J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway and J. Liang, "Convolutional neural networks for medical image analysis: Full training or fine tuning?", IEEE Transactions on Medical Imaging **35**, 5, 1299–1312 (2016).

[49] Wang, J., Y. Chen, R. Chakraborty and S. X. Yu, "Orthogonal Convolutional Neural Networks", URL `http://arxiv.org/abs/1911.12207`, arXiv:1911.12207 [cs] (2020).

[50] Yosinski, J., J. Clune, Y. Bengio and H. Lipson, "How transferable are features in deep neural networks?", URL http://arxiv.org/abs/1411.1792, arXiv:1411.1792 [cs] (2014).

[51] Yun, S., D. Han, S. J. Oh, S. Chun, J. Choe and Y. Yoo, "Cutmix: Regularization strategy to train strong classifiers with localizable features", URL https://arxiv.org/abs/1905.04899 (2019).

[52] Zhang, H., M. Cisse, Y. N. Dauphin and D. Lopez-Paz, "mixup: Beyond empirical risk minimization", URL https://arxiv.org/abs/1710.09412 (2017).

[53] Zhang, L., Z. Deng, K. Kawaguchi, A. Ghorbani and J. Zou, "How does mixup help with robustness and generalization?", URL https://arxiv.org/abs/2010.04819 (2020).

[54] Zhu, J., S. Rosset, R. Tibshirani and T. Hastie, "1-norm support vector machines", Advances in neural information processing systems **16** (2003).