Sample Complexity of Few-shot Learning in Convolutional Neural Networks

Mujin Kwun and John Wang

mujinkwun@college.harvard.edu

jwwang@college.harvard.edu

May 11, 2022

Abstract

In recent decades, Convolutional neural networks (CNNs) have become increasingly popular as the solution for image classification tasks. Though the architecture for these networks has evolved quickly, the theory has mostly lagged behind. Recent research has posited a new theory [1] regarding the geometry of object embeddings as they pass through the layers of the CNN. It has been shown that this theory can explain few-shot learning accuracy in both CNNs and animal visual pathways. What we show in our research is that this theory explains relative few-shot classification accuracies between models of varying depth, providing insight into why the visual pathway is segmented into numerous cortices.

1 Introduction

In his seminal paper, "Computing Machinery and Intelligence," Alan Turing wrote, "the idea behind digital computers may be explained by saying that these machines are intended to carry out any operations which could be done by a human computer" [10]. In the decades since, thanks to the development of powerful computing chips that can support the demands of revolutionary algorithms and models, computers have indeed emulated and even surpassed humans in many fields. Complex algorithms have been developed to surpass humans in pattern and image recognition, weather prediction, image generation/art, and even playing strategic games. However, one defining feature of human intelligence that we have yet to completely emulate with computer algorithms is "few-shot" learning, or the ability to learn new concepts with just a few examples. The success of computer algorithms thus far has largely been predicated on not only computing power but also the availability of large, comprehensive datasets on the concepts to be learned. For example, state of the art image classification models typically requires tens of thousands of training images to learn concepts. This is in stark contrast with the innate human ability to learn concepts with just a few example images or no images at all in the case where concepts can be learned with a verbal description alone [7].

The benefits afforded by developing few-shot learning algorithms are numerous. Few shot learning eliminates the need to collect and maintain extremely large supervised datasets for models to be trained on. Naturally, this yields new use cases where few shot learning algorithms can be deployed in cases where the collection of large-scale supervised data is difficult or even impossible as in cases dealing with sensitive, private medical data. In this paper, we focus on few shot learning classifiers. Existing algorithms are typically categorized as N-way-K-shot classifiers in which N concepts are learned with K examples in each concept [11]. Specifically, in our work, we will focus on exploring 2-way classifiers, and we draw inspiration from the human brain to do so.

Current theories about concept learning in humans posit that concepts are learned as sets of features rather than rigid, comprehensive descriptions and definitions. Practical applications of this idea originated from experiments on the visual cortex of cats that found the presence of different cells that detect specific features at different spatial locations or receptive fields [8]. These findings were integral in the development of the first Convolutional Neural Nets. The two primary dueling theories about human concept learning through features are exemplar and prototype theory. Prototype theory proposes that features from previously seen examples of a given class are averaged into a single prototype so that new examples can be quickly compared with the existing prototypes, intuitively conducive to quick judgement [5]. In contrast, exemplar theory posits that new examples are compared to the features of previously seen examples that are directly stored in memory.

In learning visual concepts, research has shown that visual stimuli result in high dimensional representation in various points along the ventral visual pathway (V1, V2, V4.) This ultimately leads to a rich representation in the IT cortex of the human brain, upon which a neuron infers the concept that the source of the visual stimulus belongs to depending on the pattern it elicits in the IT cortex. In this paper, we first adapt this idea to deep neural networks for image classification by extracting readouts at different layers in the models, analogous to the representations at various points in the ventral visual pathway. We build classifiers on top of these existing DNN's to use the resulting readouts at a given layer from previously seen ”training” examples to classify future examples using the corresponding readouts at the same layer, much like a neuron uses the representation at the IT cortex to classify visual stimuli. We explore both prototype and exemplar theory in building our classifier upon the linear readouts. Previous work [**?**] indicates that in few-shot learning image classifiers, using prototypes to classify future examples tends to yield higher accuracy. Thus we chose to focus on building a classifier that maintains two prototypes for each of the two concepts and classifies new examples based on euclidean distance to the existing prototypes. Readouts from previously seen examples are averaged per class, and readouts produced from inferring on new data are compared to these prototypes to generate a classification.

We explore the possibility of using this architecture to few-shot learn, empirically testing the number of examples needed from two previously unseen concept classes to train the classifier to reach a threshold accuracy on future examples. Furthermore, we hypothesize that much like the human brain, the feature representations in the readouts for deeper layers encode more information than initial layers, allowing for higher accuracy in few-shot learning if we choose to extract readouts from later layers. Naturally, we believe that this architecture will perform better for deeper, rather than shallower neural nets, and we test this hypothesis on three neural networks of varying depth.

2 Theory

2.1 Geometric Manifolds in Concept Learning

Theory posited by Sompolinsky et al. (2018) [2] suggests that viewing concept classes as manifolds in high dimensional space can provide insight into how deep convolutional neural networks perform in comparison to animal brains. Research has suggested that there is a dimensionality reduction in Macaque visual pathways that result in feature detectors, with data recorded and collected by Majaj et. al (2015) [6]. Paralleling this idea is the idea that the embedding space of objects as they go deeper through the layers of convolutional neural networks may provide insight into how classification may work in the context of object recognition.

The idea is that there is a manifold in space corresponding to all possible variants of a certain object (face, scene, etc), such as the one shown in Figure 1. Note that these manifolds can be complex - as such, to simplify calculations, for any manifold a, we can describe it via it's centroid, x_a^0, and the dimensions in which the manifolds take in space, defined by unit vectors $u_a^1, ..., u_a^D$, each with corresponding radii $R_a^1, ...R_a^D$. These manifolds can be thought of D-dimensional ellipsoid. For each manifold, the question that classification asks is if it is possible to find a rule that separates the two manifolds with high enough probability to give on average high accuracy for performing the task.

In order to perform classification well, it is necessary to have concept classes correspond to point-cloud representations that are sufficiently separated, and small enough variance such that images in these concept classes do not have representations that bleed into convex hull of the manifolds themselves. Otherwise, we encounter possibilities for error.

Note that it is possible to project the data up into arbitrarily large dimension and classify it. This can be done with kernel based SVM, or some other random projection into $N \to \infty$ neurons. However, the issue is that there are a finite number of neurons that must be able to carry out object recognition capability. In addition, there are theoretically infinite number of points within these manifolds that need to be classified. As such, we need to be able to find a way to do this without projecting into too high of dimension. A metric that is used to classify the ability for a set number of neurons to classify concept classes is known as the classification capacity, though we omit the details as it is not relevant for our experiments.

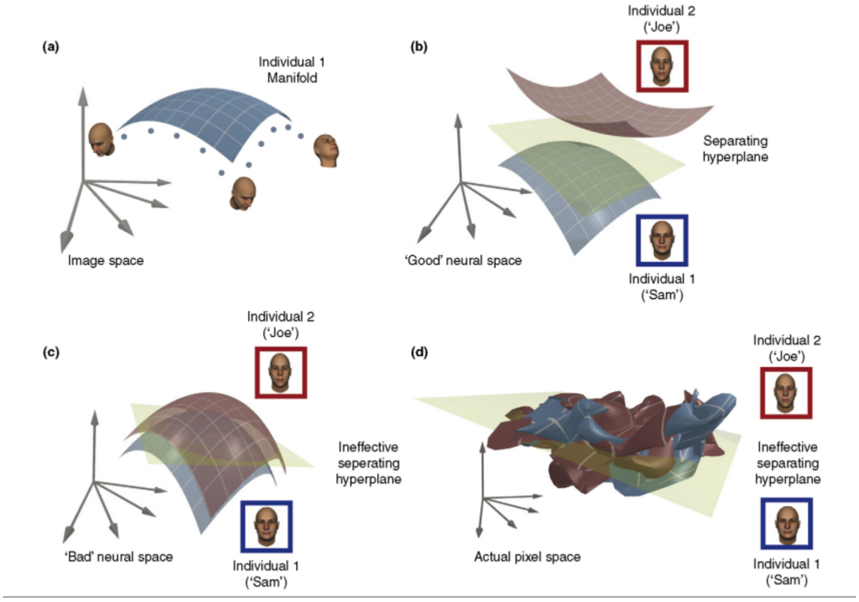


Figure 1: Manifolds for object recognition 1. a) Each individual has their own manifold within the image space (defined in our model architecture as RGB channels per pixel). Note that this manifold encompasses all possible objects in this concept class, mirroring perceptual invariance in object recognition tasks. b) The goal is to find a separating hyperplane between concepts that allow for classification between the two manifolds with low error. This figure shows a good neural space - one that seperates the two faces in such a way that we can find a good halfspace. c) On the other hand, this figure shows a bad representation, in which there is not a good way to separate the hyperplane. d) In the pixel space, we empirically see that many manifolds overlap with each other, and are not distinguishable by halfspaces.

2.2 Prototype Learning

Prototype learning has been posited as a very simple and feasible learning strategy for mammalian brains, as it only requires one downstream corticle neuron to adjust the weights for the hyperplane based on the centroids of certain manifolds. For few-shot learning, the idea is that given a set of P images that have never been seen before for concept classes a and b, we can calculate the centroid first as x_0^a and x_0^b, and find a halfspace $y = w \cdot x + \beta$ such that:

$$w = x_0^a - x_0^b$$
$$\beta = \frac{||x_0^a - x_0^b||^2}{2}$$

Another way to formulate this learning algorithm is that for some test sample ξ, classify a if and only if $||\xi - x_0^a||^2 < ||\xi - x_0^b||^2$. In biological terms, this learning rule can be implemented in the visual pathway by a downstream cortical neuron that adjusts based on the new images it has seen, using the w and β rules for new concept classes. This cortical neuron theoretically has access to all of the information of layers prior to it. With this formulation of prototype learning, the number of concept classes here can be arbitrary - for k concept classes C_k, we would determine the closest centroid x_0^k to sample test objects, and classify that object as C_k.

We focus on prototype learning in our experiments, though note that it is also possible to use SVM and Nearest Neighbors for this task as well (see future directions).

3 Methods

3.1 Models and Datasets

For our analysis, we use AlexNet, VGG11, and Resnet50. Dimensionality, mean squared radius, and other metrics have been measured in the original experiments carried out in [1], though these were carried out for Resnet50 only. We chose AlexNet as it is an architecture that is relatively shallow when compared to larger models like VGG11 and Resnet50 (in terms of convolutional layers). In practice, Alexnet also performs worse on average than both VGG11 and Resnet50. We hope to elucidate both classification accuracy and few-shot learning accuracy with the geometric manifold theory.

Regarding models, we took Pytorch's pretrained models and ran experiments with those. Each of the Pytorch's pretrained models are trained on the same set of images as shown here, allowing us to compare results across models with subsampling as explained in Methods 3.2. The images used to train are of 1000 categories, and located in the ILSVRC2012 training dataset. To be sure that we were few-shot learning correctly, we picked a couple of categories in ImageNet21K that were not a part of the training dataset, namely tulips, dandelions, bees, ants, roses, and sunflowers. The categories used can be found in the supplementals. As we were focused on few-shot learning with $m = 1$ to 25, we did not need very large datasets - as such, we used $P = 150$ for animal images and $P = 500$ for flower images.

3.2 Gathering Embeddings

To gather the embeddings, we removed the final connected layer from the classifier for each of the models and ran forward passes through batches of each concept class. In order to keep results comparable across models, we randomly subsampled $N = 2048$ neurons from each layer for AlexNet, and 7 random convolutional layers from VGG11 and Resnet50. The input layer was also subsampled, using the same neurons for each of the models. For Resnet50, 2048 is the size of the final layer so no subsampling on said layer was necessary, though intermediate layers were subsampled. Alexnet and VGG11 were subsampled at all layers, using the same subsampling per model for all the objects. We repeated the subsampling multiple times over to get results.

Layers were randomly subsampled such that deeper layers have a higher chance of being sampled. This is aligned with the findings of Chung et al. 2018, as later layers have much larger success in few-shot learning, and certain metrics such as mean squared radius and dimensionality were more pronounced during these layers. In accordance with the previous subsampling of neurons, we repeated these samplings multiple times and averaged.

The reason that we chose $N = 2048$ can be seen in [9], due to the fact that our N should be sufficiently larger than our values of P in order for the random subsampling will not have a large effect on the resulting dimensionality of the data. We could opt to go higher, and we attempted to do so, though we ran into memory issues trying to calculate the singular values, reasons for which we explain in detail below.

3.3 Calculating Metrics

Relevant metrics we calculate include mean squared radius, dimensionality, and signal. To calculate the mean squared radius of the manifold a (which we denote as R_a), we performed diagonalized the covariance matrix of any arbitrary manifold a, defined as:

$$C = \frac{1}{P} X_a X_a^T - \mu_a \mu_b^T$$

This was done via singular value decomposition. This lends itself to a good estimate of the mean distance of sampled points in the manifold to the centroid of the manifold, allowing us to perform conversions of distance metrics to dimensionless quantities. It is worth noting that the values that we extract from the covariance matrix can also be viewed as PCA on the concept space, lending itself to a nice theory of subspaces in which we can calculate dimensionality.

Dimensionality was measured by a metric called the participation ratio, mentioned in Gao et al. 2017 [4]. This metric estimates the number of dimensions that the point-cloud representation of a particular concept class a varies as a function of the singular values. The metric is defined per concept class as

$$D_a = \frac{(\sum_i R_{ia}^2)^2}{\sum_i R_{ia}^4}$$

As dimensionality decreases, we can expect the object recognition task to be easier, as there is less chance for overlap between modes [1] [2]. This also makes intuitive sense, as smaller dimension means that on average, there will be less dimensions that have overlap between differing concept classes that most likely would be due to non-overlapping features.

Lastly, we calculate the signal of two concept classes, defined as:

$$S_a = \frac{||\mu_a - \mu_b||^2}{R_a^2}$$

The signal represents the normalized distance between any two concepts, and is critical in how well a hyperplane can, on average, distinguish between objects from either class. The signal is assymetric in terms of the concept classes - if the R value of one concept class is much larger than another, the signal is much lower. An intuitive interpretation is the concept class with larger mean squared radius will be spread out further, with more points closer to the hyperplane than the concept class with a smaller mean squared radius.

3.4 Accuracy

We took the embeddings per model represented by a $L \times P \times N$ matrix, where P and N are as defined above, and L is the number of layers subsampled from the models and ran prototype learning, varying the number of test samples m fed forward, and tested accuracy with the remaining $P - m$ examples. We compared the performance of the models by benchmarking on two pairs of concept classes: Bees versus Ants, and Dandelions versus Roses. Instead of measuring total accuracy as the original paper did, we measure class accuracy:

$$\text{"Per Class" Classification Accuracy} = \frac{\text{Number Correctly Classified Examples from Class}}{\text{Total Number of Examples from Class}}.$$

We averaged results over 30 iterations to remove noise present in the data, and we did so for all models.

4 Results

Below are plots of selected metrics and accuracy of few-shot learning for novel concept classes as described in methods. We include more plots in the appendix.

4.1 Metrics

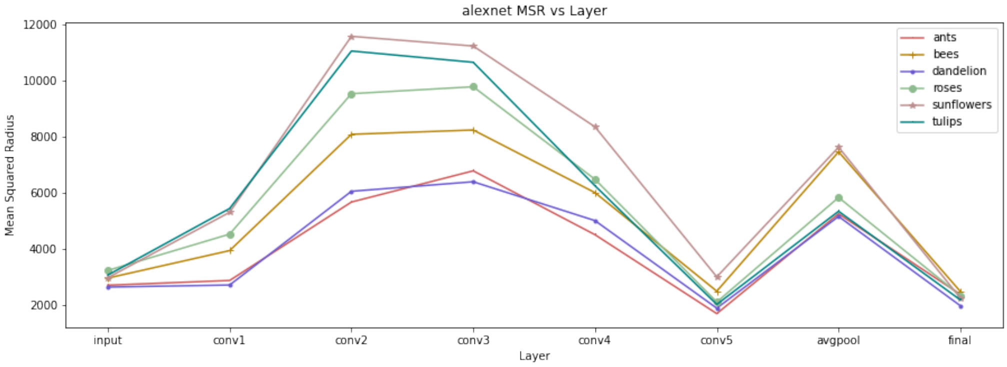


Figure 2: Mean Squared Radius for Alexnet

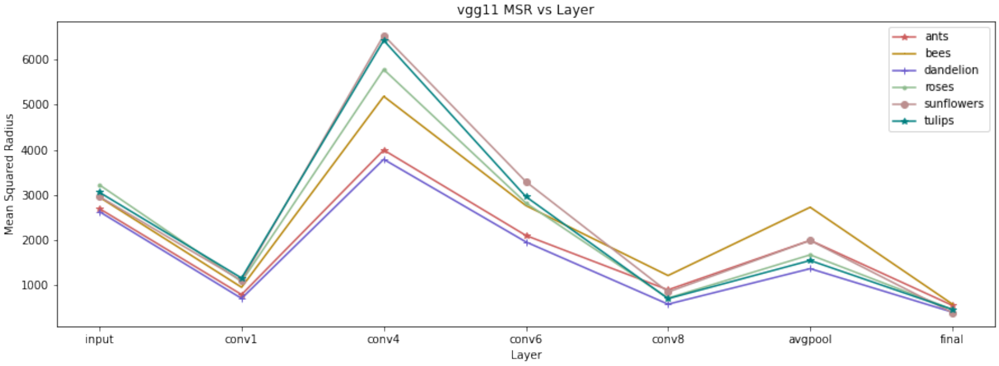


Figure 3: Mean Squared Radius for VGG11

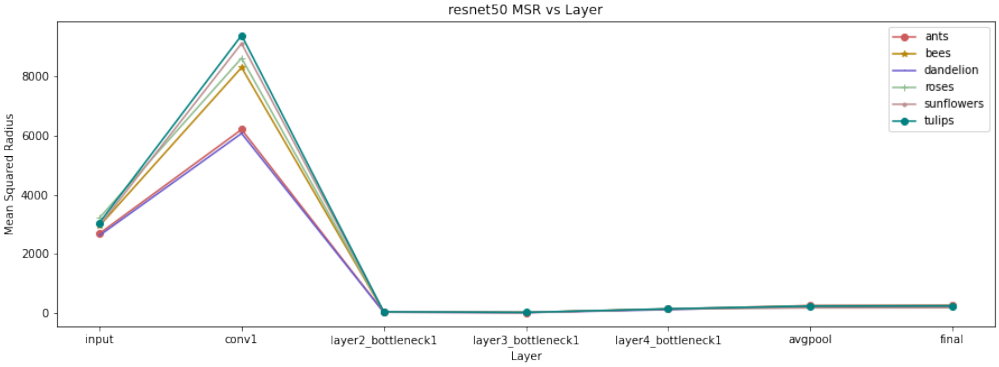


Figure 4: Mean Squared Radius for Resnet50

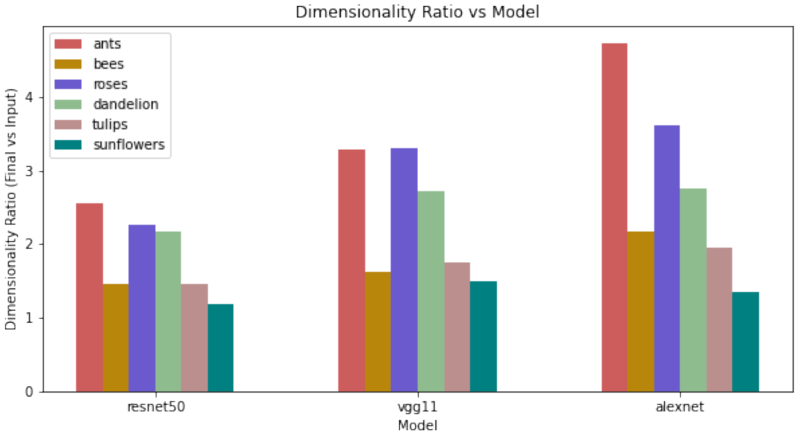


Figure 5: Dimensionality Ratio (Final Layer to Input Layer). Plotted here is the ratio of the final layer to the initial input layer of dimensionality. Note that shallower models to not compress dimensionality as well as deeper ones

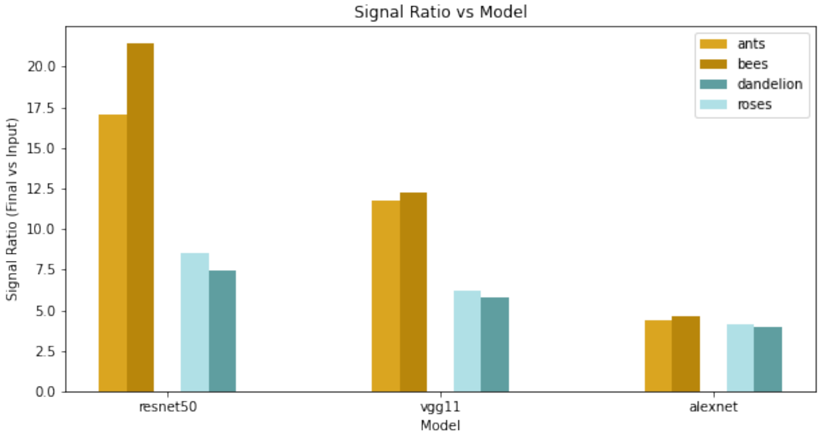


Figure 6: Signal Ratio (Final Layer to Input Layer). We plot here the signal ratio of final layer to input layer in regards to the pair of classes we test. Signal increase is directly correlated with the depth of the architecture.

4.2 Classification Accuracy

The plots below describe the changes in accuracy, per class for both ants and bees as well as total combined accuracy, as the number of training examples increases for different layers in different models. These plots support our hypothesis that few shot learning is possible using our framework of classifying based on layer readouts, albeit to varying degrees of accuracy depending on layer and model architecture.

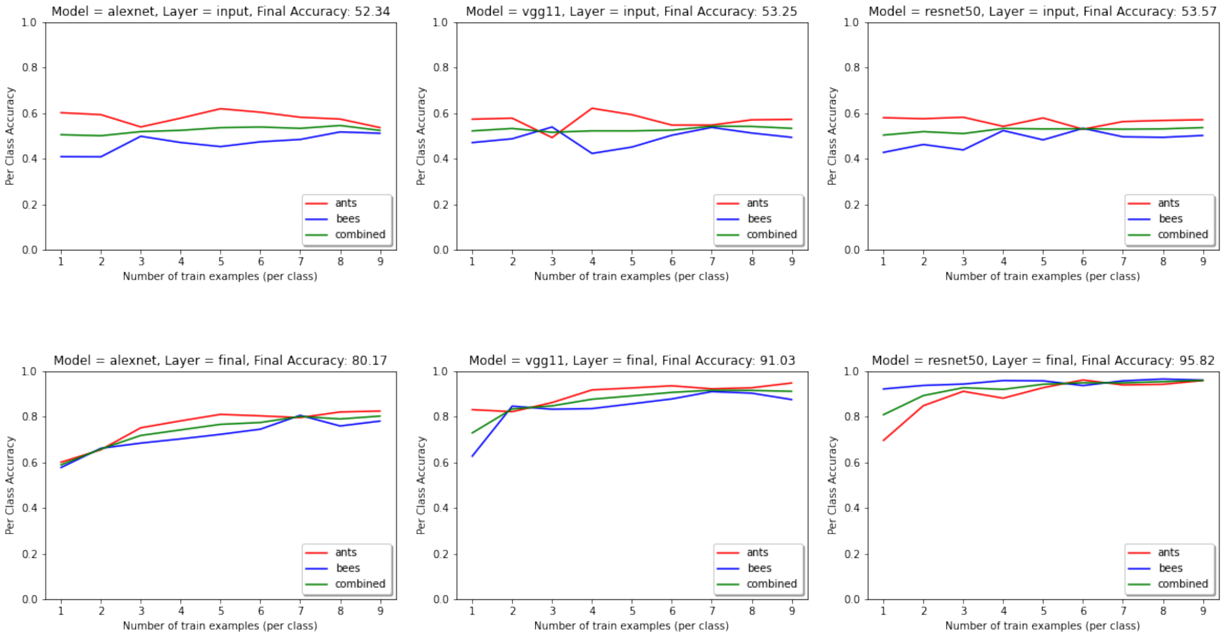


Figure 7: Classifying Ants vs Bees

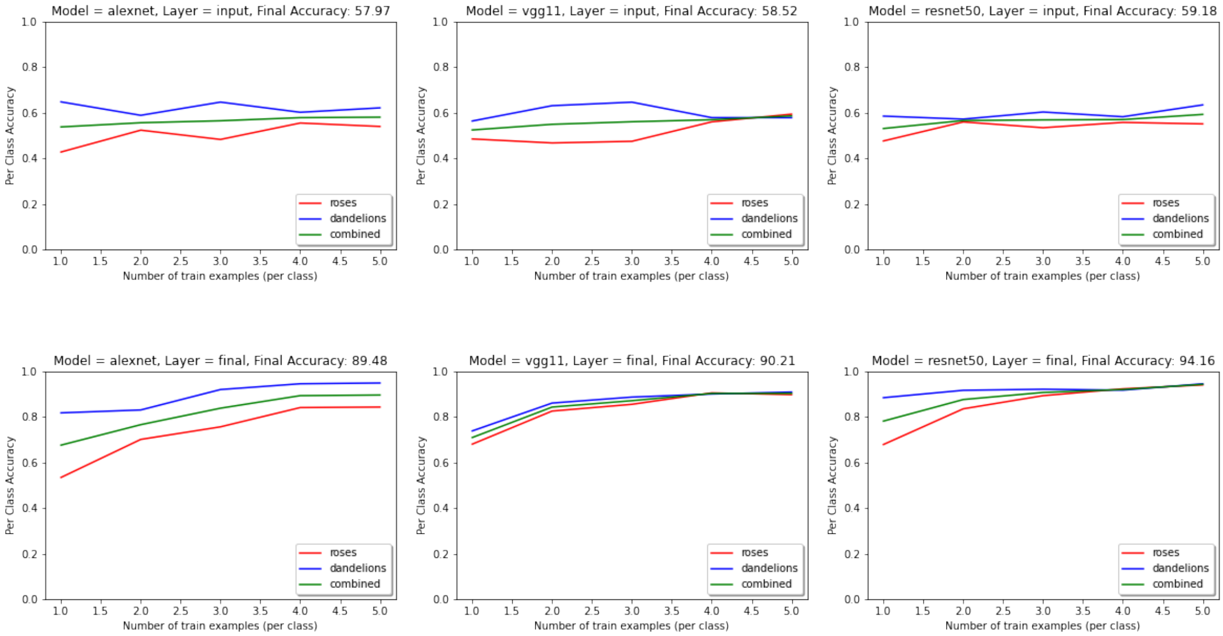


Figure 8: Classifying Roses vs Dandelions

5 Discussion

When experimenting, we first generated the metric plots, and then generated the classification accuracy plots, to verify our intuition guided by theory that as mean squared radius decreased for a given concept and layer, the classification accuracy from using that given layer would be higher. Figures 2-5 show that increase in mean squared radius and dimensionality is directly correlated with the depth of the network, which supports our hypothesis that depth of the network does in fact change these geometric manifold metrics. In addition, Figure 6 shows that the signal also increases as a result of increasing depth. This increase in signal is correlated in part with the decrease in dimensionality and mean squared radius as the manifolds are ”shrinking.” This increase in signal leads us to posit that the classification accuracy of the models should also be correlated to the depth of the architecture as a higher signal ratio should naturally correspond to higher classification accuracy. A higher signal means more separation between the manifolds - we see this as a function of depth.

Figure 7 and 8 shows the input layer versus output layer accuracy. Note that the accuracy of the input layer is the same amongst all of the models, as we subsample the same set of neurons per the input layer, which remains unchanged (perturbations due to random noise). However, the feature layer embeddings do reflect the hypothesis that we had made - Alexnet, in both cases, performs much worse than VGG11 and Resnet50, while Resnet50 tops VGG11. What is interesting to see is that there are some classes that perform better than others on average - for example, dandelions consistently outperforms roses, while ants consistently outperforms bees. Future work may include investigating this phenomenon further. We hypothesize that there may be correlation between the classification accuracy for a certain class with metrics like those we have discussed here describing the associated manifolds.

Sample complexity also plays a large factor. See that as we increase the sample complexity for the feature layer and intermediate layers (plots in the appendix), the accuracy also goes up. This matches our intuition, as with more samples, we are able to more accurately assess the true centroid of the manifold. Something more subtle is that as we traverse through the layers, the sample complexity matters in the sense that it impacts learning rate. For the input layer, no number of training examples can improve the training accuracy, though as we traverse through the convolutional neural network, even if the class accuracy is lower, the rate at which the accuracy increases in terms of sample accuracy increases. There is a direct corollary of this to the human brain is that perhaps more intricate layers added to the brain allow for faster learning. An interesting future direction of this finding could be to map the Macaque IT cortex and intermediate cortices, and see if the Macaque IT cortex also exhibits these characteristics.

6 Future Directions

6.1 Models

Future work can take a look at more models, and in particular, models that do not have good classification accuracy. We have focused on few-shot classification accuracy of decent model architectures, though the question remains to see if there are any extremely shallow architectures that perform poorly on object recognition, but perform well on few-shot learning classification. Theory posits that this is not the case, though it would be interesting to see in empirical results. In addition, the learning model can be extended to SVM and nearest neighbor classifiers. The experiments that we've ran here are easily extendable to these settings.

In addition, our paper deals with 2-way-K-shot classifiers, experimenting with different values for k, the number of training examples for each of the two concepts. In our future work, we can explore multi class few shot classifiers to see if our framework, applied to existing image classifiers, can effectively classify multiple concepts.

Furthermore, the classifier presented in this paper is derived form prototype theory, and we do not experiment with a classifier that uses exemplars to classify future examples. In the future, we can build a classifier maintains a record of previously seen training examples and classifies new images based on proximity to actual examples previously seen by the model. One goal may be to determine which theory yields the better classifier in practice for different models and datasets.

Although briefly mentioned, we do not experiment with zero shot learning. As alluded to in the introduction, one example zero shot concept learning in humans is the ability to visually identify new objects without having seen them before using language descriptions. A future area of work may be building a model that can leverage cross-modal transfer of verbal descriptions of images to classify concepts.

6.2 Model Similarities

Something of interest to us when experimenting with the signal ratio was that with certain classes, such as dandelion vs tulips in Figure 9 in the appendix, signal ratio changed uniformly across all architectures regardless of width. What this implies is that for certain classes, the features that the CNN learns is uniform, and furthermore there are certain features that all the CNNs are equally good, or equally bad, at distinguishing, regardless of width. We plan to explore this more in future experiments.

7 Bibliography

References

[1] Chung, S., Lee, D. D., and Sompolinsky, H., "Classification and Geometry of General Perceptual Manifolds", Physical Review X, vol. 8, no. 3, 2018. doi:10.1103/PhysRevX.8.031003.

[2] Cohen, Uri Chung, Sue Yeon Lee, Daniel Sompolinsky, Haim. (2019). Separability and Geometry of Object Manifolds in Deep Neural Networks. 10.1101/644658.

[3] Froudarakis, Emmanouil Cohen, Uri Diamantaki, Maria Walker, Edgar Reimer, Jacob Berens, Philipp Sompolinsky, Haim Tolias, Andreas. (2020). Object manifold geometry across the mouse cortical visual hierarchy. 10.1101/2020.08.20.258798.

[4] Gao, P., Trautmann, E., Yu, B., Santhanam, G., Ryu, S., Shenoy, K. Ganguli, S. A theory of multineuronal dimensionality, dynamics and measurement. bioRxiv, 214262 (2017).

[5] Li, G., Jampani, V., Sevilla-Lara, L., Sun, D., Kim, J., Kim, J. (2021). Adaptive Prototype Learning and Allocation for Few-Shot Segmentation. arXiv. https://doi.org/10.48550/ARXIV.2104.01893

[6] Majaj, Najib Hong, Ha Solomon, Ethan Dicarlo, James. (2015). Simple Learned Weighted Sums of Inferior Temporal Neuronal Firing Rates Accurately Predict Human Core Object Recognition Performance. Journal of Neuroscience. 35. 13402-13418.10.1523/JNEUROSCI.5181-14.2015.

[7] Quinn, Paul Eimas, Peter Tarr, Michael. (2001). Perceptual Categorization of Cat and Dog Silhouettes by 3- to 4-Month-Old Infants. Journal of experimental child psychology. 79. 78-94. 10.1006/jecp.2000.2609.

[8] Schmidhuber, Juergen. (2014). Deep Learning in Neural Networks: An Overview. Neural Networks. 61. 10.1016/j.neunet.2014.09.003.

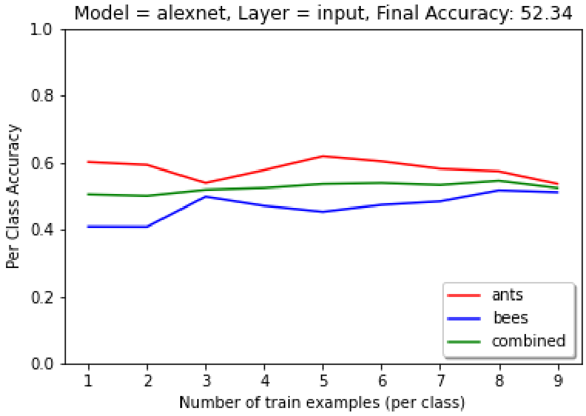
[9] Sorscher, Ben Ganguli, Surya Sompolinsky, Haim. (2021). The Geometry of Concept Learning. 10.1101/2021.03.21.436284.

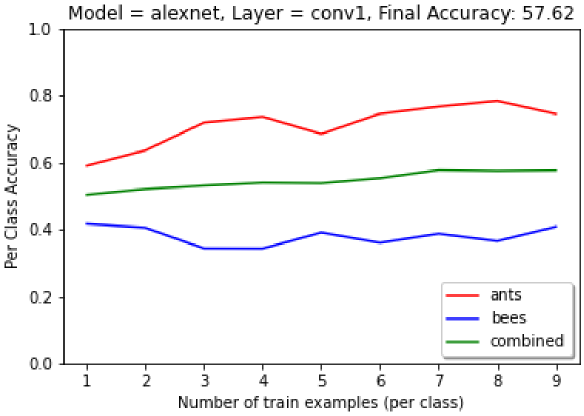
[10] Turing, A.M. I.—COMPUTING MACHINERY AND INTELLIGENCE, Mind, Volume LIX, Issue 236, October 1950, Pages 433–460, https://doi.org/10.1093/mind/LIX.236.433

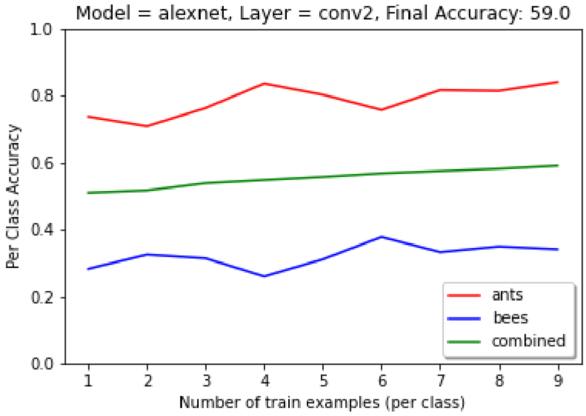
[11] Wang, Yaqing Yao, Quanming Kwok, James Ni, Lionel. (2020). Generalizing from a Few Examples: A Survey on Few-shot Learning. ACM Computing Surveys. 53. 1-34. 10.1145/3386252.

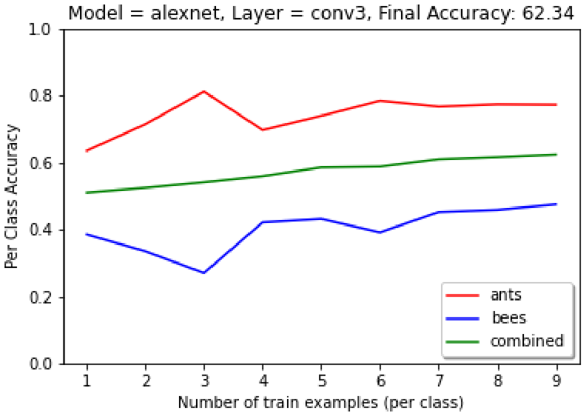
8 Appendix

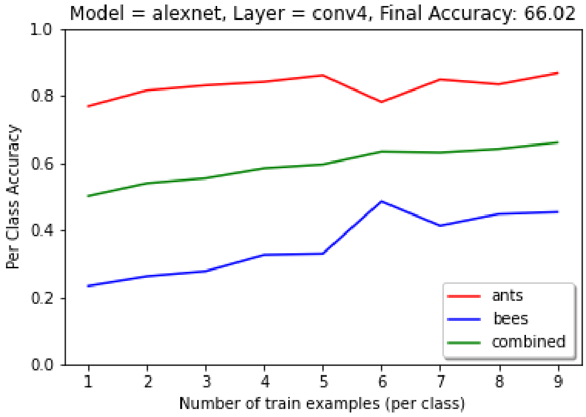
8.1 Ants vs Bees

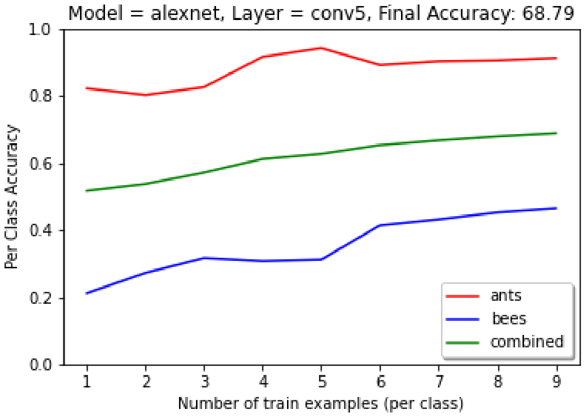


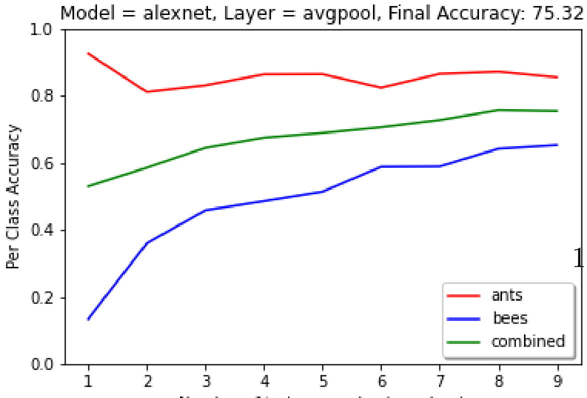


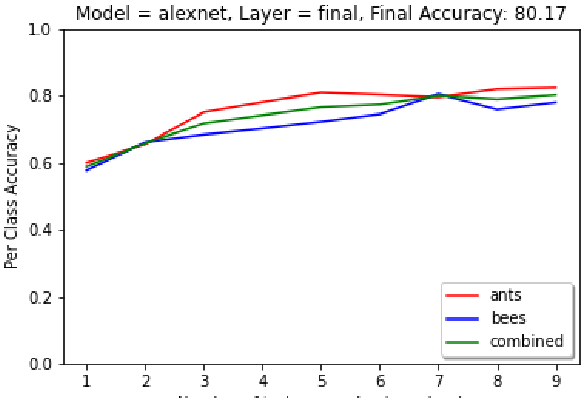


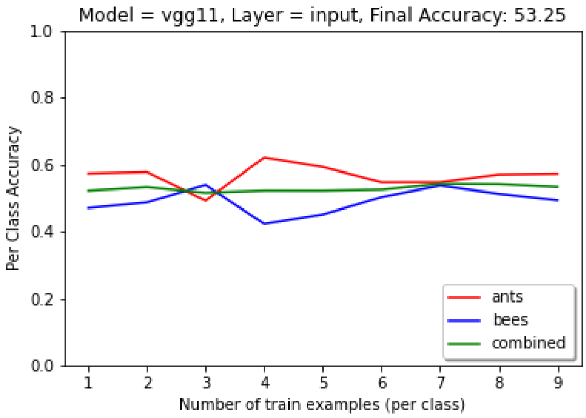


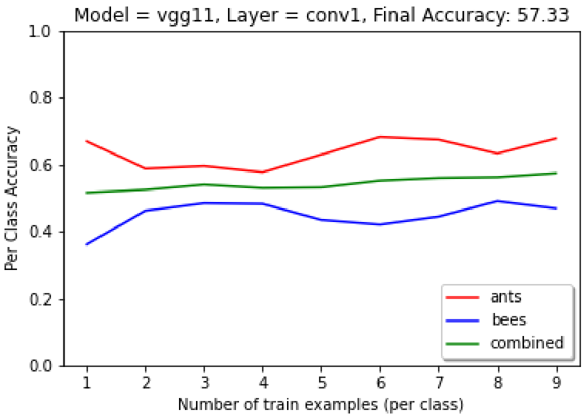


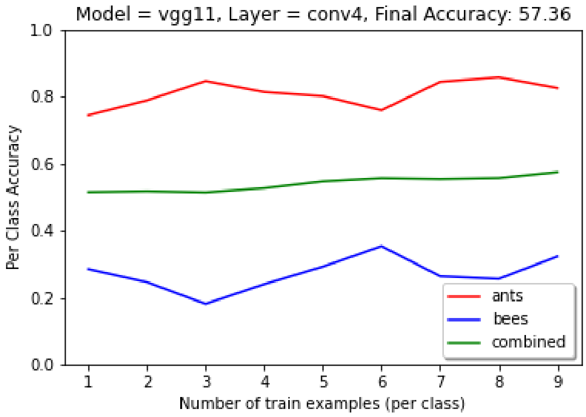


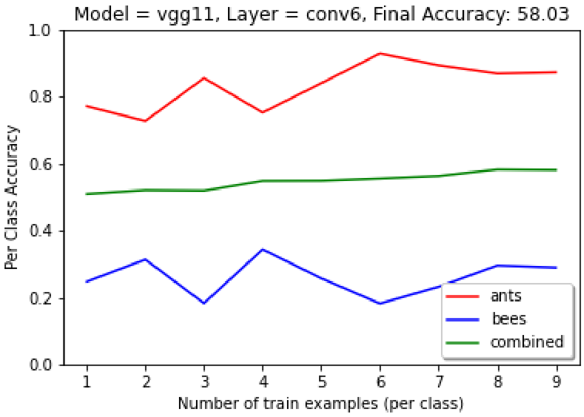


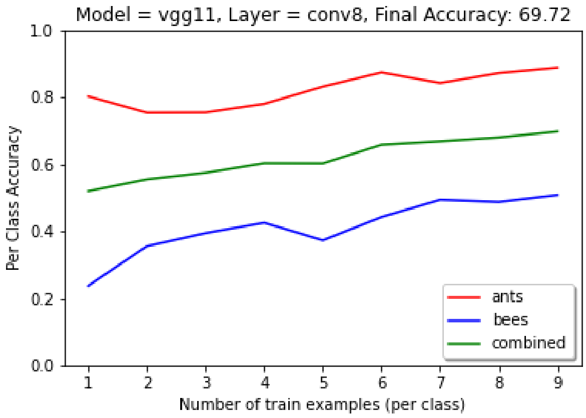


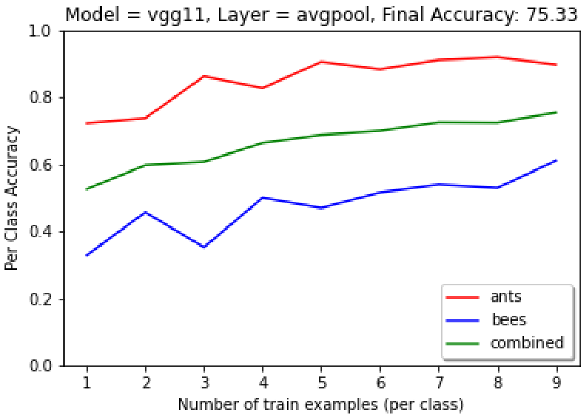

Model = vgg11, Layer = input, Final Accuracy: 53.25
Per Class Accuracy
Number of train examples (per class)
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
7
8
9
ants
bees
combined

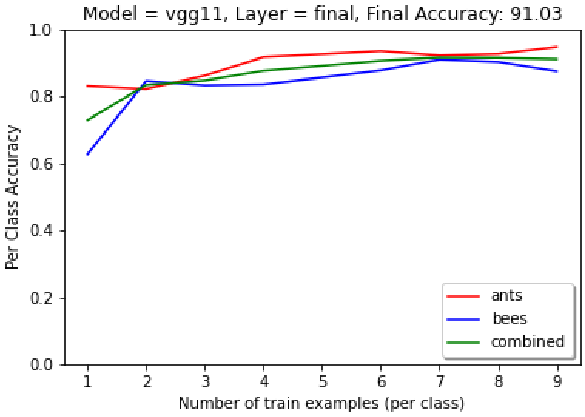

Model = vgg11, Layer = conv1, Final Accuracy: 57.33
Per Class Accuracy
Number of train examples (per class)
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
7
8
9
ants
bees
combined


Model = vgg11, Layer = conv4, Final Accuracy: 57.36
Per Class Accuracy
Number of train examples (per class)
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
7
8
9
ants
bees
combined

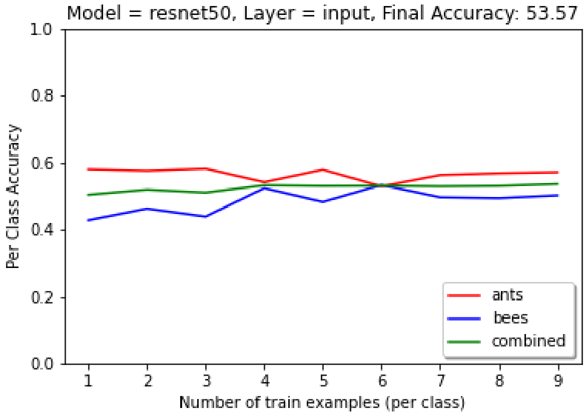

Model = vgg11, Layer = conv6, Final Accuracy: 58.03
Per Class Accuracy
Number of train examples (per class)
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
7
8
9
ants
bees
combined


Model = vgg11, Layer = conv8, Final Accuracy: 69.72
Per Class Accuracy
Number of train examples (per class)
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
7
8
9
ants
bees
combined

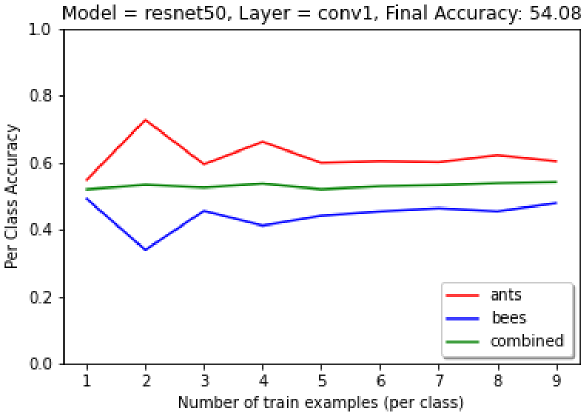

Model = vgg11, Layer = avgpool, Final Accuracy: 75.33
Per Class Accuracy
Number of train examples (per class)
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
7
8
9
ants
bees
combined


Model = vgg11, Layer = final, Final Accuracy: 91.03
Per Class Accuracy
Number of train examples (per class)
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
7
8
9
ants
bees
combined

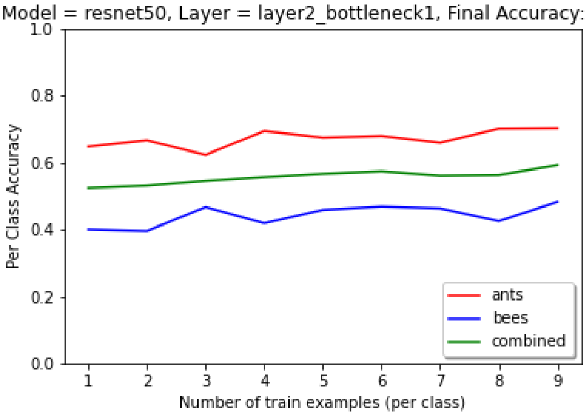
Model = resnet50, Layer = input, Final Accuracy: 53.57



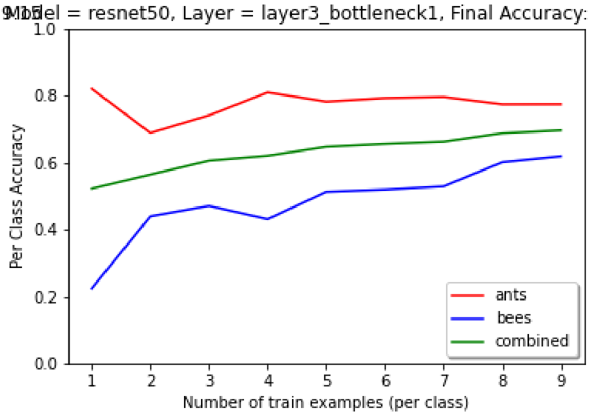
Model = resnet50, Layer = conv1, Final Accuracy: 54.08



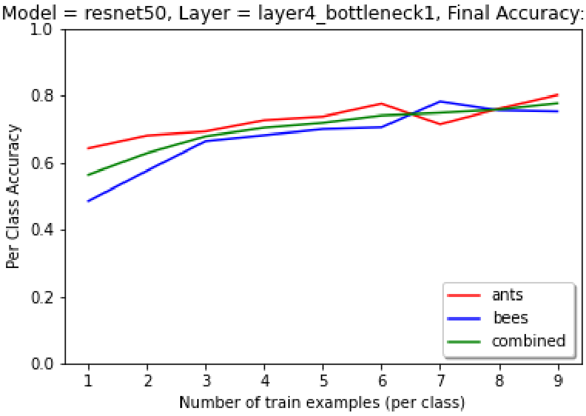
Model = resnet50, Layer = layer2_bottleneck1, Final Accuracy: 59.15



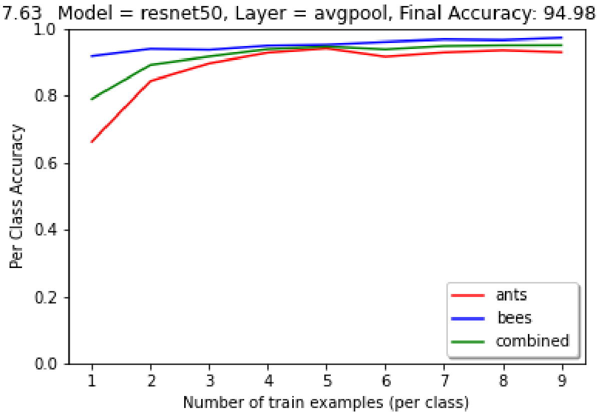
Model = resnet50, Layer = layer3_bottleneck1, Final Accuracy: 69.57



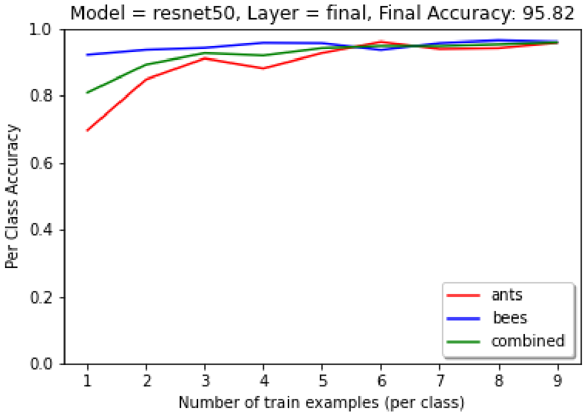
Model = resnet50, Layer = layer4_bottleneck1, Final Accuracy: 77.63



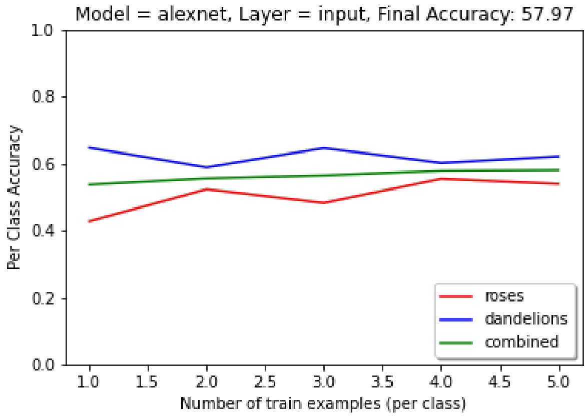
Model = resnet50, Layer = avgpool, Final Accuracy: 94.98

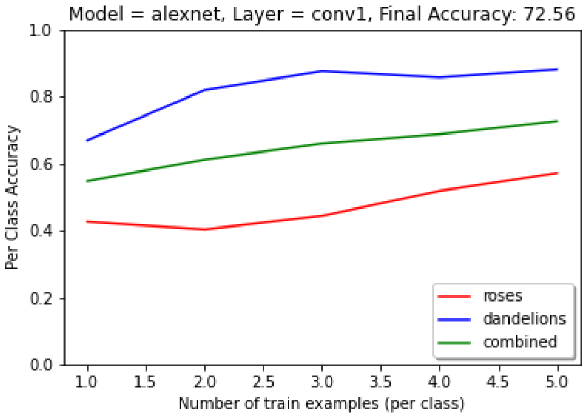


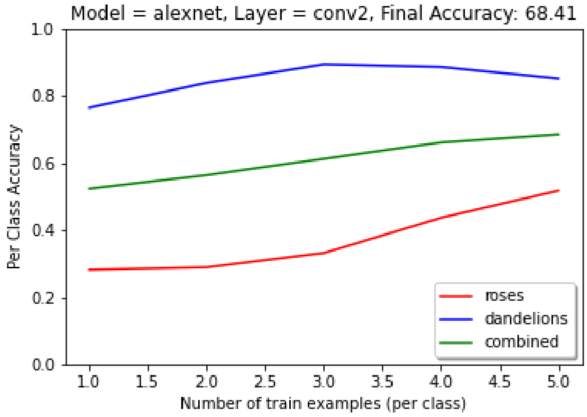
Model = resnet50, Layer = final, Final Accuracy: 95.82

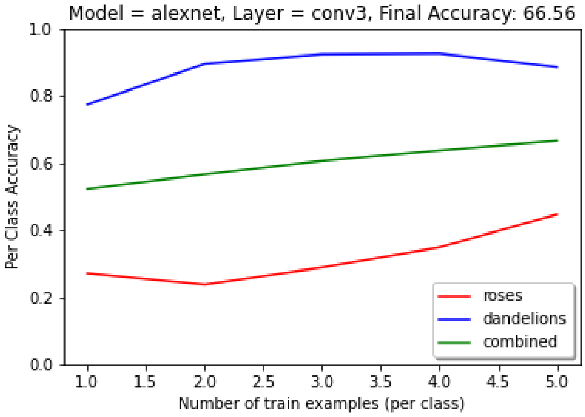


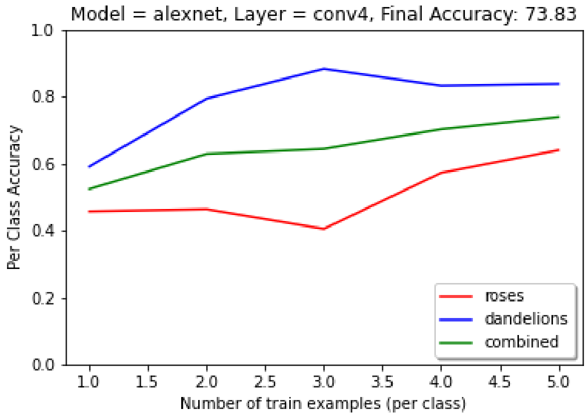
8.2 Roses vs Dandelions

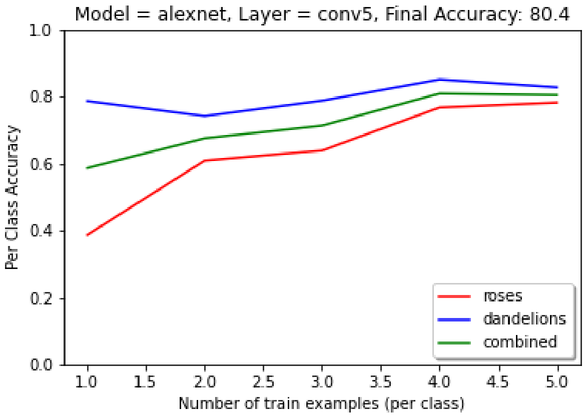


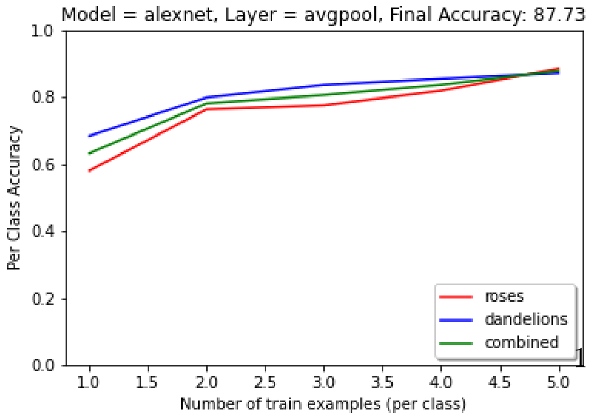


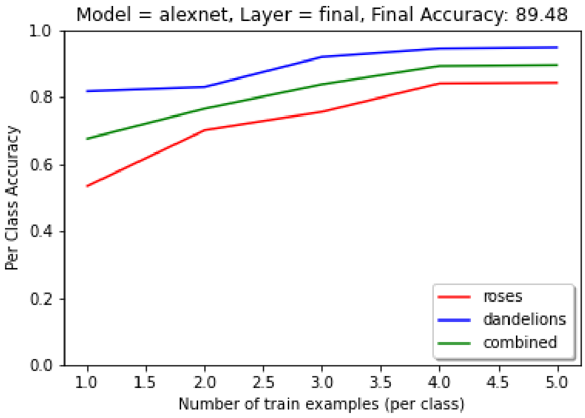


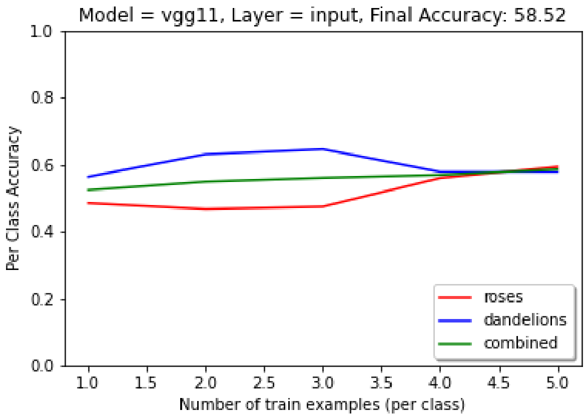


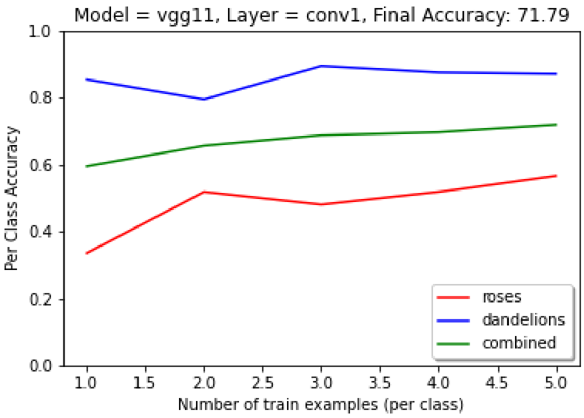


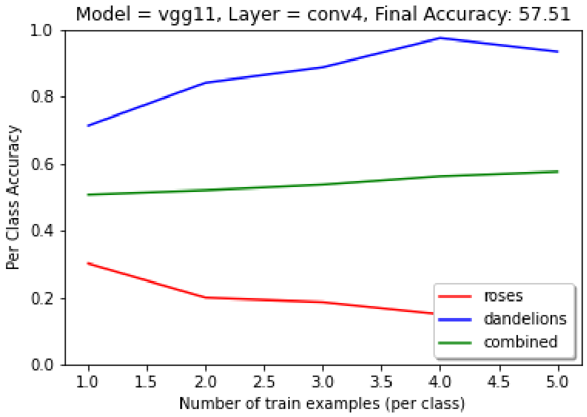


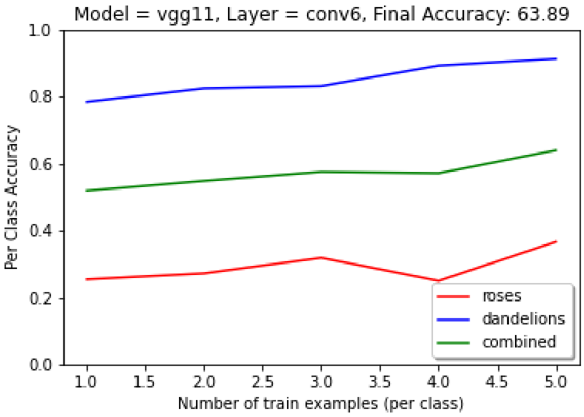


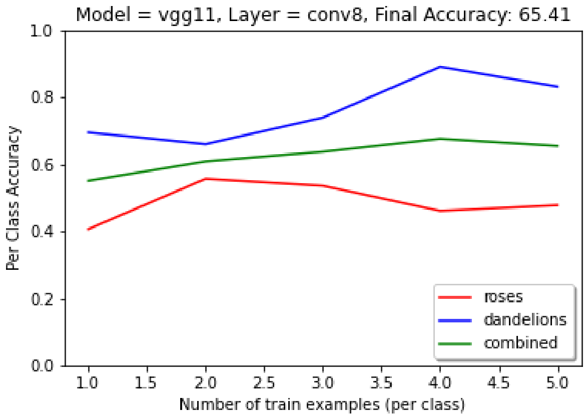


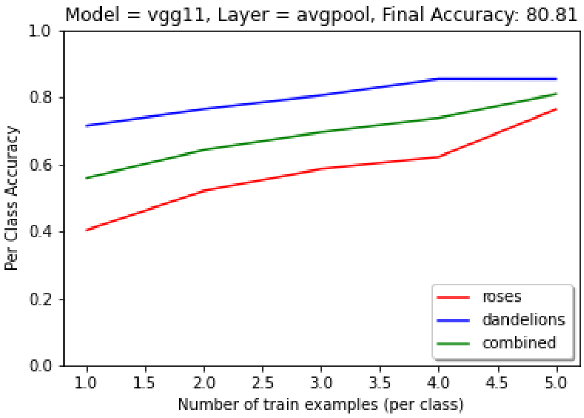

Model = vgg11, Layer = input, Final Accuracy: 58.52

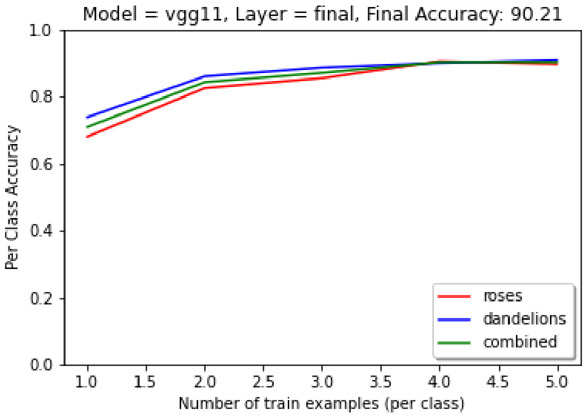

Model = vgg11, Layer = conv1, Final Accuracy: 71.79

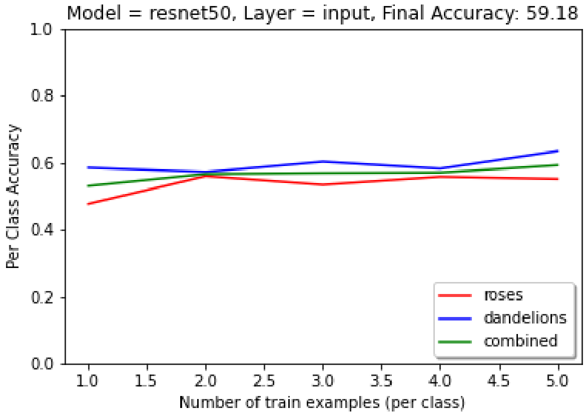

Model = vgg11, Layer = conv4, Final Accuracy: 57.51

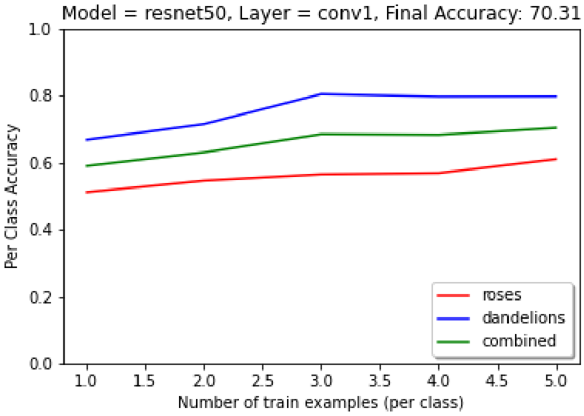

Model = vgg11, Layer = conv6, Final Accuracy: 63.89

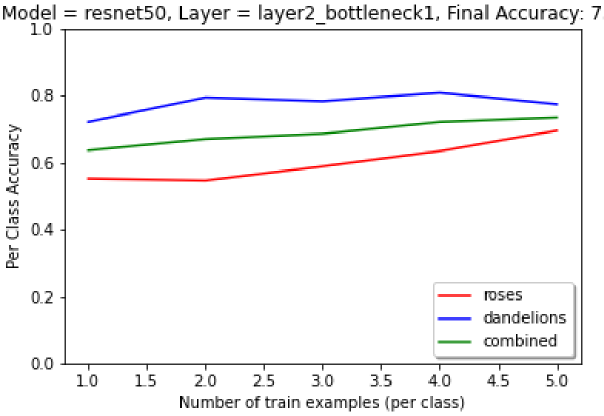

Model = vgg11, Layer = conv8, Final Accuracy: 65.41

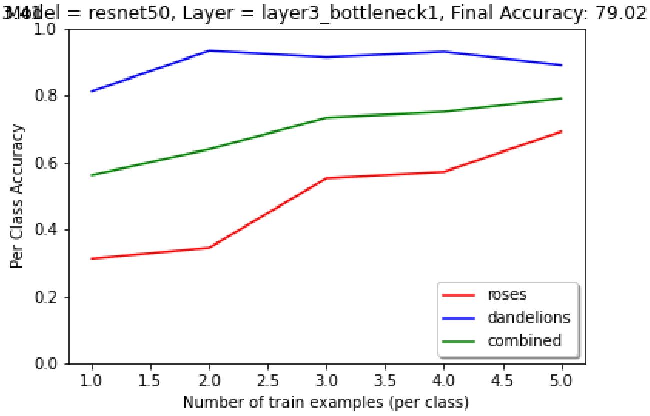

Model = vgg11, Layer = avgpool, Final Accuracy: 80.81

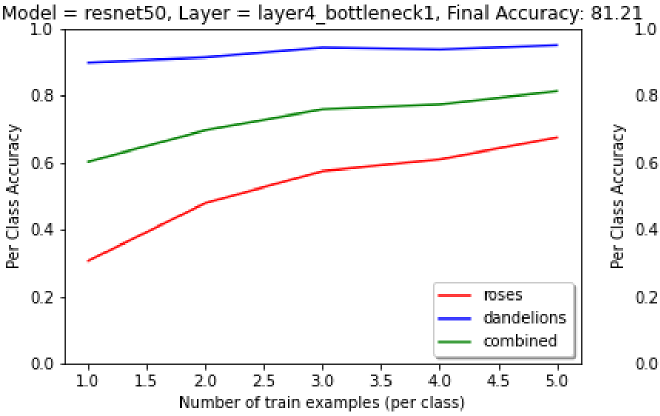

Model = vgg11, Layer = final, Final Accuracy: 90.21

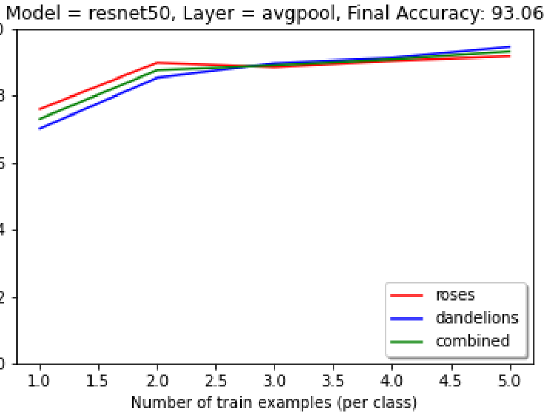

Model = resnet50, Layer = input, Final Accuracy: 59.18
Per Class Accuracy
0.0
0.2
0.4
0.6
0.8
1.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Number of train examples (per class)
roses
dandelions
combined

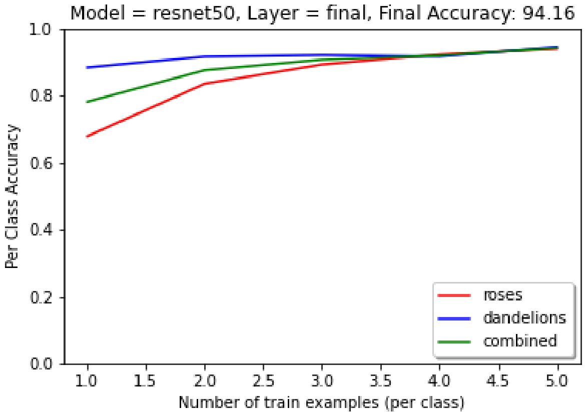

Model = resnet50, Layer = conv1, Final Accuracy: 70.31
Per Class Accuracy
0.0
0.2
0.4
0.6
0.8
1.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Number of train examples (per class)
roses
dandelions
combined


Model = resnet50, Layer = layer2_bottleneck1, Final Accuracy: 73.41
Per Class Accuracy
0.0
0.2
0.4
0.6
0.8
1.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Number of train examples (per class)
roses
dandelions
combined


Model = resnet50, Layer = layer3_bottleneck1, Final Accuracy: 79.02
Per Class Accuracy
0.0
0.2
0.4
0.6
0.8
1.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Number of train examples (per class)
roses
dandelions
combined


Model = resnet50, Layer = layer4_bottleneck1, Final Accuracy: 81.21
Per Class Accuracy
0.0
0.2
0.4
0.6
0.8
1.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Number of train examples (per class)
roses
dandelions
combined


Model = resnet50, Layer = avgpool, Final Accuracy: 93.06
Per Class Accuracy
0.0
0.2
0.4
0.6
0.8
1.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Number of train examples (per class)
roses
dandelions
combined


Model = resnet50, Layer = final, Final Accuracy: 94.16
Per Class Accuracy
0.0
0.2
0.4
0.6
0.8
1.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Number of train examples (per class)
roses
dandelions
combined

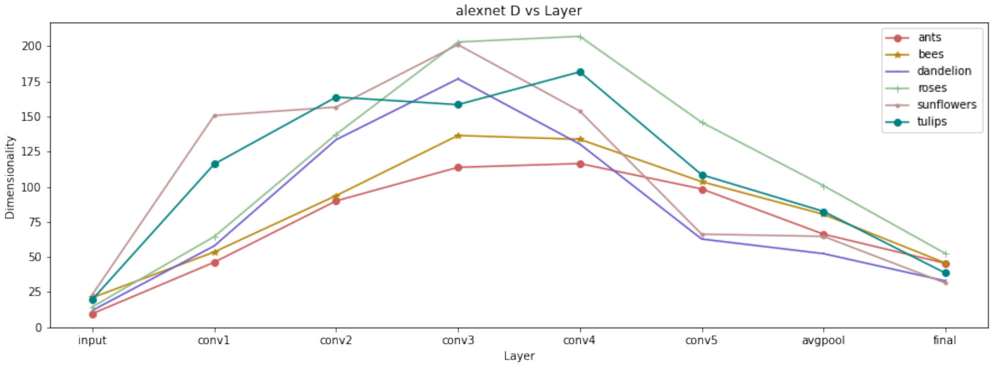


Figure 9: Dimensionality across Layers, Alexnet

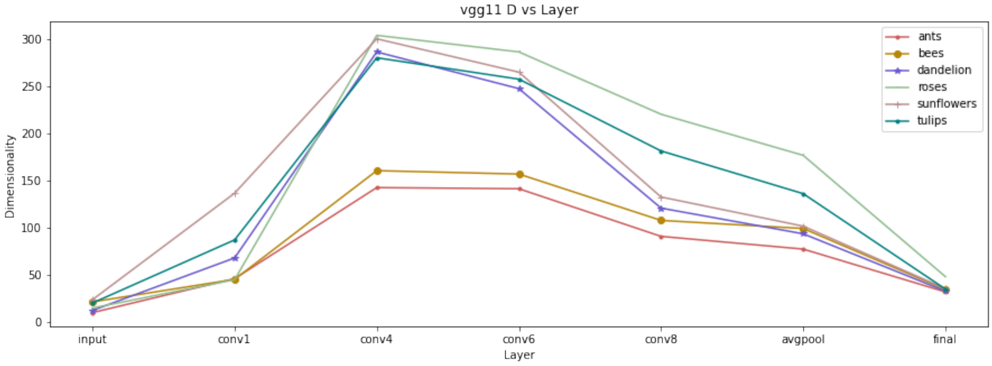


Figure 10: Dimensionality across Layers, VGG11

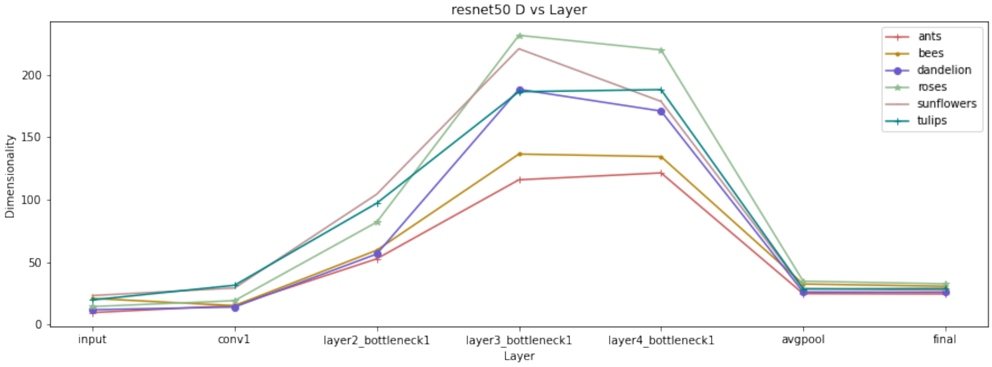


Figure 11: Dimensionality across Layers, Resnet50

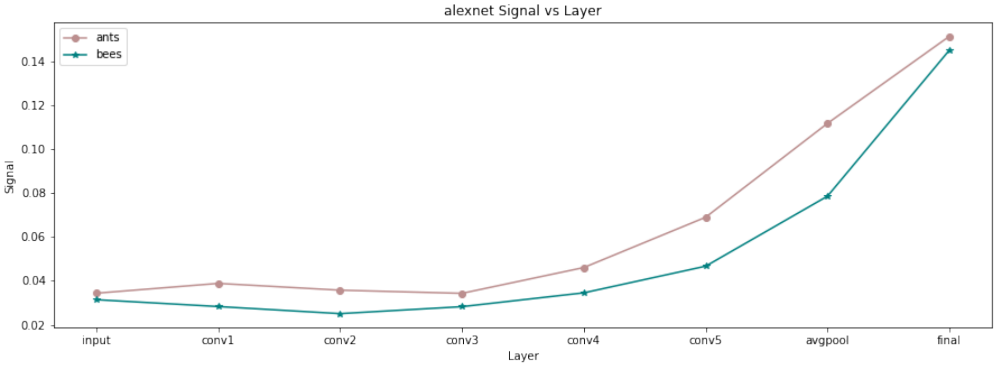


Figure 12: AlexNet Signal vs Depth, Ants and Bees

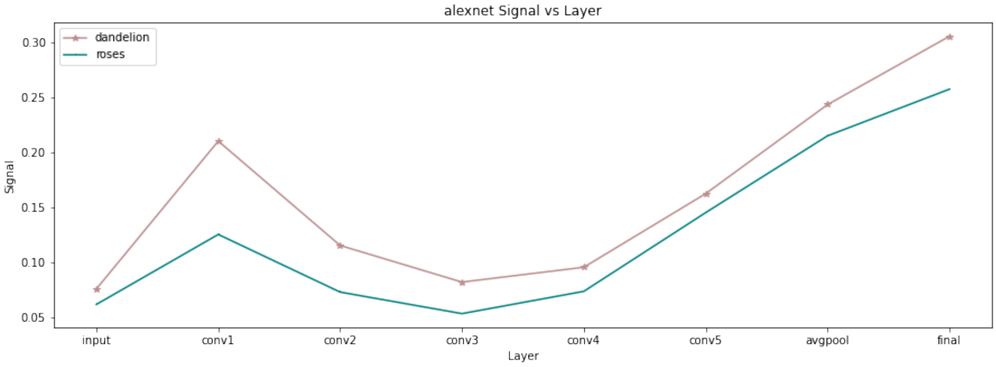


Figure 13: AlexNet Signal vs Depth, Dandelion and Roses

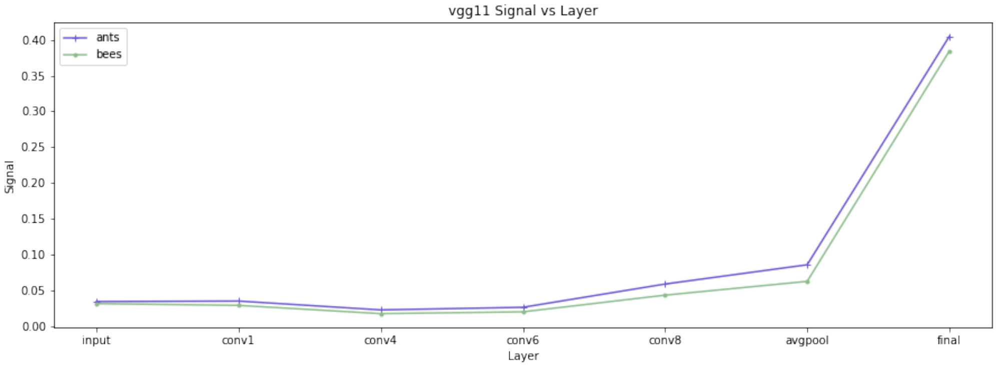


Figure 14: VGG11 Signal vs Depth, Ants and Bees

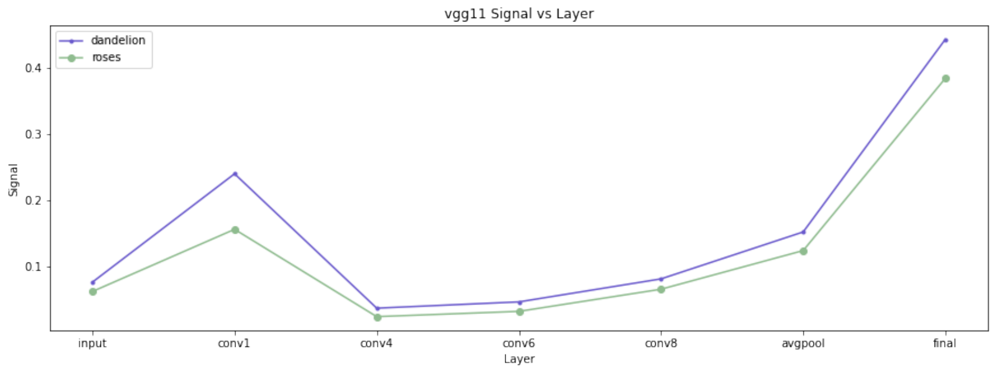


Figure 15: VGG11 Signal vs Depth, Dandelion and Roses

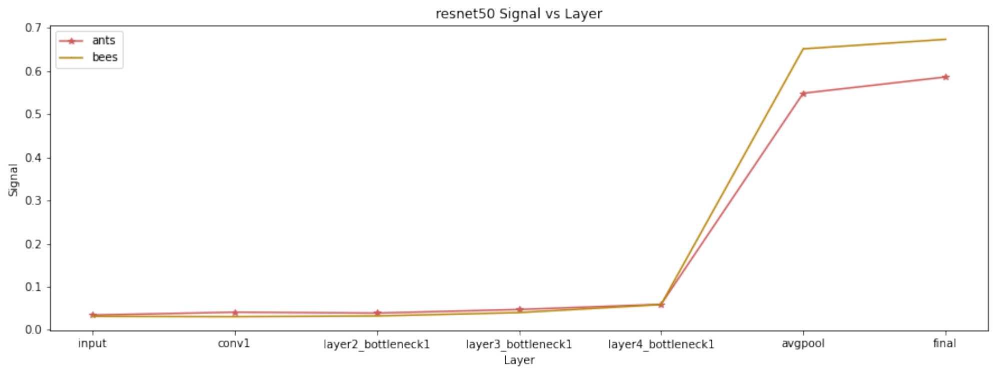


Figure 16: Resnet50 Signal vs Depth, Ants and Bees

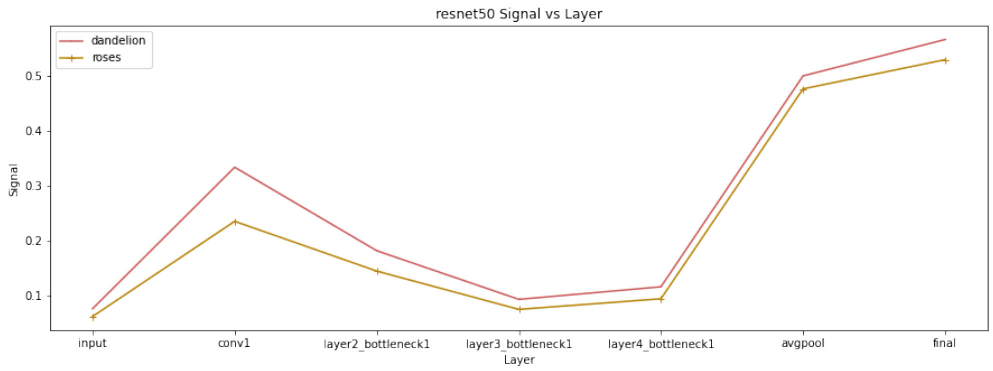


Figure 17: Resnet50 Signal vs Depth, Dandelion and Roses

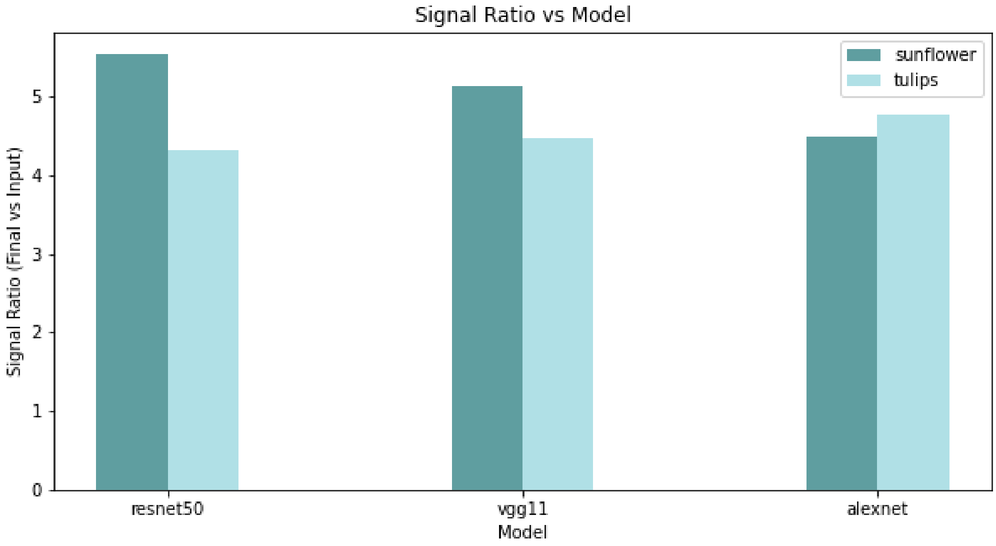


Figure 18: Sunflower Tulips Signal Ratio. For this particular pair of classes, the signal ratio changes the same

Supplementary Files

All code can be found in our Github repository linked here. Within the repository is a list of categories we used, the data we collected, and plotting code.