AlexNet's Perception of Lines and the Poggendorff Illusion

by

Matthew Nyhus

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

Bachelor of Science with Honours

in

THE FACULTY OF SCIENCE

(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

April 2018

Abstract

This thesis examines the limitations of modeling the brain's visual processing system with hierarchical convolutional neural networks (HCNNs). I approach this problem through studying AlexNet's perception of the Poggendorff illusion and, more broadly, its ability to differentiate lines from non-lines. AlexNet is a landmark hierarchical convolutional neural network (HCNN) that performs object classification with very low error [13, 12]. AlexNet has also been shown to model hierarchical neural responses in the ventral stream [4, 14], the area of the brain used for object recognition. The Poggendorff illusion is a distortion illusion that makes aligned line segments appear offset; that is, it causes a disconnected line to be interpreted as a non-line

I retrain AlexNet on new images containing both lines and non-lines with different background elements. I find that AlexNet is capable of differentiating connected lines from non-lines, but only consistently in the simplest case. Increasing the number of background lines steadily decreases AlexNet's performance, and moving to disconnected lines greatly decreased its performance. AlexNet's classification ability improves when using consistently placed vertical lines, but not enough to observe the Poggendorff effect consistently. Overall, AlexNet's susceptibility to the Poggendorff illusion is made irrelevant by the network's inability to accurately differentiate disconnected lines from non-lines, a prerequisite for experiencing the effects of the Poggendorff illusion.

I propose that AlexNet's consistently worse performance on disconnected lines is due to the lack of local differentiating information, indicating AlexNet's difficulty using global information for image classification. This is surprising given that humans use global information over local information when performing object classification [16, 1]. These results indicate that AlexNet cannot fully utilize global information like humans, and thus that AlexNet's model of the brain's visual processing system is more limited than previously thought. Furthermore, AlexNet's difficulty differentiating connected lines from non-lines in the presence of many background lines, despite their prevalence in real images, indicates a more fundamental limitation of AlexNet's capabilities for even simple abstract tasks. Together, these results indicate that current HCNNs are being trained for tasks that are too narrow and concrete to accurately model the general-purpose human visual system.

Preface

This thesis is the result of numerous discussions with, and direction from, Patrick Laflamme. The neural network used throughout this work, AlexNet, was published by Krizhevsky *et al.* at *NIPS 2012* [15]. The experiments for this thesis were run on Michael Guerzhoy's tensorflow implementation of AlexNet [11] with some modifications [17]. All the algorithms, analysis scripts, and plots are implemented in python.

Contents

Abstract i

Preface ii

Contents iii

List of Figures v

List of Tables vii

1 Introduction 1
- 1.1 Motivation . . . 1
  - 1.1.1 Hierarchical Convolutional Neural Networks . . . 1
  - 1.1.2 Visual Illusions . . . 1
- 1.2 The Poggendorff illusion . . . 2
- 1.3 AlexNet . . . 2
- 1.4 Overview of experiments . . . 4

2 Exploratory Research 6
- 2.1 Classification versus interpretation . . . 6
- 2.2 Cosine similarity . . . 6
- 2.3 Exploratory research methodology . . . 6
  - 2.3.1 Vertical line example . . . 6
  - 2.3.2 Test sets to be studied . . . 7
- 2.4 Results of exploratory research . . . 10
  - 2.4.1 Simple set . . . 10
  - 2.4.2 Centered set . . . 12
  - 2.4.3 Offset set . . . 12
  - 2.4.4 Permutations of offset set . . . 14
- 2.5 Discussion of results . . . 17

3 Classifying lines and non-lines 18
- 3.1 Data Generation . . . 18
  - 3.1.1 Generating lines and non-lines . . . 18
  - 3.1.2 Generating an image . . . 22
  - 3.1.3 Generating a data set . . . 25
- 3.2 Retraining AlexNet . . . 25
  - 3.2.1 Training methodology . . . 25
  - 3.2.2 Notes on hyperparameter abbreviations and errors . . . 25
  - 3.2.3 Layers to be retrained . . . 27
  - 3.2.4 Final layer choice and loss function . . . 29
  - 3.2.5 Learning rate . . . 30

4 Results 35
- 4.1 Experiments with large joint angles . . . 35
- 4.2 Experiments with small joint angles and no background elements . . . 37
- 4.3 Experiments with small joint angles and gamma background element distributions . . . 37
- 4.4 Experiments with small joint angles and different background element distributions . . . 38
- 4.5 Classifying Poggendorff images with retrained networks . . . 41
  - 4.5.1 Experiments with networks retrained with images containing vertical background lines 41
  - 4.5.2 Experiments with networks retrained with images containing two background lines . . 44

4.6 Summary and discussion of results 48

5 Conclusion **52**

A Appendix **54**
A.1 Graph of raw data for permutations of offset set 54
A.2 Graphs for learning rate comparison 55
A.3 Graphs for experiments with large joint angles 57
A.4 Graphs for experiments with small joint angles and no background elements 59
A.5 Graphs for experiments with small joint angles and gamma background element distributions 60
A.6 Graphs for experiments with small joint angles and different background element distributions 63

Glossary **66**

References **70**

List of Figures

1.1 The Poggendorff Illusion . . . 2
1.2 The Hering Illusion . . . 3
1.3 AlexNet architecture . . . 4
2.1 Vertical spread and control images . . . 7
2.2 Cosine similarity graphs for vertical lines . . . 8
2.3 Spread and control images for the simple set. . . . 9
2.4 Spread and control images for the centered set. . . . 9
2.5 Spread and control images for the offset set. . . . 10
2.6 Results for the simple set . . . 11
2.7 Results for the centered set . . . 13
2.8 Results for the offset set . . . 14
2.9 Cosine similarity of permutations at fc8 layer, centered . . . 16
3.1 Non-line example, labeled . . . 19
3.2 Line length gamma distribution . . . 19
3.3 Joint angle gamma distributions . . . 20
3.4 Joint angle transformation . . . 20
3.5 Background line gamma distributions . . . 22
3.6 Vertical background elements . . . 23
3.7 Error type comparisons . . . 27
3.8 Classification, loss and raw error example . . . 27
3.9 Graphical comparison of errors when not using, and using, dropout . . . 29
3.10 Graphical comparison of different final layers . . . 31
3.11 Sigmoid loss error graphs . . . 32
3.12 Graphical comparison of errors when not dropping, and dropping, the learning rate . . . 33
4.1 Comparison of local information in connected and disconnected images . . . 36
4.2 Comparison of error rates across background distributions with different means . . . 39
4.3 Examples of problematic images for $bg = const$ and $bg_{colour} = 0$. . . . 40
4.4 Comparison of local information for disconnected elements with vertical background lines . . . 41
4.5 Joint angle by index . . . 43
4.6 Classification graph of the connected set evaluated with the vertical connected network . . . 44
4.7 Classification graph of the disconnected set evaluated with the vertical disconnected network 45
4.8 Classification graph of the disconnected set evaluated with the vertical connected network . . . 45
4.9 Classification graph of the connected set evaluated with the *random connected network* . . . 46
4.10 Classification graph of the disconnected set evaluated with the *random disconnected network* 47
4.11 Classification graph of the disconnected set evaluated with the *random connected network* . . . 47

4.12 Significant test images from the experiment in Fig. 4.11 48
4.13 Classification graph of a modified disconnected set evaluated with the *random disconnected network* 49
A.1 Cosine similarity of offset set permutations at the fc8 layer 54
A.2 Graphical learning rate comparison with training down to $layer = fc7$ 55
A.3 Graphical learning rate comparison with training down to $layer = fc6$ 56
A.4 Graphs of experiments with large joint angles with connected lines 57
A.5 Graphs of experiments with large joint angles with disconnected lines 58
A.6 Graphs for experiments with small joint angles and no background elements 59
A.7 Graphs for experiments with small joint angles and gamma background element distributions with $k = 3$ 60
A.8 Graphs for experiments with small joint angles and gamma background element distributions with $k = 5$ 61
A.9 Graphs for experiments with small joint angles and gamma background element distributions with $k = 9$ 62
A.10 Graphs for experiments with small joint angles and vertical background lines at 75% opacity 63
A.11 Graphs for experiments with small joint angles and black vertical background lines 64
A.12 Graphs for experiments with small joint angles and 2 black background elements 65

List of Tables

2.1 Permutation spread and control images 15
3.1 Image set samples with $ja_k = 6$, $ja_\theta = 0.75$ 23
3.2 Image set samples with $ja_k = 6$, $ja_\theta = 0.4$ 24
3.3 Comparison of errors when not using, and using, dropout 29
3.4 Comparison of different final layers 30
3.5 Learning rate comparison 32
3.6 Comparison of errors when not dropping, and dropping, the learning rate while training . . . 34
4.1 Results of experiments with large joint angles 36
4.2 Results with small joint angles and no background elements. 37
4.3 Results with small joint angles and gamma background element distributions 38
4.4 Results with small joint angles and different background element distributions 42

1 Introduction

1.1 Motivation

1.1.1 Hierarchical Convolutional Neural Networks

Recent developments in machine learning and artificial intelligence have been widely praised for their broad applicability and impressive results. For a variety of problems, traditional algorithms that had yielded only marginal improvements have given way to tremendous advancement by shifting the approach onto training algorithms with a lot of data. An important piece of these new techniques are neural networks. Based loosely on primates' visual systems, these networks are now able to regularly outperform humans in object recognition tasks that had seemed intractable [18].

However, these networks do not merely approximate the output of the brain's visual processing system, but also approximate some components of its internal processing steps. Work in this area has shown evidence of the brain recognizing objects by using a hierarchical series of cortical areas know as the ventral visual stream [7, 21]. The brain's final classification relies on a high-level representation stored in the final layer of this stream, the inferior temporal cortex [14].

Hierarchical convolutional neural networks (HCNNs) model this stream architecturally by structuring the network into hierarchical layers of neurons. The similarities, though, extend beyond architecture. Image responses by layer in HCNNs have been shown to correlate with the brain's hierarchical representation in both time and space [4]. This correlation has been used to model the brain's ventral stream using HCNNs [22], and its modeling capabilities are significant. Though compelling models for lower ventral areas have been discovered previously [3, 23], extrapolating these results to higher cortical areas had only limited modeling success [4]. However, the use of HCNNs has allowed for much more accurate models of these previously enigmatic higher cortical areas [14, 23, 22].

Thus, HCNNs not only perform classification tasks well, but can also apparently perform these classification tasks in a brain-like manner. Training these networks can thus provide insights into the inner workings of the brain. However, HCNNs are only a model of the brain, with their own imperfections and characteristics. Thus, it is equally important to investigate the limitations of these networks' modeling capabilities in order to accurately draw insights from their study.

1.1.2 Visual Illusions

HCNNs have only recently acted as means through which researchers can study characteristics of the brain's visual processing system. Visual illusions, in contrast, have fulfilled a similar role for decades [10], helping researchers understand similar higher order visual systems. For example, the moon illusion has been used to study the ventral visual stream [20], the Müller-Lyer illusion to study the dorsal visual stream [6] and the Ebbinghaus illusion to study differences between the vision-for-perception and vision-for-action systems [8].

These illusions have helped us understand the assumptions and heuristics used by the brain to interpret the visual world [10]. In a similar way, they offer an opportunity to learn about the assumptions and models used by HCNNs when classifying images. Does their ability to model the brain's ventral stream extend to cases where the ventral stream misinterprets an image? Are HCNNs susceptible to the effects of visual illusions, and if so, are they the same illusions experienced by humans? Running experiments at the intersection of visual illusions and HCNNs can hopefully answer these questions, and, in so doing, teach us more about HCNNs' ability to accurately model the human visual system in all of its idiosyncrasies and complexities.

The Poggendorff illusion and AlexNet offer a good choice of visual illusion and HCNN, respectively, to study this intersection.

1.2 The Poggendorff illusion

The Poggendorff illusion falls into a category of "distortion illusions" [10]. These are images of figures whose construction distorts the appearance of the figure's size or shape. In the case of the Poggendorff image, the diagonal line segments appear offset, despite being aligned. See Fig. 1.1 for an example of the illusion.

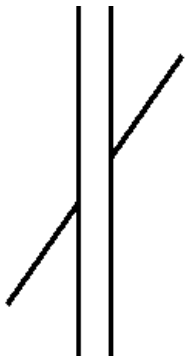


Figure 1.1: The Poggendorff Illusion; the straight line crossing the gap (the *diagonal*) appears offset.

As can be seen from Fig. 1.1, the Poggendorff illusion is relatively simple in its construction. This simplicity is part of the illusion's appeal when considering it for experiments with an HCNN. The Poggendorff illusion's four line construction allows for a more straightforward analysis with fewer confounding image feature factors compared to an illusion that requires a more complicated image, such as the Hering illusion (see Fig. 1.2).

The Poggendorff illusion has been widely studied. Multiple theories for the illusion's cause have been posited, with the most popular attributing the effects to depth processing. This theory suggests that the illusion is caused by the brain interpreting the 2D image as a 3D construct [2]. However, studies on variations of the illusion have disagreed with this explanation, indicating that there may be other mechanisms affecting the perception of the illusion [5, 9].

Despite all this research into the Poggendorff illusion, no experiments (to my knowledge) have been run with an HCNN. Regardless of what causes the illusion for humans, its existence provides an opportunity to study if HCNNs are similarly susceptible to the illusion's effects. This susceptibility, whether present or not, can help determine the extent to which the HCNN is capable of modeling the human visual system.

1.3 AlexNet

The HCNN used in this thesis to study the intersection between visual illusions and HCNNs is AlexNet, developed in 2012 by Krizhevsky *et al.* [15]. Using a pre-existing HCNN has the advantage of working with a well-studied architecture, and allows the experiments to focus on the question of the network's interpretation of the illusion.

AlexNet was chosen because it's ability to model the human visual system has been well studied. AlexNet's architecture was inspired by the ventral stream [23], and its internal image representation was shown to

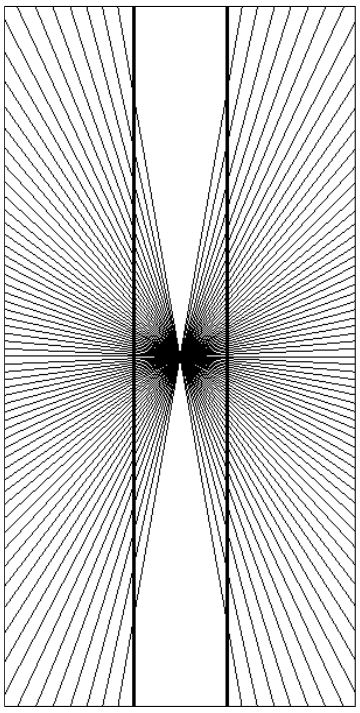


Figure 1.2: The Hering Illusion. The bold, vertical lines appear warped despite being straight. Note the much larger number of lines required for this illusion as compared to the Poggendorff.

correlate with the brains' representation at the inferior temporal cortex [14]. Additionally, a retrained version of AlexNet using the same architecture was used to show layer-level correlations across the brain's entire ventral stream in time and space [4].

AlexNet also represents a significant network outside the study of the visual processing system. It is a landmark network in the development of deep convolutional networks for object recognition tasks [13]. In 2012 it won the ImageNet Large Scale Visual Recognition Challenge with a top-5 error of 15.3%, over 10% lower than the second place network [12].

AlexNet is composed of 8 layers: 5 convolutional layers and 3 fully connected layers. 3 of the 5 convolutional layers use max pooling. See Fig. 1.3 for a visualization of the structure, as well as the sizes of all the layers and windows used for convolution. However, the "AlexNet" network is not just its architecture, but also the weights that architecture learned through training on the ImageNet dataset. Note, for example, that it was the final trained network that was shown to model the brains' inferior temporal cortex in [14].

These weights, though, were trained for the specific task of classifying images into one of 1000 categories based on the ImageNet image database. Thus, classifying susceptibility to the Poggendorff illusion is not the standard object recognition task for which AlexNet was trained. Indeed, interpreting the Poggendorff illusion is not an image recognition task at all, but a more abstract task of interpreting a particular line's straightness. Therefore, work must be done to interpret the information outputted by AlexNet in the context of the more abstract notion of susceptibility to the Poggendorff illusion.

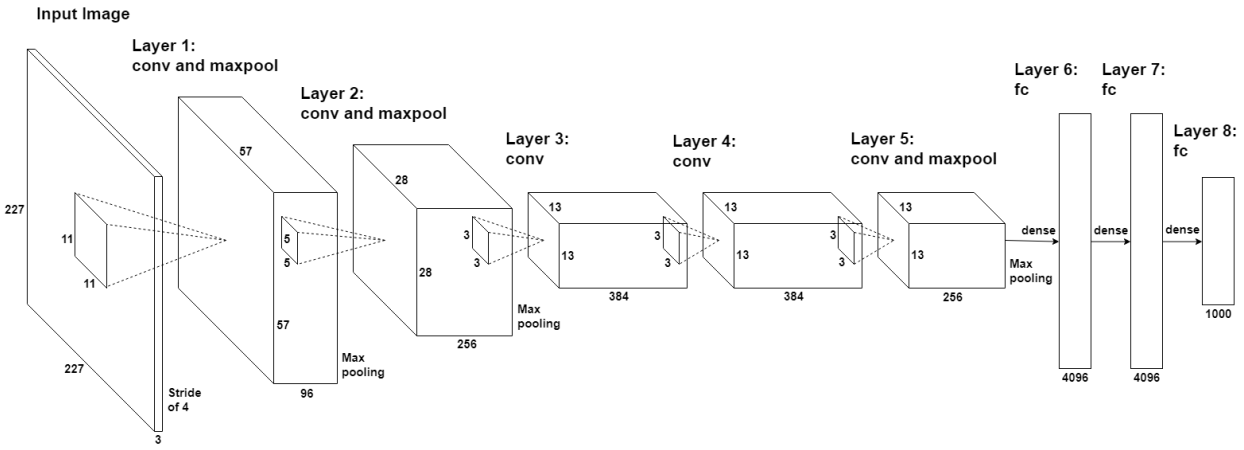


Figure 1.3: Layer and window sizes for the AlexNet implementation used for this thesis. Based off values from [11] and fig. 2 of [15]. Figure side lengths not to scale. "conv" indicates a convolutional layer, "maxpool" indicates a layer where max pooling was applied after convolution, and "fc" indicates a fully connected layer.

AlexNet was originally written in C++/CUDA. In order to work in python, Michael Guerzhoy's tensorflow implementation [11] was used. This implementation has weights directly converted from the original model, and replicates AlexNet's architecture with 2 minor changes. AlexNet originally used an optimization to allow it to be trained on two separate GPUs, but with modern hardware this is no longer necessary (and subsequent implementations have foregone this optimization, such as [13]). Secondly, the network runs on 227×227 images instead of the original 224×224, which slightly increases both the input image dimensions as well as future layer dimensions. Guerzhoy's code was then adapted for the various experiments discussed in this thesis.

It should be noted that, once trained, an HCNN's classification of an image is deterministic. During training, a network's classifications will change as it uses labeled examples to "learn" the optimal weights for the network, but once trained these weights do not change. Thus, running an image through AlexNet multiple times will result in an identical classification every time.

1.4 Overview of experiments

There are three categories of experiments in this thesis. First, exploratory testing is done on the original, unmodified AlexNet. These experiments use cosine similarity to evaluate AlexNet's interpretation of the Poggendorff illusion according only to its final classification scores. Designing such a framework is necessary because AlexNet was trained for object recognition, not Poggendorff interpretation. These experiments measure the similarity between a control image containing an aligned diagonal, and a series of test images with both aligned and offset diagonals. The Poggendorff illusion affects humans' ability to discern a line's straightness, where aligned line segments are taken to make a non-straight (offset) line. Thus, differentiating lines from non-lines is a prerequisite to being susceptible to the Poggendorff illusion. Through these exploratory tests I hope to find a set of images where the similarity score for an image corresponds to the straightness of its line segments. This would demonstrate AlexNet's ability to correctly classify lines and non-lines and provide a framework for evaluating this classification. Using this technique and image set, if it exists, I can then test AlexNet's susceptibility to the Poggendorff illusion.

Interestingly, none of the image sets tested satisfy the similarity-straightness correspondence, and based on the results it is doubtful such an image set exists. This could be because AlexNet is unable to distinguish aligned from offset line segments, or because AlexNet does not use this information during classification. To determine the cause, the second category of experiments retrain the final layers responsible for the original 1000-category classification to distinguish lines from non-lines directly. The retraining is done on new image sets generated according to a number of hyperparameters. These hyperparameters control the difficulty of the classification task, so that AlexNet's classification ability can be measured under different circumstances. From these experiments, I hope to determine AlexNet's underlying ability to distinguish lines from non-lines.

These retrained networks implicitly provide a framework for evaluating AlexNet's perception of line straightness, since they are trained to output their classification directly. The third category of experiment then uses these retrained networks to judge line-straightness in Poggendorff-like images. These are images constructed like the Poggendorff illusion but with both aligned and offset line segments. Depending on the network's underlying ability to differentiate lines from non-lines, I hope to learn if AlexNet is susceptible to the effects of the Poggendorff illusion, which I will refer to as "Poggendorff effects".

2 Exploratory Research

2.1 Classification versus interpretation

AlexNet is trained to output a normalized 1000-dimension classification vector. For a given image, each dimension in this output vector represents the probability with which AlexNet "believes" the image should be classified in the corresponding list of 1000 categories. Though this structure works well for the classification task AlexNet was built to solve, it does not translate to the more abstract task of interpreting a visual illusion like the Poggendorff Illusion. There is no category for "aligned" or "offset" line segments, nor is there any ability to specify which line segments the network should care about.

2.2 Cosine similarity

Instead, the illusion was reinterpreted in measurable terms. Line segments that are aligned are taken to be similar to an image containing a line, whereas lines segments that are offset are dissimilar to the same image. To mathematically represent the notion of similarity, I use cosine similarity. For two vectors $\vec{x}$ and $\vec{y}$, where θ is the angle between the vectors, cosine similarity is defined as:

$$\begin{aligned} similarity(\vec{x}, \vec{y}) &= cos(\theta) \\ &= \frac{\vec{x} \cdot \vec{y}}{\|\vec{x}\| \|\vec{y}\|} \end{aligned} \tag{1}$$

Cosine similarity measures the magnitude of the angle between two vectors in an arbitrary dimensional space. The similarity score is a value in the range $[0, 1]$, where 0 indicates the vectors are perpendicular (low similarity), and 1 indicates they are parallel (high similarity).

2.3 Exploratory research methodology

Cosine similarity is then used to measure how similar AlexNet perceives two images at every layer of its network. This is done by passing the image through the (original, trained) network and measuring the resulting state at each layer. Since the cosine similarity is defined for vectors, these layer states are flattened in row-major order as necessary. Note that the resulting loss of some spacial locality information is not important, since cosine similarity only compares vectors element-wise.

Cosine similarity was used to measure the similarity between a test set and a single control. Collectively, the test set and control are referred to as an image set. The relative similarity scores between the images in the test set were then used to learn what image features AlexNet cares about when making similar classifications.

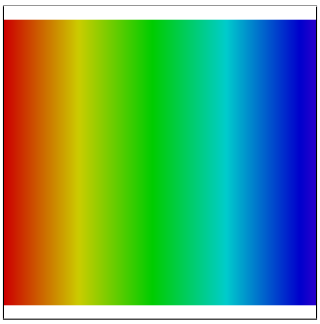
2.3.1 Vertical line example

To describe this technique, consider a simple example with vertical lines. The control is a white image with a single, vertical black line centered about the x-axis (see Fig. 2.1b). The test images are a set of 255 images from index $i = 0$ to $i = 254$. Like the control, each image is white except for a single, vertical black line positioned horizontally $i + 1$ pixels from the left of the image. (This 1 pixel offset is necessary because the

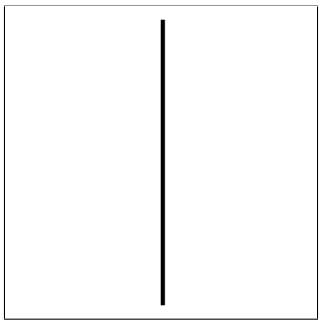
lines are 3 pixels wide. For the entire line to be visible at every index, then, requires starting the 0-index image 1 pixel from the edge of the image.)

To represent all 225 of these test set images in a single image, I introduce a spread image (see Fig. 2.1a). Since all images in these experiments are made up of only white or black pixels, spread images use colour to represent how the test set changes across indices. Any pixels that are constant across the test set (always black or always white) will appear unchanged in the *spread* image. (See the top and bottom white margin in Fig. 2.1a for an example of this).

For pixels that change, the *red* areas represent black pixels in low-index images, and *blue* areas represent black pixels in high-index images. For example, in Fig. 2.1a, the far-left of the image is red and corresponds to the vertical lines of images with small indices (and thus small offsets from the left of the image). Similarly, the far-right of the image is blue and corresponds to the vertical lines of images with large indices (and thus large offsets from the left of the image). The colours between red and blue represent the black pixels in images between low- and high-index images in the test set. The colour spread corresponds to hue values from 0 to 0.7 in the HSV colour space.



(a) Vertical spread image



(b) Vertical control image.

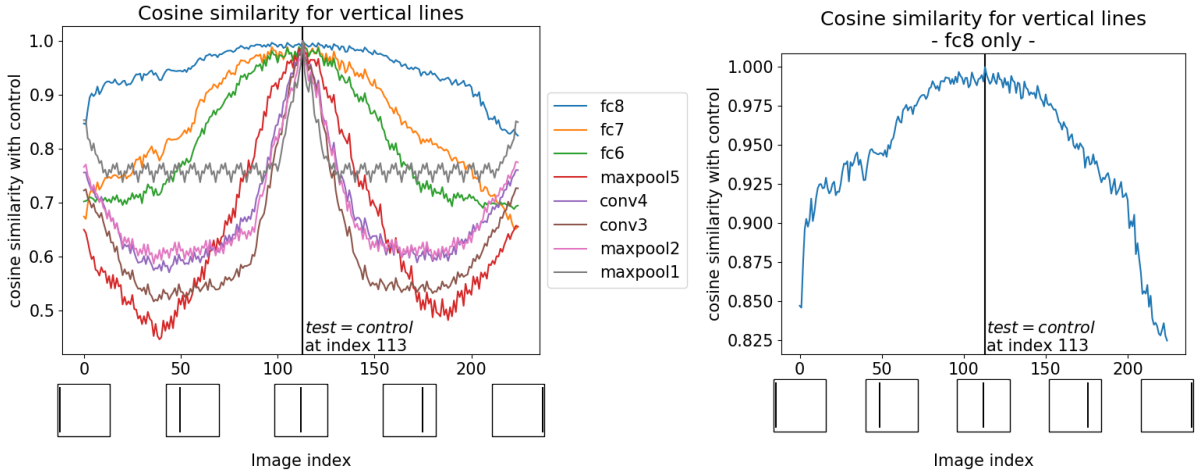
Figure 2.1: Vertical image set. Note that for these and all other control/spread images, borders are added only for clarity; they are not part of the image used for testing.

The control image and test images were then all run through the original AlexNet, and the values at each layer were collected. These values were treated as a vector, and the cosine similarity between each image in the test set and the control at each layer of the network was taken.

The image at index 113 has a similarity of 1 at all levels of the network. This is expected, since this image had its line centered about the x-axis and is thus identical to the control. At the $fc8$ layer, vertical lines near the center of the image achieve high similarity (though with some seemingly periodic perturbations), and as the line moves farther away from the center in both directions this similarity decreases, though not symmetrically. This pattern is repeated for other fully-connected layers, but the lower maxpool and convolutional layers exhibit very different, non-obvious patterns. See Fig. 2.2 for the plots of the similarity scores for the 227 images in the test set.

2.3.2 Test sets to be studied

This exploratory testing was not concerned with whether AlexNet perceived the Poggendorff illusion. Rather, it was concerned with understanding what image features AlexNet cared about, and using that to construct a framework within which AlexNet's Poggendorff perception could be measured. Thus, the "Poggendorff-like" images used were *connected* Poggendorffs: Poggendorff illusions where the two diagonal line segments are



(a) Cosine similarity for all layers

(b) Cosine similarity for the fc8 layer

Figure 2.2: Cosine similarity graphs for vertical lines. In this and all future similarity graphs the images under the axes represent a sample of the test images at those indices. The vertical line (if present) indicates where the test image exactly matches the control image.

connected by an additional line segment between the verticals. Thus, determining whether the line segments are "aligned" or "offset" becomes a matter of determining whether the three line segments make a straight line or non-line.

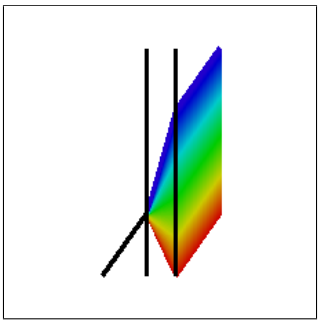
For exploratory testing, a number of image sets were created. For each image set, the test set was created by moving the rightmost line segment (the right arm) up along the rightmost vertical, from the bottom of the image to the top (as an example, see Fig. 2.3a). Thus, the diagonal line segments were only perfectly aligned (making a straight line) in one of the test set images, at index $i = 75$. The diagonal became progressively more unaligned as the indices moved further from 75 in both directions. For every image sets, the control image contained a perfectly aligned diagonal. Thus, the relative similarity scores measured, ideally, the significance AlexNet placed on line straightness when making classifications.

There are four ronds of experiments. The first experiment used an image set that simply compared centered, connected Poggendorffs against a control whose diagonal was at the bottom of the image. I will refer to this image set as the *simple set*; its spread and control images can be see in Fig. 2.3.

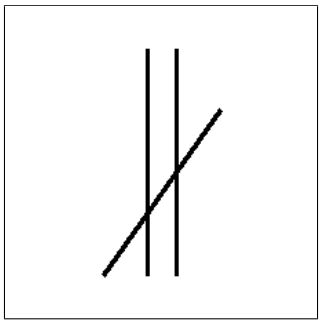
However, by placing the control's diagonal at the bottom of the image, this resulted in one of the test images matching the control image exactly. This could potential result in similarity scores that were due more to pixel alignment than line straightness.

To alleviate this problem, the second experiment used an image set that compared the same test set against a control whose diagonal was centered about the y-axis. I will refer to this image set as the *centered set*; its spread and control images can be see in Fig. 2.4.

By centering the control's diagonal, there are is perfect match between the control and any test image. However, there is still a lot of overlap between the sets. With the control's diagonal centered, there are now test set images with indices greater than 75 (the index of the image with a straight diagonal) whose right arm

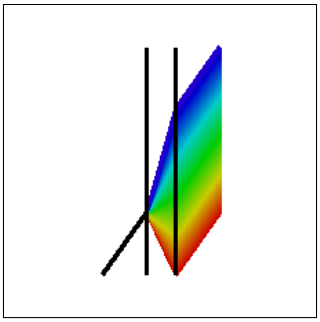


(a) Spread image

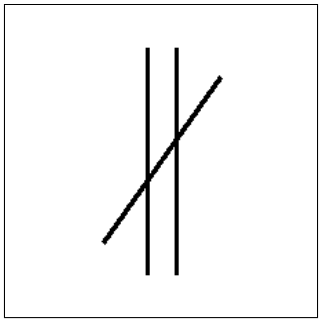


(b) Control image

Figure 2.3: Spread and control images for the simple set.



(a) Spread image

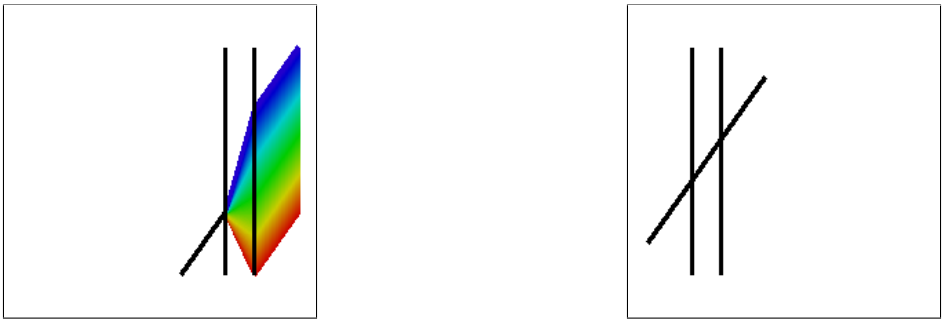


(b) Control image

Figure 2.4: Spread and control images for the centered set.

align very closely, or perfectly, with the control image's right arm. These images had higher similarity scores than the image at index 75, indicating the AlexNet placed higher significance on pixel-level correlations than the more abstract notions of line straightness.

Thus, to remove any influence of pixel-level correlation, a third experiment was run. This test used a third image set that moved all test images to the right half of the image and the control to the left half, resulting in what I will call a right-left orientation. This ensured that no test image shared any black pixel locations with the control image. I will refer to this image set as the *offset set*; its spread and control images can be see in Fig. 2.5.



(a) Spread image **(b)** Control image

Figure 2.5: Spread and control images for the offset set.

The offset set produced high similarity scores close to index 75, the index of the image with the straight diagonal. However, the max similarity score did not occur exactly at index 75. To further understand what other factors were affecting the similarity score, the fourth experiment permuted offset set in 3 binary ways, resulting in 8 sets:

- Image sets were created with a control that had, and did not have, vertical lines
- Image sets were created with a control whose diagonal had a positive, and negative, slope
- Image sets were created with a left-right orientation and right-left orientation

These permutations were studied to compare how various features unrelated to line alignment affected the similarity scores.

2.4 Results of exploratory research

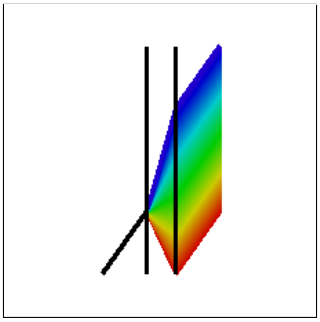
2.4.1 Simple set

The first experiment used the simple set. Since the control's diagonal is at the bottom of the image, it is important to note that the image at index 75 of the test set (the image with the straight diagonal) is identical to the control image. This resulted in perfect (1.0) similarity between the control and the test set at index 75 for all levels of AlexNet, just as index 113 achieved perfect similarity for the vertical line case in Fig. 2.2. Unlike the vertical line experiment, the similarity approached 1 from both sides relatively smoothly and without dipping, though the two approaches are asymmetric.

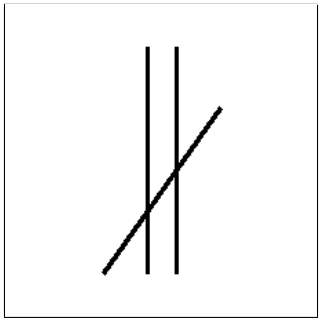
The similarity scores representing AlexNet's perception of line straightness could explain this smooth approach. As reflected in the scores, the line segments indeed make a progressively less straight "line" as the indices move farther from 75 in both directions. However, it is impossible to attribute these results entirely to line straightness because of the high similarity between the test set and the control image.

In the simple set, the control is identical to the test image at index 75 and the only difference between other test images is the position of the right arm. This right arm position differs smoothly from image to image because its vertical position is shifted by a single pixel per index. Thus, the smooth approach to perfect similarity at index 75 could also be explained by local pixel-level correlations in the images. This confounding factor must be removed from the image set in order to determine AlexNet's perception of line straightness. The first set in removing pixel-level correlations is eliminating any perfect matches between the control and test set. This is done in the centered set.

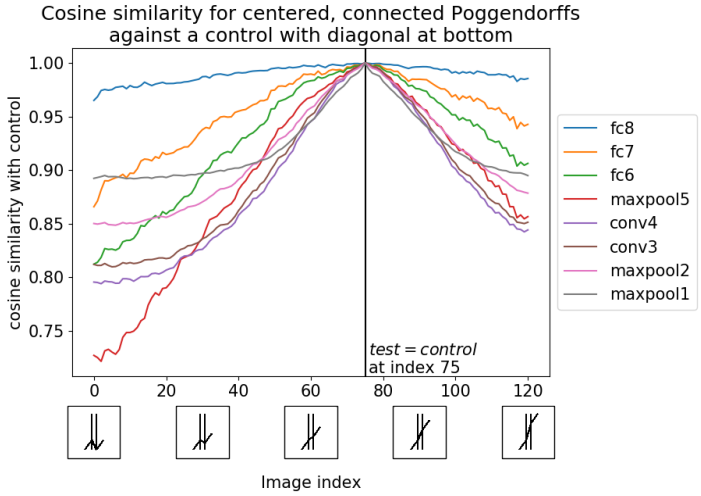
See Fig. 2.6 for the graphs of the similarity scores for the simple set.



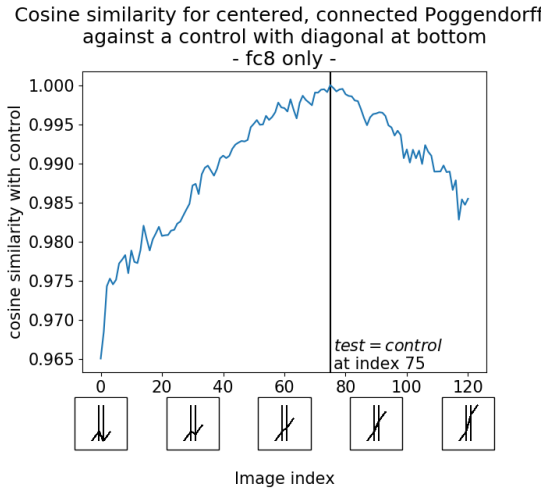
(a) Spread image



(b) Control image



(c) Cosine similarity for all layers



(d) Cosine similarity for the fc8 layer

Figure 2.6: Spread image, control image and cosine similarity graphs for the simple set.

2.4.2 Centered set

The second experiment uses the centered set. This image set improved on the simple set by eliminating any perfect matches between the test set and control image. This was done by moving the control's diagonal line to the middle of the control image. However, by doing this, the right arm of the control image now correlates exactly with the right arm of image 98 in the test set. This is the only part of the diagonal line segments that is able to correlate exactly, since the leftmost segment does not move in the test set.

This correlation in the right arm seems to have a huge effect on the similarity scores. In the centered set, the max similarity score was shifted to a much larger index at every layer. This would be expected at lower layers, when the network is still working with barely-transformed versions of the image and so exact pixels matches would greatly affect the similarity of images. However, the max similarity score even at the final layer, $fc8$, occurs at index 101, indicating that exact pixel matches seem to play a big role throughout the network. It seems, then, that AlexNet cares much more about local pixel correlations than more global features.

The centered set's similarity scores appear to only show which images are highly correlated at the pixel level. To learn about AlexNet's perception of line straightness requires fully removing any pixel-level correlations, which is done in the offset set.

See Fig. 2.7 for the graphs of the similarity scores for the centered set.

2.4.3 Offset set

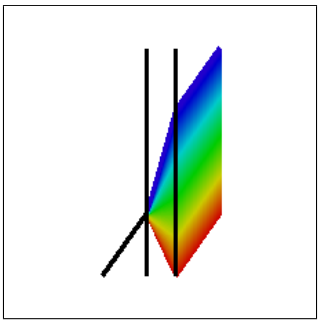
To eliminate all pixel-level matches, the placement of the line segments within the images was modified in the offset set. The lines in the test set image were moved to the right half of the image, and the lines in the control were moved to the left half. This is what will be referred to as a right-left orientation. Importantly, by moving the lines to different halves of the image, there are no pixel-level matches. That is, no black pixels are shared between any of the test set images and the control in the offset set.

Ensuring that the lines in the test set occupy a different space on the canvas from the lines in the control image appears to have fixed the issues seen with the centered set. The index of max similarity has been pushed back towards index 75, the image with the straight diagonal, 75. For the final $fc8$ layer, the max similarity score is now at index 72 instead of 101 as seen in the centered set.

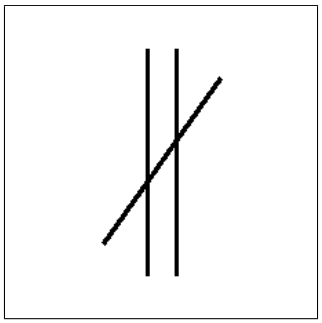
In fact, these results are similar to those seen in the first experiment with the simple set. Once again, the similarity scores approach their max value near index 75 relatively smoothly from both sides. Just as in the simple set, this shape can be explained by the similarity scores accurately representing line straightness, since the line segments become progressively less straight as the indices move farther from 75 in both directions. Additionally, unlike in the simple set, the results cannot be explained by pixel-level correlations.

However, there are some irregularities. Instead of the max similarity occurring exactly at index 75, it occurs slightly earlier, at index 72. In fact, 22 images have higher similarity scores than image 75 with all but 3 occurring at a lower index. This indicates that there may be other factors at play affecting AlexNet's scores other than simply its perception of line straightness. To further explore what these factors could be, the offset set was permuted in 3 ways, resulting in 8 different image sets.

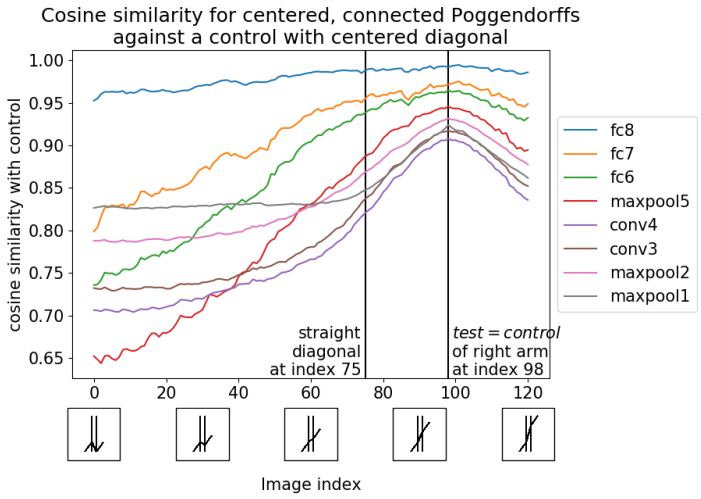
See Fig. 2.8 for the graphs of the similarity scores for the offset set. On first viewing, the larger similarities at indices less than 75 may appear to indicate effects of the Poggendorff illusion. Since the original Poggendorff illusion in Fig. 1.1 causes the right arm to appear too high, line segments that appear aligned would require



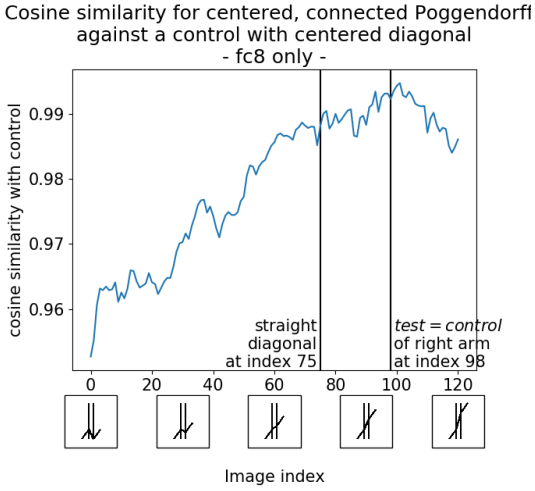
(a) Spread image



(b) Control image



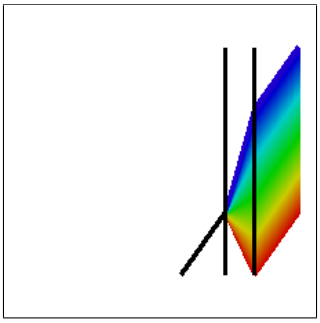
(c) Cosine similarity for all layers



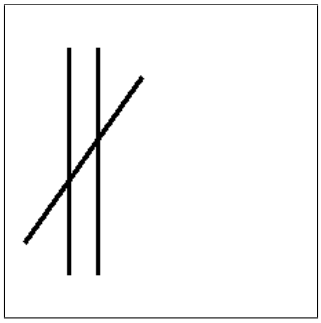
(d) Cosine similarity for the fc8 layer

Figure 2.7: Spread image, control image and cosine similarity graphs for the centered set. Note that the test set is identical to that used in the simple set.

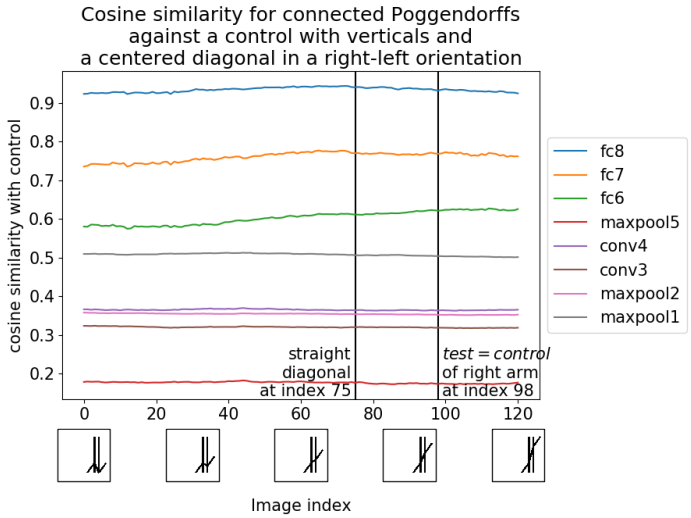
moving the right arm down, to a lower index. However, the Poggendorff illusion only applies to disconnected diagonals, and all images used for exploratory testing had connected diagonals. Thus, this shift cannot be explained by the Poggendorff effect.



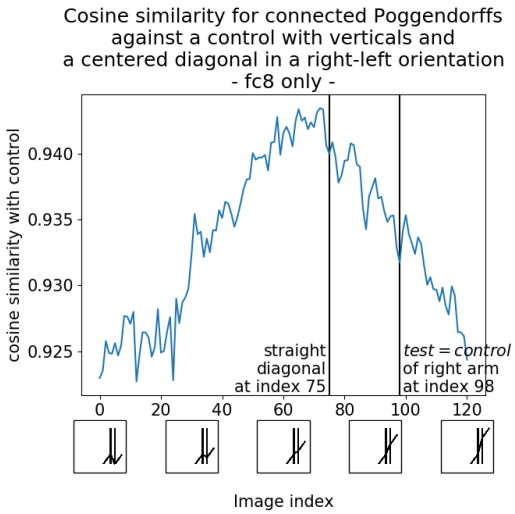
(a) Spread image



(b) Control image



(c) Cosine similarity for all layers



(d) Cosine similarity for the fc8 layer

Figure 2.8: Spread image, control image and cosine similarity graphs for the offset set.

2.4.4 Permutations of offset set

If AlexNet is able to differentiate straight from non-straight lines, then we would expect an image set to exist where a test image's similarity score with a control containing a straight diagonal reflects the straightness of the test image's diagonal. Furthermore, such an image set should not have to rely on pixel-level correlations to achieve this relationship between line straightness and similarity scores, but instead be a product of AlexNet's ability to differentiate lines from non-lines. The third experiment using the offset set nearly achieves this, but the set's max similarity does not occur at exactly index 75, the image containing the straight diagonal. To try and further understand this discrepancy, and to explore other possible image sets, offset set was permuted in 3 binary ways, resulting in 8 sets:

- Image sets were created with a control that had, and did not have, vertical lines
- Image sets were created with a control whose diagonal had a positive, and negative, slope

- Image sets were created in a left-right orientation and a right-left orientation

For simplicity, these 8 sets are named in a standard way. This naming convention uses camel-case where capital letters, instead of spaces, indicate new words. The first word is either "Vert" or "Line", and indicates whether the image set had, or did not have, vertical lines, respectively. The second word is either "Up" or "Down" indicating that the control image's diagonal has either a positive (angles up from left to right) or negative (angles down from left to right) slope, respectively. The third and final word is either "RL" or "LR" indicating that the image set has either a right-left orientation or a left-right orientation, respectively. This orientation follows the convention *[test set side]-[control side]*. The spread and control of each permutation can be seen in Table 2.1.

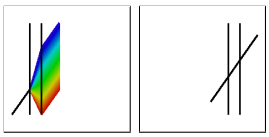
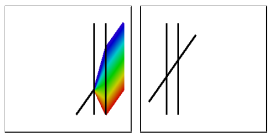
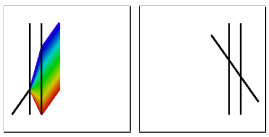
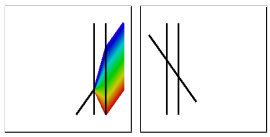
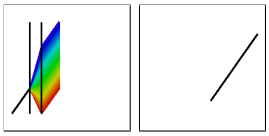
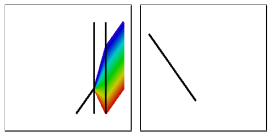
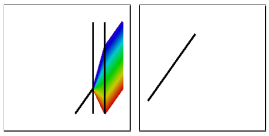
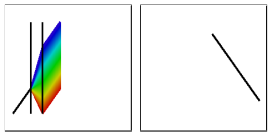
	Control Diagonal Up		**Control Diagonal Down**	
	Left-Right Orientation	**Right-Left Orientation**	**Left-Right Orientation**	**Right-Left Orientation**
With Verticals	 *VertUpLR*	 *VertUpRL*	 *VertDownLR*	 *VertDownRL*
Without Verticals	 *LineUpLR*	 *LineUpRL*	 *LineDownLR*	 *LineDownRL*

Table 2.1: The spread and control images of the 8 permutations tested. For each pair, the image on the left is the spread, and the image on the right is the control. The italicized text underneath is the label used in the legends for Fig. A.1 and Fig. 2.9.

These permutations were chosen both to test different features that were seen in the offset set and to attempt to find an image set that demonstrates AlexNet's ability to perceive straight lines. Interestingly, 19 of the 22 images with higher similarity scores than image 75 (the image with the straight diagonal) occurred at indices less than 75 in the offset set. Permuting the control diagonal's slope and the orientation of the image set tests potential causes for this trend. Removing the vertical on the control images, in contrast, attempts to test AlexNet's perception of line straightness more directly by removing potentially confounding image features.

Graphing the similarity scores of these 8 permutations at the $fc8$ layer directly is not particularly helpful. Comparing the shape of the different permutations' curves is difficult because some permutations have overall higher similarity scores than other permutations. This is particularly true for permutations whose control do not have vertical lines, which is to be expected due to the lack of similar image features. (To see a graph of the raw data, see Fig. A.1 in Appendix A.1.) To aid comparison, I centered the similarity score data about each permutation's mean.

From the centered data, it is clear that the result seen in experiment three on the offset set (labeled here as VertUpRL) is an outlier. The promising result of a max similarity score close to the 75^{th} index is not typical in these permutations. The only other permutation to exhibit a relatively centered max is the other right-left orientated image with verticals, this time with the diagonal angling down (VertDownRL). Its max similarity score occurs at index 66. Insofar as position of max similarity is concerned, this indicates that the orientation of the control's diagonal line does not have a large impact, unlike the other permutations.

For all other permutations, there is a clear positive trend in the curves. The max similarity values come at very large indices, all between 115 and 118. Interestingly, the permutations that exhibit this positive trend most strongly are those whose control images have no vertical lines. These are the permutations that were meant to examine directly how well AlexNet could abstract the notion of a straight line. In these permutations, the test image with a straight diagonal (at index 75) had much smaller similarity with the control than later, non-straight, test images.

The max similarity score for the offset set occurred at index 72 instead of the expected 75, the index of the image containing the straight diagonal. The original goal of experimenting with these permutations was to explore possible factors for this three index discrepancy. However, the three index discrepancy I was trying to remedy turned out to be the smallest discrepancy of any of the other permutations. Thus, none of these permutations are good candidates for demonstrating AlexNet's ability to differentiate lines from non-lines.

The larger goal of the exploratory experiments is to find an image set where the test images' similarity scores reflect the straightness of their lines without relying on pixel-level correlations. However, all permutations of the most promising image set performing so poorly results in a dead end in my search, and calls into question the existence of such an image set. If no such image set exists, then it calls into question AlexNet's underlying ability to differentiate lines from non-lines. Regardless of AlexNet's underlying ability, these results indicate that the unmodified AlexNet is unable to differentiate lines from non-lines using cosine similarity for Poggendorff-like images.

See Fig. 2.9 for a graph of the centered similarity scores at the $fc8$ layer.

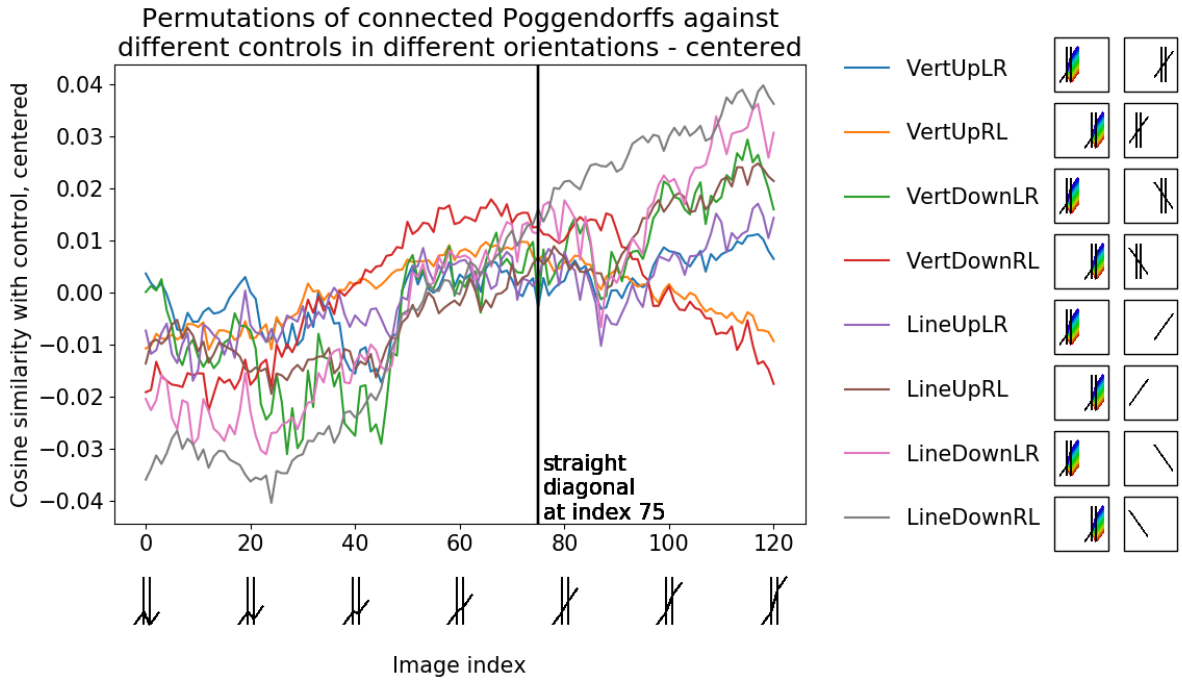


Figure 2.9: Cosine similarity at the fc8 layer for 8 the permutations in Table 2.1, with data centered about each permutation's mean. The images to the right of each legend item are the spread and control for that set. The images under the axis represent a generic sample of the test images at those indices, without any orientation.

2.5 Discussion of results

This exploratory research was intended to construct a framework for evaluating AlexNet's interpretation of the Poggendorff Illusion, given that it is a more abstract task than classifying an image into one of 1000 categories. However, these results bring into question AlexNet's underlying ability to recognize a line from a non-line.

Perceiving the Poggendorff Illusion is contingent on being able to distinguish aligned versus offset line segments, which requires being able to distinguish lines from non-lines. If AlexNet were both able to distinguish lines from non-lines, and used this information in its classification, then we would expect there to be an image set where the similarity scores of test images corresponds to the straightness of their lines. However, no such image set was found, and the permutations of the most promising image set, offset set, had max similarity scores at indices very far from image 75, the image containing the straight diagonal. Recall that AlexNet's classifications are deterministic, so running these tests multiple times will not yield different results. These experiments indicate that the unmodified AlexNet is unable to differentiate lines from non-lines using cosine similarity for Poggendorff-like images.

It is important to note that only the original, unmodified AlexNet was used during exploratory testing. This makes AlexNet's inability to differentiate lines and non-lines during these tests somewhat less surprising. AlexNet was trained to correctly classify images into one of 1000 categories, and since none of those categories are "lines" or "non-lines", its original training optimized the network's weights for a problem different than the one being asked here. Thus, its inability to differentiate lines from non-lines does not necessarily mean that the network ins fundamentally incapable of this task. It could be that the information required is present in the network, particularly at lower layers, but is being disregarded for the 1000-class classification problem AlexNet was originally trained for.

Before trying to determine if AlexNet perceives the Poggendorff illusion, then, I first ask if AlexNet is able to differentiate lines from non-lines. I answer this question by directly retraining AlexNet to distinguish more generic lines from non-lines. Directly retraining the network addresses two problems with the exploratory testing. First, it addresses the exploratory testing's limited scope of only Poggendorff-like images by retraining the network to classify much more generic lines and non-lines. Second, retraining circumvents the layers of the network optimized for a different problem (classifying an image into one of 1000 classes) and instead develops layers optimized for differentiating lines from non-lines. It does this while still using the underlying information stored in AlexNet. Retraining thus allows me to determine if the results of the exploratory testing were due to an inability to differentiate lines from non-lines, or if AlexNet *is* able to differentiate them but disregards this information when making classification decisions in the original network.

3 Classifying lines and non-lines

The results of the exploratory testing bring into question AlexNet's ability to distinguish lines from non-lines. In order to determine AlexNet's is fundamentally capable of this task, the network is retrained with the explicit goal of distinguishing lines from non-lines. This process involves retraining some number of AlexNet's final layers using newly generated training images, and measuring the network's success using different validation and test image sets.

When retraining the network, a number of hyperparameter decisions were made related to data generation, network architecture, and network retraining. I go through these decisions in order.

3.1 Data Generation

The training, validation, and test data sets are generated programmatically. 10,000 images were generated for the training set, and 2,000 each for the validation and test set.

3.1.1 Generating lines and non-lines

Each image contains one primary line or non-line (collectively referred to as elements), and some (potentially 0) number of background elements. Each element is generated as a function of at least 3 parameters: the starting point of the element, the total length of the element (either the line's length or the sum of the non-line's line segments), and the initial angle of the element (the angle the line or first line segment makes with respect to the canvas). All angles are measured counterclockwise with respect to the positive x-axis of the canvas.

To model the types of elements seen in the Poggendorff illusion, non-lines are chosen to be composed of three connected lines segments in a specific configuration. The first and last segments are always parallel, and the middle line segment makes some non-zero angle relative to the element's initial angle. (If this angle were zero, then the three segments form a line.) This angle is a non-line's 4^{th} parameter and is referred to as *joint angle*. The middle line segment's length was set to a constant 21 pixels to match the gap used in the the exploratory tests. (This is a fixed 5^{th} parameter.) See Fig. 3.1 for a labeled example of a non-line.

These parameters are randomly generated according to different distributions. The initial point and angle are chosen first and so could take on any value (modulo the size of the canvas and 2π, respectively). Thus, they are randomly chosen from a uniform distribution. One small note about these choices. Depending on the starting position, initial angle, element length, and joint angle (for non-lines), the element may extend beyond the bounds of the canvas. If a non-line is drawn off canvas, then there is a chance only one of its line segments will be visible. To any observer, this would make the element a line and cause the image to be mislabeled. To avoid this scenario, if any point lies out of bounds, it is moved back in bounds. This translation is then applied to every other point to both keep the entire element on the canvas and maintain its shape.

The length and joint angle are chosen according to a gamma distribution due to the distribution's convenience. A gamma distribution has two parameters, k and θ, and by manipulating these parameters the relative peak and tail size of the distribution can be controlled. Additionally, all gamma distributions have a a zero probability of a zero value. This is a requirement for both the length parameter, since lines must be visible, and the joint angle parameter, since non-lines require a non-zero joint angle.

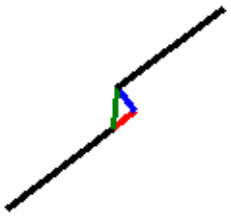


Figure 3.1: An example of a non-line. The middle line segment is green. Its parallel component (to the first line segment) is red, and its perpendicular component is blue.

For line length, over 70 possible distributions are considered before choosing $k = 9$ and $\theta = 0.65$. The resulting values are then multiplied by 16 as this provided a good length relative to the canvas size. This distribution is chosen because its mean is 94, giving most lines a length close to half a side length. This decreases the chance of a line being so long that it doesn't fit on the canvas, and minimizes the need to translate the line back in bounds. (Note that translating lines changes the distribution of lines' starting positions, so minimizing the number of translated lines keeps the true distribution of initial locations close to uniform.) The distribution also has a small number of short lines (which are difficult to interpret) and a tail that contains some longer lines up to about the side length of the canvas. See Fig. 3.2 for a graph of the distribution based on 10 million randomly generated lengths.

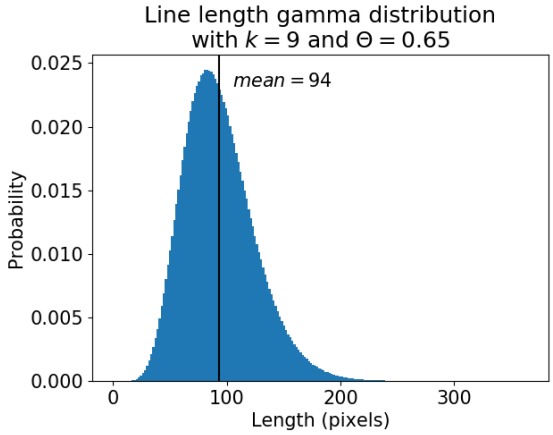


Figure 3.2: The gamma distribution used for line length in image generation.

The hyperparameters for the joint angles' gamma distribution are also selected by examining many possible distributions. For each distribution, the resulting values are multiplied by $\frac{\pi}{20}$ to give more reasonable angles. Note that the smaller the joint angle, the more difficult it becomes to differentiate lines from non-lines. For this reason, two distributions are chosen so that the network's relative performance could be compared. For both distributions $k = 6$ is chosen. For the easier classification task, $\theta = 0.75$ is used, resulting in a mean of

0.71 $rad \approx 37°$. For the harder classification task, $\theta = 0.4$ is used, resulting in a mean of 0.38 $rad \approx 20°$. See Fig. 3.3 for the distributions. These angles are then randomly multiplied by 1 or -1 with equal probability to produce non-lines that bend both to the right and left relative to the first line segment.

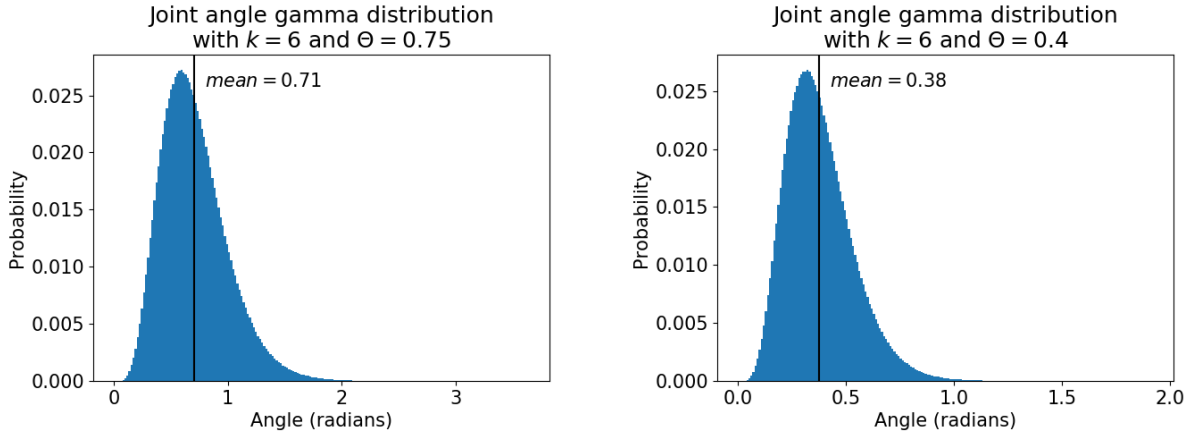


Figure 3.3: The two gamma distributions used for joint angle in image generation.

Though using the gamma distribution helps ensure that the angles lie within a reasonable range, simply using the values directly can cause problems. Compare the green and blue lines in Fig. 3.4a. Though both use the same gap length, indicated by the red circle, determining whether the line segments form a line is much more difficult for the blue segments. This is because the component of the middle line segment parallel to the first line segment is much larger for the blue case.

(a) The result of using $\frac{\pi}{2} rad$ (green) and $\frac{\pi}{18} rad$ (blue) joint angles directly, without correction.

(b) The result of using the joint angles from **(a)**, corrected according to Eq. (2).

Figure 3.4: Comparing the effect of not correcting, and correcting, joint angles. The red circle with radius *gap* represents all the possible places the third line segment could begin, based on different joint angles.

To account for this, all joint angles are transformed using Eq. (2) to guarantee that the final angle will result in the center line segment having some minimum (non-zero) component parallel to the first line segment. See Fig. 3.4b for the result of using the transformed angles. Note that the length of the line does not change (it is still 21 pixels, the constant length of the gap); the transformation simply pushes *both* the blue and green lines farther from the extreme cases (0 and $\frac{\pi}{2}$ radians).

Following is the joint angle transformation equation. ja_{new} is the new joint angle and ja is the original joint angle from the gamma distribution, both in radians. gap is the gap distance, 21 pixels, and gap_p is the parallel gap constant, which was chosen as 13 pixels. per is the length of the perpendicular component of the middle line segment.

$$\begin{aligned} ja_{new} &= tan^{-1}(\frac{gap * sin(ja)}{gap_p}) \\ &= tan^{-1}(\frac{gap * \frac{per}{gap}}{gap_p}) \\ &= tan^{-1}(\frac{per}{gap_p}) \end{aligned} \tag{2}$$

Using Eq. (2), we can explicitly examine the affect of the transformation on the resulting parallel component of the middle line segment (par_{new}).

$$\begin{aligned} par_{new} &= gap * cos(ja_{new}) \\ &= gap * cos(tan^{-1}(\frac{per}{gap_p})) \\ &= \frac{gap}{\sqrt{(1 + (\frac{per}{gap_p})^2)}} \\ &\text{Since } per \leq gap \implies \\ &\geq \frac{gap}{\sqrt{(1 + (\frac{gap}{gap_p})^2)}} \\ &\text{Substitute in the constants' values:} \\ &= \frac{21}{\sqrt{(1 + (\frac{21}{13})^2)}} \\ &\approx 11 \end{aligned} \tag{3}$$

Thus, transforming the joint angle guarantees a reasonable parallel component length of 11, ensuring some level of equitable difficulty among the generated images. A small note about the tan^{-1} function. Its range is $(-\pi, \pi)$, meaning that it excludes some quadrants if used directly. However, python has an $atan2$ function that uses the sign of the numerator and denominator to correct for this, ensuring that ja_{new} can be in any of the 4 quadrants.

Since the element requiring classification in the Poggendorff illusion is not a series of connected line segments, but rather disjoint line segments, there is one additional permutation available during line generation. The lines described above have all been *connected* lines, but *disconnected* lines were also generated. For non-lines, this simply involved omitting the middle line segment. Disconnected lines are created by drawing a disconnected non-line with a joint angle of 0. (Note that transforming a joint angle of 0 does not change the joint angle; it is still 0.)

3.1.2 Generating an image

All images contained at least one primary element that determined the image's label as either containing a line or non-line. Depending on the test, images also contained some number of background elements according to different distributions. The more background elements, the more difficult it became to determine the primary element's classification.

One distribution used to determine the number of background lines was again a gamma distribution. Since the number of background lines must be an integer, the floor of the value is taken. So, it is still possible to have an image with no background elements, if the gamma distribution produced a number in the range $[0, 1)$. Three parameter choices were used:

1. $k = 3$ and $\theta = 0.5$, resulting in an average of 1.5 background elements

2. $k = 5$ and $\theta = 0.5$, resulting in an average of 2.5 background elements

3. $k = 9$ and $\theta = 0.5$, resulting in an average of 4.5 background elements

See Fig. 3.5 for graphs of these distributions. In all three cases, these background elements are randomly assigned to be either lines or non-lines (both of which are generated as described in Section 3.1.1). In order to distinguish the line of interest from these background elements, the background elements are drawn at 75% opacity (an RGB of (192,192,192)).

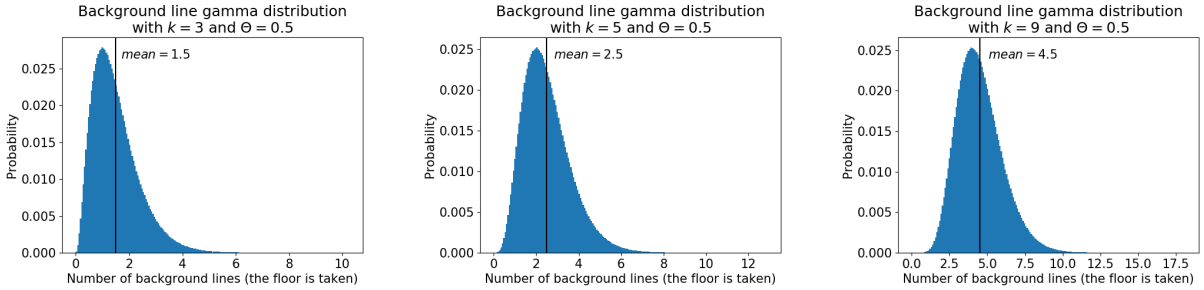


Figure 3.5: The gamma distributions used to determine the number of background elements.

To mirror the Poggendorff illusion's "background" lines, images are also generated with vertical lines placed at the joints between line segments. To ensure consistency with the Poggendorff illusion, images with too small a gap between the vertical lines are excluded (the gap_p from Section 3.1.1 value was used), as are any images whose middle line segment "criss-crossed" through the vertical gap (see Fig. 3.6). The vertical lines were drawn in at either 0% or 75% opacity, depending on the image set.

For a more generic version of the vertical distribution, images are also generated with a set, constant number of background elements at random lengths and positions (as described in Section 3.1.1). These lines are all chosen to be black (with no opacity), so the classification problem became one of finding if *any* of the lines in the image are non-lines.

Lastly, images are also generated explicitly with no background elements.

See Table 3.1 and Table 3.2 for sample line and non-line images from all the image sets used to retrain AlexNet. These sets were generated using permutations of the hyperparameters described in this section and Section 3.1.1. The abbreviations used are summarized in Section 3.2.2.

Figure 3.6: An example of a good (left) and bad (right) image with vertical background lines.

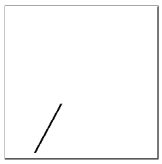
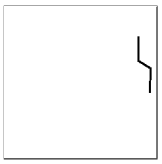
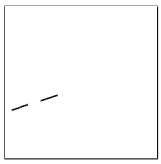
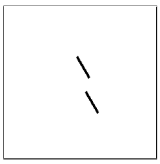
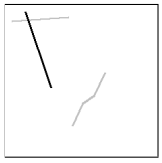
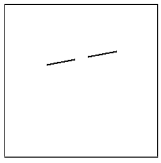
Hyperparameters				Connected lines		Disconnected lines	
ja	***bg***	bg_{colour}	***bg* properties**				
$ja_k = 6$, $ja_\theta = 0.75$	*single*	n/a	n/a				
	gamma	192 (75% opacity)	$bg_k = 3$, $bg_\theta = 0.5$				

Table 3.1: Sample line and non-line examples with all permutations that had joint angle parameters $k = 6$ and $\theta = 0.75$ used to retrain AlexNet. For each pair, the image on the left is an image containing a line, and the image on the right is an image containing a non-line.

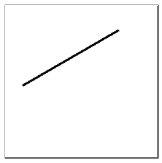
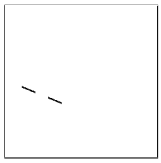
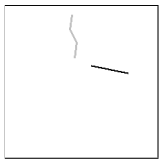
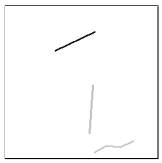
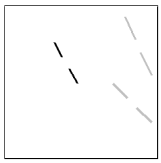
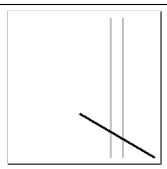
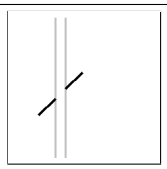
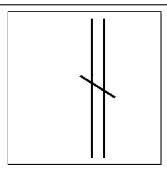
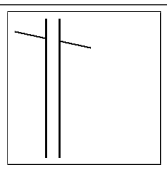
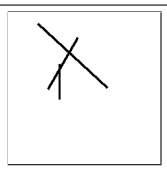
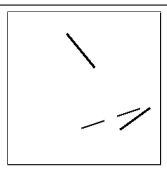
Hyperparameters				Connected lines	Disconnected lines
ja	***bg***	***bg*** $_{colour}$	***bg* properties**		
$ja_k = 6$, $ja_\theta = 0.4$	*single*	n/a	n/a		
	gamma	192 (75% opacity)	$bg_k = 3$, $bg_\theta = 0.5$		
			$bg_k = 5$, $bg_\theta = 0.5$		
			$bg_k = 9$, $bg_\theta = 0.5$		
	vert	192 (75% opacity)	n/a		
		0 (black)	n/a		
	const	0 (black)	$bg_{num} = 2$		

Table 3.2: Sample line and non-line examples with all permutations that had joint angle parameters $k = 6$ and $\theta = 0.4$ used to retrain AlexNet. For each pair, the image on the left is an image containing a line, and the image on the right is an image containing a non-line.

3.1.3 Generating a data set

Using this image generation technique, data sets are produced programmatically. Each data set is described by the line type (connected or disconnected), joint angle gamma parameters, and background distribution. Each data set is comprised of 3 sub data sets: a 10,000 image training set, a 2,000 image validation set, and a 2,000 image test set. In order to automatically label the images, the first half of each set is generated with a line as the primary element, and the second half used a non-line as the primary element. (The image order is randomly shuffled for training.)

3.2 Retraining AlexNet

3.2.1 Training methodology

Using datasets generated as described in Section 3.1, AlexNet is retrained according to 3 additional hyperparameters: the layers to be retrained; the final layer type and loss function used; and the learning rate.

The training consisted of a series of epochs. In each epoch, the network is trained on all 10,000 training images in random order in batches of 100. The training error is measured at the beginning of each epoch, on the first 100 randomly selected images. At the end of each epoch, the error on the entire set of 2,000 validation images is measured. The network is trained for a pre-set length of time, training for as many epochs as is possible before the time elapses. At that point, the network's error on the entire 2,000 test image set is measured. In cases where visual inspection of the validation error curves indicated the network could benefit from additional training epochs, training can be continued for some additional length of time.

3.2.2 Notes on hyperparameter abbreviations and errors

The charts and tables in this and later sections will use the following abbreviations when specifying the hyperparameters used during retraining:

- *layer* is the lowest layer retrained; it and all subsequent layers are retrained (see Section 3.2.3).
- lr is the learning rate (see Section 3.2.5).
- $final$ is the final layer used (either sigmoid or softmax; see Section 3.2.4).
- lt is the line type (either connected or disconnected, see Section 3.1.1).
- ja_k and ja_θ are the k and θ parameters for the gamma function used to generate the random joint angle (see Section 3.1.1).
- bg is the background distribution (see Section 3.1.2). This can be one of
  - *single* (bg_{single}): no background elements.
  - *gamma* (bg_{gamma}): gamma distribution. bg_k and bg_θ are the parameters for the gamma function.
  - *vert* (bg_{vert}): two vertical background lines placed at the joints between line segments.

– *const* ($bg_{constant}$): a constant number (bg_{num}) of randomly placed background lines.

For all *bg* values other than *single*, bg_{colour} indicates the colour of the background elements. It is an integer in the range [0,255] where 0 corresponds to black, and 255 to white. 75% opacity is taken to be $bg_{colour} = 0.75 * 256 = 192$.

To evaluate the performance of the retrained networks, three types of errors are collected:

- *loss error* is the error according to the network's loss function (see Section 3.2.4).
- *raw error* is a measure of how far the prediction probability is from the correct probability.
- *classification error* takes the most probable result (according to the model) and measures how often that is a correct prediction.

As can be seen in Fig. 3.7, $raw_{error} \approx classification_{error} \leq loss_{error}$. This inequality is expected, since the raw and classification errors are probabilities in the range [0,1], whereas the loss error does not represent a probability and thus does not have an upper bound. Since the result being measured is AlexNet's ability to correctly classify lines and non-lines, classification error is used throughout this analysis to measure the network's success at this task, and to compare performance under different hyperparameters and image sets.

When training any neural network, overfitting to the training data is a concern. Overfitting occurs when a network is trained so that it models the particular characteristics of the training data (including any noise or error present in this data), instead of modeling the actual underlying distribution. Though this decreases the training error, it causes the validation error to increase because the validation set does not have the training set's particular characteristics and noise.

Since overfitting applies only to data the network has seen during training, overfitting can be measured by looking at how the validation error curve by epoch. In particular, if validation error decreases, reaches some minimum, and then increases, this indicates overfitting. However, if the validation error decreases, reaches some minimum, and then stays relatively constant at that minimum, there is no evidence of overfitting. If a model exhibits overfitting, the state of the network at the last epoch *before* the validation error began increasing is used as the final model. The network is "rolled back" to a state before the it began overfitting. However, this rollback is not necessary if there is no evidence of overfitting.

It is important to note that in my results, loss error sometimes exhibited overfitting, but classification and raw error did not exhibited overfitting. This could be due to loss error's increased sensitive to small changes that don't greatly affect the predicted probabilities. Regardless of the cause, the performance of the network is based on its ability to correctly classify lines versus non-lines. Thus, the only error rate of concern when evaluating the final network is the classification error; the loss error is only used to learn more about *how* the network arrived at its final state.

Therefore, the decision of whether to rollback the model to an earlier epoch for final evaluations is not based on the presence of overfitting in any arbitrary error curve. Rather, the model is only rolled back if the *classification* error exhibited overfitting, since that was the quantity being measured. Since no classification error curve exhibited overfitting in any of my experiments, this is never required. This is particularly important for test error, which is only measured once at the end of each training session. Evaluating a network based on the classification error of the network's final state is only valid if there is no classification overfitting, as was the case in my tests.

See Fig. 3.8 for a typical example of error curves. Note that the validation loss curve exhibits overfitting after epoch ~ 100, but this does not affect the validation classification or raw errors, which do not exhibit overfitting.

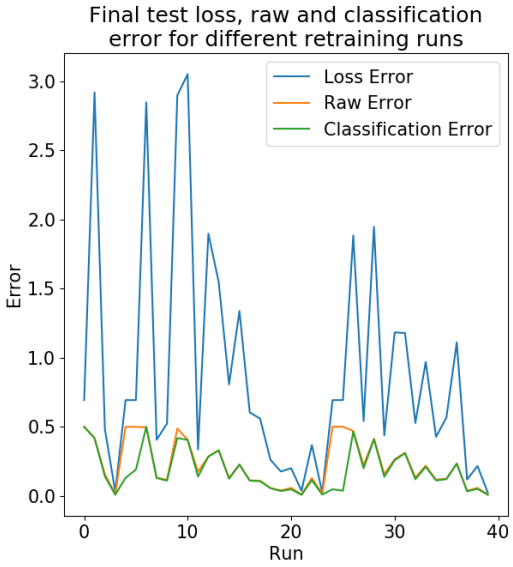


Figure 3.7: A comparison of the three types of errors across a set of runs. The runs' hyperparameters, ordering and relative error rates are not important for this graph (see Section 4 for the analysis of the results). Instead, it is just meant to demonstrate the error types' relative values.

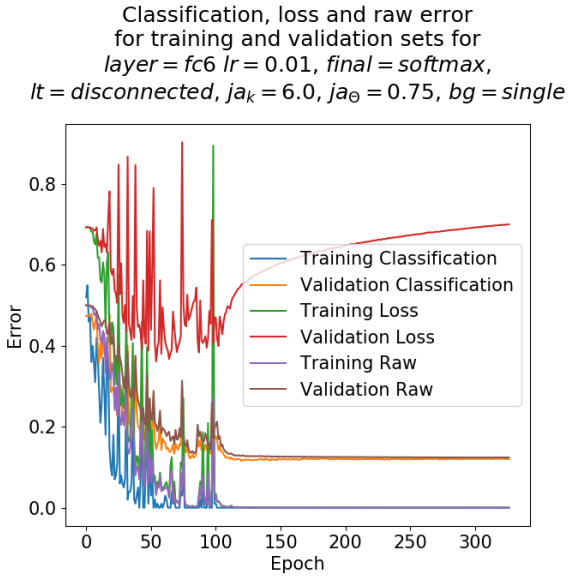


Figure 3.8: An example of the change in classification, loss and raw error for the training and validation image sets. Once again, the runs' hyperparameters and performance is not important here (see Section 4 for the analysis of the results). Instead, this figure is simply to demonstrate the presence of overfitting in the different types of error.

3.2.3 Layers to be retrained

When choosing which layers to retrain, it is important to note that the "AlexNet" network is not simply the architecture introduced in [15]. Instead, it is this architecture *as well as* the weights for that architecture learned through training on the ImageNet dataset. It was this final trained network that was shown to model the brains' inferior temporal cortex [14]. Thus, simply retraining the entire network does not show whether AlexNet is capable of classifying lines from non-lines.

Recent work has also gone into learning what type of information is stored at each level of CNNs. For AlexNet, simple elements, edges, colors and textures are processed primarily in the first layer, while region and surface information is processed in the third layer and object information is handled in the fifth layer [24, 19]. The final three fully connected layers use this information to determine classification probabilities [19]. (See Fig. 1.3 for a visualization of AlexNet's 8 layers.)

Thus, since we want to retain AlexNet's stored information while retraining it to solve a new classification problem, only the final three fully connected layers are considered for retraining. Retraining only some number of the fully connected layers is consistent with other recent research that retrained AlexNet to study its classification ability [1].

The final three fully connected layers are referred to as $fc6$, $fc7$ and $fc8$, where fc stands for *fully-connected*, and the number indicates the layer. In particular, experiments were run by retraining either layer $fc7$ ($layer = fc7$) or $fc6$ ($layer = fc6$), and all subsequent layers. So, $layer = fc7$ indicates that layers $fc7$ and $fc8$ are retrained. Similarly, $layer = fc6$ indicates that layers $fc6$, $fc7$ and $fc8$ are all retrained. In initial testing, there was no clear superiority in choosing $layer = fc7$ or $fc6$; they each outperformed the other depending on the image set used. So both permutations are tested for all experiments.

The original training method described in [15] used dropout to minimize overfitting. Dropout is a technique where hidden neurons' outputs are randomly set to 0 during training. For AlexNet, this meant randomly setting the output of each neuron in $fc6$ and $fc7$ to 0 with probability 0.5 during training. However, in my tests I found this did not significantly improve the results, and made training take much longer.

The effects of using dropout are compared on a hyperparameter choice that performed very poorly without dropout (and thus had high potential for overfitting). Without dropout, the training loss and classification error decrease to zero very quickly while the validation classification error remained around 0.4, which could indicate overfitting. However, the shape of the error curves do not change with dropout: training error once again decreases to zero while validation classification error remained around 0.4.

See Fig. 3.9a for graphs comparing the error curves with and without dropout. The most important features of this type of curve (which is used throughout this thesis) is to notice that the validation classification error has achieved a *learned state*. This means both that the curve has leveled out and that it does not exhibit overfitting. The curve being level, (ignoring inevitable noise,) indicates that no further training is required. All networks were trained in sessions of 2 hours until the network either achieves a learned state or it is clear that this was not possible.

Though these graphs are helpful for observing the curve's general shape, they do not allow for more accurate comparison between test sets' error data. For the latter, I will use tables that include three pieces of information from each experiment.

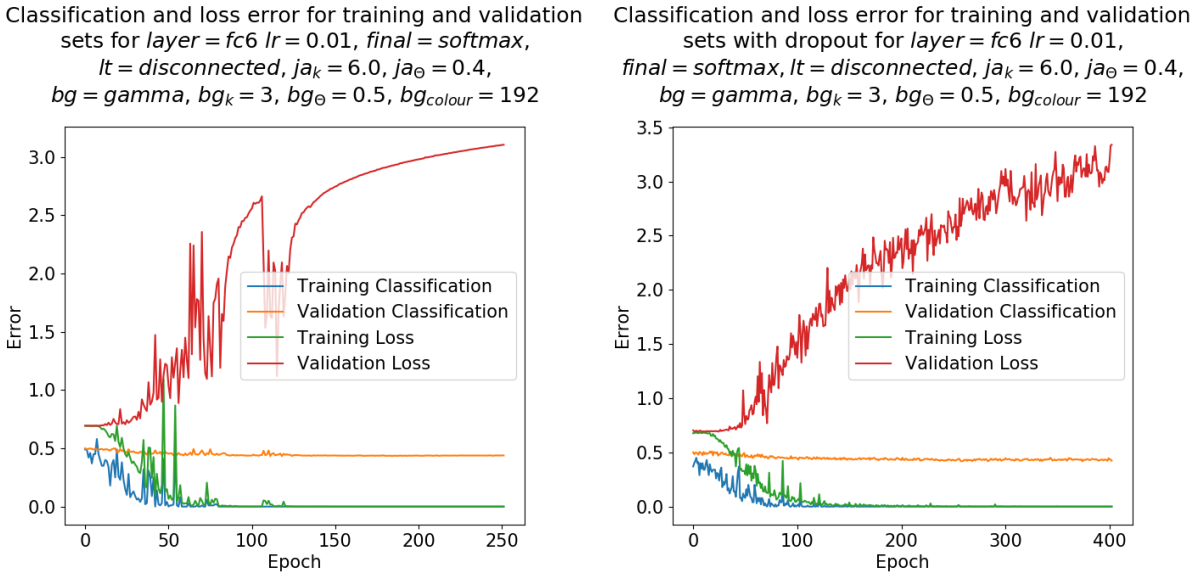
The first column, final test error, approximates the network's overall performance. It is always a classification error that is measured once at the end of the training session. Because of this, it is only valid when the validation classification curve does not exhibit overfitting, as is the case when it is in a learned state such as Fig. 3.9.

The second column, minimum validation error, shows the lowest validation error seen across all training epochs. Since it is the minimum, it is not susceptible to overfitting. However, because it is a minimum taken over hundreds of epochs, it does risk under-representing the true error of the network. It is shown with final test error to check for these effects, though in almost every case minimum validation error is only slightly smaller than final test error, indicating that the model is both not overfitting and that minimum validation error is not greatly under-representing true error.

The third column, epoch of min val error, is short for *epoch of minimum validation error*. It is the epoch at which the minimum validation error occurred. As such, it acts as an estimate for how quickly the network trained (in conjunction with the error curves).

Using these three metrics on the dropout comparison test, we can learn that the dropout network achieves slightly lower error, but at the cost of longer training. Final test and minimum validation classification errors are slightly lower in the dropout case by about 1.5%, but the minimum validation error occurred after more than 3 times as many epochs, indicating much slower training. Additionally, the 1.5% drop in test error represents only a 3.2% improvement (41.85% to 40.5%). Since the improvement is not substantial and took

much longer, dropout is not used for any experiments. See Fig. 3.9 for the error curves and Table 3.3 for the metric values.



(a) Result of not using dropout.

(b) Result of using dropout.

Figure 3.9: Comparison of errors when not using (left), and using (right), dropout. Note that the classification error curves are very similar, as are the loss curves, except for the large dip in the no-dropout validation loss curve.

	Final test error	Minimum validation error	Epoch of min val error
Without dropout	0.4185	0.4335	111
With dropout	0.4050	0.4175	374

Table 3.3: Comparison of errors when not using, and using, dropout. Hyperparameters are chosen based on the experiment from Section 4 that performed the worst, and so had the most opportunity to improve. In particular, the test is run with a learning rate of 0.01, a final softmax layer, a joint angle k of 6, and a joint angle θ of 0.4. Background lines are generated based on a gamma distribution with a k of 3 and θ of 0.5. The background element colour was 192 (75% opacity). Classification error is used throughout.

3.2.4 Final layer choice and loss function

Classifying an image as containing a line or non-line is a binary decision, (instead of a 1000 category decision,) and so the final layer's output shape must be modified. Two choices are considered: outputting a single value and using a *sigmoid* function to give a probability score, or outputting two values and using a $softmax$ function to give two probabilities that sum to 1.

In the *sigmoid* case, the model's single output value is run through the *sigmoid* function to produce AlexNet's confidence that the image is a line, and log-loss is used as the network's loss function. In the $softmax$ case, the model's 2 outputs are passed through the $softmax$ function, and represent AlexNet's confidence that the image contains a non-line or line, respectively. Here, softmax cross-entropy is used as the loss function.

The $softmax$ networks greatly outperforms the *sigmoid* networks for retraining down to both the $fc7$ and $fc6$ layer. This is despite the redundancy of the data stored in the 2-vector (since the values sum to 1) and

the *sigmoid* and $softmax$ functions mathematical similarity. In both experiments, the final test error and minimum validation error are smaller in the $softmax$ networks by factors of $5 - 21$; see Table 3.4 for the values.

Hyperparameters		**Final test error**	**Minimum validation error**	**Epoch of min val error**
layer	***final***			
$fc7$	*sigmoid*	0.0480	0.0260	373
	$softmax$	0.0075	0.0050	59
$fc6$	*sigmoid*	0.1305	0.1060	324
	$softmax$	0.0075	0.0050	32

Table 3.4: Comparison of using a *sigmoid* versus a $softmax$ layer as the final output layer. The hyperparameters match those of the charts in Fig. 3.10. Hyperparameters are chosen to make the classification task easy. All variable hyperparameters are listed; all entries had a connected line type, a joint angle k of 6, a joint angle θ of 0.75, no background elements, and a $lr = 0.01$ (see Section 3.2.5). Classification error is used throughout.

However, the error curves seem to indicate that while the $softmax$ networks achieve a learned states, the *sigmoid* networks do not. Their validation curves do not fully level out, and their minimum validation error indices are both near the end of training, indicating that the error rates could drop with further training. This is related to $softmax$'s second advantage over *sigmoid*: training speed. Both *sigmoid* networks trained for 4 hours without converging to learned states, as opposed to the $softmax$ networks which both converge in less than 2 hours (particularly the $fc6$ network). Additionally, the *sigmoid*s' training classification error is much more unstable, and the loss errors show very little movement, indicating that network is not learning well.

See Fig. 3.10 for the error curves comparing the two different final layers. In these figures, the *sigmoid* loss curves appear linear. Looking at only these loss curves shows that the training loss error is much more erratic than the validation loss error, but neither curve demonstrates much learning. Unlike the $softmax$ loss curves, neither curve decreases stably. See Fig. 3.11 for the loss-only curves.

Based off these results, all subsequent tests are run using $softmax$. The final output of AlexNet is reshaped to be a 2-dimensional vector which is passed through the $softmax$ function, and softmax cross-entropy is used as the loss function.

3.2.5 Learning rate

To train the network, stochastic gradient descent is used to minimize over the softmax cross-entropy loss function. This is the same optimizer and loss function used to train the original AlexNet in [15]. Gradient descent requires a learning rate, which is another hyperparameter. The authors of AlexNet started with a learning rate of 0.01 and manually decreased the learning rate by a factor of 10 when they found the network was no longer learning [15]. Consequently, learning rates of 0.01 and 0.001 are compared for retraining.

In three of the four experiments comparing these learning rates, a learning rate of 0.01 outperform 0.001 in every category. They achieve lower test error and minimum validation error, though the values are very close. More significant is the far fewer number of epochs in which networks using a learning rate of 0.01 achieve this minimum. For the one experiment where a learning rate of 0.001 outperforms 0.01, the error rates are again very close (0.113% versus 0.1205%) but the minimum validation is achieved almost 4 times slower for the network using 0.001. See Table 3.5 for all of these metrics.

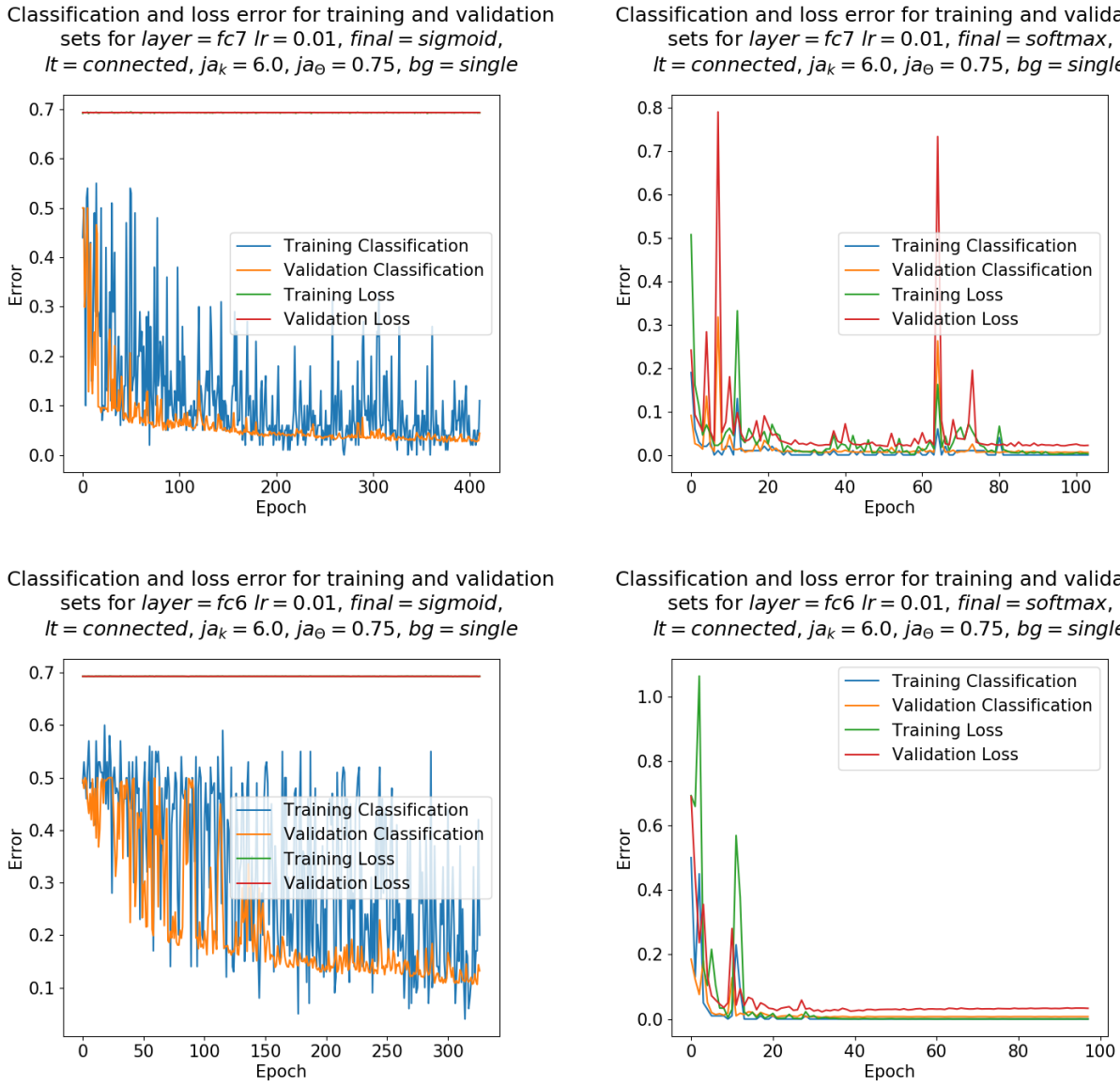


Figure 3.10: Comparison of using a *sigmoid* (left) versus a *softmax* (right) layer as the final output layer (*layer*) of the retrained network. The *sigmoid* networks are trained for 4 hours in 2 2-hour sessions, whereas the *softmax* networks are trained in only 2 hours. Note that the network using the *sigmoid* layer both has much smaller validation classification error, and converges much faster.

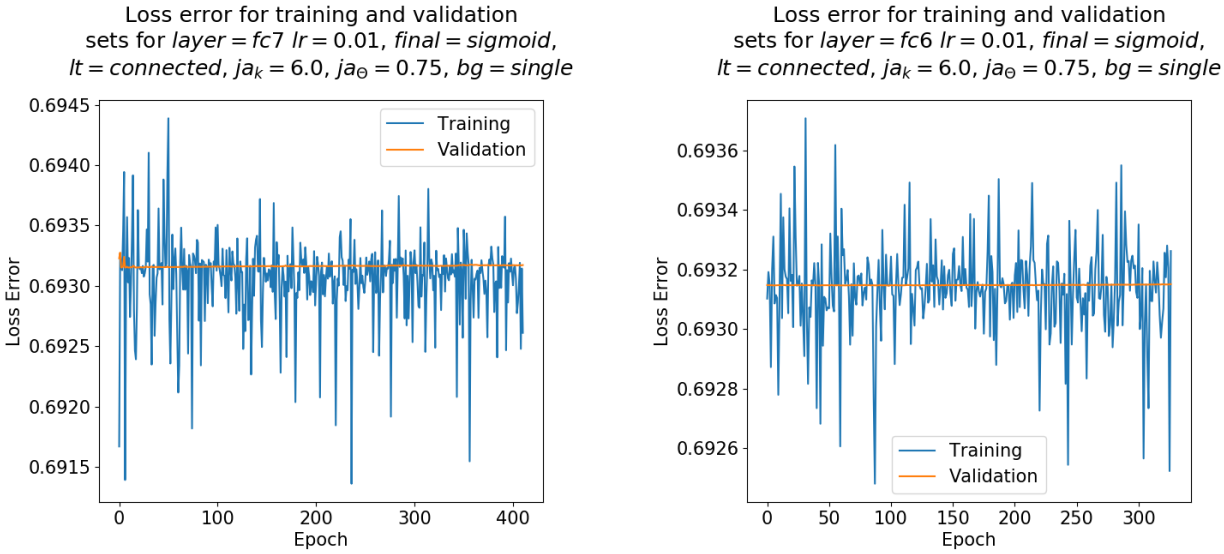


Figure 3.11: Loss error for the two *sigmoid* networks from Fig. 3.10. Note that y-axes' scales.

Hyperparameters			Final test error	Minimum validation error	Epoch of min val error
layer	***lt***	***lr***			
$fc7$	*connected*	0.001	0.0095	0.0055	143
		0.01	0.0075	0.0050	59
	disconnected	0.001	0.1130	0.1050	500
		0.01	0.1205	0.1145	129
$fc6$	*connected*	0.001	0.0080	0.0065	41
		0.01	0.0075	0.0050	32
	disconnected	0.001	0.1415	0.1390	226
		0.01	0.1240	0.1150	120

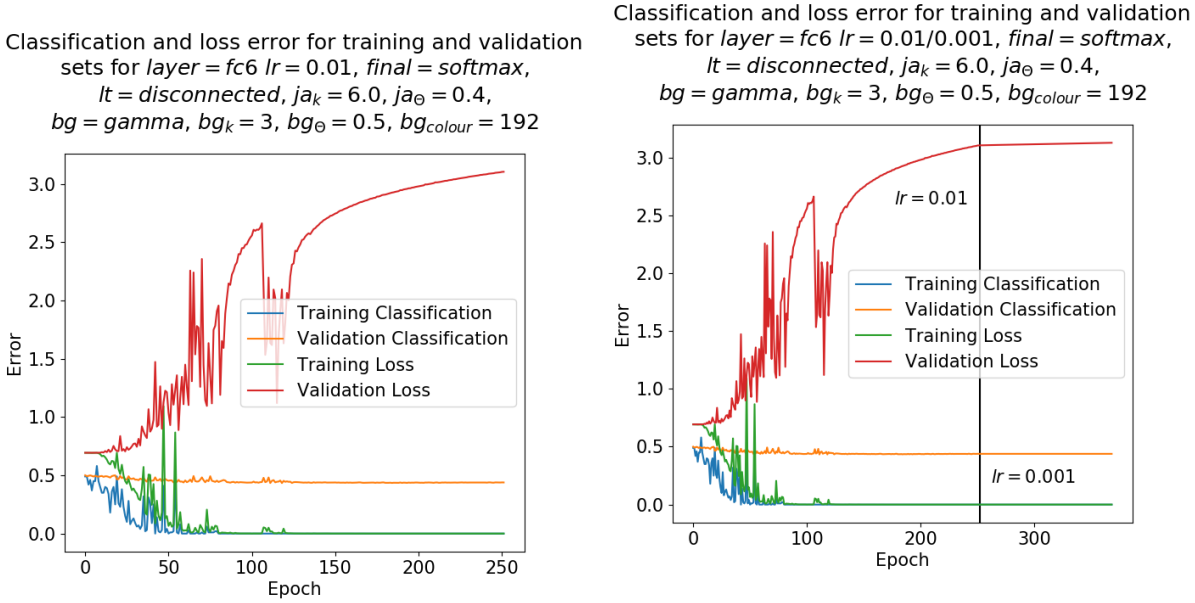
Table 3.5: Comparison of using a learning rate (lr) of 0.001 versus 0.01 during gradient descent. All variable hyperparameters are listed; all entries had a joint angle k of 6, a joint angle θ of 0.75, no background elements. Classification error is used throughout. For the error graphs, see Fig. A.2 for $layer = fc7$ and Fig. A.3 for $layer = fc6$.

Since a learning rate of 0.01 performs better in most of the tests, and always achieves its minimum validation error faster (and thus appears to have trained faster), a learning rate of 0.01 is used for all subsequent experiments. The error curves for these experiments are not particularly helpful in comparing the different learning rates. When this is the case, I have placed the error curves in the Appendix. For these experiments, the curves can be found in Appendix A.2; see Fig. A.2 and Fig. A.3. It should be noted from these graphs that all the networks achieved a learned state.

Recall that the original method used for training AlexNet described in [15] not only used a learning rate of 0.01 initially, but then dropped this learning rate by a factor of 10 manually when the network was found to no longer be learning. However, in my tests I find that this drop has a negligible impact on the results.

As in the dropout comparison from Section 3.2.3, the effects of dropping or not dropping the learning rate are compared on a hyperparameter choice that performs very poorly without dropping the learning rate, but had achieved a learned state. This would theoretically allow for large improvements if dropping the learning rate could help the network learn. After training the initial version with a learning rate of 0.01 for 2 hours, the model is trained for an additional 2 hours with the learning rate dropped to 0.001. The results indicate that this drop does not aid training. The minimum validation error does not change; it occurs during the first 2 hours of training (at a learning rate of 0.01) for both networks. The final test error only drops 0.1%, which is negligible and could be explained by random noise. In the error graphs, there is also no visible change in the classification curves. Based on this result, the learning rate is not dropped during retraining for any experiment.

See Table 3.6 for the error and epoch values comparing retraining with and without a drop, and see Fig. 3.12 for the error curves.



(a) Result of not dropping the learning rate.

(b) Result of dropping the learning rate by a factor of 10 after reaching a learned state (indicated by the vertical black line).

Figure 3.12: Comparison of errors when not dropping (left), and dropping (right), the learning rate.

	Final test error	**Minimum validation error**	**Epoch of min val error**
Without lr drop	0.4185	0.4335	111
With lr drop	0.4175	0.4335	111

Table 3.6: Comparison of errors when not dropping, and dropping, the learning rate while training. Hyperparameters are chosen based on a poorly performing network from Section 4, and so had the a lot of room to improve. In particular, the test is run with a a final softmax layer, a joint angle k of 6, and a joint angle θ of 0.4. Background lines are generated based on a gamma distribution with a k of 3 and θ of 0.5. The background element colour is 192 (75% opacity). Both tests are trained with a learning rate of 0.01 for 2 hours, with the second run then dropping the learning rate to 0.001 and training for another 2 hours. Classification error is used throughout.

4 Results

36 experiments are run on different permutations of the hyperparameters described in Section 3. These are the 18 image set permutations shown in Table 3.1 and Table 3.2 with the added $layer = fc7$ and $fc6$ permutation. In general, the results are ordered from "easier" classification problems to "harder" classification problems, based on the average joint angle size and number of background elements. This is the same order that the image sets appear in Table 3.1 and Table 3.2.

Though not all permutations are tested, every experiment is run with both *connected* and *disconnected* line types, and with retraining down to the $fc7$ and $fc6$ layer.

4.1 Experiments with large joint angles

As described in Section 3.1.1, two sets of gamma parameters for generating joint angles are used. Both have a k of 6, but one has a θ of 0.75 and the other uses 0.4. After scaling, these distributions have mean joint angles of 0.71 and 0.38 rad, respectively. Since distinguishing lines from non-lines is easier when the difference is more distinct, the distribution with the larger mean joint angle is considered the "easier" distribution. This larger distribution is used in this first experiment to determine AlexNet's baseline capabilities.

The large joint angle distribution is run with a number of different hyperparameters. It is tested with no background elements ($bg = single$) and background elements generated using a gamma distribution ($bg = gamma$) with $k = 3$ and $\theta = 0.5$. A background colour of 192 (75% opacity) is used for all background lines. As with all the experiments, they are tested with both connected and disconnected lines and with retraining down to the $fc7$ and $fc6$ layers. Thus, there are 8 tests in this experiment.

Under the simplest scenarios with connected lines and no background elements, it is clear that AlexNet can distinguish lines from non-lines; it achieves $< 1\%$ test and validation classification error. Even in the presence of background elements, the network still achieves an impressive $\approx 5\%$ error rate. Note though that with $bg_k = 3$ and $bg_\theta = 0.5$ there are only an average of 1.5 background elements, and this valued is floored to get the number of background elements in each image.

AlexNet has much more difficulty with disconnected lines. It has $\approx 12\%$ error with no background elements and achieves a learned state much slower; it takes over twice as many epochs until the minimum validation error is reached as compared to the connected case. The results with background elements are even worse, with classification errors over 30%. The disconnected line case is the permutation that mimics the setup of the Poggendorff illusion. This large error rate indicates that AlexNet cannot distinguish lines from non-lines well in even moderately difficult circumstances that mimic this setup. Distinguishing lines from non-lines is a prerequisite for the Poggendorff illusion, and these disconnected line results indicate that AlexNet is not capable of satisfying the prerequisite.

The far worse performance in the disconnected case is a pattern seen in almost every experiment. The permutation that most affects the error rates is consistently the choice of line type. This could be due AlexNet's relative ability to use local versus global information during classification.

As noted in Section 3.2.3, AlexNet's layers process progressively more abstract information. Lower layers deal with local features like texture and edges, while later layers handle more abstract concepts like objects and scenes [24, 19]. Thus, though AlexNet is clearly capable of interpreting more abstract, global features, its processing begins with local features.

From these results, it appears that AlexNet does not just start with local features, but is also better at using these local features in its final classification decision. By generating lines according to the method described in Section 3.1.1, the only difference between a line and non-line is the joint angle. This angle is zero for lines, and non-zero for non-lines. When there is a middle line segment, this angle can be determined locally; the angle can simply be measured at the joint between line segments without using any global image information. However, by removing the middle line segment, which is done for disconnected images, the information required to measure the joint angle locally is also removed (see Fig. 4.1).

(a) Connected image; the network has local information at the joints

(b) Disconnected image; the network has no local information

Figure 4.1: Comparison of local information in connected and disconnected images.

Thus, a primary difference between connected and disconnected lines is the ability to distinguish lines from non-lines using only local information. The disconnected experiments' much higher error rates, then, seem to indicate that AlexNet is worse at utilizing global information than local information when classifying images.

See Table 4.1 for the results of this experiment. The error curves for these experiments are not particularly useful, except to note that every model appears to have achieved a learned state. The curves can be found in Appendix A.3; see Fig. A.5 and Fig. A.4.

Hyperparameters			**Final test error**	**Minimum validation error**	**Epoch of min val error**
lt	***bg***	***layer***			
connected	*single*	$fc7$	0.0075	0.0050	59
		$fc6$	0.0075	0.0050	32
	gamma	$fc7$	0.0510	0.0520	48
		$fc6$	0.0475	0.0505	42
disconnected	*single*	$fc7$	0.1205	0.1145	129
		$fc6$	0.1240	0.1150	120
	gamma	$fc7$	0.3090	0.2965	148
		$fc6$	0.3295	0.3085	66

Table 4.1: Results of large joint angle experiments. All hyperparameters that vary between experiments are listed. All experiments use a learning rate of 0.01, a final softmax layer, a joint angle k of 6, and a joint angle θ of 0.75. For $bg = gamma$ experiments, a k of 3 and θ of 0.5 was used. The background element colour was 192 (75% opacity). Classification error is used throughout. For the error graphs, see Fig. A.5 for $lt = connected$ and Fig. A.4 for $lt = connected$.

4.2 Experiments with small joint angles and no background elements

All subsequent experiments in this and later section are run on image sets generated using a joint angle distribution with the smaller mean. For these experiments, the a k of 6 and a θ of 0.4 was used for the joint angle gamma distribution, resulting in a mean of 0.38 rad after scaling. This is about half the average joint angle compared to using a θ of 0.75 as in Section 4.1. This smaller joint angle makes the difference between lines and non-lines smaller, which can be expected to make the differentiation task more difficult.

The first set of experiments with these smaller joint angles is conducted with no background elements. As for all experiments, they are run with both connected and disconnected lines, and with retraining down to layer $fc7$ and $fc6$. As expected, all of the results are worse than with a $ja_\theta = 0.75$. For the connected lines, the results are still quite good, with $< 5\%$ test classification error. However, for disconnected lines, the results are much worse, with $> 25\%$ test classification error. As in Section 4.1, AlexNet has a lot more difficulty distinguishing lines from non-lines on images without local differentiating information.

See Table 4.2 for the results. The error curves for these experiments are once again not useful, except to note again that every model appears to have converged to a learned state. The curves can be found in Appendix A.4; see Fig. A.6.

Hyperparameters		**Final test error**	**Minimum validation error**	**Epoch of min val error**
lt	***layer***			
connected	$fc7$	0.033	0.0245	164
	$fc6$	0.0355	0.0325	87
disconnected	$fc7$	0.2580	0.2485	181
	$fc6$	0.2845	0.2835	168

Table 4.2: Results generated using small joint angles and no background elements. All hyperparameters that vary between experiments are listed. All experiments used a learning rate of 0.01, a final softmax layer, a joint angle k of 6, a joint angle θ of 0.4, and had no background elements. Classification error is used throughout. For the error graphs, see Fig. A.5 for $lt = connected$ and Fig. A.4 for $lt = connected$.

4.3 Experiments with small joint angles and gamma background element distributions

Background lines are added for the next round of experiments according to different gamma distributions. The distributions all use a $bg_\theta = 0.5$ but use three different bg_k values: 3, 5 and 9. This results in image sets with progressively more background elements; they have an average of 1.5, 2.5 and 4.5 background lines, respectively. Since the number of background elements has to be an integer, the floor of the random variable produced by the distribution is taken. Once again, they are run with both connected and disconnected lines, and with retraining down to layer $fc7$ and $fc6$.

The results are similar to those seen previously. Adding more background elements consistently increases the error rates, but by far the biggest impact on classification error is the use of disconnected lines. Having an average of 1.5 background elements ($bg_k = 3$) results in classification errors around 40% for disconnected lines. Increasing this background element average to 4.5 ($bg_k = 9$) increases the classification error to over 48%, very close to the logical limit of 50% where the network is randomly guessing. AlexNet is clearly unable to distinguish disconnected lines from non-lines in the presence of around 4 background elements.

Increasing the number of background lines in the disconnected case degrades AlexNet's performance up to the worst possible error rate of 50%. However, it is unclear if the same is true for connected lines. Plotting these errors against the average number of background lines shows an increasing, somewhat linear trend

that does not appear to flatten. Further testing should be done with more background lines to determine if AlexNet's error plateaus at some maximum value less than 50% for connected lines.

It is interesting to note that the worse error rate observed for connected images, 22% with $bg_k = 9$, is still better than the best error rate for disconnected lines, 26% with no background lines. Regardless of the connected case's worst case error, it is clear that the presence of local features that can help differentiate lines from non-lines significantly helps AlexNet classify images, at least for images with 0 to around 5 background lines.

See Fig. 4.2 for the graph of error rates across different number of background lines, and see Table 4.3 for the results of the experiments. The error curves for these experiments are once again not useful, except to note that again every model appears to have converged to a learned state. The curves can be found in Appendix A.5; see Fig. A.7, Fig. A.8 and Fig. A.9.

Hyperparameters			Final test error	Minimum validation error	Epoch of min val error
bg_k	*lt*	*layer*			
3	*connected*	$fc7$	0.1210	0.1205	242
		$fc6$	0.1065	0.1065	32
	disconnected	$fc7$	0.4090	0.4150	292
		$fc6$	0.4185	0.4335	168
5	*connected*	$fc7$	0.1650	0.1635	41
		$fc6$	0.1445	0.1465	33
	disconnected	$fc7$	0.4520	0.4415	96
		$fc6$	0.4680	0.4595	13
9	*connected*	$fc7$	0.2400	0.2370	36
		$fc6$	0.2230	0.2230	42
	disconnected	$fc7$	0.482	0.4650	81
		$fc6$	0.4885	0.4720	31

Table 4.3: Results with small joint angles and gamma background element distributions. All hyperparameters that vary between experiments are listed. All experiments use a learning rate of 0.01, a final softmax layer, a joint angle k of 6, and a joint angle θ of 0.4. Background lines are generated based on a gamma distribution with k as listed above and a constant θ of 0.5. The background element colour is 192 (75% opacity). Classification error is used throughout. For the error graphs, see the figures in Appendix A.5.

4.4 Experiments with small joint angles and different background element distributions

In order to return the analysis to the original question of AlexNet's perception of the Poggendorff illusion, experiments are also run with more Poggendorff-like images. This involves training on images with vertical background lines ($bg = vert$), both at 75% opacity ($bg_{colour} = 192$), and with completely black lines ($bg_{colour} = 0$). To provide a comparison with these results, experiments are also run with 2 randomly placed and orientated black background lines ($bg = const$, $bg_{colour} = 2$). Once again, all of these experiments are run with both connected and disconnected lines, and with retraining down to layer $fc7$ and $fc6$.

The image sets constructed using $bg = gamma$ and $bg_k = 5$ have an average of 2.5 background lines, which, after taking the floor, means that most images have roughly 2 background lines. Thus, it might be expected that the image sets studied in these experiments, particularly the ones with $bg_{colour} = 192$, would result in similar error scores to those seen in Table 4.3 for $bg_k = 5$. Recall that the connected images for $bg_k = 5$ results in an error of $\approx 15\%$ and the disconnected images have an error of $\approx 45\%$.

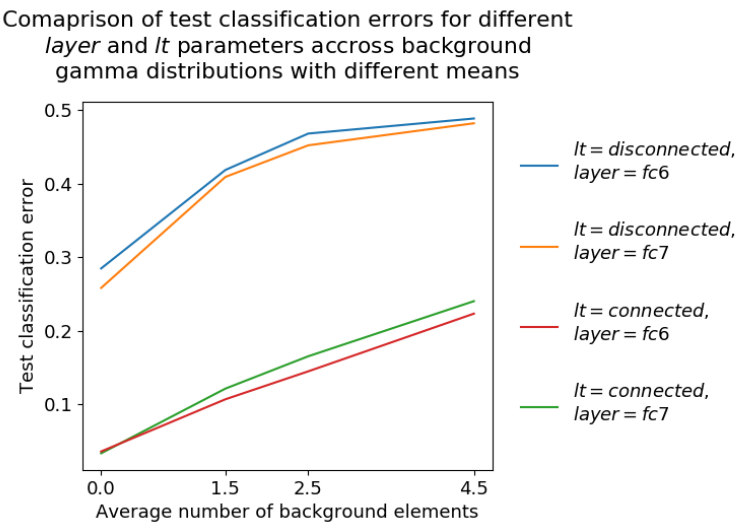


Figure 4.2: Comparison of test classification error rates across background distributions with different means. The data is from Table 4.2 and Table 4.3. All networks are trained with a learning rate of 0.01 and a final softmax layer; the lowest layer of training is listed above. The networks are trained on data sets that are generated with a joint angle k of 6 and a joint angle θ of 0.4. Data sets are generated with either no background elements (for the 0 case) or based off a gamma distribution whose mean corresponds to the value along the x-axis. These gamma distributions all use a $bg_\theta = 0.5$ and a bg_k of 3, 5 and 9 from left to right. The background element colour is 192 (75% opacity).

However, the results for vertical background lines are much better for both the connected and disconnected sets. The classification test errors are about 5% and 10%, respectively, much lower than the 15% and 40% errors seen using a gamma distribution. Even drawing the background lines in black produces better results than the $bg_k = 5$ case; the errors are 11% for both the connected and disconnected line types for the $fc6$ layer.

It should be noted that the black vertical results are outliers from the rest of the results for a couple reasons beyond their impressive performance. First, for $layer = fc6$, it is the only test where moving from connected to disconnected lines does not dramatically increase the classification error. Second, the disconnected experiment is also the only test where training down to layer $fc6$ and $fc7$ produce very different results; $fc6$ has about half the error seen for $fc7$. Third, the difference in $fc7$'s test and validation error for disconnected lines is about double what is usually seen. 2-3% is normal, but here the difference is about 5%.

There are a number of differences between the image sets in the $bg = vert$ and $bg = gamma$ image sets that could be the cause for the discrepancy in the experiments' results.

1. The *vert* experiments have a constant number of background lines; the *gamma* experiments have a random number of background lines that changes image to image.

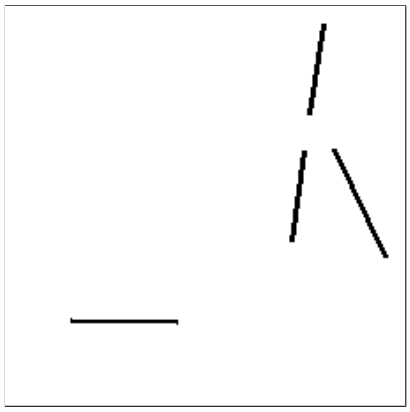
2. The *vert* experiments have images that contain only connected background lines; the background elements in the *gamma* experiments are randomly either lines or non-lines, and are connected or disconnected based on the permutation.

3. Since the vertical background lines are always placed at the joints between the line segments of the primary element, and since they always have the same height and y position in the image, these background lines are at a consistent position relative to both the canvas *and* the primary element. In the the *gamma* experiments, there is no such consistency.

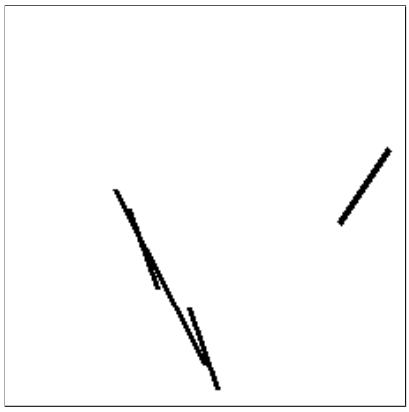
To test the significance of differences (1) and (2) on the results, experiments with $bg = const$ are run. For these tests, two connected lines are used as background elements (even when the primary element is disconnected). This results in image sets that, like in the $bg = vert$ experiments, have exactly 2 background elements that are always connected lines. These tests are run with $bg_{colour} = 0$ (so the background lines are black) to compare with the black vertical background line tests.

The experiments with these background lines perform very similarly to the connected gamma background line distribution cases with $bg_k = 9$, with an error of about 21% for connected lines and 46% for disconnected lines. With $bg_k = 9$, the average number of background lines is 4.5, more than double the constant 2 background lines present in this experiment, so it is surprising that AlexNet performs so poorly. The background lines being drawn in black rather than at 75% opacity are perhaps the cause of this drop in performance.

As an aside, the poor performance in this experiment for disconnected primary elements with black background lines can at least be partly explained by peculiarities in the image set. Black background lines can make it confusing which lines on the page are being evaluated (as in Fig. 4.3a) and black background lines can obfuscate the element being classified (as in Fig. 4.3b). Granted, these issues are not present for every image, and the consistent gap length makes finding the elements to classify possible in almost every image. However, this image set is noticeably more difficult than the previous ones studied, which helps explain the surprisingly poor performance. However, these same arguments do not apply to the connected case, which also performed poorly.



(a) An image labeled as a line. Note that the rightmost line segment makes it confusing which primary disconnected line is being evaluated.



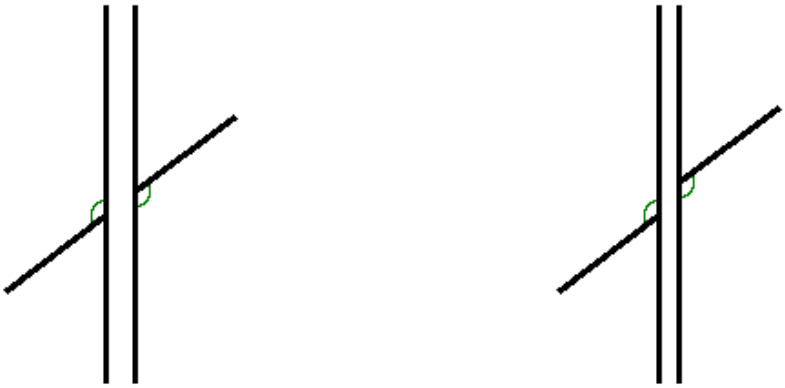
(b) An image labeled as a non-line. Note that since all lines are drawn in black, background lines can obfuscate the primary element being classified.

Figure 4.3: Examples of problematic images for $bg = const$ and $bg_{colour} = 0$.

Due to the $bg = const$ experiment's poor performance, I conclude that differences (1) and (2) from above did not cause AlexNet's unexpectedly low error rates on images with vertical background lines. Instead, it would appear as though the background lines' consistent placement relative both to the canvas and the primary element improves AlexNet's ability to classify these images. This is perhaps because the consistent placement results in the background lines containing some information useful for classification, whereas the randomly placed background elements are only random noise that can confuse the network. Additionally,

the added checks that ensure a minimum gap and no "criss-cross" through the vertical gap may result in test sets that are easier to classify. See Section 3.1.2 for the description of how the image sets with vertical background lines are generated.

It should be noted, though, that adding vertical lines does add local differentiating information for disconnected images. Due to the construction of non-lines, the angle between the vertical and the primary line segments are identical for line and non-line images; see Fig. 4.4. Thus, the impressive disconnected results cannot be easily explained. One possible explanation is that the distance between the verticals is used to determine if line segments are aligned. Though this information does not directly differentiate lines from non-lines and is present in the primary element itself, it is more readily apparent with vertical background lines. This and the other irregularities listed above for the disconnected case are an area for future investigation.



(a) Disconnected line **(b)** Disconnected non-lines

Figure 4.4: Comparison of local information for disconnected elements with vertical background lines. Note that the only local information, the angles with the verticals, are identical for both the line and non-line cases.

See Table 4.4 for the results of all the experiments discussed above. The error curves for these experiments are once again not useful, except to note that every model appears to have converged to a learned state, at least roughly. For many of the vertical background line plots, the error rate curves are very erratic, and even after many epochs do not stabilize. However, for all the graphs, the final validation classification error looks to be roughly equal to a learned state. The curves can be found in Appendix A.6.

4.5 Classifying Poggendorff images with retrained networks

4.5.1 Experiments with networks retrained with images containing vertical background lines

With these newly trained networks, it is possible to directly classify Poggendorff images to test AlexNet's perception of the illusion. The AlexNet networks retrained on connected and disconnected image sets with vertical background lines are used to classify 144 connected and disconnected Poggendorff images, similar to the test sets used during exploratory testing (see Section 2.4). There are only two differences between the image sets used for this experiment and those used during exploratory testing.

First, in Section 2.4, a horizontal gap of 21 pixels is used. Here, a gap of 21 pixels is again used, but this time measured parallel to the diagonal line (to match the method used when training the network). Since the line was at a 55° angle (the same as in Section 2.4), this resulted in a horizontal gap of 12 pixels.

Hyperparameters				Final test error	Minimum validation error	Epoch of min val error
bg	***bg$_{colour}$***	***lt***	***layer***			
vert	192	*connected*	$fc7$	0.0530	0.0490	207
			$fc6$	0.0555	0.0505	86
		disconnected	$fc7$	0.1385	0.1335	134
			$fc6$	0.1025	0.1075	106
	0	*connected*	$fc7$	0.1115	0.0920	166
			$fc6$	0.1100	0.0945	119
		disconnected	$fc7$	0.2000	0.1540	162
			$fc6$	0.1100	0.1105	143
const	0	*connected*	$fc7$	0.2105	0.1995	23
			$fc6$	0.226	0.2120	34
		disconnected	$fc7$	0.4645	0.4770	69
			$fc6$	0.5005	0.4820	13

Table 4.4: Results with small joint angles and different background element distributions. All hyperparameters that vary between experiments are listed. All experiments used a learning rate of 0.01, a final softmax layer, a joint angle k of 6, and a joint angle θ of 0.4. For $bg = vert$, vertical background lines were used with a colour indicated by bg_{colour}. For $bg = const$, 2 randomly placed and orientated background elements were used. Classification error is used throughout. For the error graphs, see the figures in Appendix A.6.

Second, the margins of the image are reduced from 31 pixels in exploratory testing to 10 pixels for this test set to again match the images used during training. (The training images use a smaller margin to minimize the chance of lines being drawn out of bounds and then requiring translation, see Section 3.1.1).

These two changes moved the test image containing the straight diagonal line from index 75 in the exploratory testing set to index 81 for these test set experiments.

Using this technique, two test sets are created. The *connected set* contains connected images like those used during exploratory testing (with the two modifications mentioned above). The *disconnected set* is identical to the connected set, except that its diagonal (primary element) is disconnected. Note that the original Poggendorff illusion in Fig. 1.1 would be classified as disconnected. Thus, only experiments using the disconnected set can be expected to exhibit Poggendorff effects.

These test sets have an important property. The image sets used to retrain AlexNet uses a gamma distribution to generate joint angles. This results in almost no joint angles very close to 0 for non-lines. However, the test sets used here are generated by simply translating the right arm of each diagonal up one pixel for each image index. This translation results in very small joint angles close to index 81 (the index of the straight diagonal). For example, the joint angle at index 80 is 0.03 rad ($\approx 2°$) and the joint angle at index 82 is 0.02 rad ($\approx 1°$). Thus, uncertainty or misclassifications can be expected near index 81 due to the image's small joint angles.

The range of this uncertainty, if present, is important. The original Poggendorff illusion in Fig. 1.1 causes the aligned line segments to appear misaligned because the right arm appears too high relative to the left arm. To make a straight line, then, would require moving the right arm down, which corresponds to a lower index in the test sets. Thus, if the range of uncertainty is disproportionately shifted to the left (to lower indices) this *could* indicate that AlexNet is susceptible to the effects of the Poggendorff illusion, but only for experiments using the disconnected set.

In the context of examining the position of uncertainty regions, it should also be noted that the joint angle does not change linearly with index in these test sets. Instead, the joint angle changes according to an inverse tan function. Since index 81 occurs to the right of the center of this function, the joint angle changes much more slowly as the index increases from 81, than it does as the index decreases from 81. This could result in uncertainty ranges that appeared skewed to the right about index 81 when examined by image index, but are symmetrical about index 81 when examined by joint angle. See Fig. 4.5 for a graph of the joint angle by index for both image sets.

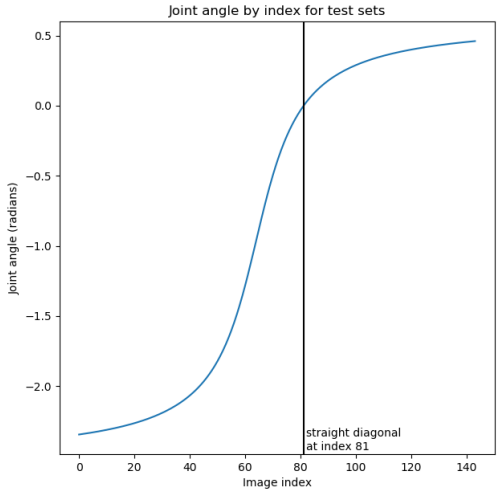


Figure 4.5: Joint angle by index for both the *connected set* and *disconnected set*.

Note, though, that the joint angle's non-linearity discussed above does not necessarily affect a right-shifted uncertainty region caused by the Poggendorff illusion. The Poggendorff illusion does not apply to images with large joint angles, and so the region over which it can be expected to affect the classification results is quite narrow. Note in Fig. 4.5 that the function, like all functions, roughly approximates a line near index 81.

Two retrained networks are used to run these experiments. The first is trained on connected lines (referred to as the *vertical connected network*) and the second on disconnected lines (referred to as the *vertical disconnected network*). Since the original Poggendorff illusion uses black vertical lines, both networks are chosen to be the ones retrained with $bg_{colour} = 0$ and bg_vert. Additionally, both networks are retrained down to layer $fc6$ since this hyperparameter performed best according to Table 4.4.

Three experiments are run. First, the connected set is evaluated with the *vertical connected network*. Second, the disconnected set is evaluated with the *vertical disconnected network*. These experiments are intended to test for AlexNet's perception of the Poggendorff illusion. Thus, to eliminate the chance of the network being trained to account for the visual illusion (as is the case with the network trained on disconnected images), a third experiment is also run. This experiment evaluates the disconnected set with the *vertical connected network*.

The first experiment produces impressive results from a classification perspective. The connected network performs quite well on the connected set, classifying all images as non-lines except for the image at index 82. This is one index off from the only image in the set that contains a straight line at index 81. Additionally, there is a 14 image range around index 81 where the network is uncertain in its classification, giving images

non-line probabilities only in the 68%-98% range. Both the misclassification and the range of uncertainty are expected, as described above, due to the very small joint angles near index 81 for this test set. Since this experiment used the connected set, effects of the Poggendorff illusion are not expected, and indeed the range of uncertainty is relatively centered about index 81. See Fig. 4.6 for the classification graph, as well as the spread image for the spread image of the connected set that the network is run on.

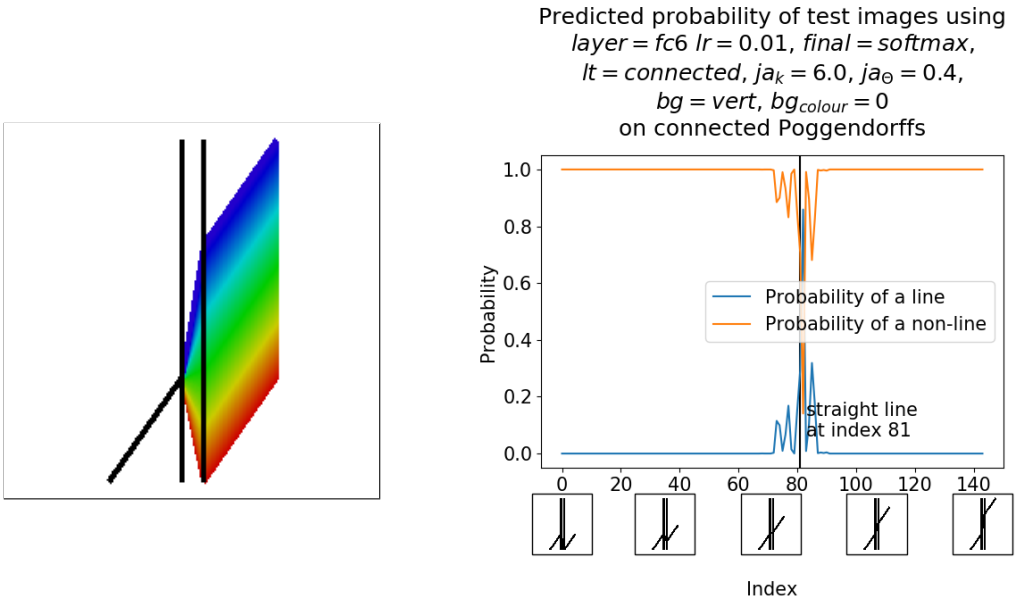


Figure 4.6: Classification graph of the first experiment: the connected set evaluated with the vertical connected network. For this and the following four figures, the image on the left is the spread image of the test set, introduced in Section 2.3.1; it represents the range of images passed through the retrained network. The graph on the right represents the predicted probabilities of each of the test set images being a line or non-line. The vertical line indicates the only test set image that contains a straight diagonal.

The second and third experiment produced less interesting results. In both cases, every single image was classified as a non-line. This classification had effectively no uncertainty; the lowest confidence of any image being a non-line in both experiments was 99.9997%. This lack of uncertainty is surprising, though from the perspective of classification error the network still does quite well, with only $\frac{1}{144} = 0.7\%$ classification error in both experiments. The classification graphs and test set spread images for these two experiments can be seen in Fig. 4.7 and Fig. 4.8.

4.5.2 Experiments with networks retrained with images containing two background lines

The three experiments in Section 4.5.1 are repeated using networks retrained on two randomly placed background lines ($bg = const$, $bg_{num} = 2$) to check for any Poggendorff-like effects. Everything else in these experiments remains unchanged, including the test sets. I will refer to the two retrained networks used during these experiments as *random connected network* (for the network retrained on connected images) and *random disconnected network* (for the network retrained on disconnected images). Recall, though, from Table 4.4 that these networks perform much worse than their $bg = vert$ counterparts. The *random connected network* has a test classification error of 0.23% as opposed to the *vertical connected network*'s 11%. The *random disconnected network* had a test classification error of 50% (the same as randomly guessing) as opposed to the *vertical disconnected network*'s 11%.

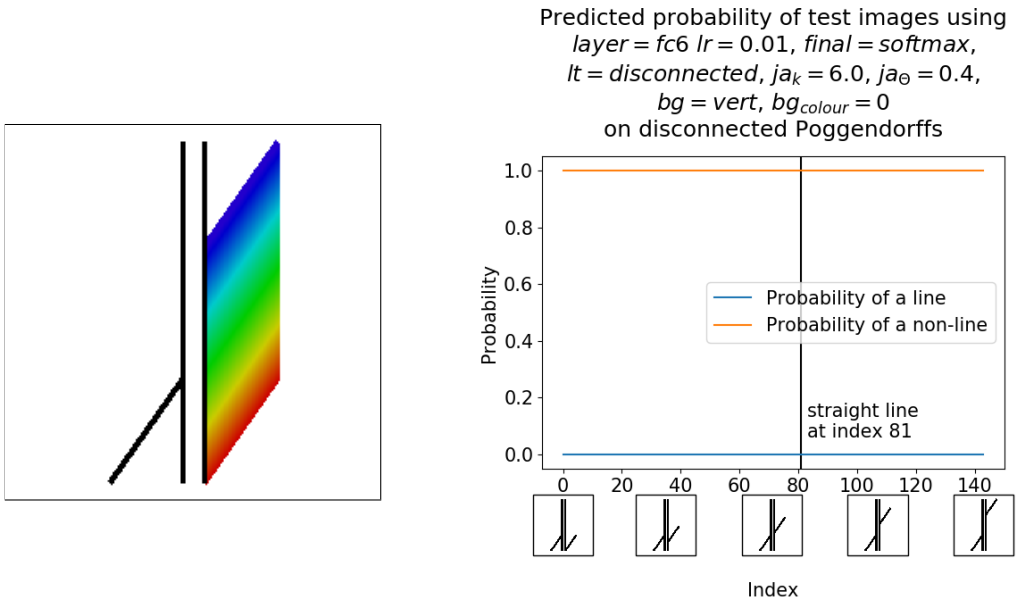


Figure 4.7: Classification graph of the second experiment: the disconnected set evaluated with the vertical disconnected network.

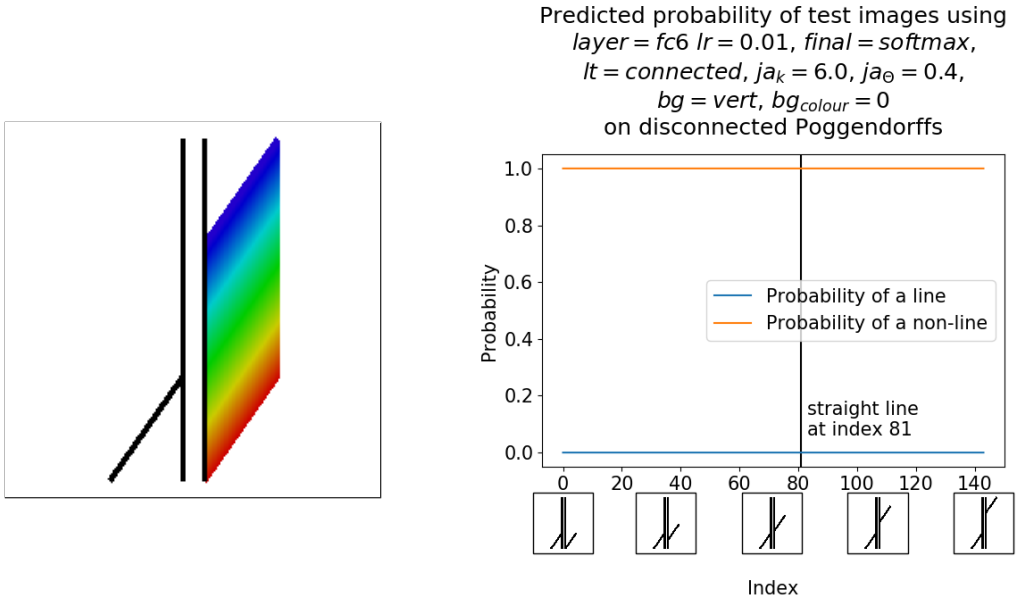


Figure 4.8: Classification graph of the third experiment: the disconnected set evaluated with the vertical connected network.

As expected from the higher error rates during retraining, the results of these three experiments are much less impressive. In the first experiment, roughly half of the images are misclassified. Interestingly, almost all of these occur after index 81 (the index of the image with a straight diagonal). This is actually the opposite trend that would be expected if AlexNet were susceptible to the effects of the Poggendorff illusion. However, even if the region of uncertainty were to the left of index 81 this could not be explained by the Poggendorff illusion because this experiment uses connected lines. More likely, the asymmetry in misclassifications is due to the test set's non-linear joint angle distribution. It could be that the network has difficulty classifying images within some symmetrical joint angle. This symmetrical range would then translate to a right-shift in image index misclassification as shown in Fig. 4.5. See Fig. 4.9 for the classification graph of the first experiment.

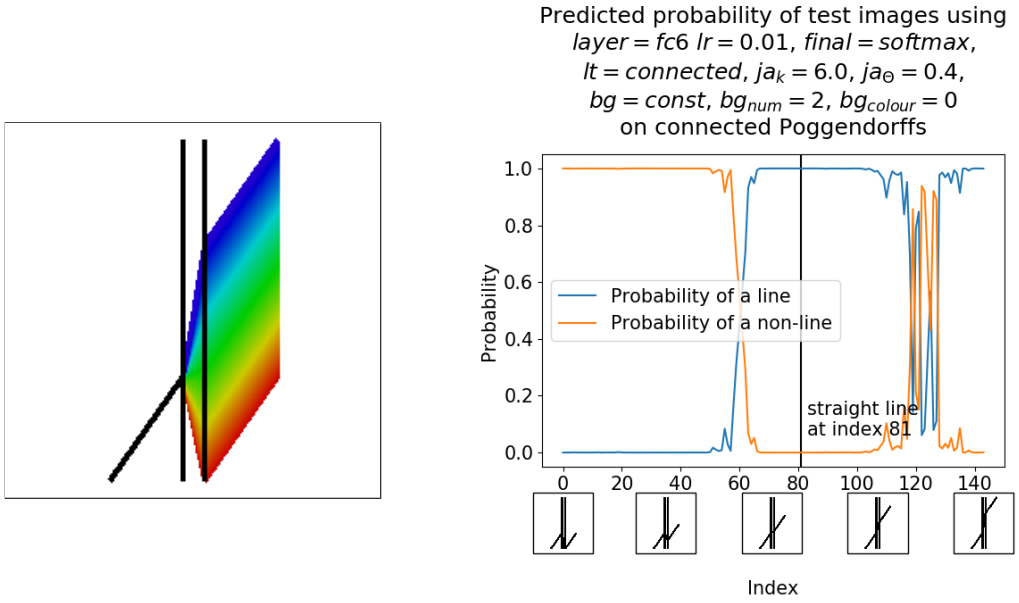
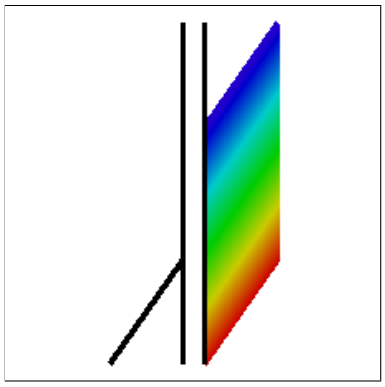


Figure 4.9: Classification graph of the first experiment: the connected set evaluated with the *random connected network*.

The second experiment does not result in much useful data. Every image was classified as a line with only moderate uncertainty in larger indices, despite almost every image in the disconnected set being a non-line. This is probably due to the *random disconnected network*'s terrible performance during retraining, where it has 50% test classification error. However, the consistency of the results indicates that the network is not randomly guessing, even though it results in the retrained network getting nearly every image's class wrong. See Fig. 4.10 for the classification graph for the second experiment.

The third experiment performs similarly to the first. Once again, there is a very wide range of uncertainty, most of which is located after index 81. Like in experiment 1, the overall poor performance is probably due to the *random connected network*'s overall difficulty with differentiating lines from non-lines. This difficulty is compounded in this experiment since disconnected lines are being passed through a network that was trained on connected lines. The lack of symmetry about index 81 for the misclassifications are, like in experiment 1, probably due to the non-linearity of the joint angle with respect to image indices. See Fig. 4.11 for the classification graph of the third experiment.

A variation of this third experiment was run with a slightly different disconnected test set. This test set was generated in the same way as the original disconnected set, except that the margins are not reduced to 10



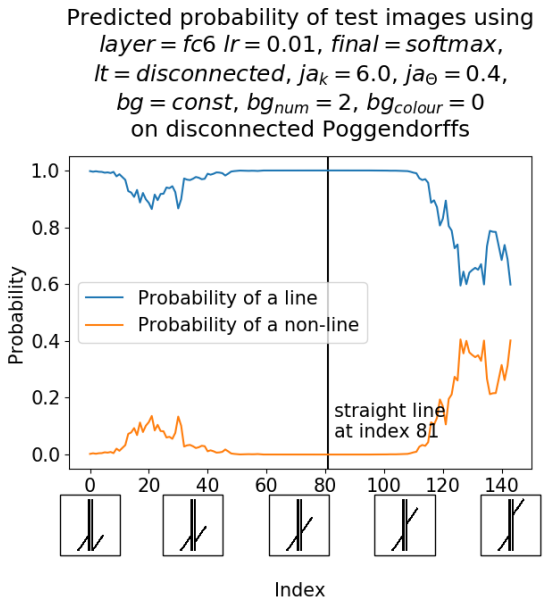
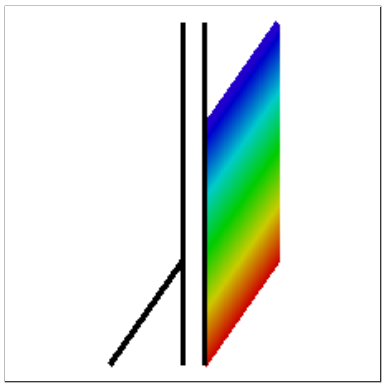


Figure 4.10: Classification graph of the third experiment: the disconnected set evaluated with the *random disconnected network*.



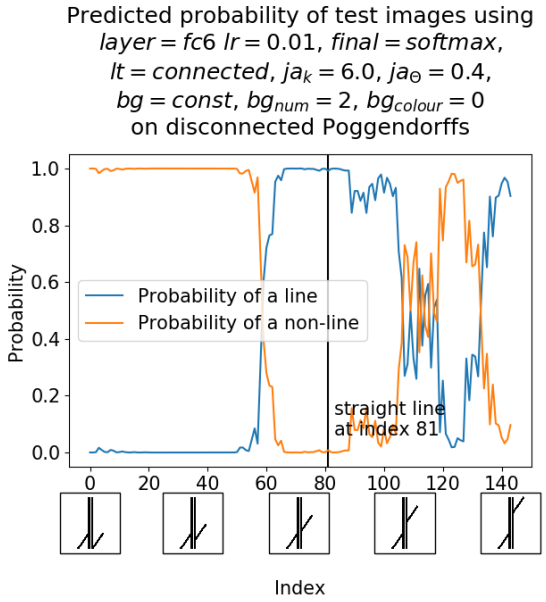


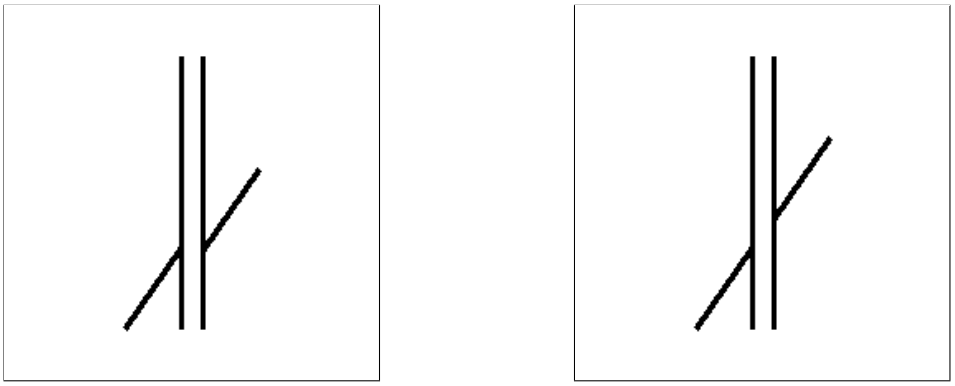
Figure 4.11: Classification graph of the third experiment: the disconnected set evaluated with the *random connected network*.

pixels; they are left at the exploratory testings' original 31 pixels. This change results in there being only 116 images in the image set, and the image with a straight diagonal occurring at index 67.

This new disconnected set is again run on the *random disconnected network*. Note that since this network is trained with two background lines of random length, there is no standard vertical line height from training that the test data should match. Thus, changing the margins is not a departure from the training data like it would be for the *vertical disconnected network* from Section 4.5.1.

Experiment four produced an uncertainty range that could most readily be explained by the Poggendorff illusion. The first image classified as a line is at index 47, and the last image classified as a line is at index 66, one index prior to straight diagonal at index 67. This makes it the first left-shifted uncertainty range seen so far.

However, though the large-index misclassifications could be explained by the Poggendorff illusion, it is difficult to argue the same for the low-index misclassifications. The image at index 66 (the last image classified as a line) has a very small joint angle of -0.03 rad ($\approx -2°$). This is not true of the image at index 47 (the first image classified as a line); this image has a very large joint angle of 1.2 rad (69°). These two images can be seen in Fig. 4.12. Thus, the Poggendorff illusion alone does not seem strong enough to account for the misclassification of image 47. See Fig. 4.13 for the classification graph of the fourth experiment.



(a) Test set image at index 47

(b) Test set image at index 66

Figure 4.12: Significant test images from the fourth experiment. 47 is the index of the first image classified as a line. 66 is the last image that was classified as a line.

Therefore, though the fourth experiment exhibited some features that might indicate AlexNet is susceptible to the Poggendorff illusion, the illusion itself is not able to explain the network's behavior at all image indices. For at least low index images near index 47 there are probably other factors causing the network's misclassification. There is not enough information from this experiment to determine if these additional factors are also causing the misclassification at larger indices near index 66, or if these misclassifications are indeed due to the Poggendorff illusion. Exploring these confounding factors further, as well as the reason for left-shifted uncertainties appearing only for the disconnected set with larger margins, are topics for future research.

4.6 Summary and discussion of results

My exploratory research indicates that the unmodified AlexNet is unable to differentiate lines from non-lines using cosine similarity in Poggendorff-like images. However, this is a narrow conclusion that applies only to

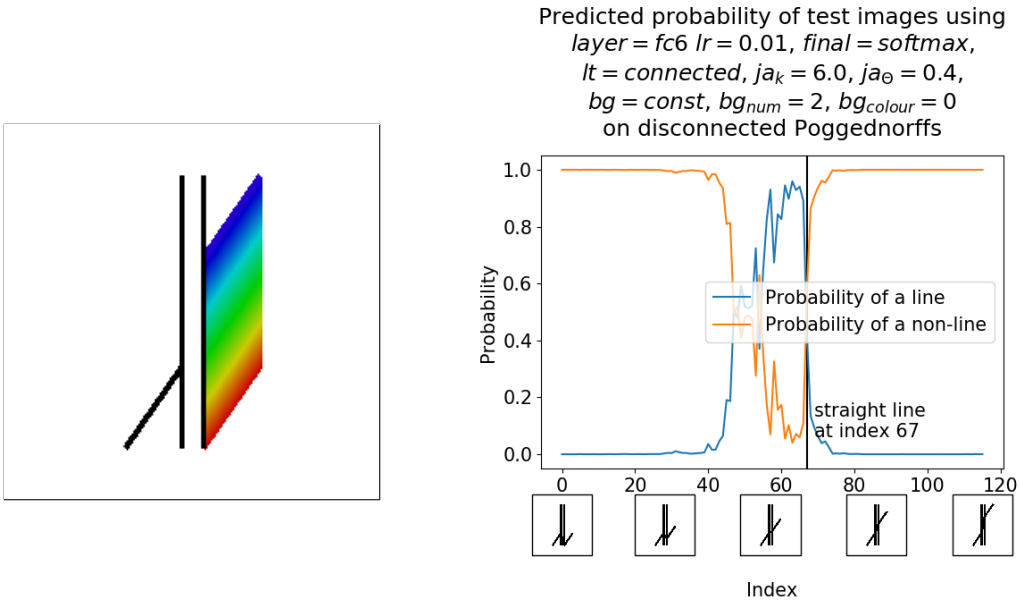


Figure 4.13: Classification graph of the fourth experiment: the modified disconnected set evaluated with the *random disconnected network*.

certain types of images evaluated using a single measure. Furthermore, by using only the original network, it is difficult to determine if AlexNet is unable to differentiate lines from non-lines, or instead does not use this information during classification. To address the limitations of exploratory testing, I directly retrain AlexNet to distinguish more generic lines from non-lines.

The experiments run with connected lines perform relatively well. In the simplest case with large joint angles and no background lines, AlexNet performs incredibly well, with less than 1% error. Clearly, AlexNet *is* capable of differentiating lines from non-lines. However, as the images became more complex, and the average joint angle is decreased, AlexNet's performance becomes steadily worse. With smaller joint angles, the test classification error increases from 3% with no background lines to 22% with an average of 4.5 background lines.

The extent of this performance degradation are not clear from these results. Note that for all of these networks, the worse performance possible is an error rate of 50%, which is the expected result of random guessing. Any error higher than this indicates that the network is learning, just with the labels flipped. Plotting the error rates for connected lines against the 4 background distributions tested shows an increasing, relatively linear trend that does not appear to flatten. Further testing should be done with more background lines to learn if this error plateaus at some error less than 50%, or degrades right to the logical limit. A smaller plateau would indicate that AlexNet has some ability to different connected lines from non-lines that is immune to background line noise.

The experiments run with disconnected lines perform far worse. In fact, the line type (whether connected or disconnected) is the permutation that consistently has the largest effect on the error rate across experiments. Even with large joint angles and no background lines, AlexNet has 12% error on disconnected lines, and this performance degrades further with the use of smaller joint angles and more background lines. With smaller joint angles, the test classification error for disconnected lines increased from 26% with no background lines to 48% with an average of 4.5 background lines. Unlike in the connected case, it is clear that AlexNet

approaches the error limit of 50%, and it does so quite quickly. For the graph of test classification errors by number of background lines for both the connected and disconnected cases, see Fig. 4.2.

The consistently worse performance in the disconnected case could be due to the lack of local information differentiating lines from non-lines. For connected elements, the joint angle can be measured locally between line segments, and this angle differentiates lines from non-lines. By removing the middle line segment (as is done for disconnected elements), the joint angle can no longer be measured directly within a small pixel window. See Fig. 4.1 for a demonstration of this difference.

Thus, a significant difference between connected and disconnected images is the ability to distinguish lines from non-lines using only local information. The disconnected experiments' much higher error rates, then, indicate that AlexNet is worse at utilizing global information than local information when classifying images.

Humans, when classifying images, use global shape information over local information like edges and texture [16, 1]. This makes AlexNet's difficulty classifying images with only global information surprising considering the research indicating AlexNet's ability to closely model the human visual system. Despite AlexNet's ability to model hierarchical neural responses in the ventral stream [4], particularly the final inferior temporal cortex that stores high-level representations [14], AlexNet's classification ability is closely tied to the presence of local differentiating features. Therefore, AlexNet cannot fully represent the brain's ability to classify images based on global shape information. This indicates that AlexNet's model of the brain's visual processing system is more limited than previously thought.

The last set of experiments are run on images with vertical and randomly placed background lines. Interestingly, AlexNet performed much better than expected on images with vertical background lines, with no performance loss when using disconnected rather than connected images with $layer = fc6$. Based on the poor results with two randomly placed background lines, the low error rate on images with vertical background lines is not due to the consistent number of background lines or that the background lines are always connected. Instead, it would appear as though the background lines' consistent placement relative both to the canvas and the primary element provide some useful information to the network that aids its classification. Admittedly, the vertical images are generated with additional checks that could result in test sets that are easier to classify; this is another factor that could contribute to the improved performance.

These good results, particularly on disconnected images with vertical background lines, motivate experiments that test for AlexNet's susceptibility to the Poggendorff illusion directly. Using networks trained with vertical background lines, there is no indication of the Poggendorff effect. However, I would attribute this non-result to the network's lack of classification ability. On disconnected images like the original Poggendorff illusion, no image was classified as containing a line. This includes both the image with a straight diagonal, and any image at a lower index that may be classified as a line due to the Poggendorff effect.

Running this same experiment with AlexNet retrained on two randomly placed background lines produces more interesting results despite (or because of) the higher classification error during training. On disconnected images, the network classifies huge swaths of images as lines. This includes the only image containing a straight diagonal at index 81, images with indices smaller than 81, *and* images with indices larger than 81. These misclassifications are not indicative of the Poggendorff illusion. Instead, the network's inability to differentiate lines from non-lines results in a loss of the classification granularity required to identify the effects of the Poggendorff illusion, if present at all.

Interestingly, shortening the height of the vertical background lines *did* produced a classification graph that could be explained (at least in part) by the Poggendorff illusion. There are a couple possible interpretations from this result. It could be that the effects of the Poggendorff illusion affected all the tests but were simply

obscured by poor classification. This latest test set, then, might have simply been easier to classify and thus exposed this ever-present effect. Conversely, the apparent effects of the Poggendorff illusion could be the result of random chance on a network that had 25% classification error.

I should note here that these experiments were run with three image sets. The first image set, as described in Section 4.5, had a gap of 12 pixels and taller vertical lines. The second image set, as again described in Section 4.5, also had a gap of 12 pixels but had shorter vertical lines. The third image set is not described in Section 4.5 and had a gap of 21 pixels and shorter vertical lines. On all three image sets, the *vertical connected network* exhibits the same behavior: every image was classified as a non-line. For the *random connected network*, only the second image set exhibits the Poggendorff effect; the third image set had a classification graph similar to the one shown for the first image set. This leads me to suspect the Poggendorff effect seen in the second image set is possibly the result of random misclassifications that happened to correspond to the Poggendorff illusion. This could also explain the misclassification of images with much larger joint angles than can be described by the Poggendorff effect. Examining additional image sets to determine whether the effect is real in AlexNet or the result of random chance is a topic for future research.

While trying to determine the cause for these results, it must be recognized that much of the confusion comes from AlexNet's inability to accurately differentiate lines from non-lines. AlexNet has $< 1\%$ classification error on connected images with no background lines. If it can achieve this error for more complicated images, then misclassifications would hold more meaning when analyzing Poggendorff-like images. Recall that differentiating disconnected lines from non-lines is a prerequisite for experiencing the effects of the Poggendorff illusion. Thus, AlexNet's inability to reliably accomplish this task in the presence of even a couple background elements means that the network's susceptibility to the Poggendorff illusion is, in some way, irrelevant.

5 Conclusion

My exploratory research indicates that the unmodified AlexNet is unable to differentiate lines from non-lines under relatively constrained conditions. By retraining AlexNet, I am able to extend this result to lines in more general images. Under the simplest circumstances, AlexNet is able to different lines from non-lines very accurately with $< 1\%$ classification error. However, increasing the number of background lines steadily degrades this performance to 22% error when there are an average of 4.5 background elements.

AlexNet's performance is much worse for disconnected images, the image class under which the Poggendorff illusion falls. For these images, the classification error falls to nearly the theoretical limit of 50%, and AlexNet has an error of 45% with only an average of 2.5 background elements. This indicates that though AlexNet is capable of differentiating lines from non-lines, it is not reliable under conditions similar to the Poggendorff illusion.

The consistently worse performance in the disconnected case could be due to the lack of local information differentiating lines from non-lines. The disconnected experiments' much higher error rates, then, seem to indicate that AlexNet is worse at utilizing global information than local information when classifying images. This is surprising knowing that humans use global information over local information to perform object classification [16, 1]. This result brings into question the accuracy of AlexNet's human visual system modeling, despite the research showing high correlation between the two [4, 14]. Since AlexNet's classification ability appears closely tied to the presence of local differentiating features, AlexNet cannot fully represent the brain's ability to classify objcts based on global shape information. This indicates that AlexNet's model of the brain's visual processing system is more limited than previously thought.

AlexNet is better able to classify images with vertical background lines, perhaps because the background lines' consistent placement relative both to the canvas and the primary element provide some useful information to the network that aids its classification. However, this improvement does not result in sufficiently accurate classifications to observe the Poggendorff effect consistently. Though AlexNet appeared susceptible to the Poggendorff effect in one test, additional image sets will need to be studied to determine whether the effect is real or the result of random chance. Overall, though, AlexNet's inability to accurately differentiate disconnected lines from non-lines, a prerequisite for experiencing the effects of the Poggendorff illusion, make AlexNet's susceptibility to the Poggendorff illusion somewhat irrelevant.

AlexNet's inability to differentiate disconnected lines from non-lines is surprising. It implies a broader inability of the network to model the brain's object classification that is based on global shape information. This result is similar to other recent work showing AlexNet's inability to use global information during classification in other contexts. In particular, AlexNet has significant difficulty classifying objects without texture, and displays a total inability to distinguish circles from ellipses [1].

AlexNet's relative difficulty differentiating connected lines and non-lines is also surprising. AlexNet's classification error for connected images increases steadily with the number of background lines, and shows no indication of flattening over the four distributions tested. This is surprising given the simplicity of the task and the prevalence of lines in real images. Lines are the simplest abstract shape possible, after points, and differentiating connected lines and non-lines appears trivial for humans. Furthermore, lines appear frequently in images of the modern world, and so would be prevalent in the ImageNet images used to originally train AlexNet. Though HCNNs rival humans for complicated object classification, it appears that even the most simple abstract task once again gives humans a large advantage.

This calls into question the techniques used to train HCNNs. It would seem that they are being over-optimized for a specific task that degrades their ability to model the general-purpose human visual system.

To develop better models for the brain, then, requires HCNNs that perform well on a variety of tasks at different levels of abstraction, and that interpret and utilize both global and local image information. Despite the promise of current networks, these HCNNs do not yet appear to satisfy these requirements. Future work is required to develop networks trained for more abstract tasks, and to discover other features of the human visual system not yet modeled by current HCNNs.

A Appendix

A.1 Graph of raw data for permutations of offset set

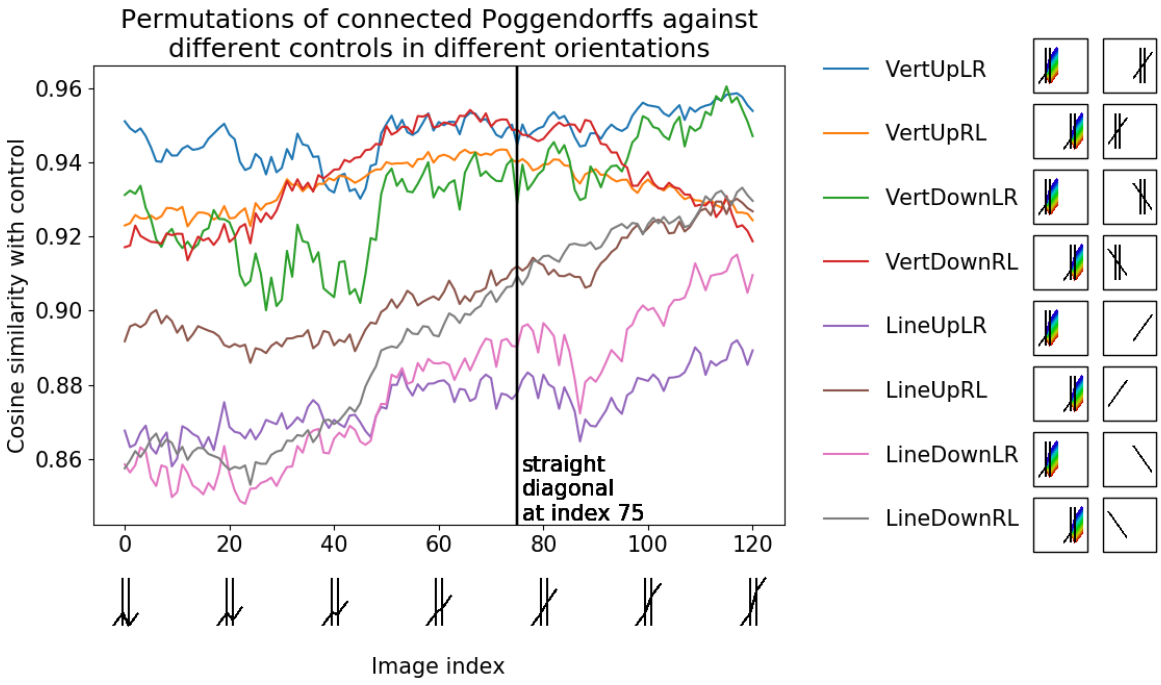


Figure A.1: Cosine similarity of the 8 offset set permutations at the fc8 layer. The images to the right of each legend item are the spread and control for that set. The images under the axis represent a generic sample of the test images at those indices, without any orientation.

A.2 Graphs for learning rate comparison

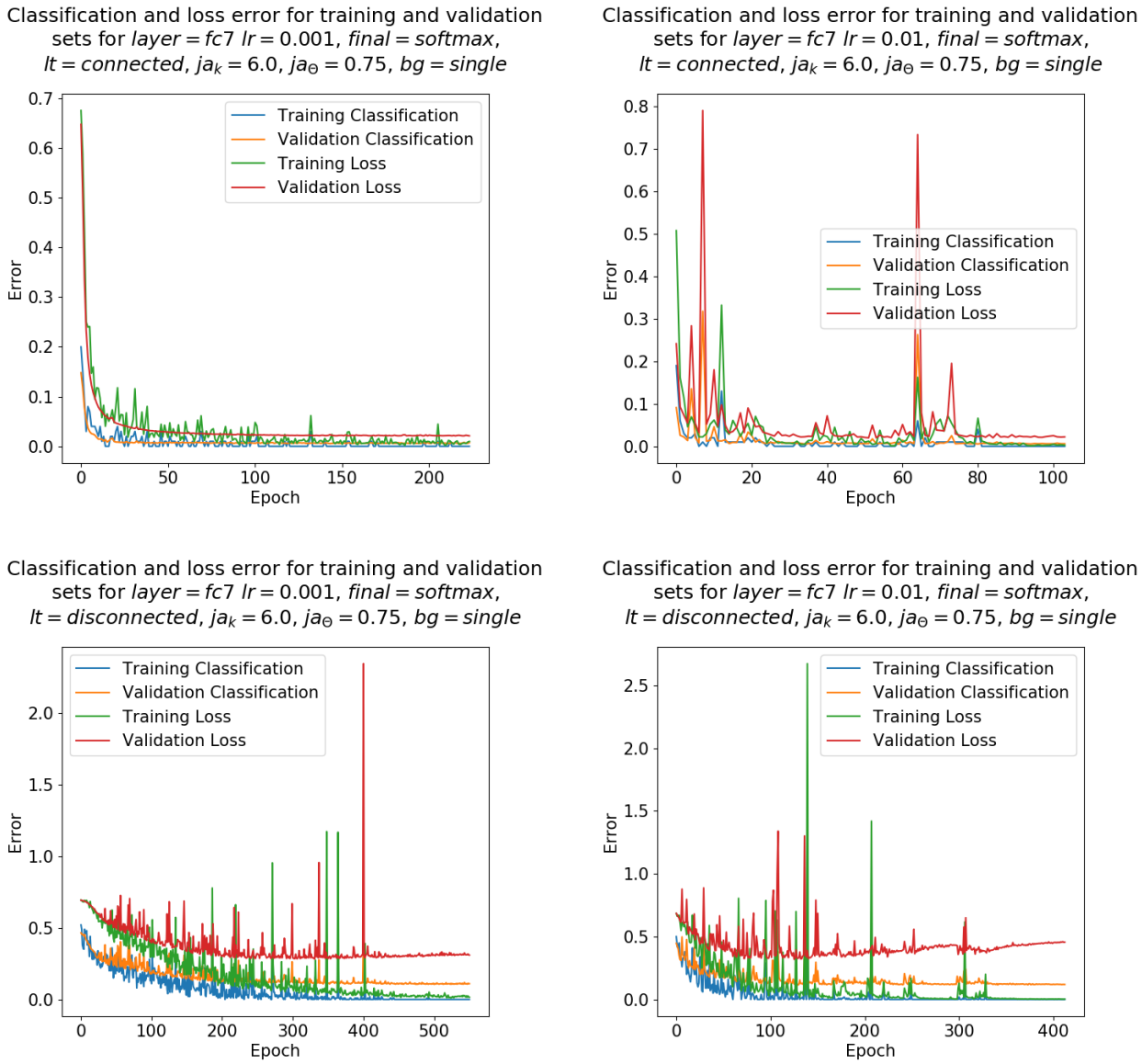


Figure A.2: Comparison of using a learning rate (lr) of 0.001 (left) versus 0.01 (right) for $layer = fc7$ with both connected and disconnected training sets.

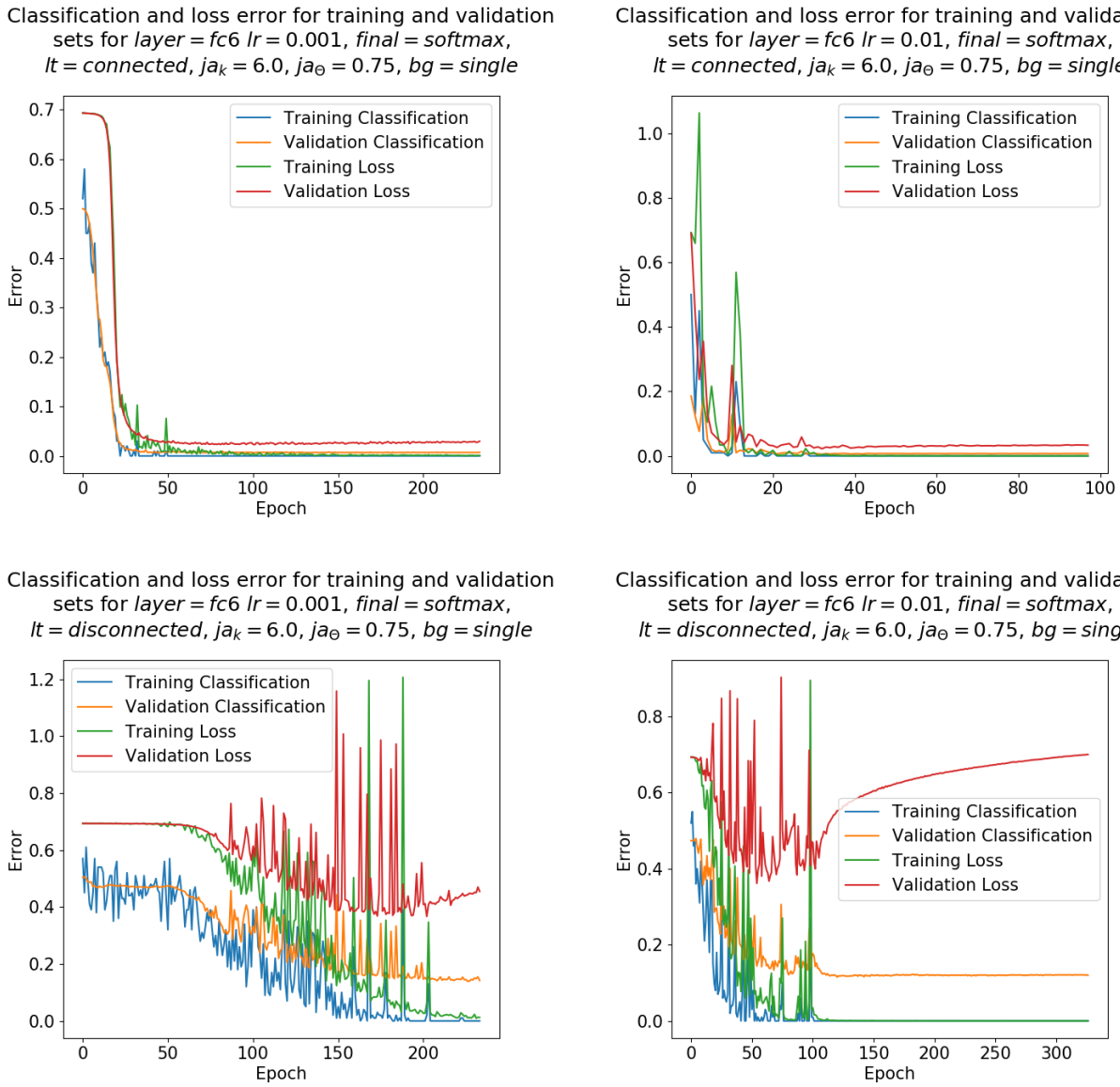


Figure A.3: Comparison of using a learning rate (lr) of 0.001 (left) versus 0.01 (right) for $layer = fc6$ with both connected and disconnected training sets.

A.3 Graphs for experiments with large joint angles

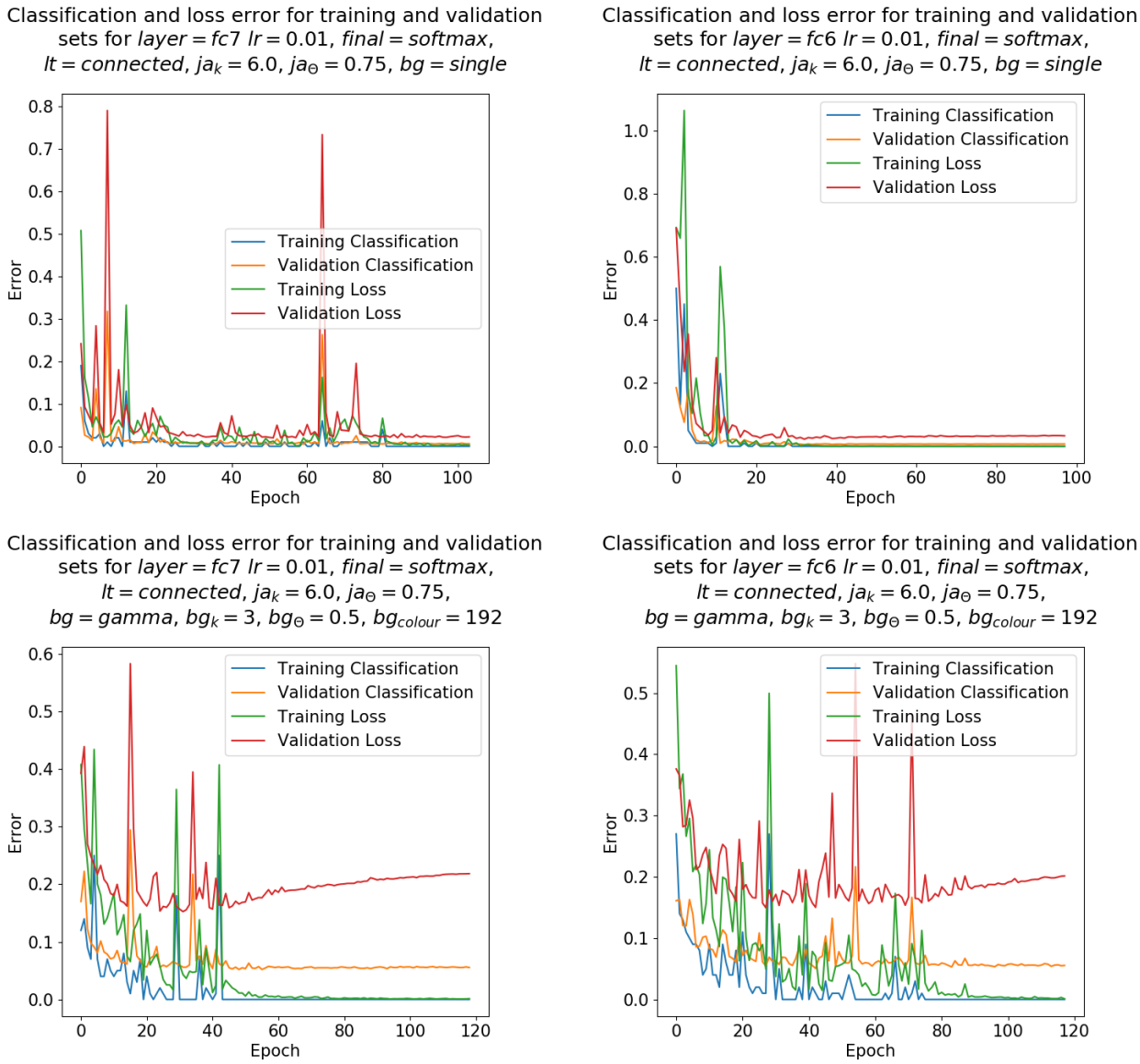


Figure A.4: Graphs of experiments with large joint angles with connected lines. Note that in all cases, the model appears to have converged to a learned state.

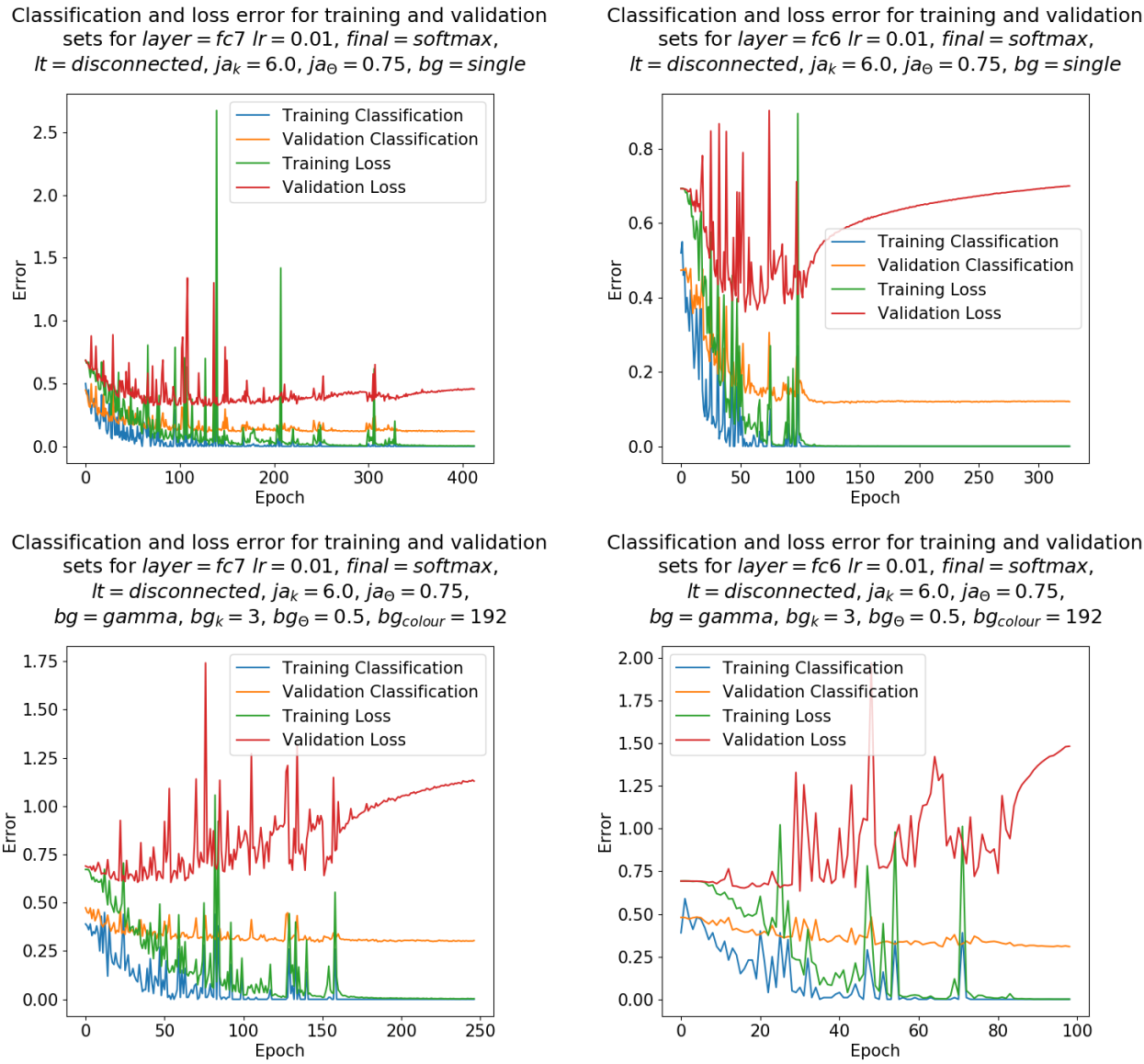


Figure A.5: Graphs of experiments with large joint angles with disconnected lines. Note that in all cases, the model appears to have converged to a learned state.

A.4 Graphs for experiments with small joint angles and no background elements

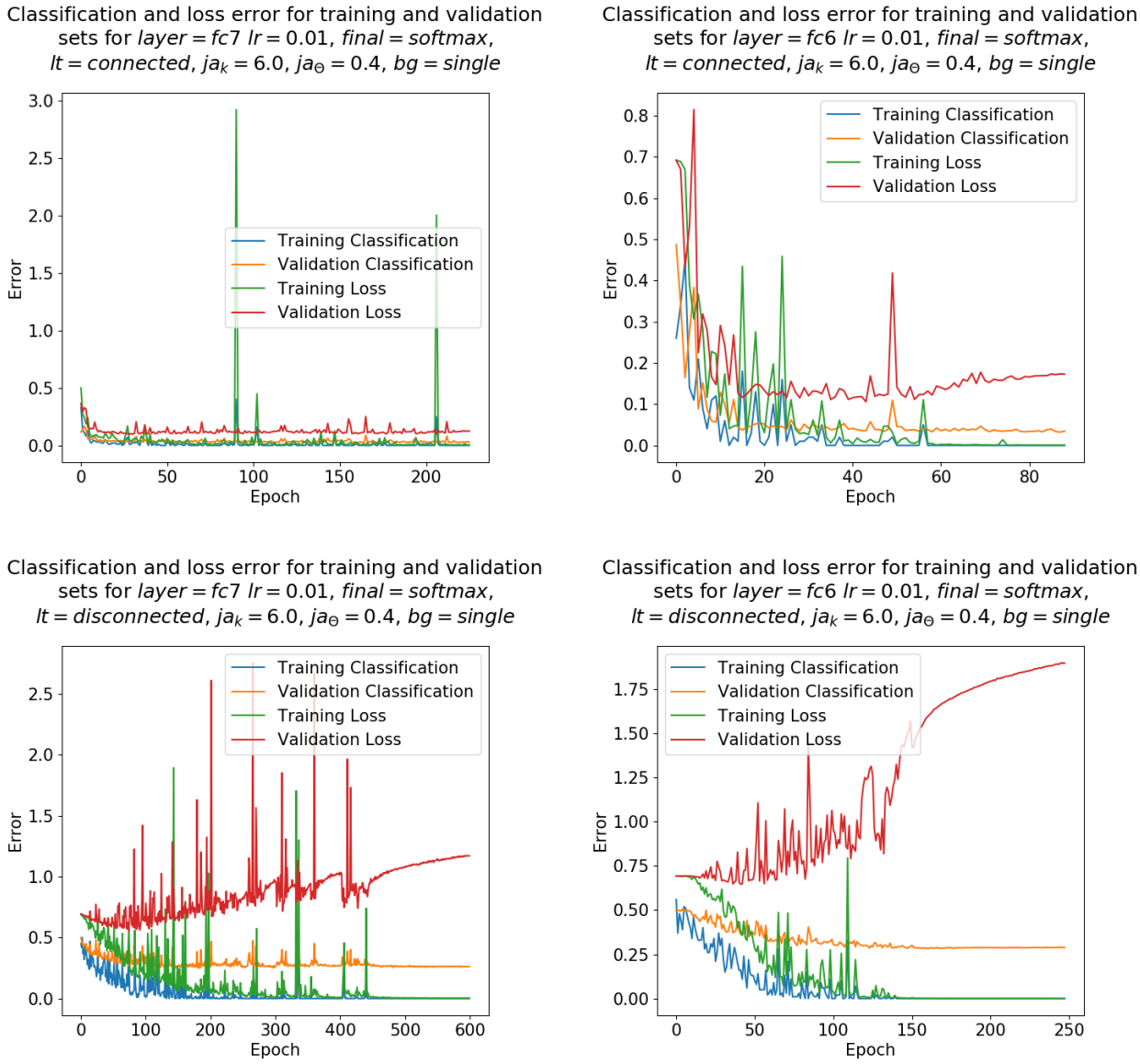


Figure A.6: Graphs for experiments with small joint angles and no background elements. Note that in all cases, the model appears to have converged to a learned state.

A.5 Graphs for experiments with small joint angles and gamma background element distributions

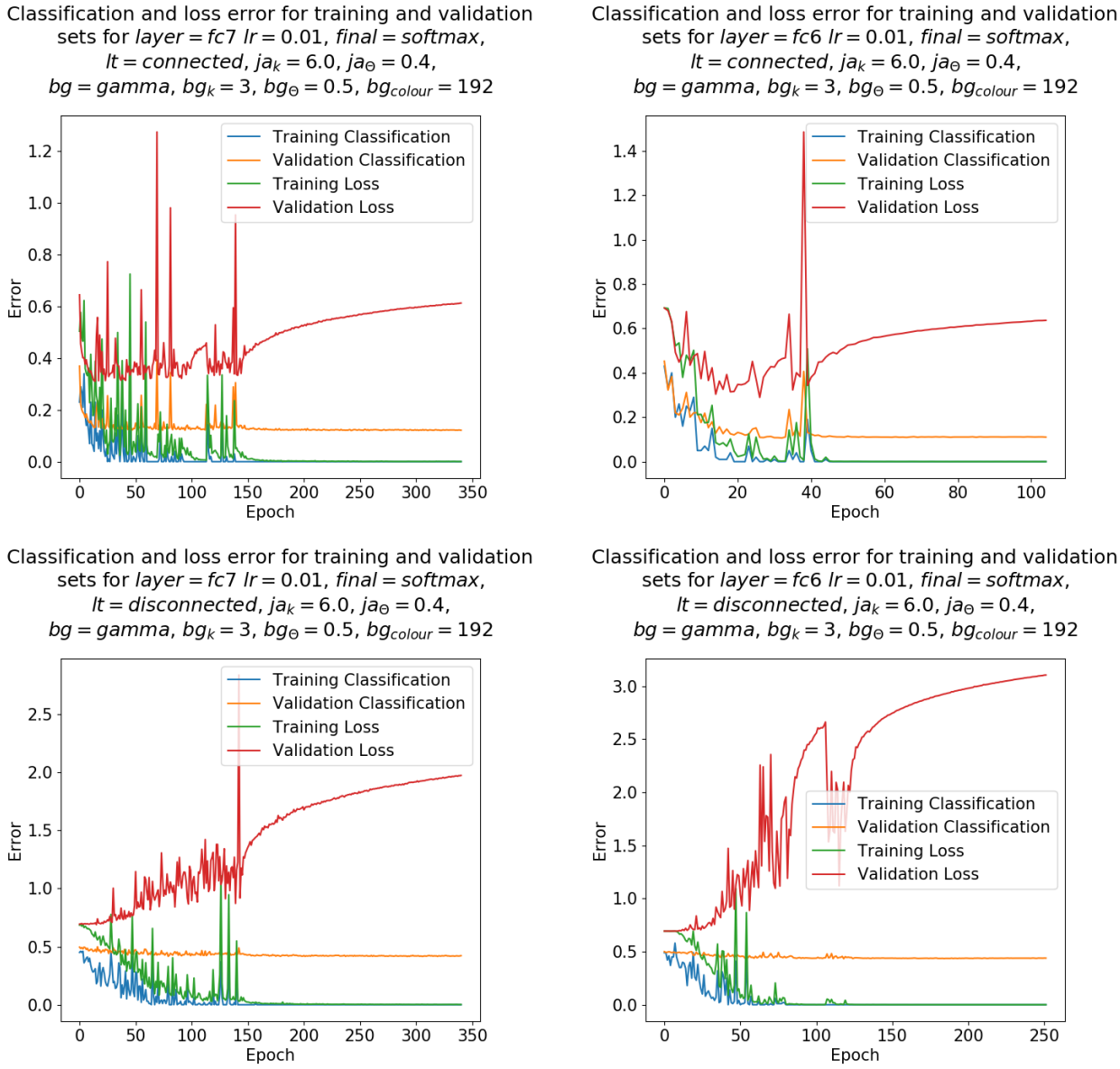
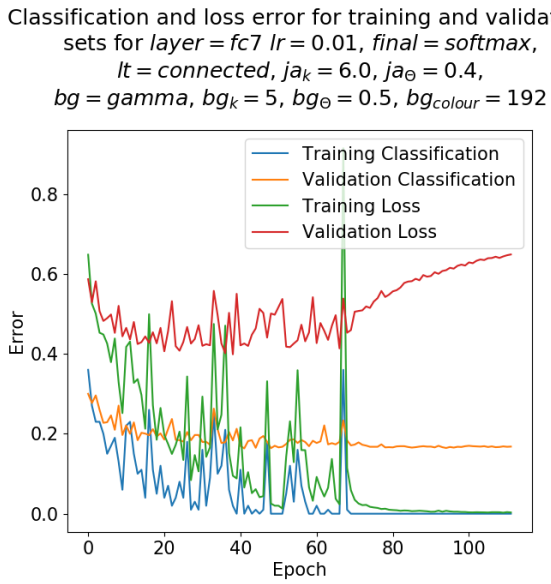
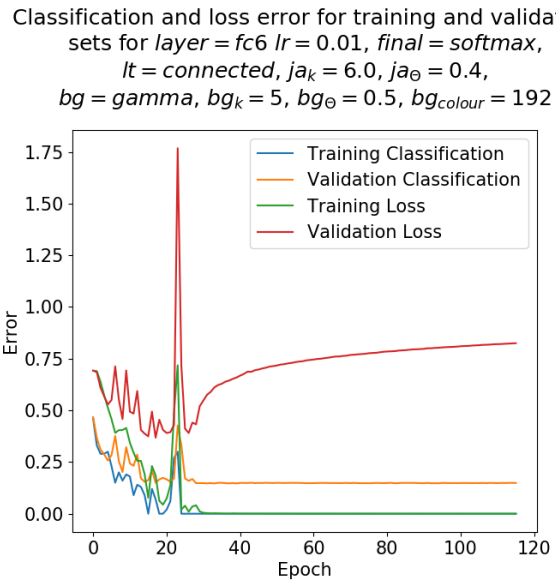
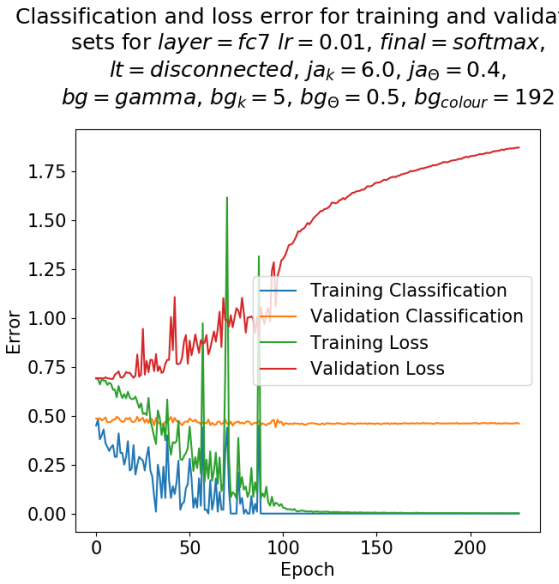


Figure A.7: Graphs for experiments with small joint angles and gamma background element distributions with $k = 3$. Note that in all cases, the model appears to have converged to a learned state.







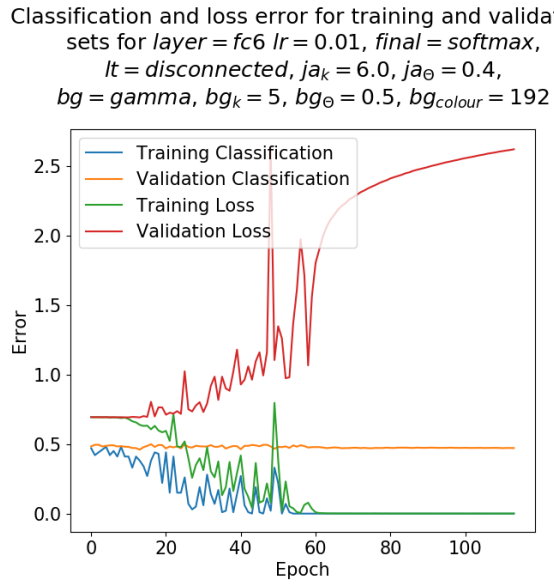


Figure A.8: Graphs for experiments with small joint angles and gamma background element distributions with $k = 5$. Note that in all cases, the model appears to have converged to a learned state.

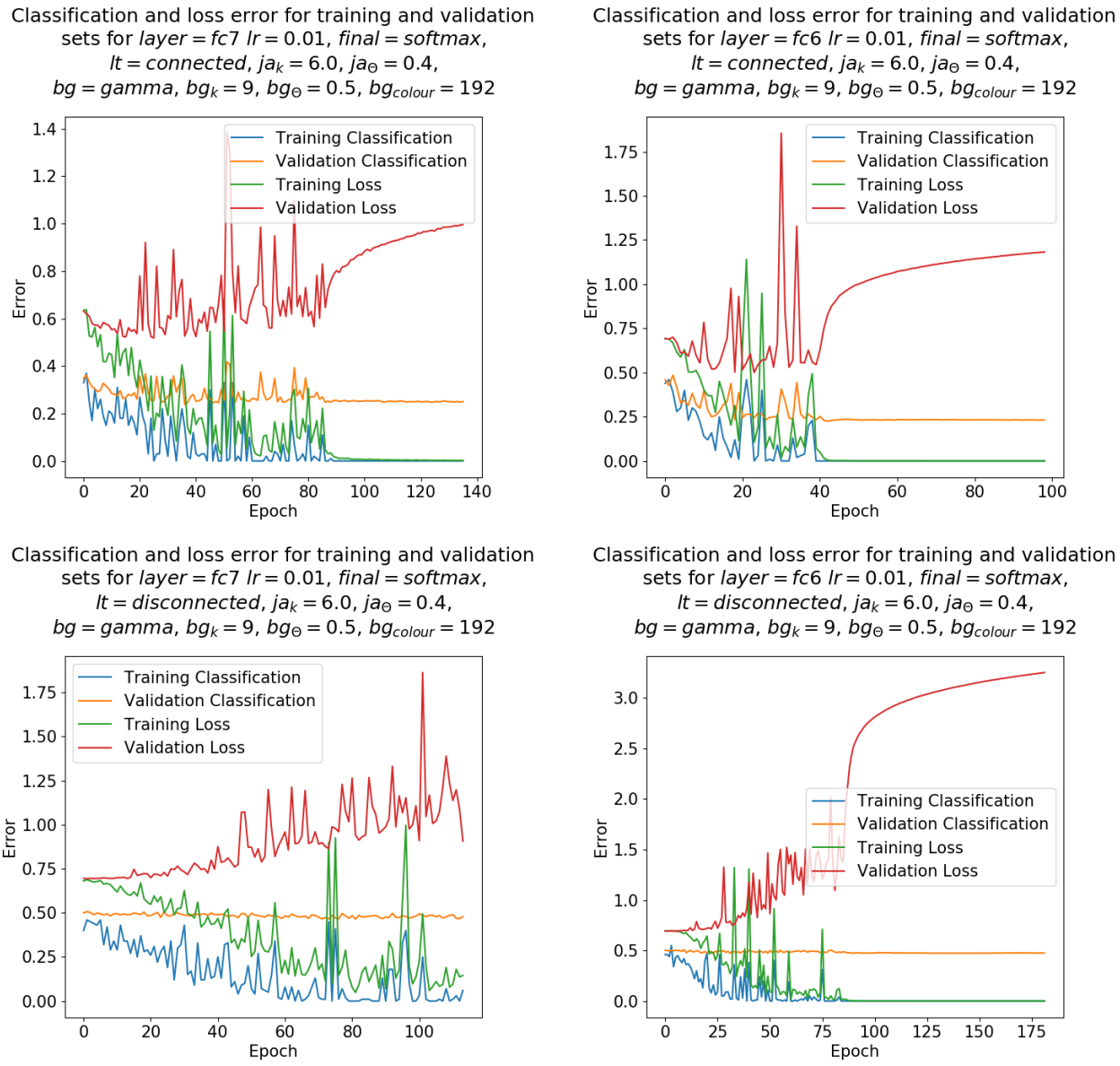


Figure A.9: Graphs for experiments with small joint angles and gamma background element distributions with $k = 9$. Note that in all cases, the model appears to have converged to a learned state.

A.6 Graphs for experiments with small joint angles and different background element distributions

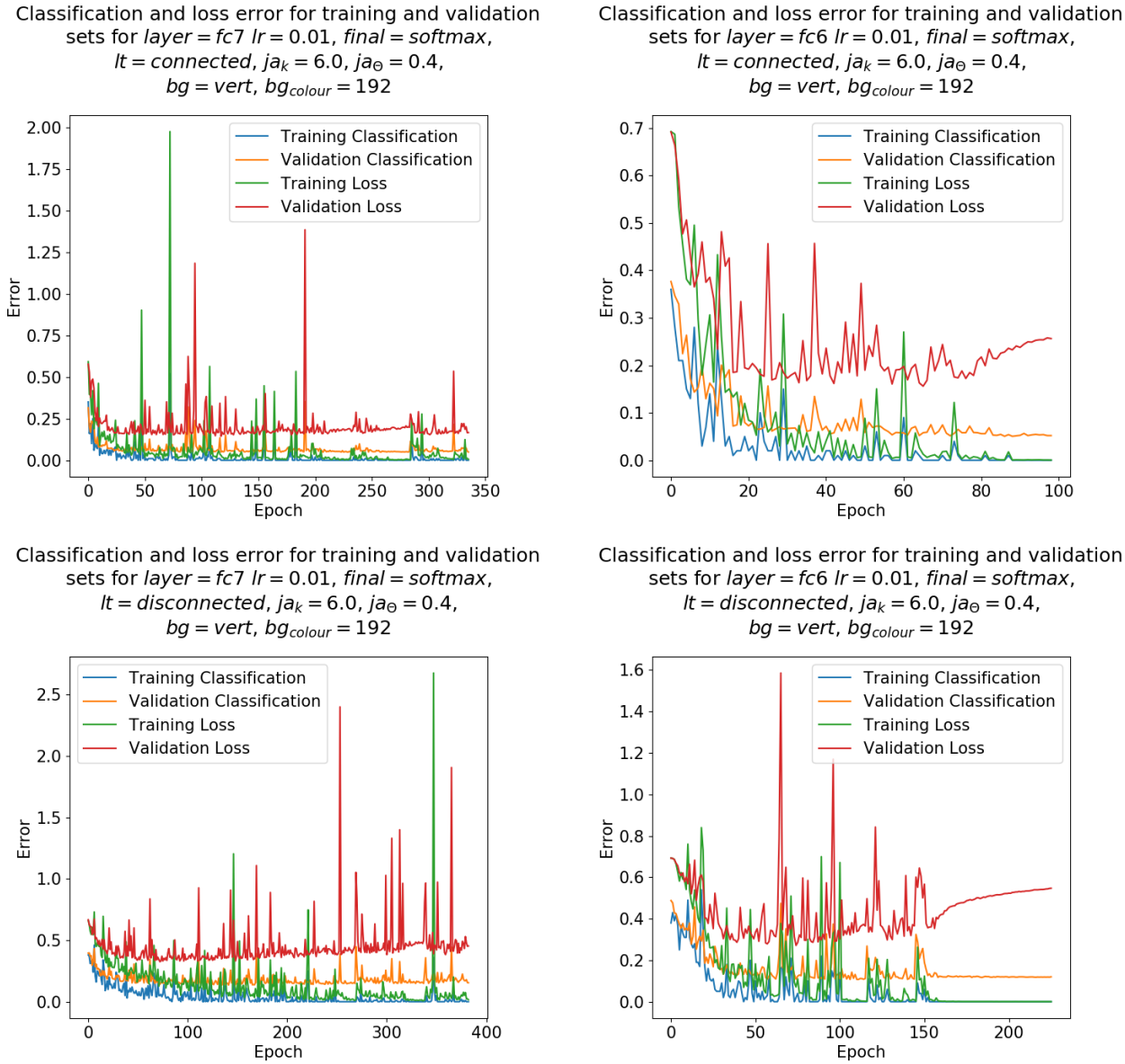


Figure A.10: Graphs for experiments with small joint angles and vertical background lines at 75% opacity. Note that in all cases, the validation classification curve appears to have (at least roughly) converged to a learned state and does not exhibit overfitting.

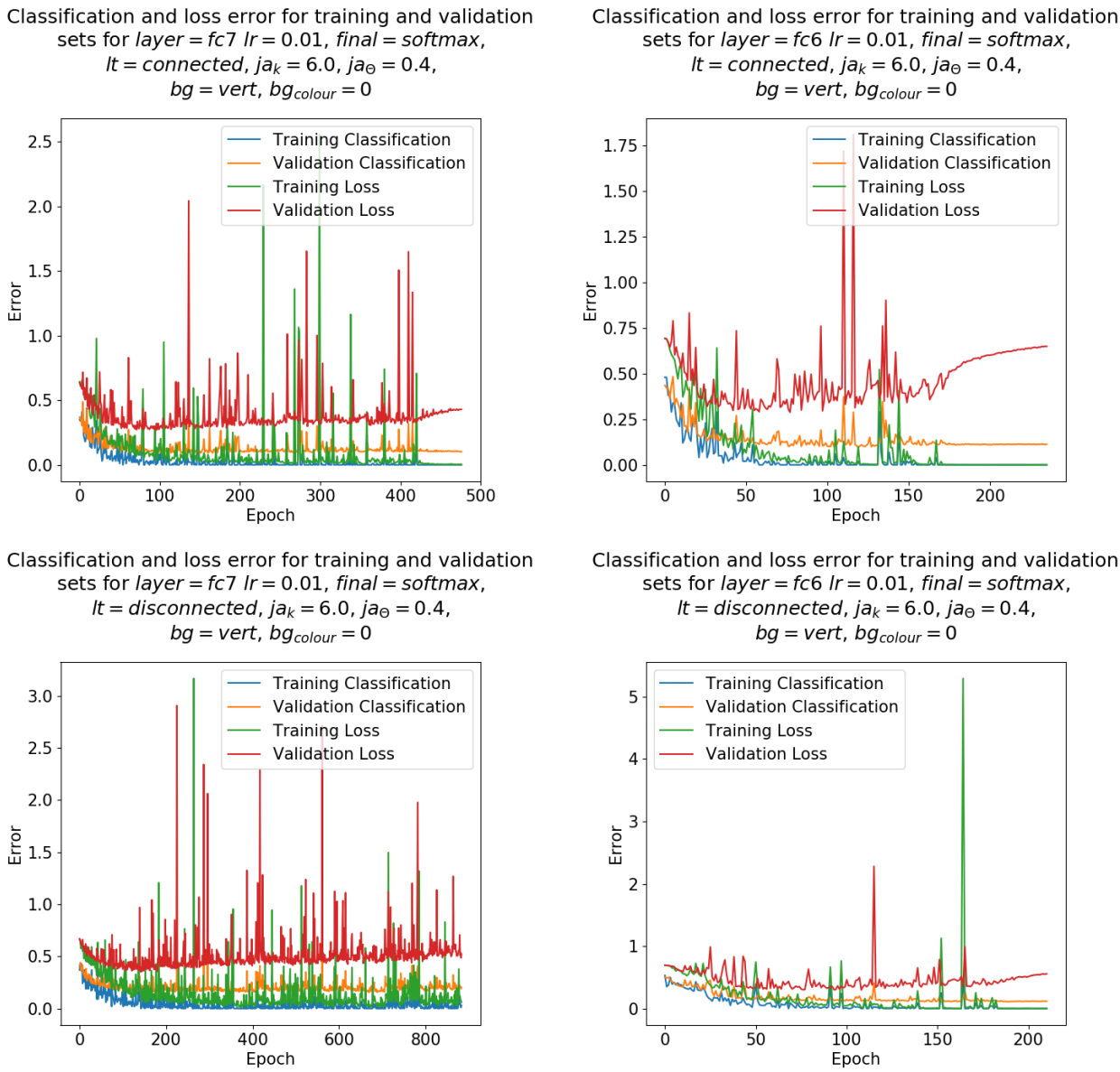


Figure A.11: Graphs for experiments with small joint angles and black vertical background. Note that in all cases, the validation classification curve appears to have (at least roughly) converged to a learned state and does not exhibit overfitting.

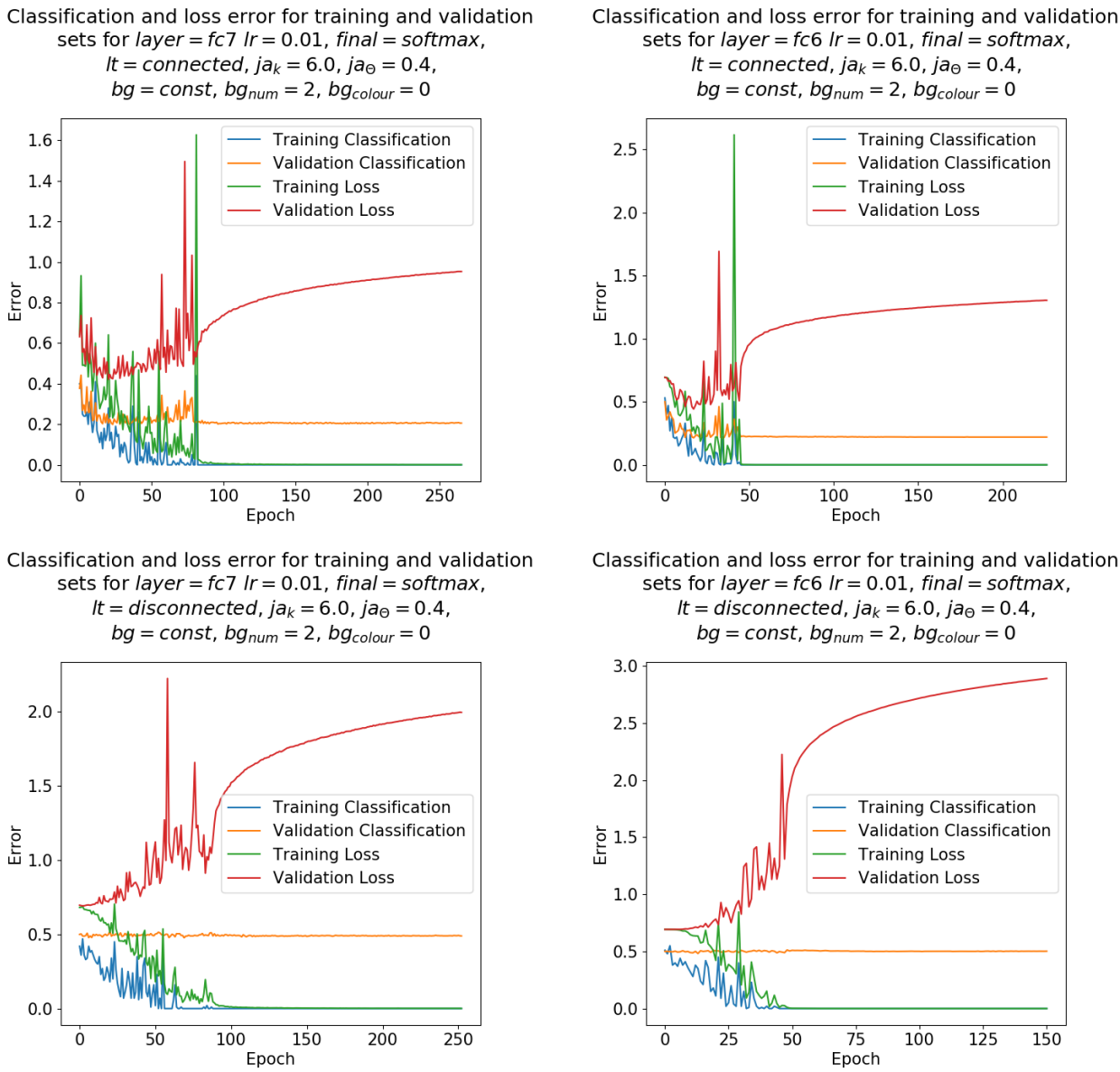


Figure A.12: Graphs for experiments with small joint angles and 2 black vertical lines. Note that in all cases, the validation classification curve appears to have converged to (roughly) a learned state and does not exhibit overfitting.

Glossary

bg A class of hyperparameter used when generating data to retrain AlexNet. *bg* is the background distribution, and can be one of bg_{single}, bg_{gamma}, bg_{vert} or $bg_{constant}$. 25, 26, 66

***bg*$_{colour}$** A hyperparameter used when generating data to retrain AlexNet. bg_{colour} is the colour of the background lines. It is an integer in the range [0,255] where 0 corresponds to black, and 255 to white. It is used for the *bg* distributions bg_{gamma}, bg_{vert} and $bg_{constant}$. 26, 66

***bg*$_{constant}$** A hyperparameter used when generating data to retrain AlexNet; it is one of the *bg* distributions. $bg_{constant}$ indicates that the images in the test set were created with a constant number of randomly placed background lines. This number is determined by bg_{num}. The colour of the background lines is determined by bg_{colour}. 26, 66

***bg*$_{gamma}$** A hyperparameter used when generating data to retrain AlexNet; it is one of the *bg* distributions. bg_{gamma} indicates that the images in the test set were created with the number of lines determined by a gamma distribution with parameters bg_k and bg_θ. The colour of the background lines is determined by bg_{colour}. 25, 66

***bg*$_k$** A hyperparameter used when generating data to retrain AlexNet. bg_k is the k parameter of the gamma function used to determine the number of background lines for bg_{gamma}. 25, 66

***bg*$_{num}$** A hyperparameter used when generating data to retrain AlexNet. bg_{num} indicates the constant number of background lines added to each image for the $bg_{constant}$ distribution. 26, 66

***bg*$_{single}$** A hyperparameter used when generating data to retrain AlexNet; it is one of the *bg* distributions. bg_{single} indicates that the images in the test set were created with no background lines. 25, 66

***bg*$_\theta$** A hyperparameter used when generating data to retrain AlexNet. bg_θ is the θ parameter of the gamma function used to determine the number of background lines for bg_{gamma}. 25, 66

***bg*$_{vert}$** A hyperparameter used when generating data to retrain AlexNet; it is one of the *bg* distributions. bg_{vert} indicates that the images in the test set were generated with two vertical background lines placed at the joints between the line segments. The colour of the background lines is determined by bg_{colour}. 25, 66

centered set An image set used for exploratory testing. It is composed of a test set of centered, connected Poggendorffs, and a control image whose diagonal is centered about the y-axis. v, 8, 9, 11–13

classification error A type of error used to evaluate the performance of the retrained AlexNet. Classification error takes the most probable result (according to the model) and measures how often that is a correct prediction. 26

connected One of the two possible *lt*s, the other being disconnected. Connected can refer to an element, image, or category of experiment. A connected element is one that has all three line segments drawn. A connected line has these three line segments aligned, forming a line; a connected non-line has these three line segments offset. A connected image contains a connected element as its primary element. As a category, it indicates that the experiment was run on connected images. i, 7, 14, 21, 25, 35–43, 46, 49–52, 67, 68

diagonal The primary element of a Poggendorff-like image. It is the line segments that crosses the two vertical lines at an angle. The perception of this line's straightness is affected by the Poggendorff illusion. 2, 4, 8, 10, 12, 14–17, 42, 44, 46, 48, 50

disconnected One of the two possible *lt*s, the other being connected. Disconnected can refer to an element, image, or category of experiment. A disconnected element is one in which only the first and last line segments are drawn, omitting the middle line segment. A disconnected line has these two line segments aligned; a disconnected non-line has them offset. A disconnected image contains a disconnected element as its primary element. As a category, it indicates that the experiment was run on disconnected images. i, 14, 21, 25, 35–43, 46, 49–52, 66, 68

element A line or non-line. i, 18, 22, 35, 37, 39, 41, 51, 52, 66–68

epoch of min val error A metric used in tables to summarize the results of a retraining experiment. It stands for epoch of minimum validation error, and refers to the epoch at which the minimum validation error occurred. As such, it acts as an estimate for how quickly the network was trained, in conjunction with the error curves. 28–30, 32, 34, 36–38, 42

final A hyperparameter used when retraining AlexNet. $final$ is the final function used during retraining. Either a $sigmoid$ or $softmax$ function was used. 25

final test error A metric used in tables to summarize the results of a retraining experiment. Final test error refers to a network's classification error over the test set, and thus approximates the overall model performance. "Final" indicates that the error was taken at the end of the training session. Because of this, final test error is is only valid when the validation classification error curve does not exhibit overfitting. 28–30, 32, 34, 36–38, 42, 67, 68

image set A set of images used for experiments. In the context of exploratory testing, this refers to both a set of test images (a test set) and the control that was compared to these test images. In the context of retraining AlexNet, this refers to the training, validation and test sets.. 6–8, 10–12, 14–17, 22, 66, 68

***ja*$_k$** A hyperparameter used when generating data to retrain AlexNet. ja_k is the k parameter of the gamma function used to generate the random joint angle. 25

***ja*$_\theta$** A hyperparameter used when generating data to retrain AlexNet. ja_θ is the θ parameter of the gamma function used to generate the random joint angle. 25

layer A hyperparameter used when retraining AlexNet. *layer* indicates the lowest layer of AlexNet that was retrained; it and all subsequent layers were retrained. 25

learned state A trained network property indicating both that the validation classification error curve has leveled out and that the curve does not exhibit overfitting. The curve being level, (ignoring inevitable noise,) indicates that no further training is required. The lack of overfitting indicates that the final test error is valid. 28, 30, 33, 35–38, 41, 57–62

left-right orientation An orientation used for some exploratory image sets so that no black pixels in the test set occurs at the same position as any black pixels in the control image. Left-right indicates that every line in every test set image is placed on the left half of the image, and all lines in the the control image are placed on the right half. Not to be confused with right-left orientation. 10, 15, 68

loss error A type of error used to evaluate the performance of the retrained AlexNet. Loss error is the error according the network's loss function. 26

lr A hyperparameter used when retraining AlexNet. *lr* is the learning rate used for stochastic gradient descent. 25

lt A hyperparameter used when generating data to retrain AlexNet. *lt* is the line type, either connected or disconnected. 25, 66, 67

minimum validation error A metric used in tables to summarize the results of a retraining experiment. Minimum validation error refers to the lowest validation error seen accross all training epochs. Since this measure is taken over hundreds of epochs, it risks under-representing true error; it should be compared to the final test error which was only measured once. 28–30, 32, 34, 36–38, 42, 67

offset set An image set used for exploratory testing. It is composed of a test set of centered, connected Poggendorffs, and a control image whose diagonal is centered about the y-axis. v, vi, 10, 12, 14–17, 54

overfitting A type of modeling error. Overfitting occurs when a network is trained so that it models the particular characteristics of the training data (including any noise or error present in this data), instead of modeling the actual underlying distribution. This causes the validation error to increase. For neural networks, overfitting usually occurs when a model is trained for too long. Though the training error decreases, the validation error may begin to increase after some number of epochs. 26–28, 67

primary element The element in an image that determines the image's label as either containing a line or non-line. 22, 25, 40–42, 50, 52, 66, 67

raw error A type of error used to evaluate the performance of the retrained AlexNet. Raw error is a measure of how far the prediction probability is from the correct probability. 26

right arm The right-most line segment of the diagonal in Poggendorff-like images. This is the line segment that lies to the right of the rightmost vertical line. 8, 10–12, 14, 42

right-left orientation An orientation used for some exploratory image sets so that no black pixels in the test set occurs at the same position as any black pixels in the control image. Right-left indicates that every line in every test set image is placed on the right half of the image, and all lines in the the control image are placed on the left half. Not to be confused with left-right orientation. 10, 12, 15, 68

simple set An image set used for exploratory testing. It is composed of a test set of centered, connected Poggendorffs, and a control image whose diagonal is at the bottom of the image. v, 8–13

spread image An image that summarizes a test set used for exploratory testing. Since all exploratory testing was done with images composed of only white or black pixels, spread images use colour to represent how the test set changes across indices. Any pixels that are constant across the test set (always black or always white) will appear unchanged in the spread image. For pixels that change, the *red* areas represent black pixels in low-index images, whereas *blue* areas represent black pixels in high-index images. 7, 9–11, 13, 14, 44, 69

test set A set of test images used for experiments. In the context of exploratory testing, these are the images compared to a control, where each image is uniquely identified by its index (starting at 0) within the set. Exploratory test sets are summarized in a spread image. In the context of retraining AlexNet, a test set is a set of images distinct from the training and validation sets used to measure the training error. 6, 8, 11–13, 41, 42, 66–69

References

[1] Nicholas Baker, Hongjing Lu, Gennady Erlikhman, and Philip J. Kellman. Deep convolutional networks do not classify based on global object shape. Configural Processing Consortium 2017, 2017.

[2] G. Barbara. A depth processing theory of the poggendorff illusion. *Perception & Psychophysics*, 10(4): 211–216, 1971.

[3] Matteo Carandini, Jonathan B. Demb, Valerio Mante, David J. Tolhurst, Yang Dan, Bruno A. Olshausen, Jack L. Gallant, and Nicole C. Rust. Do we know what the early visual system does? *Journal of Neuroscience*, 25(46):10577–10597, 2005. ISSN 0270-6474. doi: 10.1523/JNEUROSCI.3726-05.2005. URL `http://www.jneurosci.org/content/25/46/10577`.

[4] Radoslaw Martin Cichy, Aditya Khosla, Dimitrios Pantazis, Antonio Torralba, and Aude Oliva. Comparison of deep neural networks to spatio-temporal cortical dynamics of human visual object recognition reveals hierarchical correspondence. *Scientific Reports*, 6:27755 EP –, Jun 2016. URL `http://dx.doi.org/10.1038/srep27755`. Article.

[5] R.H. Day and R.T. Kasperczky. Apparent displacement of lines and dots in a parallel-line figure: A clue ot the basis of the poggendorff effect. *Perception & Psychophysics*, 38:74–80, 1985.

[6] A.J. de Brouwer, J.B.J. Smeets, T.P. Guttenling, I. Toni, and W.P. Medendorp. The mller-lyer illusion affects visuomotor updating in the dorsal visual stream. *Neuropsychologia*, 77:119–127, 2015.

[7] JamesJ. DiCarlo, Davide Zoccolan, and NicoleC. Rust. How does the brain solve visual object recognition? *Neuron*, 73(3):415 – 434, 2012. ISSN 0896-6273. doi: https://doi.org/10.1016/j.neuron.2012.01.010. URL `http://www.sciencedirect.com/science/article/pii/S089662731200092X`.

[8] M.A. Goodale. Transforming vision into action. *Vision Research*, 51:1567–1587, 2011.

[9] E. Greene. The corner poggendorff. *Perception*, 17:65–70, 1988.

[10] R.L. Gregory. A discussion on the logical analysis of cerebral functions perceptual illusions and brain models. *Proceedings of the Royal Society of London. Series B, Biological Sciences*, 171(1024):279–296, 1968.

[11] Michael Guerzhoy. tf_weights. `https://github.com/guerzh/tf_weights`, 2017.

[12] ImageNet. Imagenet large scale visual recognition challenge 2012 (ilsvrc2012), 2012. URL `http://www.image-net.org/challenges/LSVRC/2012/results.html`.

[13] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross B. Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *CoRR*, abs/1408.5093, 2014. URL `http://arxiv.org/abs/1408.5093`.

[14] S. Khaligh-Razavi and N. Kriegeskorte. Deep supervised, but not unsupervised, models may explain it cortical representation. *PLOS Computational Biology*, 2014.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. URL `http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf`.

[16] James A. Mazer and Jack L. Gallant. Object recognition: Seeing us seeing shapes. *Current Biology*, 10 (18):R668 – R670, 2000. ISSN 0960-9822. doi: https://doi.org/10.1016/S0960-9822(00)00684-9. URL `http://www.sciencedirect.com/science/article/pii/S0960982200006849`.

[17] Matthew Nyhus. AlexNetPoggendorff. `https://github.com/mdnyhus/AlexNetPoggendorff`, 2018.

[18] R. VanRullen. Perception science in the age of deep neural networks. *Frontiers in Pyschology*, 2017.

[19] Donglai Wei, Bolei Zhou, Antonio Torralba, and William T. Freeman. Understanding intra-class knowledge inside CNN. *CoRR*, abs/1507.02379, 2015. URL `http://arxiv.org/abs/1507.02379`.

[20] R. Weidner, T. Plewan, Q. Chen, A. Bunchner, P.H. Weiss, and G.R. Fink. The moon illusion and size-distance scaling evidence for shared neural patterns. *Journal of Cognitive Neuroscience*, 26(8): 1871–1882, 2014.

[21] D. Yamins and J.J. DiCarlo. Using goal-driven deep learning models to understand sensory cortex. *Nature Neuroscience*, 19:356–365, 2016.

[22] D. Yamins, H. Hong, C. Cadieu, and J.J. DiCarlo. Hierarchical modular optimization of convolutional networks achieves representations similar to macaque it and human ventral stream. *Advances in Neural Information Processing Systems*, 2013.

[23] D. Yamins, H. Hong, C. Cadieu, E.A. Solomon, D. Seibert, and J.J. DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the National Academy of Sciences*, 111(23):8619–8624, 2014.

[24] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. *CoRR*, abs/1412.6856, 2014. URL `http://arxiv.org/abs/1412.6856`.