# When is visual attention useful?

Freddie Bickford-Smith

*Primary supervisors*
Bradley C Love
Brett D Roads

*Secondary supervisor*
Edward Grefenstette

University College London
September 2019

This report is submitted as part requirement for the MSc degree in Data Science & Machine Learning at University College London. It is substantially the result of my own work except where explicitly indicated in the text. The report may be freely copied and distributed provided the source is explicitly acknowledged. See `github.com/fbickfordsmith/attention` for associated code.

# Abstract

Visual attention, the ability to prioritise the processing of selected visual information, is key to how we perceive the world. Recent work has shown that deep convolutional neural networks are good models of human vision. We use such a network, incorporating attention as a multiplicative weighting of neural representations of visual stimuli. We define a category set to be a grouping of ImageNet categories. Learning to apply attention to a particular category set represents a distinct cognitive task. We hypothesise that a category set's difficulty, size and visual similarity influence the usefulness of attention (the accuracy improvement attention provides on a task). Using results from more than 90 trained computer models, we show that usefulness correlates positively ($\beta_1 = 0.30$) and strongly ($R^2 = 0.92$) with category-set difficulty, negatively ($\beta_1 = -0.04$) and strongly ($R^2 = 0.94$) with category-set size (on a logarithmic scale), and negatively ($\beta_1 = -0.11$) and weakly ($R^2 = 0.37$) with the visual similarity within a category set. The first two relationships agree with our intuitions; the third does not (we expected a positive correlation). Future efforts could focus on building more sophisticated models of visual attention. But there are, we argue, a number of natural extensions to our work that offer more tractable steps towards a better computational account of visual attention.

## Acknowledgements

This project was the product of a collaborative effort. Thanks to Brad Love, and the Love Lab more broadly, for providing an Inclusive, Productive, Accountable$^{\text{TM}}$ research environment; to Brett Roads, for thoughtful, smart advice and unfailing moral support; to Ken Luo, for his semantic sets of ImageNet categories and an eagle-eyed code review; and to Ed Grefenstette, for cool ideas and fun conversations.

# Contents

# 1 Introduction

Opening our eyes, we are overwhelmed with information. Yet we effortlessly understand the visual world. We apply attention, separating the relevant from the irrelevant. Attention strongly shapes perception: even with a fixed retinal image, a change in attentional state can affect neural activity in the visual cortex (Carrasco and Barbot, 2019). This makes clear that vision's role is to provide a **useful**, not necessarily truthful, representation of the world.

So when is visual attention useful? It certainly feels like we need visual attention some times more than others. Finding a needle in a haystack would be easier with less hay. How can we check this intuition empirically? In this work we focus on image recognition, in which the aim is to classify images into categories as accurately as possible. We take a state-of-the-art computer model of human vision, and incorporate visual attention as a reweighting of the model's representations of visual stimuli (Section 3.2). Using this, we show that visual attention improves accuracy on image-recognition tasks. But its **usefulness** (the accuracy boost it provides) varies between tasks (Figure 10).

We aim to explain this phenomenon of non-uniform usefulness. We define a **category set** to be a grouping of image categories (Section 3.1). Learning to apply attention to a particular category set represents a distinct cognitive task. We hypothesise that there exist some key **category-set properties** that determine the extent to which attention improves performance. We consider three candidates:

1. Difficulty (average error rate of an industry-standard image-recognition system on examples in the category set)
2. Size (number of categories in the category set)
3. Visual similarity (average similarity in the way a computer model of human vision 'sees' images in the category set)

Why prioritise studying these properties? Each has a well-defined, quantitative definition. Each also intuitively seems to be important. Think of trying to spot a friend in a train station. It seems like applying attention would be more useful if the station were busy rather than empty (ie, visual clutter, a component of task difficulty, matters). Attending to particular visual features might be particularly useful if we knew that our friend was wearing a red coat, allowing us to narrow down our visual search (ie, task size matters). And it feels like an eye for detail would be more useful if, by some freak event of bad luck, everyone else in the station looked just like your friend (ie, visual similarity matters).

We show in Figure 11 that, between sets of conceptually similar categories, there is substantial variation with respect to difficulty, size and visual similarity (our category-set properties). At the same time, we see variation in the extent to which attention, when optimised for a particular set of conceptually similar categories, provides an improvement in image-classification accuracy (Figure 10). Is there a link between these observations? In other words, do these three properties of a task have an effect on how useful attention is in a task? Visual inspection of Figures 10 and 11, as well as statistical analysis (Table 2), suggests there might be. But we have insufficient evidence to confidently claim that there is a relationship. Furthermore, we seek to establish not just the **existence** of a relationship, but also to understand its **nature**.

We aim to answer this question empirically. In Section 3 we describe four experiments. Each is comprised of training a series of attention-augmented neural networks (our models of human vision). Each attention network is trained solely on examples from a single category set. Its accuracy is then compared to that of a fixed, attention-free network (our baseline). We control the properties of the category sets used to train the networks, and then observe how the impact of attention varies. By training and evaluating more than 90 attention models, we gather a large collection of evidence. With this, we can better assess our hypotheses.

Why is this important? Visual attention in humans is still not well characterised from a computational perspective (Carrasco, 2011). Recent work has shown that particular neural networks are state-of-the-art models of the human visual system (Yamins and DiCarlo, 2016). Thus, we seize the novel opportunity to model visual attention to a higher level of sophistication. With better understanding of **when** attention works (ie, in which tasks it is most and least effective), we are better equipped to work out **how** it works.

Better characterisation of attention could influence the way we apply neural networks to solve problems. When we develop a new machine-learning system, we make design decisions informed by accumulated knowledge, both empirical and intuitive. With more knowledge, we waste less time exploring the tradeoffs associated with a design feature. Furthermore, if the attention mechanism is designed to emulate its functional counterpart in the human brain, the neural network might make more human-like errors. If this improves the interpretability of a neural network's behaviour, this is valuable (Gilpin et al, 2018). In summary, this work has the potential to advance the science of intelligence in a broad sense.

# 2 Background

In our experiments we use neural networks to model attention in the human visual system. Understanding our method requires some background knowledge of key ideas. First are the principles of feedforward and convolutional networks. With a conceptual grasp of these, we can discuss VGG16, the particular instance of convolutional network that we use. Closely related is ImageNet, the dataset for which VGG16 was designed. VGG16, and networks like it, are used in computational cognitive science to model human cognitive functions like vision. Particularly connected to our contribution is work on the use of neural networks to model visual attention. A review of this shows that the research question we aim to answer is both relevant and interesting.

## 2.1 Neural networks

The study of neural networks, computer programs whose design is inspired by biological brains, dates back to at least the 1940s (McCulloch and Pitts, 1943; Hebb, 1949). Their theoretical potential was well established by the late twentieth century (Cybenko, 1989; Hornik et al, 1989). Relatively recent, however, is the ability to train a deep neural network—that is, build a many-layered network and optimise it for a specific task—with a reasonable amount of time and computing power (Hinton et al, 2006). Deep networks now represent the state of the art in tasks ranging from the automatic translation of foreign languages (Sutskever et al, 2014) to the generation of natural-sounding synthetic speech (van den Oord et al, 2016).

### Feedforward networks

A feedforward neural network is a mathematical function mapping input to output through a series of simpler functions, organised in **layers** (Goodfellow et al, 2016). Let $f_\theta$ be a network that maps a vector input, $x$, to an output, $y$, through $L$ hidden layers, $h_1, \ldots, h_L$ (Figure 1). The activity at the first hidden layer is computed by taking $x$ and applying an affine transformation (multiplying by a weight matrix, $W_1$, and adding a bias vector, $b_1$) followed by a transfer function, $g_1$:

$$h_1 = g_1(W_1 x + b_1) \tag{1}$$

The activity of each subsequent layer is a function of the layer before it:

$$h_l = g_l(W_l h_{l-1} + b_l) \text{ for } l \in [2, L] \tag{2}$$

$$f_\theta(x) = g_L(W_L h_{L-1} + b_L) \tag{3}$$

where $W_l \in \mathbb{R}^{d_l \times d_{l-1}}$ and $b_l \in \mathbb{R}^{d_l}$, with $d_l = \dim(h_l)$. A **deep network** is simply one with many layers.

The parameters of a network, $\theta = \{(W_1, b_1), (W_2, b_2), \ldots, (W_L, b_L)\}$, control the input-output mapping, and are tuneable. The performance of a network is measured using a **loss function**, $J$. Learning is the process of tuning $\theta$ such that $J$ is minimised. This is typically achieved through gradient-based optimisation. First, automatic differentiation is used to compute $\nabla_\theta J$, the gradient of the loss with respect to the network parameters. Second, a small change is made to $\theta$ in the direction of greatest reduction in $J$:

$$\theta_{t+1} = \theta_t - \alpha \nabla_\theta J \tag{4}$$

where $t$ denotes the time step and $\alpha$ denotes the learning rate. If the exact gradient is computed using all examples in the **training set** (the collection of $(x, y)$ pairs used to train the network), this update to $\theta$ is called gradient descent. If an approximate gradient is computed using only a sample of training examples, it is called **stochastic gradient descent**.

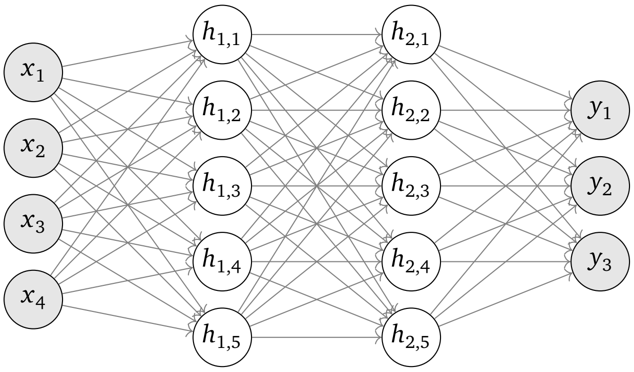


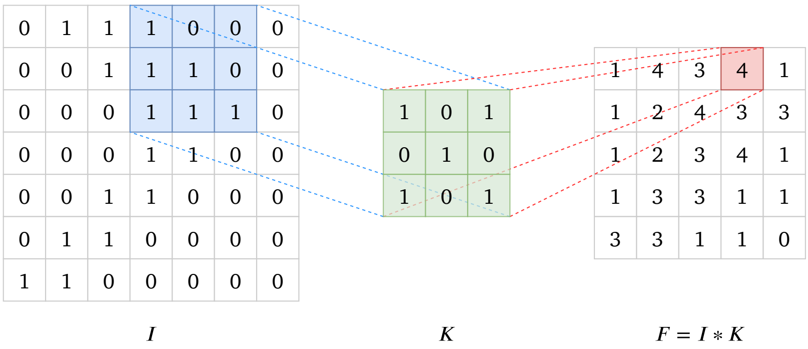
Figure 1: A feedforward neural network. This is a mathematical function transforming $x$ to $y$ through a series of hidden layers, $h_1, \ldots, h_L$. The activity at each layer is computed by taking the activity of the previous layer and applying an affine transformation followed by a (typically non-linear) transfer function. The parameters of the network, $\theta = \{(W_1, b_1), (W_2, b_2), \ldots, (W_L, b_L)\}$, control the input-output mapping, and are tuneable.

### Convolutional networks

A convolutional neural network is a network specialised for input data that has a grid-like structure (Goodfellow et al, 2016). This type of network derives its name from the convolution operation that it uses in at least one layer. Let $I$ be a two-dimensional array of values representing an image, and $K$ be a two-dimensional filter (an array of tuneable weights). The **feature map**, $F$, produced by the convolution of $I$ with $K$ is given by

$$F_{i,j} = (I * K)_{i,j} = \sum_m \sum_n I_{i-m,j-n} K_{m,n} \tag{5}$$

Intuitively, convolving $I$ with $K$ involves sliding the filter across the width and height of $I$. At each position the dot product is taken between $K$ and the elements of $I$ over which $K$ floats (Figure 2). As in feedforward networks, this linear operation is typically followed by adding a bias and applying a nonlinear transfer function.



|  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 |

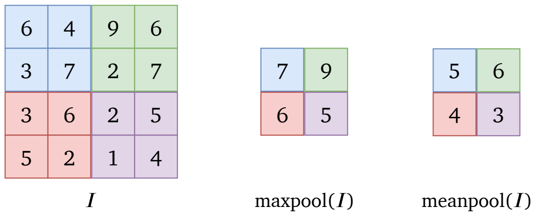
|  |  |  |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

|  |  |  |  |  |
|---|---|---|---|---|
| 1 | 4 | 3 | 4 | 1 |
| 1 | 2 | 4 | 3 | 3 |
| 1 | 2 | 3 | 4 | 1 |
| 1 | 3 | 3 | 1 | 1 |
| 3 | 3 | 1 | 1 | 0 |

Figure 2: The convolution operation in two dimensions. Computing the output feature map, $F = I * K$, involves sliding an array of weights, $K$, to each position on the input array, $I$. At each location the dot product is performed between $K$ and the elements of $I$ over which $K$ floats. Drawing inspired by Velickovic (2018).

It is common to also use **pooling operations** (Figure 3) in convolutional networks. Pooling is similar to convolving: a window slides to each position in an input array, $I$, and performs an operation on the elements of $I$ that it covers. The difference is that the operation is nonparametric, typically an aggregation such as the maximum or mean value in a local area.

Feedforward networks canonically operate on vectors. In contrast, at layer $l$ in a convolutional network, a two-dimensional image is typically represented as a $H_l \times W_l \times C_l$ tensor, where $H_l$ is the height, $W_l$ is the width and $C_l$ is the number of channels. Each channel represents a **distinct visual feature** of the input image. Slicing the tensor along the third dimension, it can be viewed as a set of



| | | | |
|---|---|---|---|
| 6 | 4 | 9 | 6 |
| 3 | 7 | 2 | 7 |
| 3 | 6 | 2 | 5 |
| 5 | 2 | 1 | 4 |

$I$

| | |
|---|---|
| 7 | 9 |
| 6 | 5 |

maxpool($I$)

| | |
|---|---|
| 5 | 6 |
| 4 | 3 |

meanpool($I$)

Figure 3: Pooling operations in two dimensions. A window slides to each position an input array, $I$. At each location a statistical summary of the covered elements of $I$ is computed.

$C_l$ feature maps, each representing how strongly a feature is present at different locations. A convolutional network typically comprises a series of convolutional layers followed by at least one standard feedforward layer (Figure 4).

The design of convolutional networks encodes assumptions about the input data. In vision applications, two assumptions are commonly cited (Krizhevsky et al, 2012). First is that locality matters: the information content of one location in the input tells us something about the content of an adjacent location. This property is reflected by the use of locally, rather than fully, connected neurons in convolutional layers. In a feedforward network each layer is fully connected: each neuron in layer $l$ is connected to every neuron in layer $l-1$. In a convolutional layer, each neuron in layer $l$ is only connected to a local rectangular area (receptive field) in layer $l-1$. Second is translation invariance, meaning that an object's appearance is independent of location. This is realised by the use of weight sharing in convolutional layers. Each feature map is computed using a single filter applied to the whole input, not a collection of separate filters assigned to distinct locations.

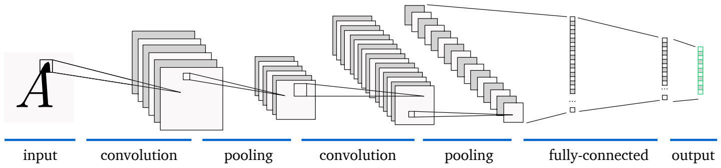


Figure 4: A convolutional neural network for processing images. Convolutional layers represent two-dimensional images as three-dimensional tensors. Slicing a tensor representation along the third dimension, it can be viewed as a collection of feature maps. Each map encodes the variation of a visual feature with respect to space. Typically at least one fully-connected layer precedes the output neurons. Drawing from Hill (2017).

Empirical studies have demonstrated how strong the built-in bias of convolutional networks is, and how appropriate it is for images (Saxe et al, 2011). Even with randomised filters, a single convolutional layer—comprising convolution, nonlinearity, normalisation and mean pooling—extracts features with which a linear classifier can achieve 53% accuracy on Caltech101, an image-recognition dataset with 101 classes (Jarrett et al, 2009).

## 2.2 ImageNet and VGG16

The popularity of convolutional networks today can be traced back to the introduction of a new image-recognition benchmark in 2009. In its full form, **ImageNet** is a dataset comprising over 14 million images, each belonging to one of roughly 22,000 categories (Deng et al, 2009). Categories are organised according to the conceptual hierarchy of WordNet, a language dataset (Figure 5; Miller, 1995).

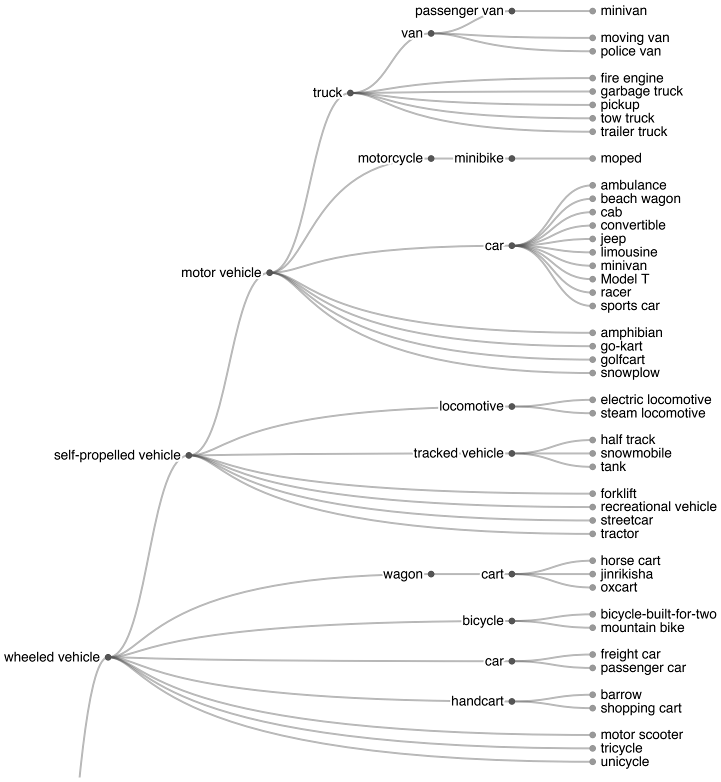


Figure 5: A subset of ImageNet categories, organised according to the hierarchical semantic structure of WordNet. Each leaf node is a category. Drawing from Bostock (2018).

Approximately 10 percent of these images, belonging to 1000 categories, were used for the annual ImageNet Large Scale Visual Recognition Challenge (ILSVRC; Russakovsky et al, 2014) from 2010 to 2017. The ILSVRC dataset is divided into three parts. The **training set**, with around 1.2 million images, is intended for training algorithms. The 50,000-image **public test set** is readily accessible and so can be used at any time to measure the performance of trained algorithms. A final 150,000 images form the **private test set**, which is reserved for evaluating performance in the competition setting. Henceforth the 2012 version of the ILSVRC dataset is what we mean when we refer to ImageNet.

**VGG16** (Figure 6) is a convolutional neural network proposed by Simonyan and Zisserman (2015) to compete in the 2014 ImageNet challenge. Compared to top-performing networks in previous competitions, this has two key design features. First, it is deep. It has 16 layers, twice as many as AlexNet (Krizhevsky et al, 2012). Second, it is simple. Whereas AlexNet uses convolutional filters ranging in size from $11 \times 11$ to $3 \times 3$, VGG16 uses $3 \times 3$ throughout.

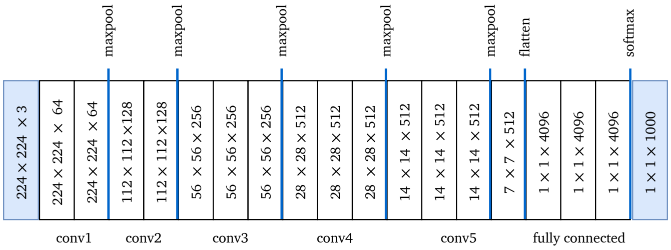


Figure 6: The structure of VGG16. This is a convolutional neural network whose distinctive features are its depth (16 layers) and its use of small $3 \times 3$ convolutional filters throughout. VGG16 takes as input an image 224 pixels tall, 224 pixels wide and with three channels (red, green, blue). Moving forwards through the convolutional layers, the tensor representation of the input becomes smaller in its spatial dimensions but larger in its channel dimension. The output of the fifth convolutional block is flattened into a vector before being passed through three fully-connected layers. The output of the last fully-connected layer is passed to a softmax function. The result is a probability distribution over the 1000 ImageNet category labels.

Combining these design features leads to more efficient use of network parameters. A stack of three $3 \times 3$ convolutional layers, without pooling in between, has an effective receptive field of $7 \times 7$. But whereas a $7 \times 7$ convolutional layer has a single nonlinearity, the stack of $3 \times 3$ layers has three. Thus, for an equivalent receptive field, the stack can achieve greater representational capacity. It also uses fewer parameters. If each filter in this example has $C$ channels, the stack uses $27C^2$ weights, and the single $7 \times 7$ layer uses $49C^2$.

## 2.3 Computational models of cognition

Following the successful application of neural networks to image-recognition tasks like the ImageNet challenge, a growing body of work has analysed how such networks represent visual stimuli and how these **representations** relate to those recorded in biological brains (Yamins and DiCarlo, 2016). Neural networks have been found to produce remarkably similar representations to those in primate brains (Schrimpf et al, 2017). These computer programs can thus serve as a useful model of the biological systems that originally informed their invention. This is a clear demonstration of a virtuous cycle in which neuroscience informs machine learning and vice versa (Hassabis et al, 2017).

It is important to clarify what it means for a neural network to model the brain. Marr and Poggio (1977) argued that information-processing systems can be decomposed into levels of abstraction that are mostly independent of each other and should be studied as such. Marr (1982) identified three levels of analysis. First is the computational, specifying what the system does—the mapping it performs of one form of information to another—and why. Second is the algorithmic, describing how the system represents and manipulates information to perform its function. Third is the implementational, concerning how the system is realised in hardware.

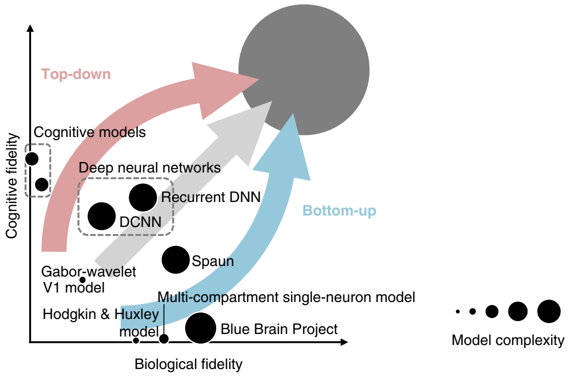


Figure 7: Computational models of cognition. A model can vary in its level of abstraction, its complexity (number of parameters), and its biological and cognitive fidelity. Bottom-up approaches to modelling emphasise the implementational details of neurons. Top-down approaches focus on the algorithmic form of cognitive functions. With current models there is a tradeoff to be made between high-level and low-level realism. The long-term aim of computational cognitive science is to create a model that sits in the upper-right corner. This will necessarily have the high complexity of the brain, represented by a large grey dot. Drawing from Kriegeskorte and Douglas (2018).

Even if a neural network does not closely resemble a brain in its implementation, it can be a useful **proxy for cognition** at higher levels of abstraction (Figure 7; Kriegeskorte and Douglas, 2018). Indeed, this is the core pursuit of computational cognitive science: to build computer models that perform real-world cognitive tasks. The aim is to explain measured neural activity and human behaviour.

## 2.4 Visual attention

Attention shapes visual perception by modulating neural activity (Lindsay and Miller, 2018). Viewed as a single mechanism, it is highly flexible. We can attend by location (Eriksen and St James, 1986), by visual feature (Liu and Hou, 2011), by object (Olson, 2001), by moment in time (Nobre, 2001). A vast body of literature covers this topic (Carrasco, 2011). This includes neural-network-based modelling of attention since at least as far back as the 1990s (Kruschke, 1992). Most relevant to our work are examples of using **deep networks** to model visual attention.

**Spatial attention**, in which focus is directed according to location, has been a popular area of research. A common way of implementing this is to frame vision as a task of combining information collected in a series of high-resolution glimpses, with an attention mechanism determining where to look (Larochelle and Hinton, 2010; Denil et al, 2012; Mnih et al, 2014; Ranzato, 2014; Ba et al, 2015). The spatial-transformer module proposed by Jaderberg et al (2015) is related to these approaches in the sense that it adaptively manipulates incoming information and is controlled with respect to location.

Non-spatial forms of attention have been studied significantly less. One example is **feature-based attention**, in which information is emphasised according to the attributes of an image instead of location. Stollenga et al (2014) developed a system that learns to sequentially process an image, using feedback loops between neural-network layers to dynamically weight the feature maps it extracts. In a related but purely feedforward approach, Lindsay (2015) incorporated feature-based attention into a convolutional network. In the proposed network, category-specific feature weightings are computed using the average response of features to images in that category. These are then used to modulate the network's representations at runtime. Chen et al (2017) blended both location-based and feature-based modulation of neural representations. When applied to VGG19 and ResNet, two popular convolutional networks, their model produced state-of-the-art performance on image-captioning tasks.

## 2.5 Moving forward

What can we conclude from what we have discussed? On the one hand, visual attention is still not fully characterised from a computational perspective. On the other, computer models of human vision are better than they have ever been. In this work, we aim to address the first fact with the second. Our approach to modelling attention builds on the work of others, particularly Chen et al (2017) and Lindsay and Miller (2018). The question we aim to answer—when is visual attention useful?—is a pertinent one. To our knowledge, the resulting work is a novel contribution to the study of visual attention.

# 3 Method

Our work is centred on a simple question: when is visual attention useful? Our approach is to work out how the nature of a task determines the extent to which attention can improve image-classification accuracy. The task we study is that of **learning to apply attention** in an image-recognition setting, where examples are drawn from a particular category set (grouping of ImageNet categories). By manipulating the properties (difficulty, size, visual similarity) of a category set, we control the nature of the task. For each category set, we train an attention network to classify images from the category set. We then compare the classification accuracy of the attention network to a standard VGG16. Our approach is summarised in Figure 8.

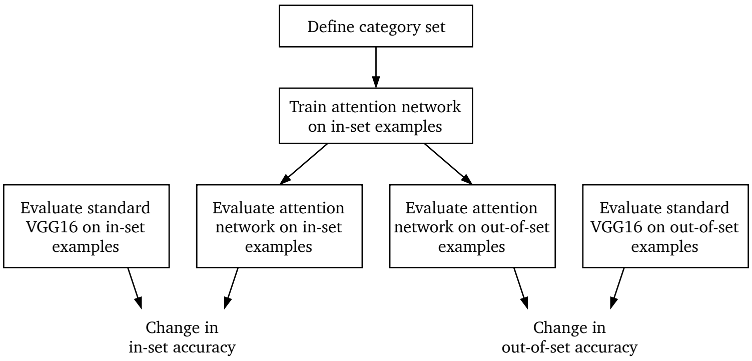

Define category set
Train attention network on in-set examples
Evaluate standard VGG16 on in-set examples
Evaluate attention network on in-set examples
Evaluate attention network on out-of-set examples
Evaluate standard VGG16 on out-of-set examples
Change in in-set accuracy
Change in out-of-set accuracy

Figure 8: Experiment design for characterising how the nature of a task (controlled by the choice of category set) determines the extent to which visual attention is useful (how much effect it has on image-classification accuracy). This procedure is repeated for each category set we define in Section 3.1.

## 3.1 Category sets

Let a category set be a grouping of ImageNet categories. One way of forming a category set is to group together categories that are conceptually similar. The result is a **semantic category set**. We can alternatively form category sets such that they have desired quantitative properties. We define three types of category set in this way: one type for each property that we study. First, we choose a collection of **difficulty-based category sets** such that they vary substantially in difficulty but not in size or visual similarity. Second are **size-based category sets**, which are diverse in size but are approximately fixed in difficulty and similarity. Third, **similarity-based category sets** have a range of visual similarity values but nearly constant difficulty and exactly constant size.

### Semantic sets

We define six semantic category sets (Table 1), each of which being a grouping of conceptually similar ImageNet categories.

| Category set | Example categories |
|---|---|
| kitchen items | spatula, frying pan, coffee mug, refrigerator, mixing bowl |
| wearable items | sweatshirt, fur coat, knee pad, bikini, backpack |
| cats | tabby, Siamese cat, lion, snow leopard, Persian cat |
| land transport | golf cart, moped, fire engine, unicycle, school bus |
| birds | magpie, hen, ostrich, albatross, pelican |
| dogs | beagle, Chihuahua, coyote, Siberian husky, red fox |

Table 1: Examples members of six semantic category sets. Each set contains conceptually similar ImageNet categories.

### Difficulty-based sets

Difficulty-based category sets vary in difficulty but are of the same size and approximately the same visual similarity. Let the **difficulty** of a generic category set, $\mathcal{C}$, be the mean error rate of a standard VGG16 on categories in that category set:

$$\text{difficulty}(\mathcal{C}) = \frac{1}{|\mathcal{C}|} \sum_{c_i \in \mathcal{C}} (1 - \text{accuracy}(c_i)) \tag{6}$$

where accuracy is measured on a scale from 0 to 1. We vary this property across 20 initial difficulty category sets, $\mathcal{A}_1, \ldots, \mathcal{A}_{20}$. To form these, we first arrange the 1000 ImageNet categories into a list, $c_1, \ldots, c_{1000}$, sorted by accuracy. We then split this list into 20 disjoint sets, each containing 50 categories:

$$\begin{aligned} \mathcal{A}_1 &= \{c_1, \ldots, c_{50}\} \\ \mathcal{A}_2 &= \{c_{51}, \ldots, c_{100}\} \\ &\vdots \\ \mathcal{A}_{20} &= \{c_{951}, \ldots, c_{1000}\} \end{aligned} \tag{7}$$

This is such that

$$\text{difficulty}(\mathcal{A}_1) \leq \text{difficulty}(\mathcal{A}_2) \leq \cdots \leq \text{difficulty}(\mathcal{A}_{20}) \tag{8}$$

Category sets $\mathcal{A}_1, \ldots, \mathcal{A}_{20}$ span a large range of difficulties, but the coverage of difficulty in the $[0.5, 0.8]$ interval is sparse (Figure 9). To improve coverage, we randomly sample 5 additional difficulty-based category sets, $\mathcal{A}_{21}, \ldots, \mathcal{A}_{25}$, using Algorithm 1 (Appendix A). This procedure is designed to minimise variation in visual similarity across category sets. The sets are equally sized.

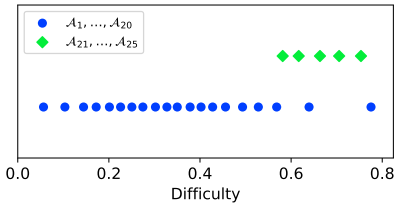


Figure 9: Difficulty values of 25 difficulty-based category sets. 20 initial difficulty category sets, $\mathcal{A}_1, \ldots, \mathcal{A}_{20}$, provide poor coverage of the difficulty range between 0.5 and 0.8. In order to improve coverage, we use Algorithm 1 to sample an additional five difficulty category sets, $\mathcal{A}_{21}, \ldots, \mathcal{A}_{25}$.

### Size-based sets

Size-based category sets vary in the number of categories they contain but have approximately the same difficulty and visual similarity. We sample 10 size-based category sets, $\mathcal{B}_1, \ldots, \mathcal{B}_{10}$, where

$$|\mathcal{B}_i| \in \{2, 4, 8, 16, 32, 64, 96, 128, 192, 256\} \tag{9}$$

We use (approximately) geometric spacing of sizes based on the intuition of diminishing marginal effect. If category-set size does impact the usefulness of attention, we expect the difference between a 2-category set and a 3-category set to be greater than the difference between a 200-category set and a 201-category set. Based on this, a natural choice of sizes is $|\mathcal{B}_i| \in \{2^1, 2^2, \ldots, 2^8\}$. But this has undesirable gaps in size towards the upper end of the range. So, in addition to this set of sizes, we include 96 and 192. With Algorithm 2 (Appendix A), we aim to ensure that size-based category sets have approximately equal difficulty and visual similarity.

### Similarity-based sets

Visual similarity can be measured in many ways (Zhang et al, 2018). Our approach uses VGG16 representations of ImageNet examples. To find the representation of an image, we pass the image forwards through VGG16 and find the activation of the penultimate layer (a 4096-dimensional vector). This is the same approach as followed by Socher et al (2014) and Birodkar et al (2019). For category $c_i$ we compute the representations of all examples, $x_{i,j}$, in it. Then we average these to obtain $r_i$, a **category representation** for $c_i$:

$$r_i = \frac{1}{|c_i|} \sum_{j}^{|c_i|} \mathrm{VGG}'(x_{i,j}) \tag{10}$$

where $|c_i|$ denotes the number of examples in $c_i$, and VGG$'$ denotes VGG16 with its final layer removed. We define the visual similarity, $s_{i,j}$, of categories $i$ and $j$ as the cosine similarity between $r_i$ and $r_j$:

$$s_{i,j} = \frac{r_i \cdot r_j}{||r_i||||r_j||} \tag{11}$$

We define the **visual similarity** of a generic category set, $\mathcal{C}$, as the mean pairwise similarity of the categories in it:

$$\mathrm{similarity}(\mathcal{C}) = \frac{1}{|\mathcal{C}|^2 - |\mathcal{C}|} \sum_{(i,j) \in \mathcal{C} \times \mathcal{C}} s_{i,j} \mathbb{1}(i \neq j) \tag{12}$$

where $\mathcal{C} \times \mathcal{C}$ denotes the Cartesian product of $\mathcal{C}$ with itself and $\mathbb{1}$ represents an indicator function. Using Algorithm 3 (Appendix A), we randomly sample 20 similarity-based category sets, $\mathcal{E}_1, \ldots, \mathcal{E}_{20}$, such that there is a substantial range in similarity while maintaining approximately equal difficulty. For all $i$, $|\mathcal{E}_i| = 50$.

## 3.2 Neural networks

Following Lindsay and Miller (2018), we use an ImageNet-pretrained VGG16 (Section 2.2) as a base neural network. This network computes a probability distribution over category labels, $y_1, \ldots, y_{1000}$, for a given image, $x$:

$$p(y|x) = \mathrm{VGG}(x) \tag{13}$$

We decompose the network into two parts. The convolutional layers, VGG$_1$, transform $x \in \mathbb{R}^{224 \times 224 \times 3}$ to a hidden representation, $h \in \mathbb{R}^{7 \times 7 \times 512}$. The fully-connected layers, VGG$_2$, transform $h$ to $p(y|x)$. That is,

$$\mathrm{VGG}_1 : x \rightarrow h \tag{14}$$

$$\mathrm{VGG}_2 : h \rightarrow p(y|x) \tag{15}$$

We define attention as a linear weighting of $h$ by nonnegative attention weights, $z \in \mathbb{R}_{\geq 0}^{7 \times 7 \times 512}$. Our **attention network** computes

$$p(y|x,z) = \mathrm{VGG}_2(z \odot \mathrm{VGG}_1(x)) \tag{16}$$

where $\odot$ denotes an elementwise multiplication. We treat the pretrained $\mathrm{VGG}_1$ and $\mathrm{VGG}_2$ as fixed functions. The attention weights, $z$, are the only trainable parameters.

## 3.3 Training

For each category set we train a separate attention network solely on examples from that set's categories. All attention weights are initialised to 1. This means that, at the beginning of training, the attention network has the same input-output mapping as a standard VGG16 (an elementwise multiplication by an array of ones has no effect). The heterogeneity of the pretrained weights in our attention network ensures that weight-space symmetry (Goodfellow et al, 2016) should not be an issue.

Gradients are computed using automatic differentiation. To update the attention weights we use Adam (Kingma and Ba, 2015), a variant of stochastic gradient descent, with a learning rate of 0.0003 (Karpathy, 2019). Each update to the attention weights is computed using a batch of 256 images. The images are preprocessed using the same procedure as used by Simonyan and Zisserman (2015). 10 percent of the training set is reserved as a validation set for evaluating the loss function after each epoch (full pass through the training set). Letting $J_t^v$ be the validation-set loss at epoch $t$, we stop training if $(J_{t-1}^v - J_t^v)/J_{t-1}^v < 0.001$ for two consecutive epochs. Unless this condition is met first, training ends after 300 epochs.

## 3.4 Evaluation

We measure performance on the publicly available ImageNet test set (Section 2.2) using two measures. For a generic category set, $\mathcal{C}$, the **in-set accuracy** is the rate of correctly classifying examples from $\mathcal{C}$:

$$\text{in-set accuracy} = \frac{1}{\sum_i^{|\mathcal{D}|} \mathbb{1}(y_i^t \in \mathcal{C})} \sum_i^{|\mathcal{D}|} \mathbb{1}(\operatorname{argmax}(p(y|x_i^t, z)) = y_i^t) \mathbb{1}(y_i^t \in \mathcal{C}) \tag{17}$$

where $\mathcal{D} = \{(x_1^t, y_1^t), \dots, (x_N^t, y_N^t)\}$ denotes the public ImageNet test set ($N = 50,000$). The **out-of-set accuracy** is the rate for all other ImageNet examples:

$$\text{out-of-set accuracy} = \frac{1}{\sum_i^{|\mathcal{D}|} \mathbb{1}(y_i^t \notin \mathcal{C})} \sum_i^{|\mathcal{D}|} \mathbb{1}(\text{argmax}(p(y|x_i^t, z)) = y_i^t) \mathbb{1}(y_i^t \notin \mathcal{C}) \tag{18}$$

For both measures, 0 is the minimum value and 1 is the maximum.

# 4 Results

In our experiments we modelled the cognitive task of learning to apply visual attention for a limited set of object types. To do this, we first defined a series of category sets (groupings of ImageNet categories). These were designed to vary with respect to three category-set properties (difficulty, size, visual similarity), which we expected to influence the performance-altering effect of visual attention. For each category set we trained a new attention network. Then we compared each attention network to a standard VGG16 in terms of in-set accuracy (the rate of correctly classifying examples from within the category set used for training) and out-of-set accuracy (the rate for all other ImageNet examples).

## 4.1 Semantic sets

In Section 3.1 we defined six semantic category sets. Each of these is a grouping of conceptually similar ImageNet categories. In Figure 10 we see that visual attention's impact on image-classification accuracy is highly dependent on the category set to which it is applied. The reason for this is not obvious. Indeed, in Section 1 we referenced this finding as part of the motivation for this work.

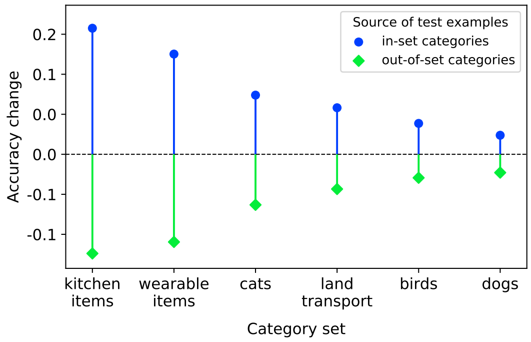


Figure 10: Classification accuracy (relative to a standard VGG16) of attention networks trained on six semantic category sets. Each semantic category set is a grouping of conceptually similar ImageNet categories. We see substantial variation in the boost in in-set accuracy that attention produces: a standard deviation of 0.05, where accuracy is measured on a scale from 0 to 1.

In Section 1 we suggested that differences in the difficulty, size and visual similarity of semantic category sets might explain the variation. In Figure 11 we show that these properties vary substantially between the sets. Table 2 shows that there might be a link between the results in Figure 10 and the properties in Figure 11. But, with high associated $p$ values, it is not possible to confidently rule out the possibility that the correlations are simply due to unrelated coincidence.

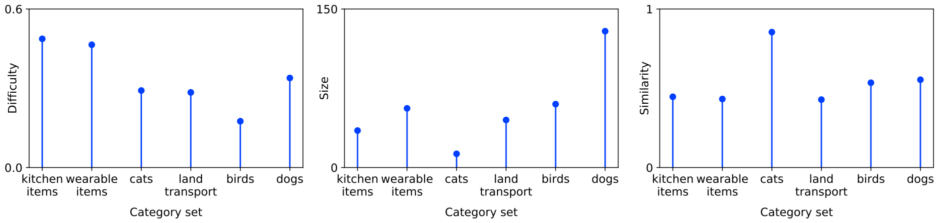


Figure 11: Properties of the semantic category sets referenced in Figure 10. Difficulty is the average error rate of a standard VGG16 on examples in the category set. Size is the number of categories in the set. Similarity is the mean cosine similarity of VGG16 representations of images in the set.

| **Property** | $\rho$ | $p_\rho$ | $\tau_b$ | $p_\tau$ |
|---|---|---|---|---|
| difficulty | -0.66 | 0.16 | -0.60 | 0.09 |
| size | -0.71 | 0.11 | -0.60 | 0.09 |
| similarity | -0.31 | 0.54 | -0.20 | 0.57 |

Table 2: Rank-order correlation coefficients (Spearman's $\rho$ and Kendall's $\tau_b$), along with associated $p$ values, computed using the results in Figure 10 and the properties in Figure 11. These coefficients suggest that the difficulty, size and visual similarity might help to explain the differential impact of visual attention in image recognition. But there is high uncertainty in these estimates. We cannot confidently exclude the possibility that these relations occurred by unrelated coincidence.

## 4.2 Difficulty-based sets

In Section 3.1 we defined the difficulty of a category set as the average error rate of a standard VGG16 on examples in the category set. We defined 25 difficulty-based category sets. These vary in difficulty but are of the same size and approximately the same visual similarity. In Figure 12 we see that higher category-set difficulty corresponds to a greater performance impact of visual attention (correlation reported in Table 3).

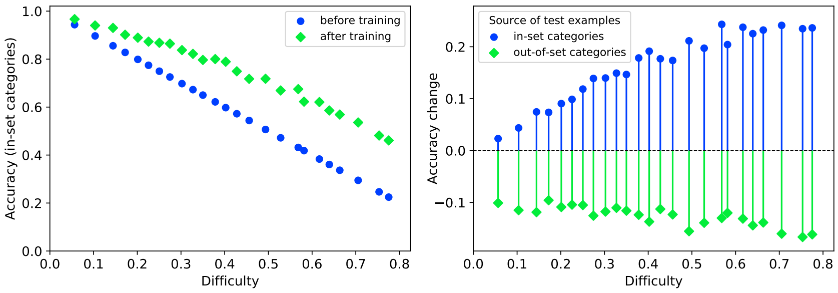


Figure 12: Classification accuracy of attention networks trained on 25 difficulty-based category sets. These sets vary in difficulty but are of the same size and approximately the same visual similarity. Left: absolute accuracy of the attention networks before and after training. Right: relative accuracy (with respect to a standard VGG16) of the attention networks after training.

## 4.3 Size-based sets

In Section 3.1 we defined 10 sized-based category sets. These range in size (number of categories) but are approximately fixed in difficulty and visual similarity. Our results suggest that visual attention's impact on classification accuracy is less for larger category sets (Figure 13; Table 3).

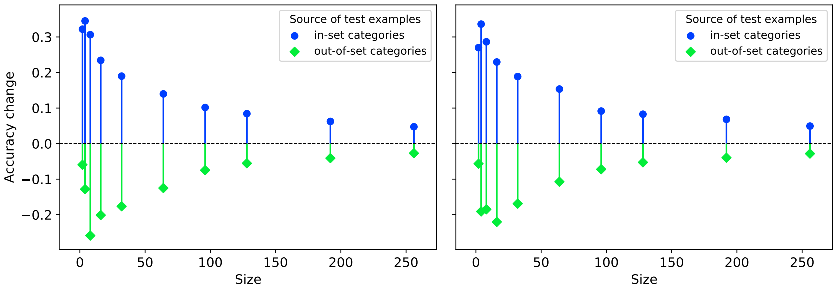


Figure 13: Classification accuracy (relative to a standard VGG16) of attention networks trained on 20 size-based category sets. These sets range in size (number of categories) but are approximately fixed in difficulty and visual similarity. Left: original experiment (10 sets). Right: repeat experiment with a different collection of category sets (10 sets).

## 4.4 Similarity-based sets

In Section 3.1 we defined the visual similarity of a category set as the mean cosine similarity of VGG16 representations of images in the category set. We defined 20 similarity-based category sets. These differ in visual similarity but are of the same size and approximately the same difficulty. The evidence presented in Figure 14 suggests that visual attention is decreasingly impactful as the visual similarity of a category set increases (correlation reported in Table 3).

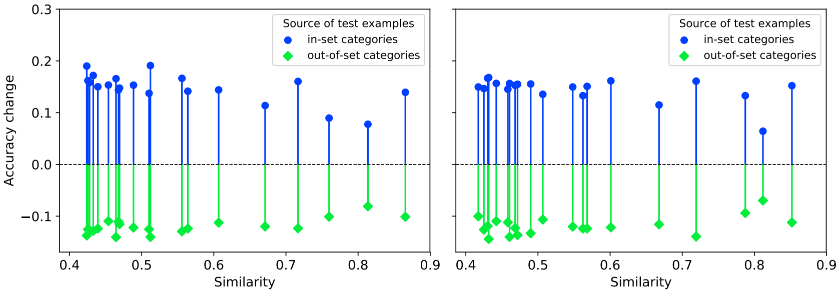


Figure 14: Classification accuracy (relative to a standard VGG16) of attention networks trained on 40 similarity-based category sets. These differ in visual similarity but are of the same size and approximately the same difficulty. Left: original experiment (20 sets). Right: repeat experiment with a different collection of category sets (20 sets).

Figure 15 shows least-squares linear regression applied to the in-set results presented in Figures 12-14. The results of repeat experiments were incorporated in this. We perform a base-2 logarithmic transformation of category-set size before fitting a linear curve.

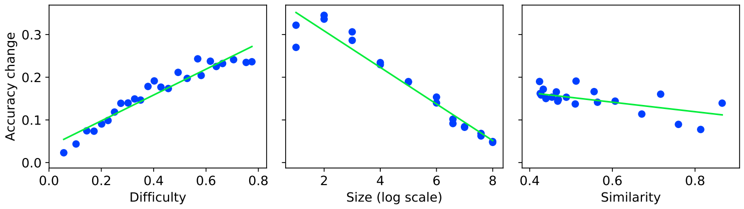


Figure 15: Least-squares linear regression applied to the changes in in-set accuracy shown in Figures 12-14. Data from repeat experiments is combined with that from original experiments. Size is transformed logarthmically (with base 2).

Table 3 presents the parameters of the least-squares lines shown in Figure 15. It also includes rank-order correlation coefficients, which can be compared with those in Table 2. We note stronger correlations and higher confidence in Table 3. Thus, the results in Figures 12-14 help us to better characterise when visual attention is useful.

| **Property** | $\rho$ | $\tau_b$ | $\beta_0$ | $\beta_1$ | $R^2$ |
|---|---|---|---|---|---|
| difficulty | -0.96 | -0.85 | 0.04 | 0.30 | 0.92 |
| size | -0.97 | -0.91 | 0.39 | -0.04 | 0.94 |
| similarity | -0.52 | -0.38 | 0.21 | -0.11 | 0.37 |

Table 3: Correlation statistics (Spearman's $\rho$; Kendall's $\tau_b$; $R^2$) and regression parameters (intercept, $\beta_0$; slope, $\beta_1$) for the in-set data shown in Figures 12-14. Data from repeat experiments is combined with that from original experiments. Size is transformed logarthmically (with base 2). For every coefficient, $p < 0.001$.

# 5 Discussion

Our empirical observations appear to show that the nature of a task has significant influence on the usefulness of attention. We suggest that two of our sets of results (from the difficulty and size experiments) have intuitive explanations, and one set (from the visual-similarity experiment) does not.

## 5.1 Difficulty-based sets

The more difficult a category set is (as judged by how frequently a standard VGG16 misclassifies in-set images), the greater the performance benefit of introducing attention. To understand why, we need to think about the behaviour of the neural networks we use, and also about the ways in which images vary. What is it about images from a given category that makes VGG16 classify them with high or low accuracy?

Training a neural network involves tradeoffs between weights (Sutton et al, 2006). We explain this with reference to the simple feedforward network shown in Figure 1. Let us assume the network is a three-way classifier. For a given input, $x$, it computes a probability distribution over three possible category labels, $y_1, y_2, y_3$. The activity, $h_2$, of the second hidden layer is a representation of $x$, and $h_{2,1}$ is a single **feature** (element of this vector).

Suppose that, for $x$ belonging to category $y_1$, $h_{2,1}$ is the single most useful feature for identifying $x$'s true label. But for inputs from other categories, $h_{2,1}$ provides no clue, and simply contributes random noise that drowns out useful information. Thus, there is a **weight tradeoff**: increasing the weights on $h_{2,1}$ improves accuracy for some examples and reduces accuracy for others. During training, the classifier tunes its weights so as to maximise average accuracy across the whole training set. Suppose that the network learns to place low weight on $h_{2,1}$, which we have said is the strongest predictor of examples from category $y_1$. As a result, if the true label of $x$ is $y_1$, the network classifies it with low accuracy.

By analogy, we can see why a convolutional network's accuracy can vary substantially across categories (Russakovsky et al, 2014). For a given category, $y_i$, there is some set of features that most strongly indicate that an image is from $y_i$. A network's accuracy on $y_i$ is low if emphasising $y_i$'s most indicative features is incompatible with good overall performance.

This helps to explain the difficulty results (greater benefit of attention for more difficult category sets; Figure 12). We have established that a category set's high difficulty (low VGG16 accuracy) is partly due to the suppression of features that would aid classification of images belonging to that category set. The task we study, in which attention weights are tuned on a small subset of ImageNet categories (most category sets contained fewer than 100), represents a **relaxation of the weight tradeoff**. With fewer categories to account for, weights can emphasise useful features that were suppressed before. This improves the network's discriminative power on images from the training category set. The more difficult the category set, the more pronounced this boost is.

With reference to human cognition, this explanation seems to make sense. Our experiments represent the cognitive task of learning to apply attention so as to be more discerning with respect to a restricted set of objects. This is analogous to the way an art critic develops an eye for assessing paintings (Koide et al, 2015). Our results suggest that the payoff of specialising is not equal for all subjects of expertise. This seems true in reality. Compared to a layperson, a radiologist might be many times more accurate at classifying medical images. In contrast, we have no trouble discerning bicycles from bananas, and more experience will not change this. In between these two examples there lies a spectrum of tasks offering **varying returns to expertise**. This is what we see in the difficulty experiments.

Difficulty is a broad measure with many contributors. For example, Russakovsky et al (2014) identified eight ways in which ImageNet examples can vary (eg, shape distinctiveness, real-world size). These are likely to affect difficulty. It can be argued that our result is especially compelling in light of this. Despite many possible complicating factors, we see a strong correlation between task difficulty and the performance boost of attention. In Section 6 we suggest teasing apart these factors to better understand what is going on. For the purpose of interpreting the current work, it should be considered that our results could capture idiosyncrasies of the dataset rather than general phenomena. But we have no reason to believe these are dominant.

More prosaic factors might underlie the difficulty results. For example, a **ceiling effect** might be at play. Suppose a category set's images are classified with high accuracy by VGG16. There is not much room for improvement in accuracy: even if incorporating attention is useful, accuracy has an upper limit of 100 percent. A ceiling effect would be identifiable by a concave regression curve (if we did not constrain it to be linear). Our data might hint at this, but rather ambiguously: the point cloud in the left-hand plot of Figure 15 appears to bend slightly in the middle. If a ceiling effect were the full explanation, this would probably be more pronounced.

## 5.2 Size-based sets

The bigger a category set is (the more categories it contains), the smaller the performance benefit of introducing attention. Again, the weight-tradeoff phenomenon (Section 5.1) is an important explanatory effect. If there are only two categories of images to classify, then an attention network can focus on the features that allow it to discern between those two categories. The more categories there are, the stronger the tradeoffs between features.

This appears to agree with our intuitions about human cognition. There seems to be a **breadth-depth tradeoff** in expertise: the broader our task, the more thinly we spread ourselves. Assuming all topics are of equal size and difficulty, it is easier to prepare for a two-topic exam than a ten-topic exam. Likewise, in the experiments we see that the performance boost produced by attention is greater for narrower tasks (category sets with fewer categories).

Another explanatory factor is noteworthy. Across all our experiments, the in-set accuracy of an attention network exceeds that of a standard VGG16. At the same time, out-of-set accuracy is lower for attention networks. To some extent, this can be interpreted as the attention network **learning the category imbalance** in its training set. That is, it works out that some categories do not appear in the training data. Thus it can improve classification accuracy by systematically biasing predictions away from those categories. In Section 6 we suggest a way of measuring this effect.

## 5.3 Similarity-based sets

The more visually similar a category set is (as judged by how similarly VGG16 represents images from the set's categories), the smaller the performance benefit of introducing attention. In our view, this seems odd. The discriminative ability of a neural network is determined by how it extracts and manipulates representations of an input (Section 5.1). In our implementation, attention allows a network to reweight features of a representation so that it can better discern between categories. Suppose a category set has high visual similarity. By our own definition, this means there is relatively high consistency in the way VGG16 represents images from the category set. In other words, **in a high-similarity category set, there is low variation in which features are most important**. It should, therefore, be straightforward for the attention network to learn how to optimally weight its features.

If visually similar categories have similar mean representations, then in vector space the representations for dog categories should cluster, as should the representations for cat categories, and so on. One way of checking this is to inspect the clusters computed using t-SNE, a data-visualisation technique (van der Maaten and Hinton, 2008; Karpathy, 2014; Wattenberg et al, 2016). We see in Figure 16 that members of the same cluster are sometimes visually similar (according to human judgement) and sometimes quite dissimilar. This suggests that, in some sense, the category representations do not capture visual appearance in the way we expect. Related to this, Geirhos et al (2019) showed that ImageNet-trained convolutional networks prefer to use texture, not shape, to recognise images. This has implications for the way they represent images.

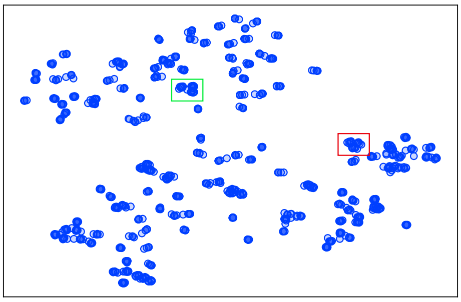


Figure 16: t-SNE visualisation of the representations used to compute the visual similarity of categories. Each point represents a 4096-dimensional vector projected into two dimensions by t-SNE. Members of the same cluster are sometimes visually similar (according to human judgement) and sometimes quite dissimilar. The green box highlights an example of the former case; the red box, the latter. Green box: (killer whale, dugong, sea lion, Chihuahua, Japanese spaniel, Maltese dog, Pekinese, Shih-Tzu, Blenheim spaniel, papillon, toy terrier, Rhodesian ridgeback, Afghan hound, basset, beagle bloodhound, bluetick, black-and-tan coonhound, Walker hound, English foxhound, redbone, borzoi, Irish wolfhound). Red box: (barracouta, eel, coho, rock beauty, anemone fish, sturgeon, gar, balloon, ballpoint, Band-Aid, banjo, bannister, barbell, barber chair, bolo tie, bonnet, bookcase, bookshop, bottlecap, bow, bow tie).

We note that our approach has precedent, both in its use of final-layer representations (Socher et al, 2014; Birodkar et al, 2019), and in the averaging of representations within a category (Karpathy et al, 2014; Lindsay and Miller, 2018). It also has justification with respect to neuroscience (Kriegeskorte, 2015). As we go deeper into the layers of a convolutional neural network, the representations become better correlated with the representations in the inferior temporal cortex of primates (Khaligh-Razavi and Kriegeskorte, 2014). This region of the brain is known to produce representations with strong categorical divisions (Kriegeskorte et al, 2008). Thus, we expect the penultimate-layer representation of VGG16 to have highly category-specific features, making them useful for comparison. But it appears that our approach does not capture the effect that our intuition implies. In Section 6 we suggest an alternative way of measuring visual similarity.

# 6 Future work

We propose natural extensions of our work that seem both promising and tractable. Some are motivated by the pursuit of greater breadth of understanding; others by a desire for more detail.

### Assessing the generalisation of our results

With VGG16 and ImageNet, we found statistically significant ($p < 0.001$) variation in the usefulness of attention when we vary the properties of an image-recognition task. Using this combination of convolutional architecture and dataset is accepted to be a state-of-the-art approach to modelling human visual perception. So our findings are noteworthy in their own right. Confirming this effect in other experimental setups (with different neural networks and image datasets) would strengthen our confidence in the existence of general phenomena.

### Allowing covariance between category-set properties

We designed our experiments to minimise covariance in the category-set properties we varied (difficulty, size, visual similarity). That is, while varying one property, we aimed to keep the others constant. This was in an effort to isolate the effect of each property on the usefulness of attention. Now that we have established the effects of these properties independently, we could consider the more complex case of when they vary together.

### Automatically selecting semantic category sets

In our experiments, the semantic category sets were six manually-selected groups of conceptually similar ImageNet categories. These allowed us to establish that visual attention seems to be more performance-enhancing for some semantic category sets than others. If desired, we could scale up the semantic-set experiment by automating the process of grouping ImageNet categories into semantic sets. With more results, we could more fully characterise how the usefulness of attention differs between conceptually distinct tasks.

We suggest two possible ways of doing this. In the first approach, we would take advantage of the hierarchical organisation of ImageNet categories. This tree structure is encoded as an Extensible Markup Language (XML) file, and is available from the ImageNet website (`image-net.org`). By parsing the tree, we would automatically select groups of categories with common conceptual ancestors. In the second approach, we would measure the conceptual similarity of categories using a word-representation technique like word2vec (Mikolov et al, 2013) or GloVe (Pennington et al, 2014). We would compute a vector representation of each category label. Then we would quantify the semantic similarity of two categories by comparing the representations of their labels (eg, with cosine similarity). In this approach, semantic similarity would be a continuous quantity. Thus, it would allow us to characterise effects more finely than the first approach would.

### Decomposing difficulty

We noted in Section 5.1 that difficulty is determined by a number of factors. If we can quantitatively measure, isolate and control the constituent components of difficulty, we could better understand the mechanisms at play. Visual clutter is an example of such a factor. It seems to be a key reason why we need to apply visual attention (Walther et al, 2005; Mnih et al, 2014). It is also readily quantified: Alexe et al (2012) demonstrated a category-agnostic object detector that evaluates the probability that a given window in an image contains an object (rather than background). If we desire a more fine-grained understanding of the difficulty results, this would be a good place to start.

### Quantifying the effect of category-imbalance-learning

In Section 5.2 we suggested that a category-imbalance effect probably influenced the size-experiment results to some extent. That is, attention networks achieved accuracy improvements simply by noticing the absence of some categories in the training set. We propose a method for quantitatively estimating the size of this effect for a given category set, $\mathcal{C}$. First, we would compute the predictions of VGG16, namely $p(y|x)$ for all $x$. Second, for each prediction, $p(y|x)$, we would remove weight from all $y_i$ corresponding to categories not in $\mathcal{C}$. We would redistribute this weight uniformly across all $y_i$ corresponding to categories in $\mathcal{C}$. The result would be a set of modified predictions. Third, we would compute the accuracy of the modified predictions. When compared to the accuracy of the original predictions, this would estimate the size of the category-imbalance effect. In other words, it would tell us how much of a performance boost can be achieved with a trivial adjustment.

### Using an alternative measure of visual similarity

In our experiments, we defined the visual similarity of two ImageNet categories as the cosine similarity of their mean final-layer VGG16 representations (Section 3.1). Perhaps the most promising alternative to this is the perceptual similarity measure proposed by Zhang et al (2018). Like ours, their method uses representations extracted by VGG16. But, unlike our approach, theirs uses representations from early layers in the network. For each image, activations at multiple layers are computed. When comparing two images, a distance function compares the two corresponding sets of activations. This distance function includes a learnt linear reweighting of features. The method produces human-like judgements of visual similarity.

We were aware of this approach when designing our experiments. But we chose not to use it. It ran too slowly in our preliminary tests to be scalable to the problem in which we were interested. With some work, it might be possible to make this approach usable on a reasonable timescale.

# 7 Conclusion

When is visual attention useful? We set out to answer this by studying the cognitive task of learning to apply attention to a restricted set of image categories. We show that incorporating a trainable explicit attention mechanism into an otherwise fixed deep convolutional neural network enables enhanced performance on this task. Manipulating the nature of the task—controlling which categories the network is exposed to during training—reliably produces variation in the performance impact of visual attention. According to our results, attention becomes more useful with increasing task difficulty, less useful with increasing task size, and less useful with increasing visual similarity within a task.

These findings have implications for cognitive science. Deep convolutional networks are state-of-the-art models of the human visual system. Thus the behavioural patterns that they display can be useful for two purposes. First, they can help explain pre-existing empirical data. Second, they can inform new hypotheses about human cognition, which can then be tested on human subjects. At the same time, our work contributes to the machine-learning community's understanding of visual attention. We systematically tested an attention mechanism in a series of easily interpreted experiments. Thus we provide information that can inform neural-network designers deciding whether to incorporate attention for visual tasks. For example, the results suggest that for every doubling of task size (the number of categories on which we want attention to specialise; we assume average difficulty and visual similarity) visual attention's expected impact on accuracy is reduced by 0.04 (4 percentage points if measuring on a 0-100 scale). Our contribution, therefore, pushes forward both basic and applied science.

# 8 References

Alexe, Deselaers, Ferrari (2012). Measuring the objectness of image windows. IEEE Transactions on Pattern Analysis and Machine Intelligence.

Ba, Mnih, Kavukcuoglu (2015). Multiple object recognition with visual attention. International Conference on Learning Representations.

Birodkar, Mobahi, Bengio (2019). Semantic redundancies in image-classification datasets: the 10% you don't need. arXiv.

Bostock (2018). ImageNet hierarchy. `observablehq.com/@mbostock/imagenet-hierarchy`.

Carrasco (2011). Visual attention: the past 25 years. Vision Research.

Carrasco, Barbot (2019). Spatial attention alters visual appearance. Current Opinions in Psychology.

Chen, Zhang, Xiao, Nie, Shao, Liu, Chua (2017). SCA-CNN: spatial and channel-wise attention in convolutional networks for image captioning. Conference on Computer Vision and Pattern Recognition.

Cybenko (1989). Approximations by superpositions of sigmoidal functions. Mathematics of Control, Signals, and Systems.

Deng, Dong, Socher, Li, Li, Fei-Fei (2009). ImageNet: a large-scale hierarchical image database. Conference on Computer Vision and Pattern Recognition.

Denil, Bazzani, Larochelle, de Freitas (2012). Learning where to attend with deep architectures for image tracking. Neural Computation.

Eriksen, St James (1986). Visual attention within and around the field of focal attention: a zoom lens model. Perception and Psychophysics.

Geirhos, Rubisch, Michaelis, Bethge, Wichmann, Brendel (2019). ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. International Conference on Learning Representations.

Gilpin, Bau, Yuan, Bajwa, Specter, Kagal (2018). Explaining explanations: an overview of interpretability in machine learning. International Conference on Data Science and Advanced Analytics.

Hassabis, Kumaran, Summerfield, Botvinick (2017). Neuroscience-inspired artificial intelligence. Neuron.

Hebb (1949). The Organization of Behavior.

Hill (2017). Deep learning for emotion recognition in cartoons. `hako.github.io/dissertation`.

Hinton, Osindero, Teh (2006). A fast learning algorithm for deep belief nets. Neural Computation.

Hornik, Stinchcombe, White (1989). Multilayer feedforward networks are universal approximators. Neural Networks.

Jaderberg, Simonyan, Zisserman, Kavukcuoglu (2015). Spatial transformer networks. Neural Information Processing Systems.

Jarrett, Kavukcuoglu, Ranzato, LeCun (2009). What is the best multi-stage architecture for object recognition? International Conference on Computer Vision.

Karpathy (2014). t-SNE visualization of CNN codes. `cs.stanford.edu/people/karpathy/cnnembed`.

Karpathy, Joulin, Fei-Fei (2014). Deep fragment embeddings for bidirectional image sentence mapping. Neural Information Processing Systems.

Karpathy (2019). A recipe for training neural networks. `karpathy.github.io/2019/04/25/recipe`.

Khaligh-Razavi, Kriegeskorte (2014). Deep supervised, but not unsupervised, models may explain IT cortical representation. PLOS Computational Biology.

Kingma, Ba (2015). Adam: a method for stochastic optimization. International Conference on Learning Representations.

Koide, Kubo, Nishida, Shibata, Ikeda (2015). Art expertise reduces influence of visual salience on fixation in viewing abstract paintings. PLOS One.

Kriegeskorte, Mur, Ruff, Kiani, Bodurka, Esteky, Tanaka, Bandettini (2008). Matching categorical object representations in inferior temporal cortex of man and monkey. Neuron.

Kriegeskorte (2015). Deep neural networks: a new framework for modeling biological vision and brain information processing. Annual Review of Vision Science.

Kriegeskorte, Douglas (2018). Cognitive computational neuroscience. Nature Neuroscience.

Krizhevsky, Sutskever, Hinton (2012). ImageNet classification with deep convolutional neural networks. Neural Information Processing Systems.

Kruschke (1992). ALCOVE: an examplar-based connectionist model of category learning. Psychological Review.

Larochelle, Hinton (2010). Learning to combine foveal glimpses with a third-order Boltzmann machine. Neural Information Processing Systems.

Lindsay (2015). Feature-based attention in convolutional neural networks. arXiv.

Liu, Hou (2011). Global feature-based attention to orientation. Journal of Vision.

Marr, Poggio (1977). From understanding computation to understanding neural circuitry. Neuronal Mechanisms in Visual Perception.

Marr (1982). Vision: A Computational Investigation into the Human Representation and Processing of Visual Information.

McCulloch, Pitts (1943). A logical calculus of ideas immanent in nervous activity. Bulletin of Mathematical Biophysics.

Mikolov, Chen, Corrado, Dean (2013). Efficient estimation of word representations in vector space. International Conference on Learning Representations.

Miller (1995). WordNet: a lexical database for English. Communications of the ACM.

Mnih, Hees, Graves, Kavukcuoglu (2014). Recurrent models of visual attention. Neural Information Processing Systems.

Nobre (2001). Orienting attention to instants in time. Neuropsychologia.

Olson (2001). Object-based vision and attention in primates. Current Opinions in Neurobiology.

Pennington, Socher, Manning (2014). GloVe: global vectors for word representation. Empirical Methods in Natural Language Processing.

Ranzato (2014). On learning where to look. arXiv.

Russakovsky, Deng, Su, Krause, Satheesh, Ma, Huang, Karpathy, Khosla, Bernstein, Berg, Fei-Fei (2014). ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision.

Saxe, Koh, Chen, Bhand, Suresh, Ng (2011). On random weights and unsupervised feature learning. International Conference on Machine Learning.

Schrimpf, Kubilius, Hong, Majaj, Rajalingham, Issa, Kar, Bashivan, Prescott-Roy, Schmidt, Yamins, DiCarlo (2018). Brain-Score: which artificial neural network for object recognition is most brain-like? bioRxiv.

Simonyan, Zisserman (2015). Very deep convolutional networks for large-scale image recognition. International Conference on Learning Representations.

Socher, Karpathy, Le, Manning, Ng (2014). Grounded compositional semantics for finding and describing images with sentences. Transactions of the Association for Computational Linguistics.

Stollenga, Masci, Gomez, Schmidhuber (2014). Deep networks with internal selective attention through feedback connections. Neural Information Processing Systems.

Sutskever, Vinyals, Le (2014). Sequence to sequence learning with neural networks. Neural Information Processing Systems.

Sutton, Sindelar, McCallum (2006). Reducing weight undertraining in structured discriminative learning. Human Language Technology Conference of the NAACL.

van den Oord, Dieleman, Zen, Simonyan, Vinyals, Graves, Kalchbrenner, Senior, Kavukcuoglu (2016). WaveNet: a generative model for raw audio. arXiv.

van der Maaten, Hinton (2008). Visualizing data using t-SNE. Journal of Machine Learning Research.

Velickovic (2018). TikZ. `github.com/PetarV-/TikZ`.

Walther, Rutishauser, Koch, Perona (2005). Selective visual attention enables learning and recognition of multiple objects in cluttered scenes. Computer Vision and Image Understanding.

Wattenberg, Viegas, Johnson (2016). How to use t-SNE effectively. Distill.

Xu, Ba, Kiros, Courville, Salakhutdinov, Zemel, Bengio (2015). Show, attend and tell: neural image caption generation with visual attention. arXiv.

Yamins, DiCarlo (2016). Using goal-driven deep learning models to understand sensory cortex. Nature Neuroscience.

Zhang, Isola, Efros, Shechtman, Wang (2018). The unreasonable effectiveness of deep features as a perceptual metric. Conference on Computer Vision and Pattern Recognition.

# A Algorithms

**Algorithm 1:** Sampling difficulty-based category sets (Python 3)

```
thresholds = [0.35, 0.4, 0.45, 0.5, 0.55]
intervals_covered = False
best_score = infinity
best_sets = None

for i in range(10000):
    candidate_sets = [sample_difficulty_set(t) for t in thresholds]
    intervals_covered = assess_difficulty_coverage(candidate_sets)
    candidate_score = max_similarity_deviation(candidate_sets)

    if intervals_covered is True and candidate_score < best_score:
        best_sets = candidate_sets
        best_score = candidate_score

if best_sets is not None:
    return best_sets
```

Algorithm 1 uses the following functions.

`sample_difficulty_set`($t$) randomly samples, without replacement, 50 categories from the subset satisfying accuracy$(c_i) \leq t$.

`assess_difficulty_coverage`$(\mathcal{A}_{21}, \ldots, \mathcal{A}_{25})$ is a Boolean function, returning True if each of the difficulty intervals

$$\{[d_0, d_0 + 0.05] \text{ for } d_0 \in [0.20, 0.35] \text{ with step } 0.05\} \tag{19}$$

contains at least one category set.

Using the visual similarity definitions from Section 3.1,

$$\begin{aligned} &\texttt{max\_similarity\_deviation}(\mathcal{A}_{21}, \ldots, \mathcal{A}_{25}) = \\ &\qquad \max\left(\left\{\left|\frac{\text{similarity}(\mathcal{A}_i) - \mu_s}{\sigma_s}\right| \text{ for } i \in [21, 25]\right\}\right) \end{aligned} \tag{20}$$

where, with $C = 1000$ denoting the number of ImageNet categories, and $\mathbb{1}$ denoting an indicator function,

$$\mu_s = \frac{1}{C^2 - C} \sum_i^C \sum_j^C s_{i,j} \mathbb{1}(i \neq j) \tag{21}$$

$$\sigma_s = \sqrt{\frac{1}{C^2 - C} \sum_i^C \sum_j^C (s_{i,j} - \mu_s)^2 \mathbb{1}(i \neq j)} \tag{22}$$

**Algorithm 2:** Sampling size-based category sets (Python 3)

```
set_sizes = [2, 4, 8, 16, 32, 64, 96, 128, 192, 256]
best_score = infinity
best_sets = None

for i in range(10000):
    candidate_sets = [sample_size_set(s) for s in set_sizes]
    candidate_score = max_difficulty_similarity_deviation(candidate_sets)

    if candidate_score < best_score:
        best_sets = candidate_sets
        best_score = candidate_score

if best_sets is not None:
    return best_sets
```

In Algorithm 2, difficulty is defined as in Section 3.1, and the following definitions are used.

$$\begin{aligned}&\texttt{max\_difficulty\_similarity\_deviation}(\mathcal{B}_1,\ldots,\mathcal{B}_{10}) = \\ &\max\left(\left\{\left|\frac{\text{difficulty}(\mathcal{B}_i)-\mu_d}{\sigma_d}\right| \text{ for } i \in [1,10]\right\} \cup \right. \\ &\left.\left\{\left|\frac{\text{similarity}(\mathcal{B}_i)-\mu_s}{\sigma_s}\right| \text{ for } i \in [1,10]\right\}\right)\end{aligned} \tag{23}$$

$$\mu_d = \frac{1}{C}\sum_i^C (1-\text{accuracy}(c_i)) \tag{24}$$

$$\sigma_d = \sqrt{\frac{1}{C}\sum_i^C ((1-\text{accuracy}(c_i))-\mu_d)^2} \tag{25}$$

where $C = 1000$ is the number of ImageNet categories, and accuracy$(c_i)$ denotes the mean accuracy of the base network on category $c_i$. The function `sample_size_set`$(s)$ randomly samples, without replacement, $s$ ImageNet categories.

**Algorithm 3:** Sampling similarity-based category sets (Python 3)

```
sampling_window_ends = [50, 366, 682, 999]
best_score = infinity
best_sets = None

for i in range(10000):
    seed_categories = sample_seeds()

    for seed in seed_categories:
        sorted_categories = sort_categories_by_similarity(seed)
        candidate_sets = [
            sample_similarity_set(sorted_categories, k)
            for k in sampling_window_ends]
        intervals_covered = assess_similarity_coverage(candidate_sets)
        candidate_score = max_difficulty_deviation(candidate_sets)

    if intervals_covered is True and candidate_score < best_score:
        best_sets = candidate_sets
        best_score = candidate_score

if best_sets is not None:
    return best_sets
```

Algorithm 3 uses the following functions.

`sample_seeds()` randomly samples, without replacement, 5 ImageNet categories.

`sort_categories_by_similarity(seed)` returns $c_1, \ldots, c_{999}$, a list of the ImageNet categories (excluding the seed) ordered by decreasing similarity to the seed category.

`sample_similarity_set(`$[c_1, \ldots, c_{999}], k$`)` samples, without replacement, 50 categories from the first $k$ elements of $[c_1, \ldots, c_{999}]$. This can be viewed as a sampled $k$-nearest-neighbour procedure.

`assess_similarity_coverage(`$\mathcal{E}_1, \ldots, \mathcal{E}_{20}$`)` is a Boolean function, returning True if each of similarity intervals

$$\{[s_0, s_0 + 0.05] \text{ for } s_0 \in [0.10, 0.55] \text{ with step } 0.05\} \tag{26}$$

contains at least one category set.

Using the definitions from Section 3.1,

$$\begin{aligned}\texttt{max\_difficulty\_deviation}(\mathcal{E}_1, \ldots, \mathcal{E}_{20}) = \\ \max\left(\left\{\left|\frac{\text{difficulty}(\mathcal{E}_i) - \mu_d}{\sigma_d}\right| \text{ for } i \in [1, 20]\right\}\right)\end{aligned} \tag{27}$$

# B Loss curves

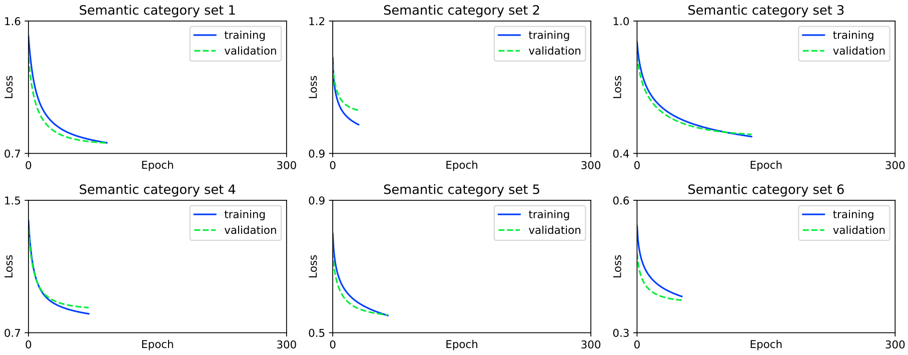


Figure 17: In-training loss curves for the attention networks referenced in Figure 10 (semantic category sets).

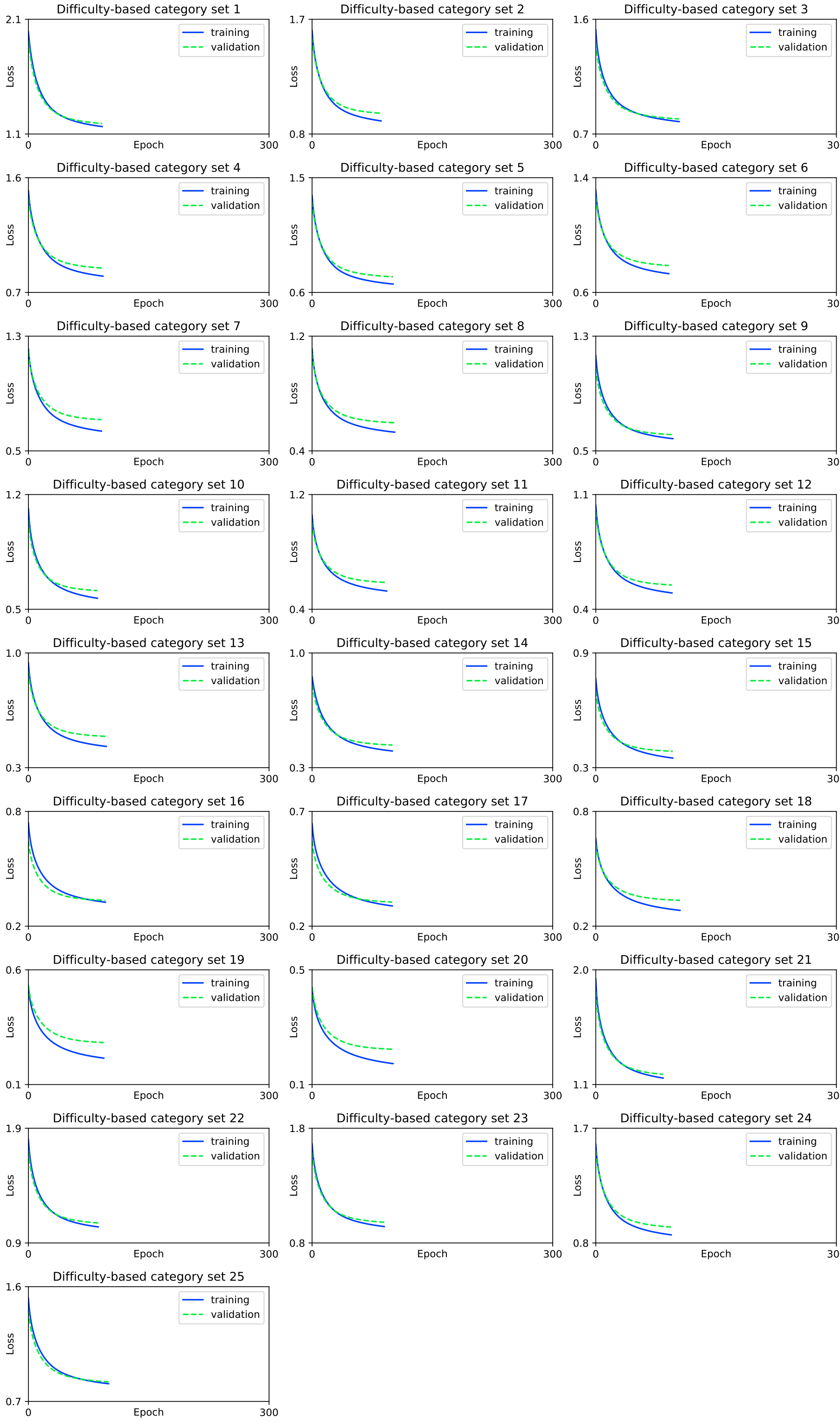


Figure 18: In-training loss curves for the attention networks referenced in Figure 12 (difficulty-based category sets).

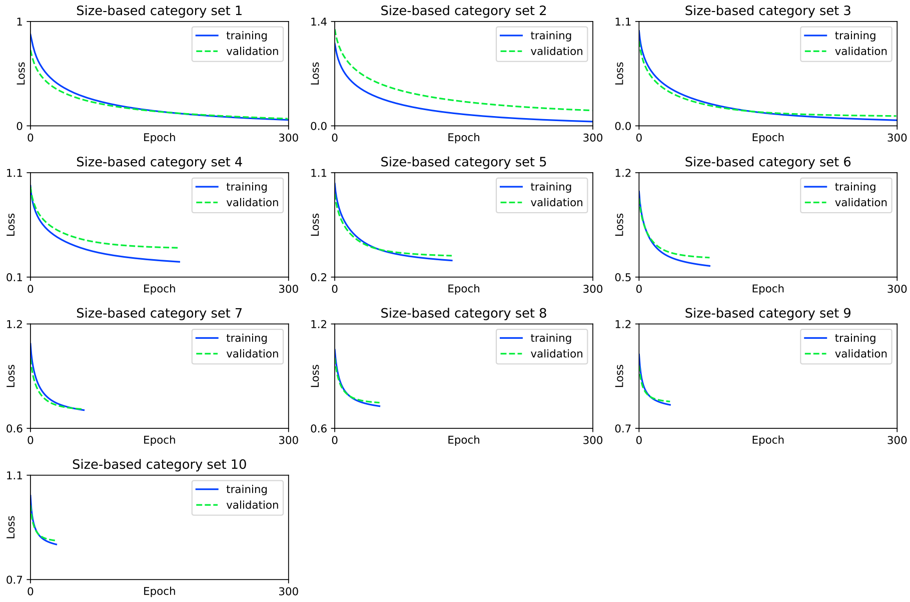


Figure 19: In-training loss curves for the attention networks referenced in the left-hand plot of Figure 13 (size-based category sets; original experiment).

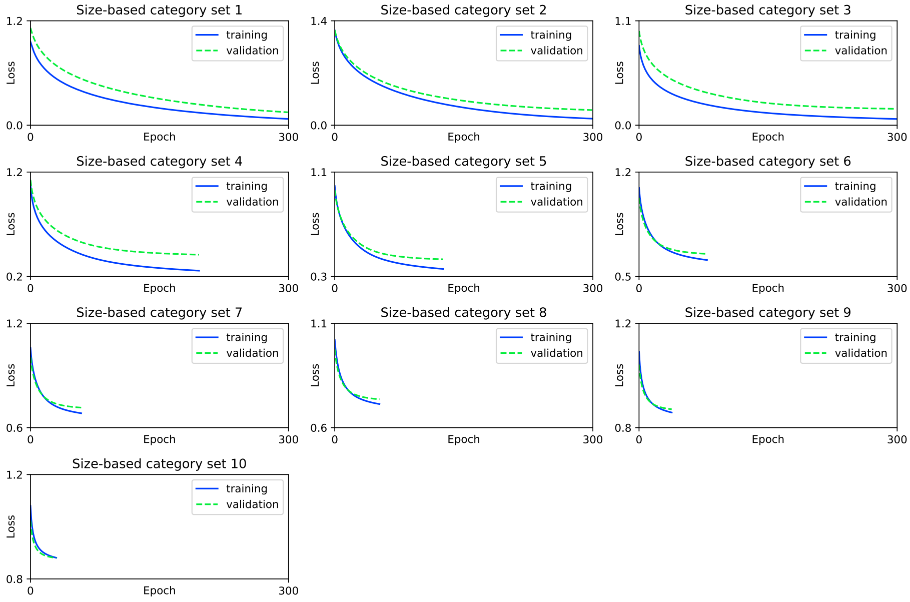


Figure 20: In-training loss curves for the attention networks referenced in the right-hand plot of Figure 13 (size-based category sets; repeat experiment).

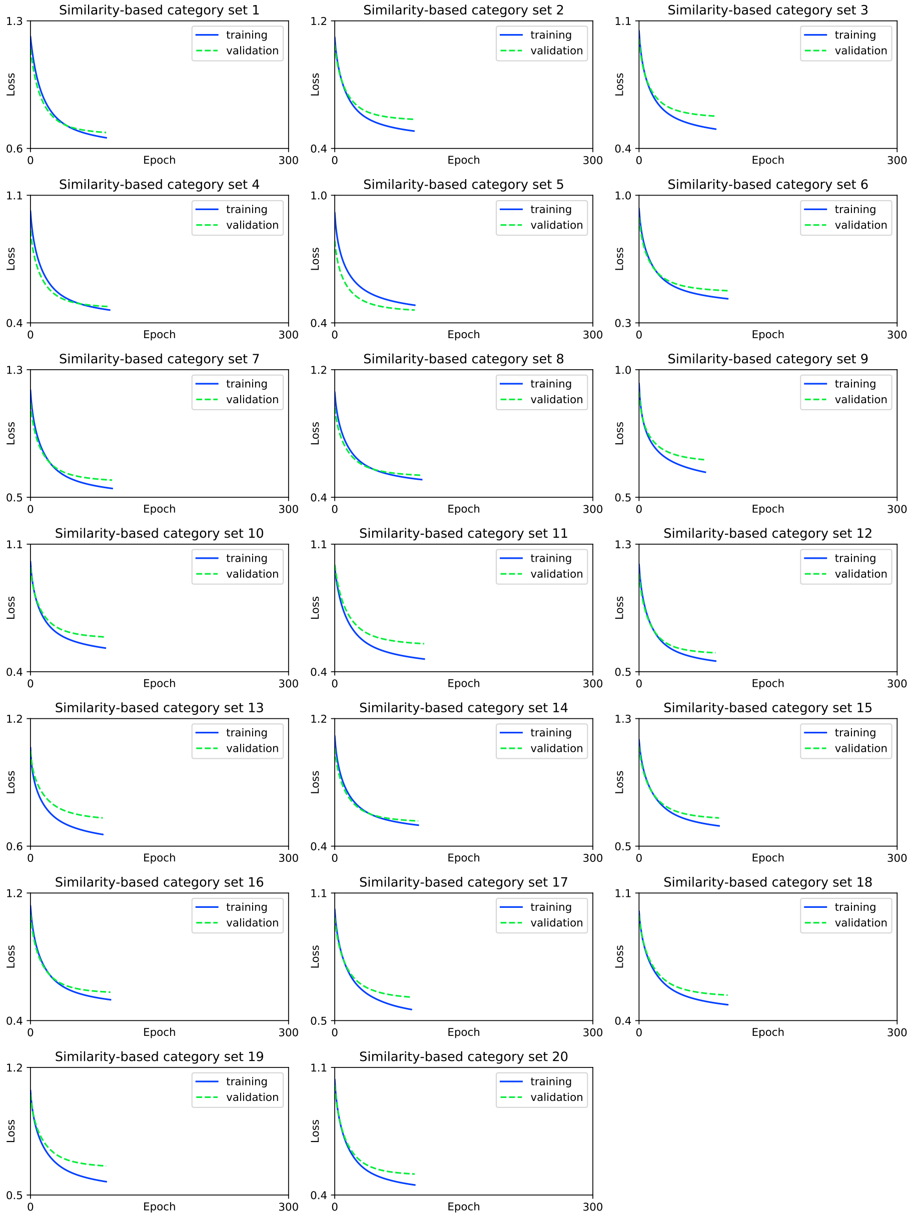


Figure 21: In-training loss curves for the attention networks referenced in the left-hand plot of Figure 14 (similarity-based category sets; original experiment).

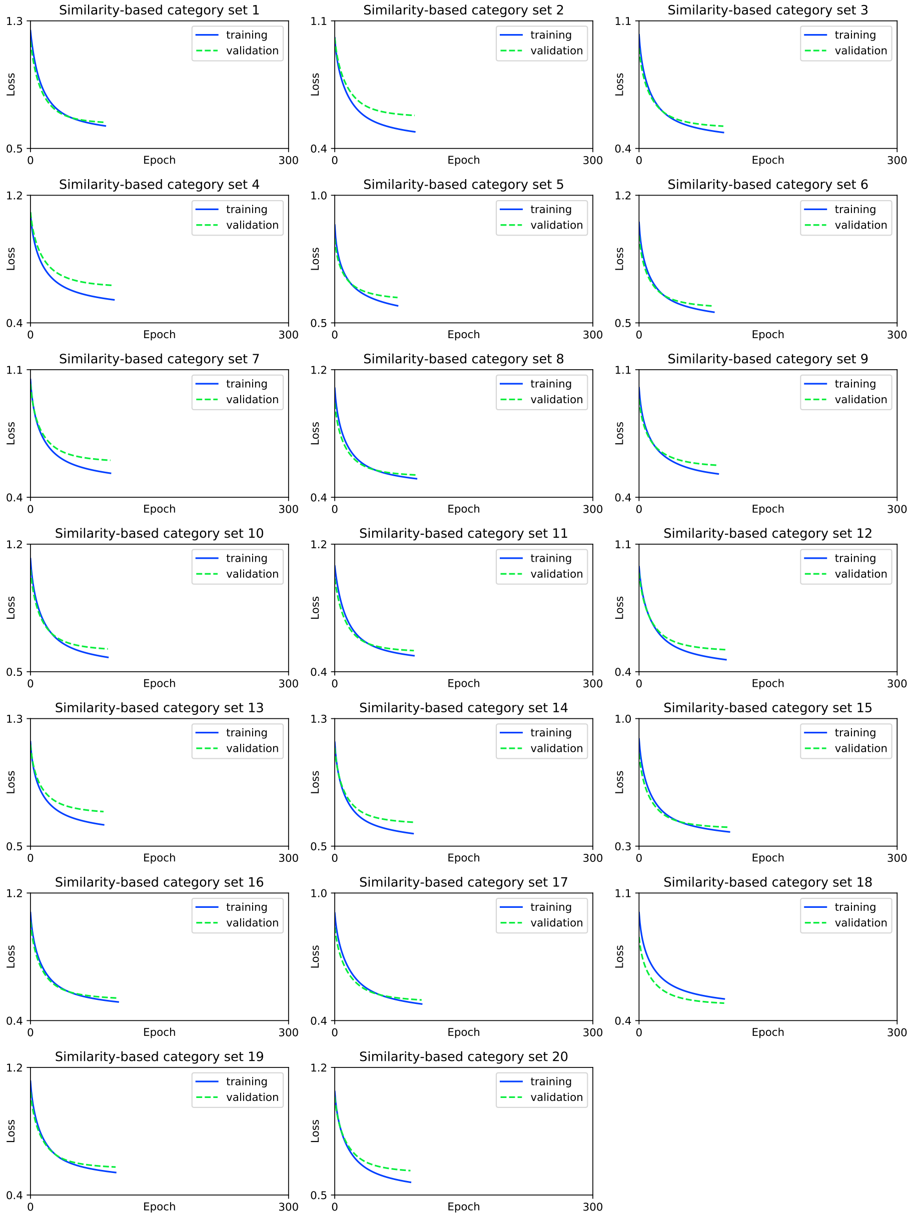


Figure 22: In-training loss curves for the attention networks referenced in the right-hand plot of Figure 14 (similarity-based category sets; repeat experiment).