LAKE FOREST COLLEGE

Senior Thesis

Dog Breed Classification Using Convolutional Neural Networks: Interpreted Through a Lockean Perspective

by

Xavier Higa

April 7, 2019

The report of the investigation undertaken as a Senior Thesis, to carry two courses of credit in the Department of Mathematics & Computer Science and the Department of Philosophy.

Davis Schneiderman Krebs Provost and Dean of the Faculty	Sugata Banerji, Co-Chairperson
	Chad McCracken, Co-Chairperson
	Jennifer Jhun
	Matthew R. Kelley

Abstract

The problem of fine-grained classification is one in which traditionally humans have fared better than computers. Only recently, with the advent of complex Machine Learning techniques, we have seen systems that can compete with or beat humans at this problem. In this work, we trained two Convolutional Neural Networks (CNNs) on the Stanford Dogs dataset and made them recognize dog breeds. We also analyzed the response maps of the CNNs with the aim of determining which breed-specific features the networks had learned in order to classify the images. Upon obtaining these features, we attempted to gain an insight into them for comparison with the human understanding of breeds under a Lockean interpretation.

Acknowledgments

A special thanks to all of my committee members, for not only agreeing to take part in my senior thesis as committee members, but also the time and assistance they have offered. Especially, Dr. Sugata Banerji and Dr. Chad McCracken, who have provided the knowledge and guidance which made this research possible. It is my hope that this work provides proof of your efforts.

I would also like to thank my family, particularly my parents Ken and Crystle Higa. You have supported me throughout my various endeavors, and my research would not have been possible if it weren't for you.

Table of Contents

List of Figures

1 Introduction

Object recognition and classification using convolutional neural networks (CNN)s has been the topic of many research projects in recent years. The popularity of CNNs can be attributed to the fact that they are capable of recognizing and classifying a wide variety of objects and images. A lot of research has gone in to the training of CNNs on a variety of datasets, most focusing on the optimization of a particular network's performance. Such research has extended into what is known as fine-grained classification tasks. In these problems, the dataset used contains classes which can have minor, or few, differences between them. Additionally, such datasets may also contain classes which vary in a number of different ways, within themselves. Fine-grained classification can be useful in a number of ways, such as in the study of botany and zoology. In the current age, when nearly everyone carries a high-resolution camera in their pockets, such systems could be of great use in recognizing dog or other animal breeds, and even plant species from photos taken in real time and uploaded as queries.

In this research, we will be training two CNNs on the Stanford Dogs Dataset, using transfer learning (or fine-tuning). However, the focus of this research, is to analyze the features used by the two trained CNNs. These features will be used to determine what the CNNs have found to be important when classifying the images. It will then be determined if these features are meaningful, or even humanly understandable. This will be crucial in comparing how the CNNs classify these dog breeds, and the way in which humans perform the same task.

The features extracted, the analysis of them, along with the networks themselves, will be interpreted through a Lockean understanding of concepts. Looking at these networks, from the viewpoint of Locke's theory of ideas and words, we will propose what can be said about this work. We intend to show how well this work fits within Locke's theory of ideas and words.

What follows, is an outline of this thesis. Previous research on CNNs and dog-breed classification, conducted by other researchers, will be discussed in Section 2. We will then discuss artificial neural networks (ANN)s, along with CNNs, in detail in Section 3. The proposed method for the experiment will be detailed in Section 4. The experiments will be discussed in Section 5. The discussion of results, and that of the findings, will follow in Section 6. We will provide a philosophical understanding of the experiment's results in Section 6.4. Finally, the Conclusions section will detail the work done in this thesis, as well as any future directions for this research.

2 Related Work

A substantial amount of research has gone into fine-grained classification problems the majority of which has focused on increasing the performance or accuracy of the classification by various approaches. Of these works, some have approached the problem similar to [3], where image processing is employed at the beginning of the process. [3] used the provided annotations of the Stanford Dogs dataset, which had locations of bounding boxes that outlined the useful information of each image. Specifically, this meant that the information pertaining to the dog could be found inside these boxes. Using this, the images were all cropped to the bounding boxes, and [3] removed any resulting images smaller than 256X256. Once this image preprocessing was completed, [3] used LeNet and GoogLeNet architectures. However, [3] noted that transfer learning was not used.

[10] researched fine-grained classification of dog breeds using part localization. The method of [10] employed a number of computer vision topics, focusing on the use of dog faces to improve accuracy in classifying the various breeds.

[10] found improved performance through the use of their method, however, this method requires a good number of steps and [16] aimed to reduce the

complexity of this sort of approach. [16] used the Grassmann manifold to represent the geometry of dog breeds. Specifically, [16] focused on the geometry of dog faces and found that their method performed on par with other, more complex, approaches.

The main take-away here, is that in all these works, the goal was to improve performance on a fine-grained classification problem. This is an important issue, as discussed in [3], [10], and [16] as it poses a number of problems. Each related work discussed here approaches the problem in a slightly different way, however the main idea behind their approaches are all the same. Each work focused on reducing the amount of information analyzed, hoping to reduce the amount to just what is important. Focusing on just what is hoped is the useful information in classifying dog breeds. We have elected to go a different direction with our research. We have decided not to look into a way for optimizing the networks used, instead, we will be investigating what networks find important in classifying the breeds in the Stanford Dogs dataset.

3 Background

3.1 Perceptrons

A perceptron is based on the neuron [13], though it should be noted that a perceptron is based on a very basic depiction of a neuron. To function, a perceptron takes in inputs, multiplies them by weights they have associated with each input, and adds to a bias [13]. The results of these are then summed and put into an activation function. A very common activation function used by a number of networks is the sigmoid function as seen in 1 [15]. This provides the output of a perceptron. An example of a perceptron can be seen in figure 1.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{1}$$

Training a network with a dataset that contains the correct labels for the

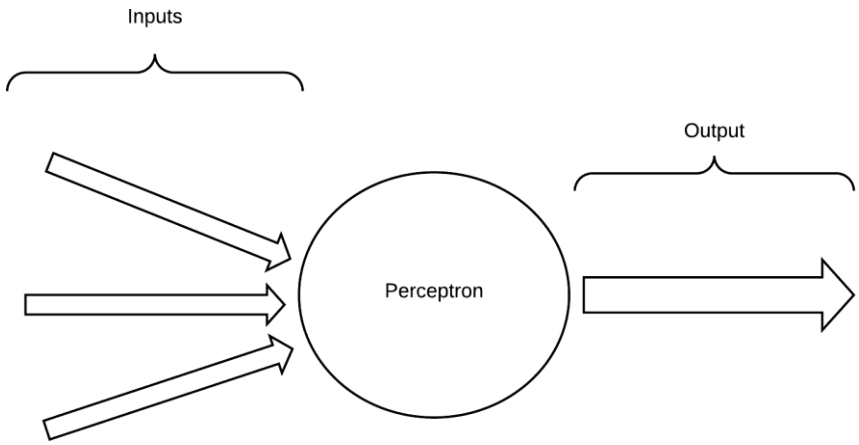

Fig. 1: Example of a perceptron.

images is called supervised learning. In supervised learning, a perceptron learns by comparing its output to the correct label. The error is calculated by finding the difference between the two. A perceptron attempts to reduce its error by going back to and updating the weights and biases that most likely caused the error. Barring data that is identical, the perceptron does this for the varied data that it encounters in training, attempting to find weights and biases that allow it to generalize for the entire dataset. From this generalized pattern, a perceptron can hopefully make correct predictions on new data. However, a single perceptron is only capable of achieving so much.

3.2 Artificial Neural Networks

ANNs were created to help deal with larger amounts of input, and different kinds of data. ANNs have multiple perceptrons lined up together, in what is called

a layer. The first layer of an ANN is called the input layer. Not much is usually done in the input layer, as it mostly just takes in the input so that the next layers can use it. A hidden layer of an ANN is a layer of perceptrons that is anywhere in between the input and output layers. These are called hidden layers, because you never really see the immediate output they produce. The output layer is the final layer of an ANN. This layer provides the output of the entire network. The output of this layer is what the network compares to the correct answers when training [5]. For a while, research of ANNs was stalled. There were

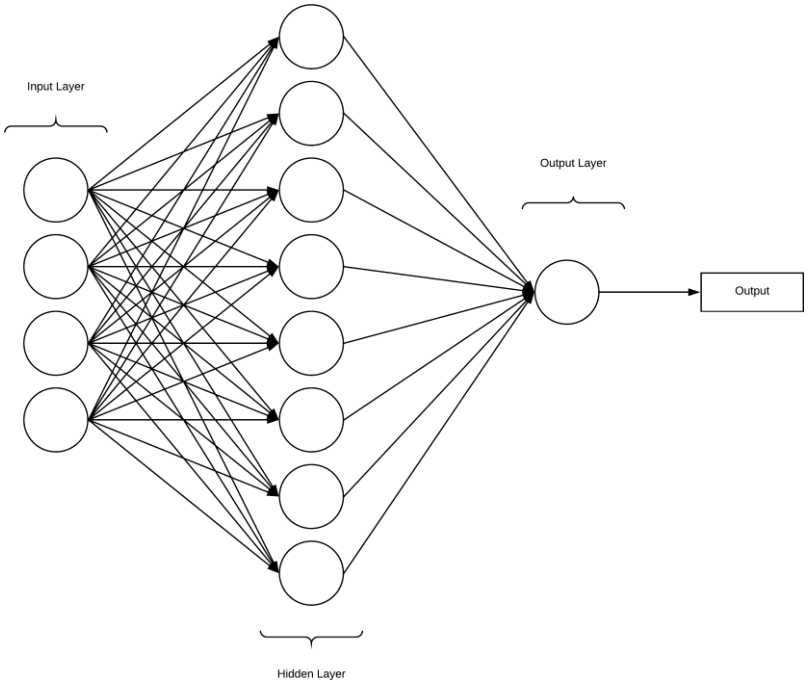

Fig. 2: Example of a single-layer ANN

a number of reasons, but one noticeable reason was that the weights and biases could not be updated for networks that were bigger than single-layer networks.

An example of a single-layer network can be seen in figure 2. Single-layer refers to the single hidden layer of the network. The issue with this, was that these single- layer networks were incapable of accomplishing classification tasks that weren't linear. However, as mathematics, as well as computers, improved, a method called back-propagation was developed to train networks that had more than one hidden layer. Thus, making it possible for ANNs to work on classification tasks that did not fit into a linear model.

The idea of training ANNs is fairly similar to the training of a perceptron. The network attempts to classify an input (i), finds its error by comparing its output (y_i) to the expected output (y_h). Then, going back, the network updates its weights (w) and biases (b) using functions seen in 2 and 3, respectively. The way in which this is carried out, is through back propagation [7].

$$w_{new} = w_{old} + (y_h - y_i) \times x_i \tag{2}$$

$$b_{new} = b_{old} + (y_h - y_i) \tag{3}$$

3.3 Convolutional Neural Networks

CNNs are based on a number of concepts within image processing and computer vision. As the name suggests, convolution is a major component to these neural networks. We will now discuss the various layers of convolutional neural networks, and what they are used for.

Convolution Layer The convolution (CONV) layer takes matrices, or images, which are just matrices of integer values, as input [12]. Then a feature matrix, convolves the input image, creating a feature map. There are multiple feature matrices, or detectors, in each CONV layer [12]. Additionally, there are usually

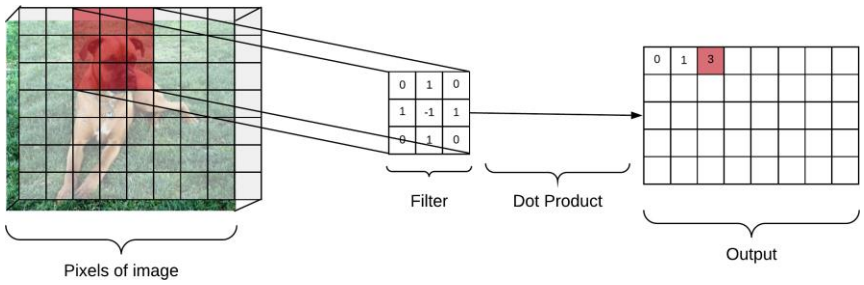

Fig. 3: CONV Example.

multiple CONV layers in a single network. An example of a simplified convolution (showing a single filter) can be seen in figure 3.

Rectified Linear Unit Layer The rectified linear unit (ReLU) layer's main purpose is to remove any negative values that result from the CONV layer's feature map. This just takes the CONV layer's output as its input, and performs the rectifier function seen in 4, producing its output [1].

$$f(x) = \begin{cases} 0, & \text{for } x < 0 \\ x, & \text{for } x \geq 0 \end{cases} \tag{4}$$

Max Pooling Layer Max pooling (MaxPool) layers traverse the input matrices, which usually come from a ReLU layer. While traversing the matrices, it looks at a portion of each matrix, and creates a smaller matrix with only the maximum value for each section [12]. This is done to help prevent over-fitting, as not all values can be memorized past the MaxPool layer [12]. Additionally, MaxPool helps reduce the computational complexity of a network by reducing the size of matrices. Though information is lost in the process, the most important

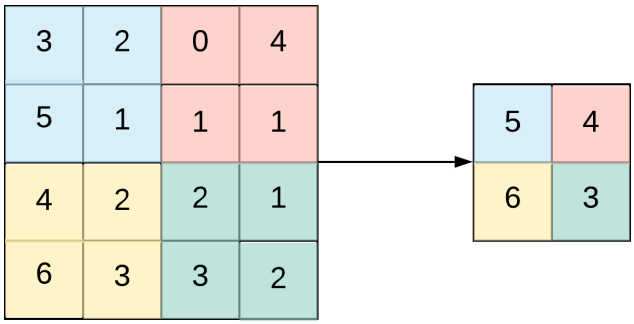

Fig. 4: Example of a 2x2 MaxPool.

information is passed through a MaxPool layer [12]. A representation of what MaxPool looks like can be seen in figure 4.

Fully Connected Layer A fully connected (FC) layer is, essentially, the same setup as an ANN [8]. This is usually included at the end of a CNN [8]. Before this can work, the result of a ReLU layer is usually put through a MaxPool layer to produce a vector. The resulting vector is then used with a fully connected layer, which usually includes dropout layer, and a softmax layer for a classification task that has more than two classes [8].

Dropout Layer The dropout layer disables half of the perceptrons from the previous FC layer [9]. This not only helps prevent over-fitting, but also helps to make the network more robust [9]. As the disabled perceptrons are chosen at random, no single perceptron can be relied upon every time [9]. The network is also less likely to adjust deeper layers for the errors of previous ones [9]. Dropout is chosen at random every time, and effectively removes the chosen perceptrons from consideration in the following layer [9]. An example of dropout can be seen in figure 5.

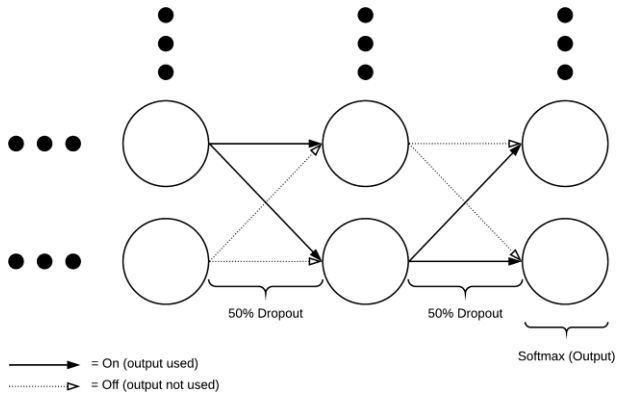

Fig. 5: Dropout Example.

Softmax Layer The softmax layer is used for classification tasks that have more than two categories, or classes [1]. This layer allows the network to provide a percentage, or confidence for each class. In a classification task, the output that has the highest confidence, is the output chosen, and corresponds to a particular class. We used the softmax layer in our experiments, since our dataset contains 120 classes. Softmax is the normalized exponential function, and converts a vector from their current real values, to real values between 0 and 1, with a sum of 1 [1]. The equation can be seen in 5, with x being the input to the softmax layer and k being the number of inputs [1]. This is what gives the percentage, or confidence for each class [1].

$$f(x) = exp(x_k) / (\sum_{n=1}^{k-1} exp(x_n)) \tag{5}$$

3.4 Fine-Tuning

Fine-tuning is a method for training a neural network that has been previously trained on a different dataset [17]. The architecture of the network is kept the

same, however, the weights and biases of the last three layers are removed. In doing this, the network maintains what has been learned at lower levels, from the previous dataset. When training by fine-tuning, the network can be more efficiently trained to learn higher-level features of the new dataset. However, if training is performed in the normal manner, after replacing the final three layers of the network, the weights and biases of the lower level features previously learned by the network will be changed as well. To prevent this, all layers, except the final three layers, have their weights and biases frozen.

4 Proposed Method

In this work, we decided to analyze VGG-16[14] and Densenet-201[4]. MATLAB was used for all of the experiments performed. We wanted to see what features these networks focused on after training them with a modified version of the Stanford Dogs dataset. To accomplish this, we fine-tuned these networks, which were both previously trained on the ImageNet dataset [2]. After fine-tuning each network, we tested the networks on a test set, constructed a confusion matrix, and collected the correctly classified images into a smaller dataset, individually for each network. We then had each network classify the images in its correctly classified dataset, taking note of the top 20 activations per class. Following this, we obtained the frequencies of each activation, across all classes, and removed any activations which were present more than 30 times. This gave us the most common activations for each class, that were also unique to that specific class. Finally, we had each network classify the images in its correctly classified dataset, this time retrieving the activations that were noted as common to, and unique for each class. We then combined the activations, individually, with the original input images, saving them for further analysis.

4.1 VGG-16 Structure

VGG-16[14] has a total of 41 layers. The overall structure of VGG-16[14] with fine-tuning implemented, can be seen in figure 6. The network takes an image of size 224x224x3 as input. The layers of VGG-16[14] can be found in table 1.

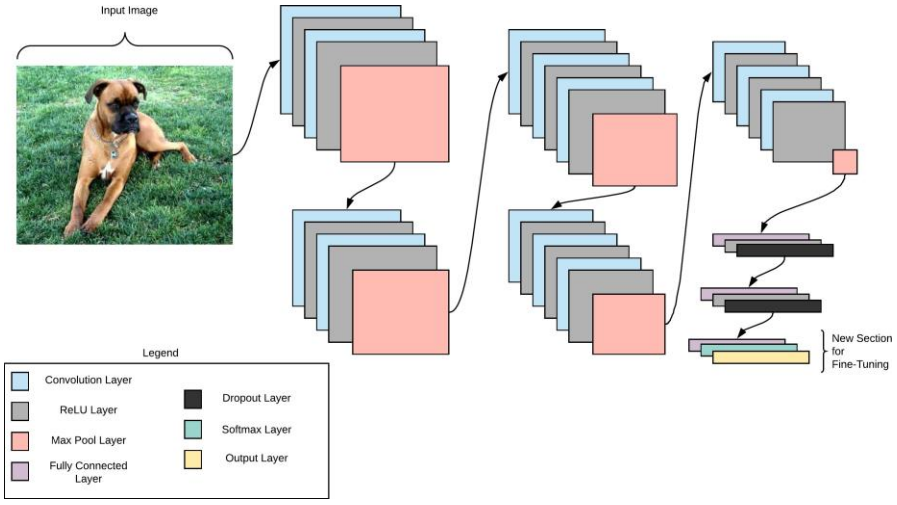

Fig. 6: A schematic representation of VGG-16.

Table 1: The structure of VGG-16 used.

Layer	Type	Activations
1	Image Input	224x224x3
2	Convolution	224x224x64
3	ReLU	224x224x64
4	Convolution	224x224x64
5	ReLU	224x224x64
6	Max Pooling	112x112x64
7	Convolution	112x112x128
8	ReLU	112x112x128
9	Convolution	112x112x128
10	ReLU	112x112x128
11	Max Pooling	56x56x128
12	Convolution	56x56x256
13	ReLU	56x56x256
14	Convolution	56x56x256
15	ReLU	56x56x256
16	Convolution	56x56x256
17	ReLU	56x56x256
18	Max Pooling	28x28x256
19	Convolution	28x28x512
20	ReLU	28x28x512
21	Convolution	28x28x512
22	ReLU	28x28x512
23	Convolution	28x28x512
24	ReLU	28x28x512
25	Max Pooling	14x14x512
26	Convolution	14x14x512
27	ReLU	14x14x512
28	Convolution	14x14x512
29	ReLU	14x14x512
30	Convolution	14x14x512
31	ReLU	14x14x512
32	Max Pooling	7x7x512
33	Fully Connected	1x1x4096
34	ReLU	1x1x4096
35	Dropout	1x1x4096
36	Fully Connected	1x1x4096
37	ReLU	1x1x4096
38	Dropout	1x1x4096
39	Fully Connected	1x1x120
40	Softmax	1x1x120
41	Output	–

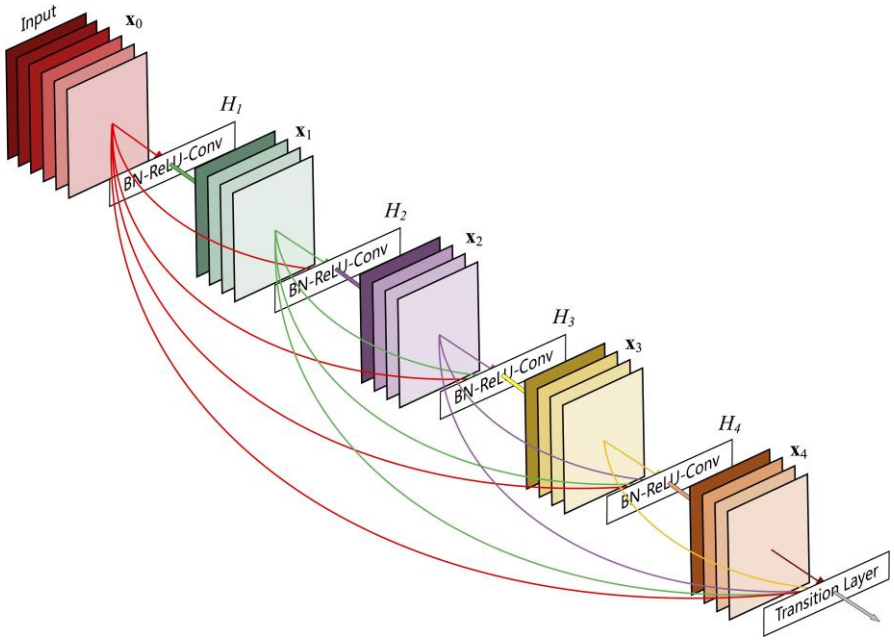

Fig. 7: A schematic representation of a dense block with five layers [4].

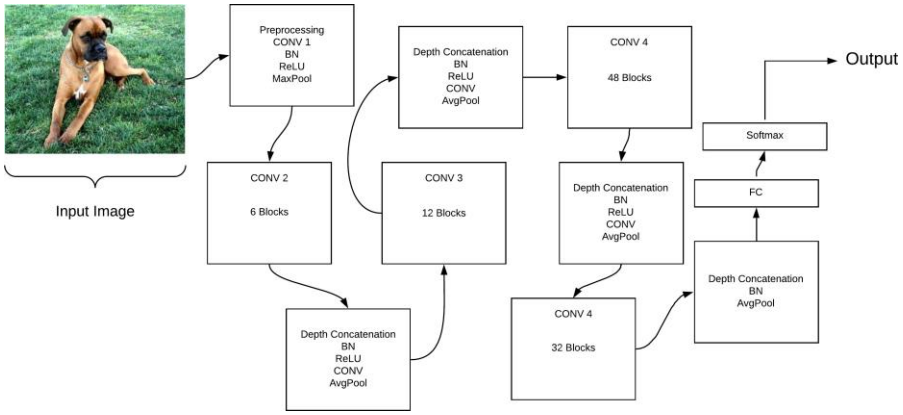

Fig. 8: A schematic representation of DenseNet-201.

4.2 DenseNet-201 Structure

DenseNet-201 [4] has a total of 201 layers. This network also has an image input size of 224x224x3, however, this network has a different connection pattern than VGG-16[14]. Instead of being completely sequential, each layer passes its feature maps, as input, to all further layers, and each layer takes, as input, each previous layer's feature maps [4]. A schematic representation of an individual dense block, taken directly from [4], can be seen in fig. 7, and a schematic representation of the full network can be seen in fig. 8. A table describing the full network has been omitted from this paper, as it is approximately 709 rows.

5 Experiments

For each network, the experiments were conducted separately.

5.1 Dataset

We used the Stanford Dogs dataset [6], which contains 120 classes of dog breeds. Each class had approximately 150 images for a total of 20580 images. Since, such a small dataset can lead to over-fitting, especially with such large CNNs, we decided to modify the dataset by converting the images to high-contrast and adding them to the original dataset. This gave us, approximately 300 images per class. We then split the dataset into a test set, containing 100 images per class, and a remaining set, containing approximately 200 images per class. Before training, we further divided the remaining set into a validation set, containing 30 percent of the remaining set, and a training set, containing 70 percent of the remaining set.

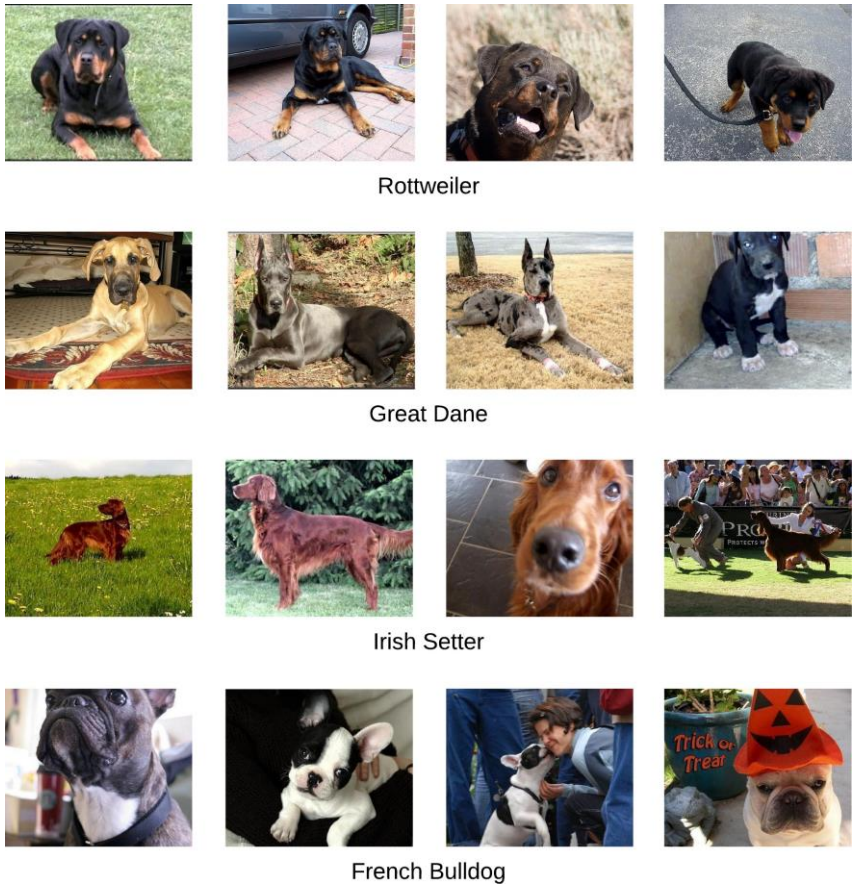

Fig. 9: Sample images from five breeds of our dataset.

5.2 Training

Each network was fine-tuned on the training set, using an image augmenter that had a pixel range of -30 to 30. The training consisted of mini-batches of 25, and a maximum number of 10 epochs. However, both networks were stopped before the maximum number of epochs was reached. The initial learning rate was set at 3e-4. Validation was conducted following every 50 iterations using the

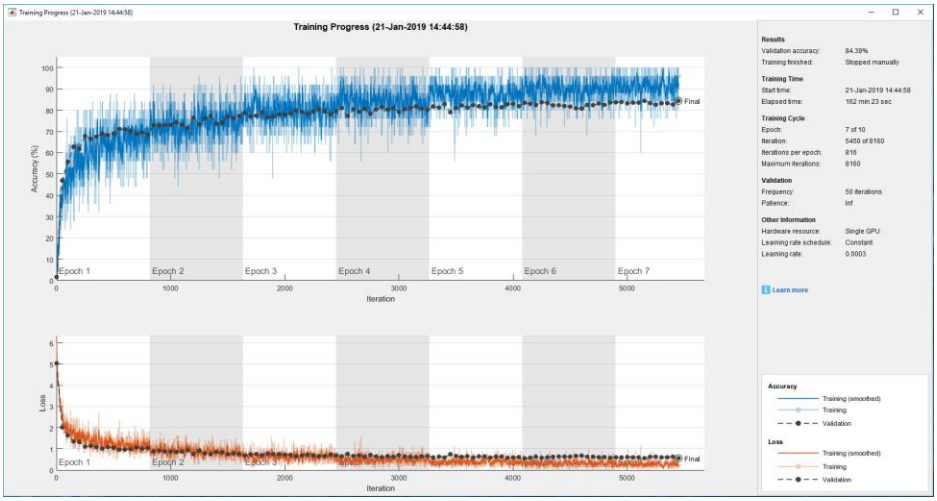

Fig. 10: VGG-16 Training Progress

validation set. Training data was shuffled after every epoch, and the training progress plotted for VGG-16 can be seen in figure 10 and for DenseNet-201 can be seen in figure 11.

5.3 Testing

Testing each network on the test set, we created a confusion matrix. The rows of the confusion matrix signify the correct class, while the columns signify the network's output class. The confusion matrix for VGG-16 in figure 12 and DenseNet-201 in figure 13 show a clear diagonal line, demonstrating a good performance on the test set. We can tell which classes proved more difficult than others, for each network, by looking at the diagonal line on the confusion matrix. The class that lines up with a spot that is not red, is a class that the network did not perform as well on.

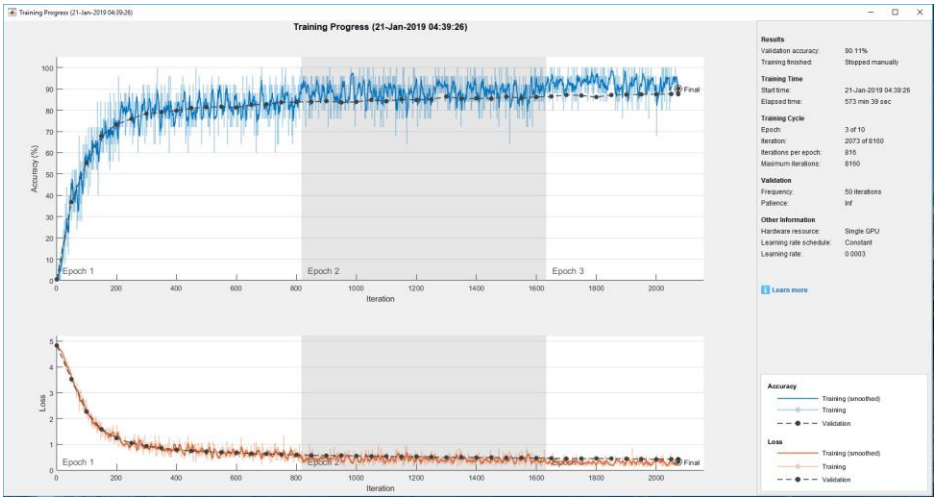

Fig. 11: DenseNet-201

6 Results

With each network fine-tuned and tested, we analyzed the features of each network's final CONV layer/block, to see what high-level features the network had detected when classifying the images.

6.1 Classification

Class Name	VGG-16 (%)	DenseNet-201 (%)
Affenpinscher	88	94
Afghan Hound	92	98
African Hunting Dog	95	100
Airedale	87	97
American Staffordshire Terrier	83	77
Appenzeller	56	72
Australian Terrier	85	87
Basenji	92	95

Basset	84	94
Beagle	88	90
Bedlington Terrier	98	100
Bernese Mountain Dog	86	96
Black and Tan Coonhound	84	91
Blenheim Spaniel	86	99
Bloodhound	94	98
Bluetick	94	95
Border Collie	83	70
Border Terrier	97	96
Borzoi	81	95
Boston Bull	88	90
Bouvier des Flandres	78	87
Boxer	81	88
Brabancon Griffon	90	94
Briard	78	88
Brittany Spaniel	77	92
Bull Mastiff	88	88
Cairn	90	93
Cardigan	63	92
Chesapeake Bay Retriever	78	94
Chihuahua	78	86
Chow	95	96
Clumber	95	98
Cocker Spaniel	75	90
Collie	66	74

Curly-Coated Retriever	91	94
Dandie Dinmont	83	98
Dhole	94	97
Dingo	83	86
Doberman	68	98
English Foxhound	82	73
English Setter	80	90
English Springer	94	90
Entlebucher	82	91
Eskimo Dog	56	34
Flat-Coated Retriever	91	95
French Bulldog	85	98
German Shepherd	79	91
German Short-Haired Pointer	93	96
Giant Schnauzer	92	92
Golden Retriever	84	94
Gordon Setter	92	99
Great Dane	87	89
Great Pyrenees	81	99
Greater Swiss Mountain Dog	92	89
Groenendael	88	96
Ibizan Hound	94	97
Irish Setter	81	96
Irish Terrier	85	83
Irish Water Spaniel	85	91
Irish Wolfhound	84	78

Italian Greyhound	88	93
Japanese Spaniel	91	96
Keeshond	93	98
Kelpie	73	79
Kerry Blue Terrier	86	91
Komondor	95	99
Kuvasz	84	82
Labrador Retriever	82	89
Lakeland Terrier	80	82
Leonberg	99	100
Lhasa	83	82
Malamute	79	78
Malinois	95	94
Maltese Dog	79	96
Mexican Hairless	97	96
Miniature Pinscher	91	92
Miniature Poodle	42	73
Miniature Schnauzer	52	87
Newfoundland	89	92
Norfolk Terrier	83	88
Norwegian Elkhound	93	94
Norwich Terrier	74	77
Old English Sheepdog	68	96
Otterhound	88	88
Papillon	91	97
Pekinese	82	91

Pembroke	87	88
Pomeranian	93	96
Pug	90	99
Redbone	79	79
Rhodesian Ridgeback	79	92
Rottweiler	99	100
Saint Bernard	99	97
Saluki	86	97
Samoyed	85	95
Schipperke	93	96
Scotch Terrier	88	95
Scottish Deerhound	87	98
Sealyham Terrier	90	96
Shetland Sheepdog	66	89
Shih-Tzu	55	81
Siberian Husky	67	87
Silky Terrier	68	78
Soft-Coated Wheaten Terrier	82	93
Staffordshire Bullterrier	74	81
Standard Poodle	66	86
Standard Schnauzer	69	71
Sussex Spaniel	91	92
Tibetan Mastiff	78	85
Tibetan Terrier	85	90
Toy Poodle	73	75
Toy Terrier	69	79

.

Vizsla	77	84
Walker Hound	52	73
Weimaraner	91	99
Welsh Springer Spaniel	73	89
West Highland White Terrier	78	95
Whippet	68	92
Wire-Haired Fox Terrier	70	86
Yorkshire Terrier	83	89

Table 2: Class Accuracies for Both Networks

VGG-16 performed at an overall accuracy of about 82.71 percent, while DenseNet-201 performed at an overall accuracy of about 89.33 percent. However, this is across all classes, and not representative of how each network performed for each class. As we have shown, the diagonal for the confusion matrix of both VGG-16 in figure 12 and that of DenseNet-201 in figure 13 are not perfect. Therefore, there were classes for which, each network performed better than their overall accuracy, as well as those in which they performed worse. This can clearly be seen if we look at the class accuracies for each network, which can be seen in table 2.

6.2 Over-fitting

Over-fitting is an issue that must be addressed when working with a dataset of a size similar to the Stanford Dogs dataset. Although there are 20580 images in the original dataset, those images span 120 classes. This makes for the small class size of approximately 150 images. Even if we assume the approximate class size, we needed to split the dataset into three smaller sets. Further reducing

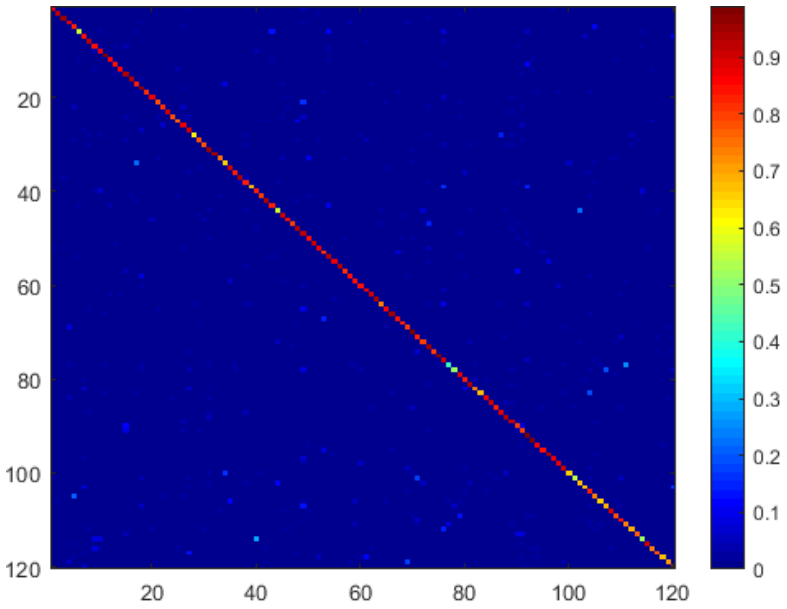

Fig. 12: Confusion Matrix for VGG-16

the amount of data to train with. This scenario often results in over-fitting. That is, when the network stops attempting to find a generalizing pattern, and begins to memorize the training images. We wanted to avoid over-fitting, since the networks would show good performance on only the training data, since this would be what the networks had learned.

The size of the dataset was not the only concern, as the size of the network in relation to the dataset can also determine if over-fitting will occur. With both networks being on the larger side, it should be noted that VGG-16 has far fewer layers than DenseNet-201. Though over-fitting can be seen in VGG-16 (fig. 10) and DenseNet-201 (fig. 11), we can see that DenseNet-201 may be more affected by over-fitting than VGG-16. We can see this, because DenseNet-201 reaches perfect accuracy on the training set, within the first epoch. However,

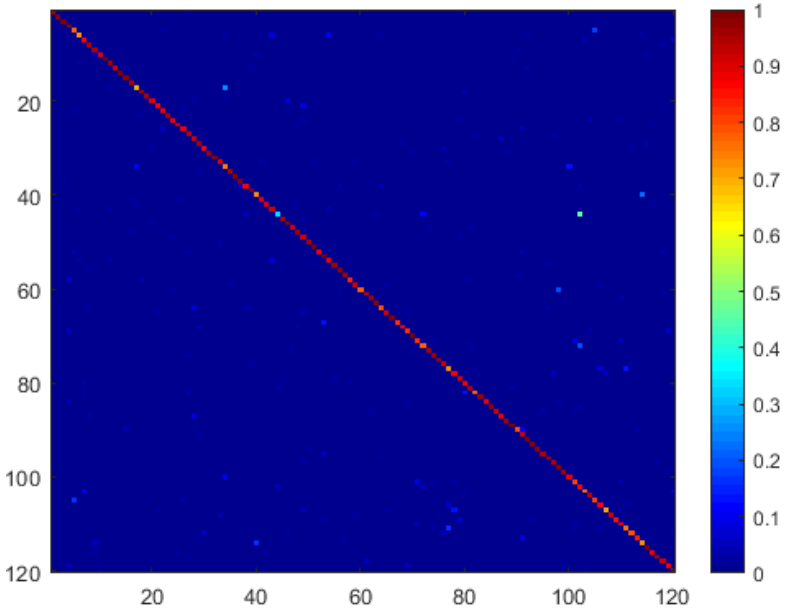

Fig. 13: Confusion Matrix for DenseNet-201

VGG-16 does not approach perfect accuracy on the training set until its third epoch of training. This may be a result of a small dataset size, matched with the large sizes of these networks. A possibility is that DenseNet-201 is more affected by over-fitting, even with our modified version of the Stanford Dogs dataset, because of its size with relation to the size of the dataset. As for VGG-16, it is still apparently affected by over-fitting, but to a lesser extent, possibly due to the smaller size of the network, compared to that of DenseNet-201.

6.3 Feature Extraction

Feature extraction was conducted on the final convolution layer of each network, using images from the test set which were correctly classified. We had setup the initial step for this process, when we tested the networks and created the

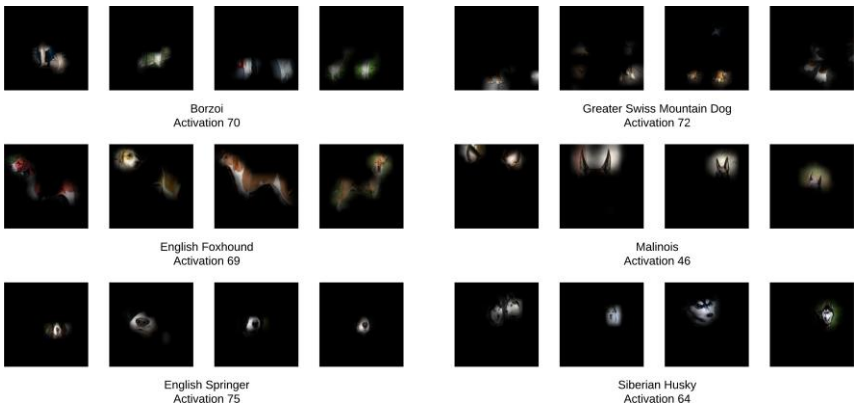

Fig. 14: Sample output from the feature extraction of our fine-tuned VGG-16.

confusion matrices, by recording all of the top activation used for each breed. Once we had that information, we wrote a program to determine the frequencies of each activation used across all breeds and removed any activations from our records that had a frequency of more than 30. Thus, we removed the activations which could be considered common among more than 25% of the breeds. With this information, we hoped to find those activations which were common to, at the very least, a handful of breeds. Using the activations that remained, we combined the images used with the activations, to create visual representations of what each activation was recognizing. We then manually analyzed these images to gain insight into how the network was distinguishing between the breeds. Sample output of our method can be seen in fig. 14 and fig. 15 for our fine-tuned VGG-16 and Densenet-201, respectively.

As can be seen in the figures, there are certain features that these activations light up, which correspond to features we would expect to be used when classifying dog breeds. For instance, we can say that the legs of the dog are a focal point for activation 70 of VGG-16 for the Borzoi class, as well as activation 31

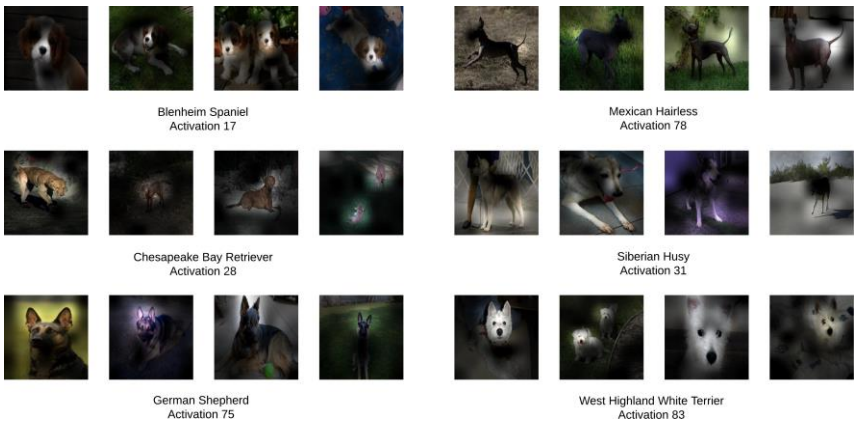

Fig. 15: Sample output from the feature extraction of our fine-tuned Densenet-201.

of Densenet-201 for the Siberian Husky class. We can say the same for the nose of the dogs (activation 75 of VGG-16 for English Springer and activation 17 of Densenet-201 for Blenheim Spaniel), ears of the dogs (activation 46 of VGG-16 for Malinois and activation 75 of Densenet-201 for German Shepherd), and even the faces of the dogs (activation 64 of VGG-16 for Siberian Husky and activation 83 of Densenet-201 for West Highland White Terrier). However, we did find some activations more difficult to interpret than others, including activation 69 of VGG-16 for English Foxhound and activation 28 of Densenet-201 for Chesapeake Bay Retriever. For these, we might be able to say that the activations are focused on the color, color patterns, or even the shapes of the breeds. We have decided that activation 69 of VGG-16 appears to be more focused on the color pattern correlating with the brown and white of the English Foxhound. Whereas activation 83 of Densenet-201 appears to be focusing more on the shape, possibly strictly the color, of the Chesapeake Bay Retriever. Even though these have been troublesome in determining the exact feature the network is focusing on,

we can say that there appears to be a definite feature that each has learned, in distinguishing between the various breeds.

6.4 Discussions

Fig. 16: Example of incorrectly classified Miniature Poodle as Toy Poodle (Example of Toy Poodle on right).

Confusion Examples By looking at the overall accuracies of both networks on the test sets, keeping in mind the diagonals found in the confusion matrices, we can say that both networks performed well in general. However, as we have noted previously, there were certain classes, or breeds, for which each network had trouble with. Looking at the accuracies for each class, found in the class accuracies table (table 2), we have found that the class which caused the most trouble for VGG-16 was the Miniature Poodle. For DenseNet-201, the most troublesome class was the Eskimo Dog. Additionally, we determined that the class which VGG-16 confused Miniature Poodle images for, was the Toy Poodle class. Miniature Poodle images were classified as Toy Poodle 26% of the time

Fig. 17: Example of incorrectly classified Eskimo Dog as Siberian Husky (Example of Siberian Husky on right).

by VGG-16. As for DenseNet-201, it classified Eskimo Dog images as Siberian Husky 46% of the time. An example of a Miniature Poodle image incorrectly classified by VGG-16 as a Toy Poodle can be seen in fig. 16. Included in fig. 16, is an example image of the Toy Poodle class. In fig. 17, we included an example of an Eskimo Dog image which was incorrectly classified by DenseNet-201 as a Siberian Husky, as well as an example image of the Siberian Husky class. Intuitively, these mistakes make sense, as it is difficult to differentiate between these classes (or breeds) when only looking at these images. The similarities between these classes can easily be seen, as they share a number of characteristics. So, we can see why the networks have incorrectly classified these images.

Philosophical Interpretation To interpret our results, and the networks themselves, we will be using Locke's understanding of concepts, or ideas as he calls them. We will also take note of how Locke understands words, what we use them for, and what they actually mean. Using this, and taking our networks,

and the results as our evidence, we will propose what can be said about this work, from the standpoint of a Lockean theory of concepts and words. From hereafter, we shall use the term 'idea' in place of 'concept', to reduce the need for repetition or explanation, as well as to keep consistent with Locke.

In discussing what an idea is, Locke says, "Idea is the object of thinking" [11, p. 33]. This meaning that there is something within our thinking which we take as the object of our thoughts, and that is what idea is, the things which we think about, and use in our thinking. Locke claims that we get our ideas from sensation and reflection [11, p. 33]. As such, ideas for Locke, do not come from native (innate) ideas, or are not to be considered "stamped" on our minds when they are created [11, p. 33]. Rather, ideas come from our experiences, be they sensation or reflection [11, p. 33]. We will now look into Locke's understandings of sensation and reflection, as they pertain to ideas.

Sensation is the source of the ideas which come from our senses, according to how external objects interact with our senses [11, p. 33]. The senses, "convey into the mind," [11, p. 33] perceptions of the external objects, conveying that which produces the perceptions (the external object which interacts with our senses) into the mind [11, p. 34]. Put another way, this is a forming of an internal representation of the external objects, through our senses. Reflection, on the other hand, does not deal with the senses [11, p. 34]. However, reflection can be considered the internal equivalent of sensation [11, p. 34]. Locke points out that reflection refers to the, "operations of our minds" [11, p. 34]. As such, the ideas which come from reflection, are those that come from, "the perception of the operations of our own minds within us," used with the ideas we have [11, p. 34]. Locke takes note that there is no comprehensible unconscious thought that we can have [11, pp. 36–37]. Additionally, Locke makes it clear that, although he does not find there to be innate ideas, our minds have a certain capacity

to receive impressions through sensation or reflection [11, p. 39]. Which just means that we have the ability, or, it is possible for us, to have the ideas which we do.

Continuing on, Locke distinguishes simple ideas from complex ideas. Concerning the former, Locke designates these as those ideas that represent, "one uniform appearance, or conception in the mind" [11, p. 40]. Simple ideas are not formed by any sort of compounding, or combination of ideas, and thus, cannot be broken down or separated into any smaller ideas [11, p. 40]. Further, looking into ideas, Locke makes clear that our ideas are internal representations of external objects, and just that [11, p. 48]. Taking this into consideration, we can note that, our ideas are not exact images or, "resemblances of something inherent in," these external objects [11, p. 48]. These internal representations have, as Locke notes, a similar relation to the external objects they represent, as names have with the ideas they signify [11, p. 48]. Concerning the latter of the types of ideas mentioned above, Locke says that they are the combination of several simple ideas [11, p. 66]. In addition to creating complex ideas, we can get relations between ideas by bringing them together, but not combining them into a single idea, as well as, form general ideas by separating them from ideas they accompany in, "their real existence" [11, p. 66]. This separating of ideas, which brings us to general ideas, is what Locke refers to as abstraction [11, p. 66]. Locke also explains that complex ideas are either modes (dependences or affections of substances), substances (combinations of simple ideas representing distinct things subsisting by themselves), or relations (coming from our considering or comparing of two ideas) [11, pp. 67–68]. Understanding the basics of Locke's ideas as such, we move to Locke's understanding of the signification of words.

To begin, Locke mentions that words are sensible signs which we use for communicating with others [11, p. 178]. Following this, Locke adds that "Words

are the sensible signs of his ideas who uses them" [11, p. 178]. Words, when used to communicate, are used to signify the ideas of the speaker, with the intention, and hope, of being understood by the listener [11, p. 178]. There are two underlying principles, or thoughts, when words are used this way. The first, is that the words refer to the same idea in the listener, as they do in the speaker [11, p. 179]. The second, is that the words refer to the reality of things [11, p. 179]. We hope,

that when we speak, we can convey our ideas, which are in our mind, to someone else, through the use of our words. This being the basis of our communicating. We also hope, that what we are talking about, is real, rather than some sort of hallucination, or fantasy. Though there are a number of word types, words of particular interest for this work are general words, which we will take a look at next.

Locke describes general words as the signs of general ideas [11, p. 181]. Further developing this, general ideas are formed by the removal of time, place, and any idea which distinguishes particular instances [11, p. 181]. Locke points out that this removal of ideas is removing, "that which is peculiar to each," and retaining, "only what is common to all" [11, p. 181]. From this, we can see that general words signify the sorts of things, or, the essence of a sort [11, p. 184]. This essence of a sort, that which makes anything to be of that sort, is the general, abstract ideas that the words are taken as signs of [11, p. 184]. These general words are formed by our understanding, which abstracts ideas, but are also founded in the, "similitude of things" [11, p. 184]. It is these sorts (the abstract, general ideas), to which we are capable of comparing particular instances of things, to find similarities between, and thus determine if a particular instance of a thing belongs to a particular sort [11, p. 184]. It will be important to understand what we have been referring to, when referring to the essence of a sort.

The essence referred to above, has the possibility of two meanings. The first is what Locke calls the real essence, or the internal constitution of substances, which is unknown to us [11, p. 184]. The second possible meaning, Locke calls the nominal essence, or the abstract ideas that a name (general word) of a sort signifies [11, p. 184]. As such, the name of a sort signifies the nominal essence of the sort, and can only be attributed to things which have that essence [11, p. 184]. Locke
makes clear, that we distinguish sorts by their nominal essence (our abstract, general ideas), and not by their unknown real essence [11, p. 186]. For simple ideas and modes, Locke notes that the real and nominal essences are the same [11, p. 186]. However, substances are unlike simple ideas and modes in that, a substance's real and nominal essences differ [11, p. 186].

Taking a look at the names of substances in particular, Locke points out that common names for substances stand for sorts [11, p. 192]. These common names are the signs of the complex ideas of the sort they stand for [11, p. 192]. Following this, the essence of a sort is the abstract ideas essential to that sort, which the name signifies [11, p. 192]. Locke then explains that this essence, which bounds the sorts, is the nominal essence [11, p. 195]. Meaning, these sorts, are nothing more than our ranking things (or grouping things) by names that signify the complex, abstract, general ideas within us [11, p. 195]. This talk of sorts has been mainly to point out that, though we think there are real reasons for why we sort things (or group them), the only reasons we have, are the abstract ideas which we form through our experiences.

Now that we have an understanding of Lockean ideas and words, we can ask, what could be said about these networks from this view? To begin, we will suppose that these programs have some form of consciousness, and therefore are capable of thinking (though it may be vastly different from our own). We want to point out, that we do believe supporting such a position would prove

very difficult, especially with the type of networks used in this work. Even so, proceeding from here, we will try to formulate a Lockean interpretation of these networks, taking the output and our analyses of the programs as evidence.

We begin, by noting that, by the very nature of these networks, and our programs, they are basing their 'sorts' on our human abstract ideas, even if only initially. We claim this, because we have determined, within the dataset, and the possible outputs for each network, what the classes are. Based on these classes, the networks are attempting to match the images to the classes which we have designated. This would mean, that any abstract ideas formed by the networks, have followed the organization of the abstract ideas which we have formed.

Next, we claim that, surely, there are no innate ideas present in these networks. The ideas formed by these networks, come from their experiences, which can be further divided into their sensation and reflection. For sensation, these networks are taking in input images, and thus experiencing them. Additionally, these networks perform calculations, combine information, and consider what class an image belongs to, which can be considered their reflection. However, we note that an argument could be brought against this claim, though we believe it to be nothing more than a superficial one. It could be said, that innate ideas are introduced, because we are applying fine-tuning (transfer-learning) on the networks. In response to this, we claim that, rather than viewing these networks as entirely new ones (thus, relating fine-tuning to being born), we should view them as remaining one, consistent network. Through this understanding of how fine-tuning is working, this work is a case of the networks forming new complex ideas for the current (new) experiences. Additionally, the ideas, or knowledge, that come from fine-tuning a network, were previously learned through experiences, and not innate ideas. As such, we find that the fine-tuning of these networks should not truly support the presence of innate ideas in this work.

When looking at what these networks output, we should note that the class names can be given. As such, these class names signify an internal representation for each of the networks. These internal representations can be found in the activations, or features, that determine what class a particular instance (input image) belongs to. As such, these internal representations, for each class, could be said to be the abstract, general ideas that are the nominal essences of the classes for the networks. Getting somewhat more technical, we could say that the first few layers of the networks could be holding the simpler, if not simple ideas. We think this is plausible, as this is where lines, curves, simple shapes, colors, and patterns are detected. Closer towards the outputs of these networks, their final layers could be said to have the complex ideas which are formed by the calculations and combinations of the previous layers' simpler ideas. Meaning, these networks sort of demonstrate the way in which Locke claimed complex ideas were formed. Lastly, the outputs of these networks give the names of the classes, and therefore, through the combination of complex ideas, and after learning how to generalize for each class, these networks have abstract, general ideas of each class, signified by the class name.

7 Conclusions

In this work, we have found that the two networks used, VGG-16 and DenseNet-201, are capable of finding patterns that are humanly recognizable when fine-tuned on the Stanford Dogs dataset. Though over-fitting was present in both networks, we took necessary precautions to prevent and reduce the effects of over-fitting, and produced results and analysis to prove that patterns were recognized by both networks, even in the presence of over-fitting. With our analyses of the response maps (or feature maps) of both networks, we were able to find breed specific features. By combining our understanding of both networks, and

the features, we were able to interpret the networks with respect to Locke's understanding of ideas and words. Finding that, though we do not consider these networks to be conscious, the networks themselves, fit well into a Lockean understanding. We were able to demonstrate in this work, that internal representations were present, the output represented general words, and the networks lack innate ideas, learning based on experiences alone.

For further research, it would be interesting to take a look at how these networks would perform on a dataset that had mixed-breed (mutt) dogs. If the output of the networks were changed to provide the top four or five classes, and confidence values for each, would the networks' outputs contain the dog breeds that constitute the mixed-breed dog? The reason this would be interesting to test, is because it would mean that a network trained on a specific dataset, for a particular classification task, with few modifications (i.e. the output), could possibly perform a task other than its originally intended task. The reason we believe this may be possible, is because the main function of these networks is to find patterns. If this worked, it would prove that patterns had been recognized by the networks, which could be applied to multiple tasks. This, of course, is assuming that the characteristics of mixed-breed dogs is based on a combination of the breeds which constitute it, and that these can be seen in an image.

8 References

[1] A. F. Agarap, "Deep Learning Using Rectified Linear Units (ReLU)," *arXiv.org*, 22 Mar. 2018 [Revised 7 Feb. 2019]. [Online]. Available: https://arxiv.org/abs/1803.08375.

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large scale hierarchical image database," in Proc. Computer Vision and Pattern Recognition [CVPR09], 2009. [Online]. Available: www.image-net.org/papers/imagenet_cvpr09.pdf.

[3] D. Hsu, "Using convolutional neural networks to classify dog breeds," CS231n: Convolutional Neural Networks for Visual Recognition [course webpage]. 2015. [Online]. Available: http://cs231n.stanford.edu/reports/2015/pdfs/fcdh_FinalReport.pdf.

[4] G. Huang, L, Zhuang, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2017. [Online]. Available: https://ieeexplore.ieee.org/document/8099726.

[5] A. K. Jain, J. Mao, and K. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 29, no. 3, pp. 31-44, 1996.

[6] A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei, "Novel dataset for fine-grained image categorization: Stanford dogs," In: First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition, 2011. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.255.6394.

[7] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, Efficient BackProp. In *Neural Networks: Tricks of the Trade*, G. B. Orr and K.-R. Müller, Eds. London: Springer, 1998, pp. 9-50.

[8] F. F. Li, J. Niebles, and S. Savarese, "Convolutional neural networks (CNNs / ConvNets)," CS231n: Convolutional Neural Networks for Visual Recognition [course webpage]. 2018. [Online]. Available: http://cs231n.github.io/convolutional-networks/ .

[9] F. F. Li, J. Niebles, and S. Savarese, "Setting up the data and the model: regularization," CS231n: Convolutional Neural Networks for Visual Recognition [course webpage]. 2018. [Online]. Available: http://cs231n.github.io/neural-networks-2/#reg.

[10] J. Liu, A. Kanazawa, D. Jacobs, and P. Belhumeur, "Dog breed classification using part localization," in Proc. European Conference on Computer Vision [ECCV], 2012, pp. 172-185.

[11] J. Locke, *An Essay Concerning Human Understanding*. Hackett Publishing, 1996. [E-book] Available: Amazon.

[12] "Cntk 103: Part d-convolutional neural network with mnist," *Microsoft*, accessed 15 Apr. 2018. [Online]. Available: https://cntk.ai/pythondocs/CNTK_103D_MNIST_ConvolutionalNeuralNetwork.html.

[13] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

[14] K. Simonyan, and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in ICLR 2015, 2015. [Online]. Available: https://arxiv.org/pdf/1409.1556.pdf.

[15] B. Karlik, and A. V. Olgac, "Performance analysis of various activation functions in generalized MLP architectures of neural networks," *International Journal of Artificial Intelligence and Expert Systems*, vol. 1, no. 4, pp. 111–122, 2011.

[16] X. Wang, V. Ly, S. Sorensen, and C. Kambhamettu, "Dog breed classification via landmarks," In Proc. 2014 IEEE International Conference on Image Processing, 2015, pp. 5237–5241.

[17] J. Yosinski, J. Clune, Y. Bengio, Y., and H. Lipson, "How transferable are features in deep neural networks?," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. pp. 3320–3328. Curran Associates, 2014, http://papers.nips.cc/paper/5347-how-transferable-are-features-in-deep-neural-networks.pdf.