Combining Convolutional Neural Networks and Cognitive Models to Predict Novel Object Recognition in Humans

Jeffrey Annis, Isabel Gauthier, & Thomas J. Palmeri

Vanderbilt University

in press, *Journal of Experimental Psychology: Learning, Memory, and Cognition*

Correspondence should be addressed to Jeffrey Annis, 111 21st Ave S., 301 Wilson Hall, Nashville, TN 37240. E-mail: jeff.annis@vanderbilt.edu

The data and code used in this work can be found at https://osf.io/dsqct/.

Abstract

Object representations from Convolutional Neural Network (CNN) models of computer vision (CNN; LeCun, Bengio, & Hinton, 2015) were used to drive a cognitive model of decision making, the Linear Ballistic Accumulator (LBA) model (Brown & Heathcote, 2008), to predict errors and response times (RTs) in a novel object recognition task in humans. CNNs have become very successful at visual tasks like classifying objects in real-world images (e.g., He, Zhang, Ren, & Sun, 2015; Krizhevsky, Sutskever, & Hinton, 2012). We asked whether object representations learned by CNNs previously trained on a large corpus of natural images could be used to predict performance recognizing novel objects the network has never been trained on; we used novel Greebles, Ziggerins, and Sheinbugs that have been used in a number of previous object recognition studies. We specifically investigated whether a model combining high-level CNN representations of these novel objects could be used to drive an LBA model of decision making to account for errors and RTs in a same-different matching task (from Richler et al., 2019). Combining linearly-transformed CNN object representations with the LBA provided reasonable accounts of performance not only on average, but at the individual-subject level and the item level as well. We frame the findings in the context of growing interest in using CNN models to understand visual object representations and the promise of using CNN representations to extend cognitive models to explain more complex aspects of human behavior.

We investigated whether object representations formed by a *Convolutional Neural Network* (CNN) model of computer vision (CNN; LeCun et al., 2015) when combined with a cognitive model of decision making, the *Linear Ballistic Accumulator* (LBA) model (Brown & Heathcote, 2008), could predict behavioral performance in visual recognition. We tested this combined model on a particularly rich data set (Richler et al., 2019) that includes a large number of human subjects who performed a within-category same-different perceptual matching task with novel objects (*Greebles*, *Ziggerins*, *Sheinbugs*, Figure 2). We tested this combined model's ability to predict errors and distributions of response times (RTs) at both the subject and item level, where subject-level variability is captured by differences in cognitive model parameters and item-level variability is captured by differences in CNN object representations. Our work demonstrates the viability of using CNN models as a perceptual front end to cognitive models, providing a potential avenue for extending cognitive models to more complex domains.

In computer vision, there has been an explosion of interest in deep learning models (Hinton, Osindero, & Teh, 2006; LeCun et al., 2015), especially CNNs. CNNs are able to reach near human levels of performance at certain visual tasks, such as classification of objects in real-world images (He et al., 2015; e.g., Krizhevsky et al., 2012; Simonyan & Zisserman, 2014a). Much of their success has been attributed to their ability to learn a hierarchy of useful representations from a large corpus of complex training images (Bengio, Courville, & Vincent, 2013).

CNNs have also generated significant interest in vision researchers who have shown, for example, that certain CNN representations can predict object representations in human and monkey visual cortex (Khaligh-Razavi & Kriegeskorte, 2014; Yamins et al., 2014), and that CNN representations can predict human behavior, such as ratings of object similarity (Jozwik,

Kriegeskorte, Storrs, & Mur, 2017; Kubilius, Bracci, & Op de Beeck, 2016; Peterson, Abbott, & Griffiths, 2016), object memorability (Dubey, Peterson, Khosla, Yang, & Ghanem, 2015), and category typicality (Lake, Zaremba, Fergus, & Gureckis, 2015).

Although aspects of CNN architectures have been inspired by the primate visual system, namely having neural-like units in a deep hierarchy implementing convolutional receptive field operations (e.g., Kriegeskorte, 2015), CNNs are computer vision model*s*, not models of primate vision. CNNs were developed with the goal of achieving high levels of accuracy on computer vision benchmarks[1], not predicting cognitive processes or patterns of human (or non-human primate) errors and RTs.

Cognitive models, by contrast, are developed to predict, explain, and understand the mechanisms underlying human behavior. Understanding of visual cognition, the ability to recognize, remember, and categorize visual images, has been significantly enhanced by the development of formal cognitive models. Such models have been used to make quantitative predictions of errors and RTs in domains such as visual recognition memory (Annis & Palmeri, 2019) and categorization (Nosofsky & Palmeri, 1997, 2015) and capture individual differences in visual performance (Bartlema, Lee, Wetzels, & Vanpaemel, 2014; Ratcliff & Childers, 2015; Shen & Palmeri, 2016).

---

[1] The *ImageNet Large Scale Visual Recognition Challenge* (Russakovsky et al., 2015) is one such benchmark many computer vision models are evaluated against, including the CNN models we used in this project. Given a large set of training and validation images used to train and tune a model, it tests the image classification performance of a model, with the winning model the one with the highest level of classification accuracy. Models with a deep convolutional architecture first won this competition in 2012 (Krizhevsky et al., 2012), which launched the current excitement and heavy investment in CNNs for computer vision, machine learning, and artificial intelligence.

Compared to computer vision models, cognitive models are relatively abstract in nature. When modeling performance in a visual cognition task, the input to a cognitive model is usually *not* the images of objects presented to subjects in an experiment (but see Dailey & Cottrell, 1999; Palmeri & Cottrell, 2009; Riesenhuber & Poggio, 1999; Ross, Deroche, & Palmeri, 2014). Rather, many models assume a more abstract representation of an object. This abstract representation could be a vector of features based on physical measurements, multidimensional scaling (MDS; e.g., Hout, Papesh, & Goldinger, 2013), or a statistical distribution thought to capture the statistics of variation in object representations (e.g., see Mack & Palmeri, in press). In the case of MDS, derived from pairwise similarity ratings of the full set of objects, a geometric representation of objects as points in a multidimensional psychological space is generated, with distance between points inversely related to their similarity. This representation can then be used to create evidence in favor of a particular response (e.g., Nosofsky, 1986; Nosofsky & Palmeri, 1997, 2015). Unfortunately, methods like MDS typically do not scale well with the number and complexity of images[2]. Furthermore, MDS does not model representation learning, nor the psychophysical transformation.

In a sense, many cognitive models can be thought of as starting in the middle of visual processing, leaving out the front-end perceptual processing that transforms the raw visual input into a representation for a given task. We ask whether a CNN model can provide an effective model of the front-end perceptual processing that, when combined with a cognitive model, can predict both subject-level and item-level variability in errors and RTs.

---

[2] Recent work has begun to explore the viability of large stimulus sets using MDS (Nosofsky, Sanders, & McDaniel, 2018; Nosofsky, Sanders, Meagher, & Douglas, 2017), using deep learning models to extrapolate beyond what can be derived from pairwise similarity ratings (Sanders, 2018; Sanders & Nosofsky, 2018).

Figure 1 illustrates our approach. The left and middle panels show the general architecture of a cognitive model and CNN, respectively. The right panel shows how we combine a CNN representation of objects with a cognitive model of performance. We exposed the top-level representation of a pre-trained CNN (at the layer prior to the classification layer) and used its representation of an object as the input to the cognitive model to drive a decision process (in this case the LBA). We evaluate model variants using a Bayesian hierarchical framework that allows us to simultaneously estimate group- and individual-level parameters and compare alternatives (e.g., Annis & Palmeri, 2018a). We fully describe the details of the modeling approach and its Bayesian implementation in later sections.

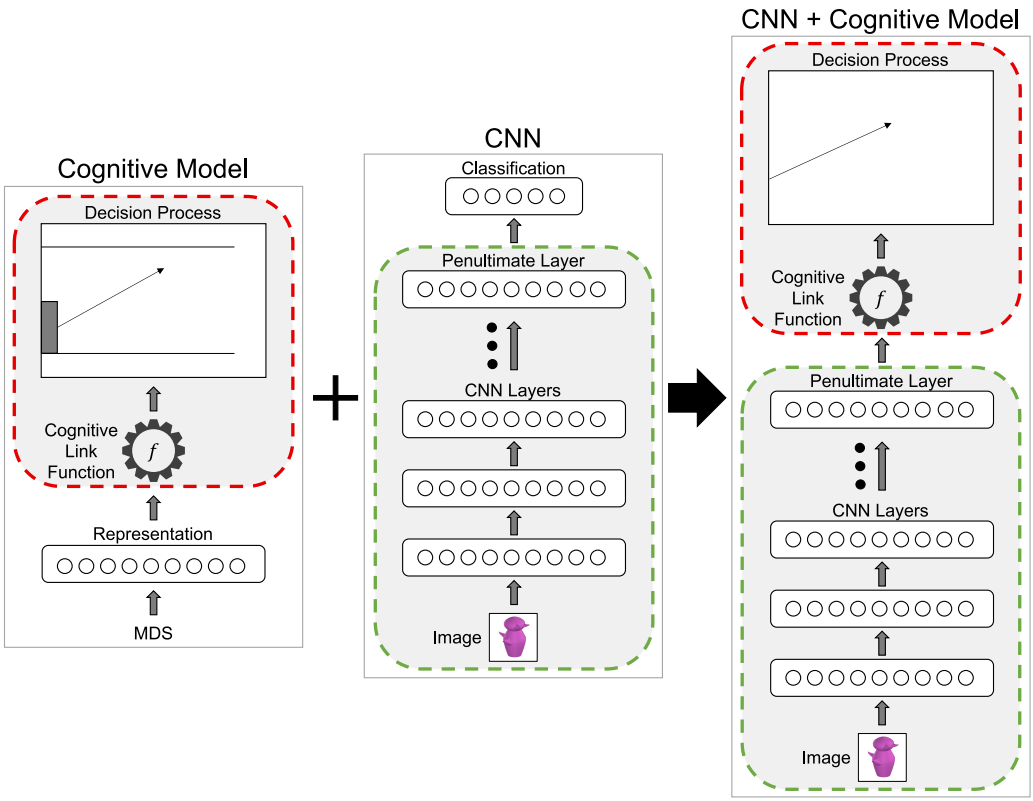

Figure 1. Framework for connecting convolutional neural networks with cognitive models. The decision process of the cognitive model is combined with the CNN via the cognitive link function such that rate of evidence accumulation in the decision process is a function of CNN object representations.

To evaluate this modeling approach quantitatively, we fitted behavioral data from Richler et al. (2019). They had subjects judge whether pairs of novel, computer-generated objects, drawn from the same category, were the same or different. On each trial of this *same-different sequential matching task* subjects were presented one object, followed by a brief pattern mask, followed by a second object. The two objects could be different, requiring a "different" response by keypress, or the two objects could be the same, possibly differing in size or viewpoint or both, requiring a "same" response by keypress (Figure 4 illustrates one trial). Errors and response times in the matching task were recorded and constitute the data we aimed to account for at the subject level and the item level; additional details of the experiment and its results are reported in later sections.

The types of novel objects (Figure 2) used by Richler et al. (2019), and that we use here, have been used in a range of other experiments to understand visual object recognition, perceptual expertise, and individual differences in visual cognition, both behaviorally and neurally (e.g., Gauthier & Tarr, 1997; Gauthier, Williams, Tarr, & Tanaka, 1998; Richler, Wilmer, & Gauthier, 2017; Wong, Palmeri, & Gauthier, 2009; Wong, Palmeri, Rogers, Gore, & Gauthier, 2009). Images of the objects were originally rendered from three-dimensional computer models, allowing the objects to be presented from different viewpoints. Greebles (Gauthier & Tarr, 1997) of the same type all share the same central body shape with four parts that are each unique in shape to one and only one Greeble; two types of Greebles were used that had different central body shapes, with one type having an asymmetric arrangement of parts (Greeble 1) and the other type a symmetric arrangement of parts (Greeble 2). Ziggerins (Wong, Palmeri, Rogers, et al., 2009) of the same type all share the same central base and the same general configuration of three or four parts, with each part unique to one and only one Ziggerin;

two types of Ziggerins were used, with different central bases and different configurations of parts. Sheinbugs (Richler et al., 2017) are insect-like creatures with a roughly common base body shape with five to eight parts arranged in a roughly symmetric fashion, with each part unique to one and only one Sheinbug; one type of Sheinbug was used.

While these objects may look somewhat creature-like (Greebles), tool-like (Ziggerins), or insect-like (Sheinbugs), they do not match any basic-level categories that subjects have had experience with prior to entering the experiment. Furthermore, these objects are artificial (but not in the sense of an artifact vs. natural kind distinction). It is in this sense that we refer to them as *novel* objects, in contrast to familiar objects. The use of novel objects rather than familiar objects in object recognition and categorization experiments provides a natural control for domain experience (Palmeri & Gauthier, 2004; Richler & Palmeri, 2014); this control can prove particularly useful for work aimed at understanding individual differences in visual cognition (e.g., Chua, Richler, & Gauthier, 2014; Gauthier et al., 1998; Richler et al., 2017; Wong, Palmeri, & Gauthier, 2009).

The added emphasis that the Greebles, Ziggerins, and Sheinbugs all had sets of features unique to each particular instance was intentional. Some types of artificial stimuli used in experiments assume instantiations of features that are shared across different instances. In some cases, this is because the stimuli are composed of simple features, like Gabor patches that vary in orientation and spatial frequency. Other times, the stimuli are more complex, for example computerized drawings of rockets that vary in the shape of their nose, tail, wings, and porthole (e.g., see Palmeri, 1999; Richler & Palmeri, 2014) where two different rockets might share exactly the same nose and tail but have different wings and porthole. An advantage of such relatively simple stimuli, at least from the perspective of modeling behavior, is that stimuli can

be represented in terms hand-coded, low-dimensional feature representations, like [1, 2, 2, 1] for one rocket and [1, 2, 1, 2] for another rocket, making the similarity between the two readily apparent. Greebles, Ziggerins, and Sheinbugs were not rendered this way, making such hand-coded, low-dimensional feature representations impossible to produce. Indeed, one motivation for exploring the combination of CNN representations with cognitive models is to allow those models to be extended to more complex objects with unique feature instantiations, which are arguably characteristic of many real-world objects.

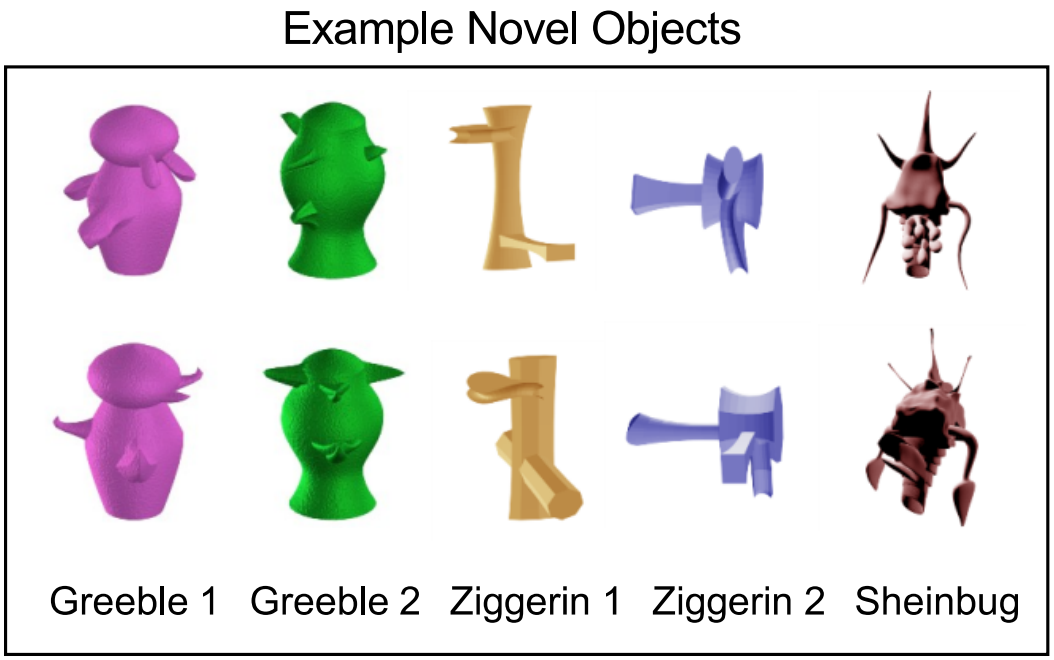

Figure 2. Example novel object from each category (two types of Greebles, two types of Ziggerins, and Sheinbugs).

The outline of the article is as follows: First, we describe the general modeling approach. Second, we report a preliminary investigation into how well CNNs represent the novel objects used in Richler et al. Because the CNNs we used were not trained on these novel objects, it is an open question as to whether they would be able to produce sensible representations at all. Third, we describe how we combined the CNNs with the LBA model of decision making, how this combined model was simulated and fitted to data, and present the fits of the model to the Richler et al. data and model selection results. To preview, the model selection procedure preferred combining the CNN with the cognitive model over the LBA alone. However, there were several key limitations of the combined CNN + cognitive model at both the subject and item level. The final modeling section discusses potential remedies for these failures.

GENERAL MODELING APPROACH

Here, we outline the general modeling approach illustrated in Figure 1. The full combined model can be divided into three subcomponents: *CNN object representation*, *cognitive link function*, and *decision process*, which we describe in turn. Details of the CNN object representations will be provided below, as will a general overview of the cognitive link function and decision process for any generic application of our modeling approach. Additional task-specific application details of the particular cognitive link function and decision process we used here will be described later, closer to where those details will need to be referenced.

CNN Object Representation

The middle panel of Figure 1 illustrates the general architecture of a Convolutional Neural Network (CNN). CNN models are complex, multi-layer versions of standard neural

networks from decades ago. They are composed of units that have some specified nonlinear activation function based on the net input from other units that are connected via weights learned by some modern variant of error-driven backpropagation (Rumelhart & McClelland, 1986). Like older backpropagation networks, CNN models are composed of layers of units. Unlike older networks, which were often limited to an input layer, a single hidden layer, and an output later, CNNs are deep networks (Krizhevsky et al., 2012) with many hidden layers. The first successes of deep networks were in computer vision (Krizhevsky et al., 2012). Such deep networks not only mirror the hierarchy of processing in the primate visual system (Felleman & Van Essen, 1991), but use convolution operations that mirror the receptive field properties of visual neurons, with more complex feature maps at higher levels of the hierarchy, and with larger receptive fields at higher levels emerging from the inclusion of so-called max-pooling (or average pooling) operations taken from earlier models of primate visual object recognition (e.g., Riesenhuber & Poggio, 1999).

The CNNs we used are models trained to do object classification. Such models often have five main types of layers: an *input layer*, *convolutional layers, pooling layers, fully-connected layers,* and an *output layer*. The input layer of the CNN corresponds to the raw or pre-processed pixel representation of an image; models of vision preserve the 2D (grayscale) or 3D (color) structure of the image on the input. Convolutional layers process the image by convolving multiple learned filters (with relatively small receptive fields) over their input to produce multiple *feature maps*. A trained CNN learns a set of filters that produces useful features for the task the network is trained to do (in this case, object classification). Visualization techniques reveal that CNNs learn filters in early layers of the network that often look a lot like the orientation and spatial frequency tuning of neurons in early visual cortex, features of

intermediate complexity in intermediate layers, and more complex object-like features in later layers (for recent reviews, see Gauthier & Tarr, 2016; Kriegeskorte, 2015). Feature maps are passed to the next convolutional layer in the hierarchy or to a pooling layer that downsamples each activation map by taking the average or the maximum within a small region. Deep CNNs will have many successive convolutional and pooling layers. CNNs may also include fully-connected layers between the convolutional and output layers; each unit of a fully-connected layer is connected to each unit in the layer preceding it, much like traditional neural networks. The final layer in the CNN is the *output layer*, with each unit fully connected to the units in the penultimate layer. In the case of a network trained to do object classification, each unit corresponds to a category label. The unit with the highest activation corresponds to the category output.

All of the CNN architectures we investigated were previously trained on object classification using massive image sets from the ImageNet Challenge (Russakovsky et al., 2015). We did not create or train the CNNs ourselves. We chose three CNN architectures that were implemented in the easy-to-use `Keras` package (Allaire & Chollet, 2018), a high-level API running on top of TensorFlow (Abadi et al., 2016): `VGG-16` (Simonyan & Zisserman, 2014a), `ResNet-50` (He, Zhang, Ren, & Sun, 2016), and `Inception-v3` (Szegedy et al., 2015). All are popular in the computer vision literature and were at one point a top competitor in the ImageNet Challenge. The `VGG-16` implementation we used had trained network weights that were ported from the Visual Geometry Group at the University of Oxford (Simonyan & Zisserman, 2014b), the `ResNet-50` implementation used weights from its original repository (He, 2016), and the `Inception-v3` implementation used weights trained by the developers of the R interface to Keras (Allaire & Chollet, 2018).

While `VGG-16`, `ResNet-50`, and `Inception-v3` share the same general architecture of all CNNs, there are differences. `VGG-16` is a basic deep network with successive convolutional and pooling layers followed by two fully-connected layers (each with 4096 units), followed by 1000 classification output units (the number of categories in the ImageNet Challenge). `ResNet-50` also has convolutional and pooling layers but includes a residual learning framework in which the input from a previous layer is mapped via a shortcut connection to a later layer. The network then only needs to learn the residual features beyond the available input, allowing extremely deep networks to be effectively trained. `ResNet-50` has no fully-connected layers. The final layer has 1000 classification output units. `VGG-16` and `ResNet-50` both utilize 3x3 convolutional filters, shown to be sufficient in practice to result in good performance without being overly demanding computationally. `Inception-v3` instead computes an array of convolutions at different filter sizes simultaneously. It then uses a convolutional layer of filter size 1x1 to downsample the output. This enables the network to increase its depth as well as its width (the number of parallel operations). This brief discussion leaves out a number of technical details, so we refer the reader to the original papers describing these CNNs for further discussion. In part, we leave out these technical details because we are treating the CNNs as black boxes that we obtained "off the shelf," as representative of popular deep learning computer vision models. This selection of models appeared sufficient to ask whether the approach of marrying a CNN object representation with a cognitive model is viable and promising for future research.

As noted, `VGG-16`, `ResNet-50`, and `Inception-v3` were all pre-trained to classify images of objects from the 1000 non-overlapping categories (e.g., mug, leopard, mushroom, motor scooter) in the ImageNet Challenge (Russakovsky et al., 2015). We were interested in the

representations created by these networks for *novel* objects these networks were *not* trained on: instances of the two types of Greebles, two types of Ziggerins, and Sheinbugs (Figure 2) used by Richler et al. (2019). When an image of a Greeble is presented to one of the CNN models, we are not interested in how it classifies the object, but in how it represents the object prior to the classification layer. In a deep network, there are many layers that could be chosen as *the* object representation. We chose the simplest approach, which was to expose the highest-level representation of the CNN, the *penultimate layer*[3], by severing and discarding the output classification layer from each network. For example, in the case of `VGG-16`, an image of some particular Greeble would be presented as input to the network and the object representation of that Greeble would be the 4096-dimensional vector on the penultimate layer of the network. For ResNet and Inception, which do not have fully-connected layers, we used instead a flattened version of the last layer of each network to create a vector representing the object. It is such penultimate representations that have been used, for example, to relate object representations in CNNs with object representations in visual cortex revealed by fMRI or neurophysiology (e.g., Khaligh-Razavi & Kriegeskorte, 2014; Kriegeskorte, 2015). Picking some other layer in a network would make it difficult to directly compare different networks because, say, the fifth layer in VGG does correspond to the fifth layer in Inception; in fact, Inception has many more layers than VGG. Picking the penultimate layer representation was a simple and justifiable choice for this initial exploration. The penultimate representation is used as input to the cognitive link function and this will drive the decision process.

---

[3] Note, we performed a brief exploration of the performance of lower-level layers and found they did worse than the penultimate layers so we did not consider lower layer representations further.

Cognitive Link Function

We use the term *cognitive link function* to refer generically to computations performed on the object representation to create the evidence that drives the decision process for the particular task being modeled (see Figure 1). In the case of modeling recognition memory for objects, this could involve computing the overall summed similarity of the object to other object representations stored in memory, whereas in the case of modeling categorization, this could involve computing the relative summed similarity to objects belonging to one category in memory to those belonging to another category in memory (e.g., Battleday, Peterson, & Griffiths, 2019; Mack & Palmeri, in press; Nosofsky & Palmeri, 2015). In our case, for modeling same-different discrimination, this will simply involve computing the similarity between the first object (study object) and the second object (test object) and assessing the magnitude of this similarity relative to a criterion to make a same-different decision. The cognitive link function is not an ancillary assumption. Different cognitive link functions instantiate different theories of cognitive mechanisms (e.g., Battleday et al. , 2019; Nosofsky, 1992; Ross et al., 2014).

Decision Process

The decision process is driven by the evidence computed by the cognitive link function; as noted above, the evidence calculated by the link function would vary depending on the task, “old” versus “new” in the case of recognition, “dog” versus “cat” in the case of categorization,

"same" versus "different" in the case of discrimination[4]. CNN models, as well as neural network models more generally, can make decisions, for example by assuming a deterministic max rule on the output activations or a more probabilistic softmax rule. While these response rules can predict response probabilities (correct and error when there is a known answer), they do not predict response times. To predict both errors and response times, we used evidence from the cognitive link function to drive an accumulation of evidence, sequential sampling model of decision making (Ratcliff & Smith, 2004).

The particular sequential sampling model we used was the *Linear Ballistic Accumulator* (LBA) model (Brown & Heathcote, 2008; for a tutorial see Donkin, Brown, & Heathcote, 2011). Like all models in this class, LBA assumes that decisions are made by an accumulation of evidence over time to a threshold, with strong evidence leading to fast and accurate responses and weak evidence leading to slow and inaccurate responses, and high thresholds leading to slow but accurate responses and low thresholds leading to fast but inaccurate responses. Variability in model parameters over trials produces variability in predicted response times and response probabilities. Details of the mathematical formalization of LBA are provided later.

We chose LBA over alternative models such as the diffusion model (Ratcliff & Rouder, 1998) or the leaky competing accumulator model (Usher & McClelland, 2001) because LBA is computationally less expensive to simulate and fit than other sequential sampling models (Brown & Heathcote, 2008). This was especially important given our use of a Bayesian hierarchical approach, which gives a principled and robust way to estimate model parameters at a group level

[4] Decision processes are not limited to two choices, of course, and many models easily extend to N choices; two choices is arguably most common in behavioral experiments, especially those that measure both response time and response probabilities.

and individual level simultaneously (Katahira, 2016). A Bayesian approach comes at a cost of increased computational burden; LBA has a likelihood function that can be easily and quickly computed, decreasing that burden.

ASSESSING CNN REPRESENTATIONS OF NOVEL OBJECTS

Before exploring whether a model combining CNN representations with cognitive model mechanisms can account for human behavior, we needed to explore if CNNs trained on a range of real-world categories could produce sensible representations of novel objects (Greebles, Ziggerins, and Sheinbugs) they had never been trained on. CNNs learn representations that maximize their performance on the tasks they have been trained on – in the present case classification of real objects in ImageNet (Russakovsky et al., 2015).

The goal in this section is simply to visualize the representations of Greebles, Ziggerins, and Sheinbugs produced by the CNNs to see if their representations cluster by object category. Greebles, Ziggerins, and Sheinbugs are visually dissimilar from one another; within-category similarity is much higher than across-category similarity. Would this also be the case for their CNN representations? The representations that CNNs produce are high-dimensional, making them difficult to visualize directly. We explored the penultimate CNN representations in two ways. First, we computed the pairwise Euclidean distances between CNN representations for all novel objects to produce a Representational Dissimilarity Matrix (RDM) (e.g., Kriegeskorte & Kievit, 2013)[5]. We expected that similar novel objects from the same category would have

[5] Other metrics, like cosine similarity (angle between representational vectors), produced qualitatively similar results.

smaller distances than those that are less similar. Second, we visualized CNN representations by reducing dimensionality using multidimensional scaling (MDS) (e.g., Hout et al., 2013). In the MDS solution, if CNNs are representing novel objects sensibly, Greebles should cluster with other Greebles, not with Ziggerins.

Visualization Methods

Images

Inputs to the networks were the set of images from Richler et al. (2019). Each of the 5 categories (two types of Greebles, two types of Ziggerins, and Sheinbugs) consisted of 50 novel objects, each from 2 viewpoints, for a total of 500 images. The original images were 400 x 400 color (RGB) images. Images were converted to greyscale, rescaled, and preprocessed as outlined below. Margins of the objects ranged from roughly 0 to 70 pixels and varied between image and margin side.

For these initial visualizations, we converted the original color images to greyscale images so that network representations could not cluster on the basis of color as a distinguishing feature. Greyscale conversion was performed using the luma transform: $L = 0.30R + 0.59G + 0.11B$, where L is the output greyscale pixel value and R, G, B are the input color channel pixel values (Allaire & Chollet, 2018)[6]. We used standard approaches for rescaling the original images to the maximum allowable image size for each network. For `VGG-16` and `ResNet-50`, images

[6] We acknowledge that this is not an optimal setup as the CNNs were trained on color images and preprocessing is usually done channel-wise. But given the need to remove color as a distinguishing category feature and our use of pretrained CNNs, we did not see another option. Leaving color as a distinguishing feature could allow the CNNs to cluster by category for trivial reasons.

were rescaled to 224 x 224 using their default interpolation; for `Inception-v3`, images were rescaled to 299 x 299, using its default interpolation. Images were preprocessed according to default preprocessing steps required for each network. For `VGG-16` and `ResNet-50`, preprocessing involved zero-centering the pixel values with respect to the ImageNet dataset; for `Inception-v3`, preprocessing involved scaling the pixel values between -1 and 1 on an image-by-image basis.

One advantage of CNNs as models of vision is that they are largely invariant to translation and have some tolerance to scale variations. For comparisons between CNN representations and raw pixel representations that we report later, we randomly rescaled and translated the novel object images. Each novel object was randomly (by a uniform distribution) rescaled between 75% and 100% of its original size. The rescaled object was randomly (by a uniform distribution) placed on the same white background in a random location with the only constraint that the object did not come within 10 pixels of the image border. Image cropping, rescaling, and translating were performed with the `magick` package in R (Ooms, 2018).

CNN Representations

To allow each CNN to create representations of novel objects, we stripped off the output classification layer of each network to expose their *penultimate layer*. In the R implementations of the networks in Keras (Allaire & Chollet, 2018), the name of the output layers (1000-dimensional) are called `predictions` for `VGG-16` and `Inception-v3` and `fc1000` for `ResNet-50`; the name of the penultimate layers in these networks are called `fc2` for `VGG-16` (4096-dimensional), `avg_pool` for `Inception-v3` (2048-dimensional pooling layer) and `flatten_1` for `Inception-v3` (also 2048-dimensional pooling layer). Whereas the output

layer represents possible category membership, the penultimate layer represents the object relatively abstractly in a distributed form. We presented all 500 greyscale, re-scaled, preprocessed novel-object images (100 from each of the 5 categories) from Richler et al. (2019) to each network and obtained the penultimate layer representation.

In one of our analyses, we compared the high-level, high-dimensional representation of each object passed through each CNN to its pixel-level representation, with each image flattened to (one-dimensional) vector. To maintain a similar vector size to the CNN, we reduced each image to 64 x 64 pixels, which, after flattening, resulted in a 4096-dimensional vector for each object. We refer later to this representational method as the *pixel-based method*.

Representational Dissimilarity Matrix

After obtaining the vector representations for all of the novel objects from each network (and the pixel-based method), we computed the Euclidean distance between each pair of novel object vector representations. Computing all pairwise distances yields a 500 x 500 matrix (one for each of the three CNNs and one for the pixel-based method). This matrix of representational distances is often referred to as a *Representational Dissimilarity Matrix* (RDM); it is analogous to a confusion matrix. We visualize these matrices by color coding them according to pairwise distance value.

Assessing Representations with MDS

In addition to displaying the raw RDM, we also performed dimensionality reduction to visualize graphically the novel object representations in a multidimensional space using

multidimensional scaling (MDS)[7]. In an MDS solution, each point in a space represents an object. Objects are positioned in the space by the MDS algorithm so that the distance between objects preserves as much of the distance information in the RDM as possible. Because we were simply interested in assessing the relationships between objects in the same or different category – visualizing, to see if they clustered by category or not – we did not attempt a detailed investigation to provide a best-fit to the distance data and simply chose to represent the space in two dimensions. For each RDM, we obtained a two-dimensional MDS solution using the `smacof` package in R (de Leeuw & Mair, 2009), which performed a metric MDS on each distance matrix.

Results and Discussion

The top row (row A) of Figure 3 shows a colorized representation of the 500 x 500 RDM for `Inception-v3`, `ResNet-50`, `VGG-16`, and the pixel-based method (left to right). Bold borders demarcate novel object classes. For example, the top-left corner of each RDM shows the distances between all 100 Type-1 Greeble images and the bottom-right corner of each RDM represents the distances between between all 100 Sheinbug images. Each colored element in the RDM represents the Euclidean distance between the representations of two novel objects. Distance was linearly rescaled such that maximal similarity is obtained when the distance is equal to 0 and is represented by dark blue; maximal dissimilarity is obtained when the (suitably

[7] We also visualized the representations using a 2-dimensional t-SNE solution (van der Maaten, 2014; van der Maaten & Hinton, 2008) with the `Rtsne` R-package (Krijthe, 2015) with cosine similarity as a distance metric, and observed qualitatively similar clustering and linear separability.

rescaled) distance is equal to 1 and is represented by dark red. For all three CNNs and the pixel-based method, within-category distances were generally smaller than between-class distances, as revealed by generally cooler colors for regions along the diagonal and warmer colors for regions that were off-diagonal. The Sheinbug partition showed a slightly more complex pattern of distances, likely due to these objects' more intricate structure and greater heterogeneity.

The second row (row B) of Figure 3 shows the two-dimensional MDS solutions generated from each corresponding RDM described above it. All the MDS solutions show sensible clustering by category. Type-1 and Type-2 Greebles are clustered close together in space but maintain linearly separable clusters. The same is true for Type-1 and Type-2 Ziggerins. Sheinbugs are linearly separable from other categories but appear to be more spread out in the space, owing to their greater complexity and heterogeneity.

To illustrate that the CNNs are doing something beyond what can be done with a mere pixel-based representation, we calculated RDMs and MDS solutions for the same novel objects for the same three CNNs and pixel-based method after subjecting the images to the random size scaling and translation described earlier in the Methods section. The third row (row C) of Figure 3 shows the colorized RDMs. While all show a decrease in crispness of the distinction of within-category compared to between-category distances, the decrease is most clear in the case of the pixel-based method. This is easily apparent when visualizing the RDMs using MDS (row D of Figure 3). The novel object representations produced by the three CNNs largely maintain their category clustering in the face of scale and translation variation. This was not the case for the pixel-based method. Its MDS solution no longer shows linear separability between novel object categories.

CNNs have thus crossed the most-minimal bar of sensibly clustering representations of novel objects. Having shown the CNNs can provide seemingly sensible representations, we proceed to combining the CNNs with a cognitive model and attempting to fit the behavioral data from Richler et al. (2019).

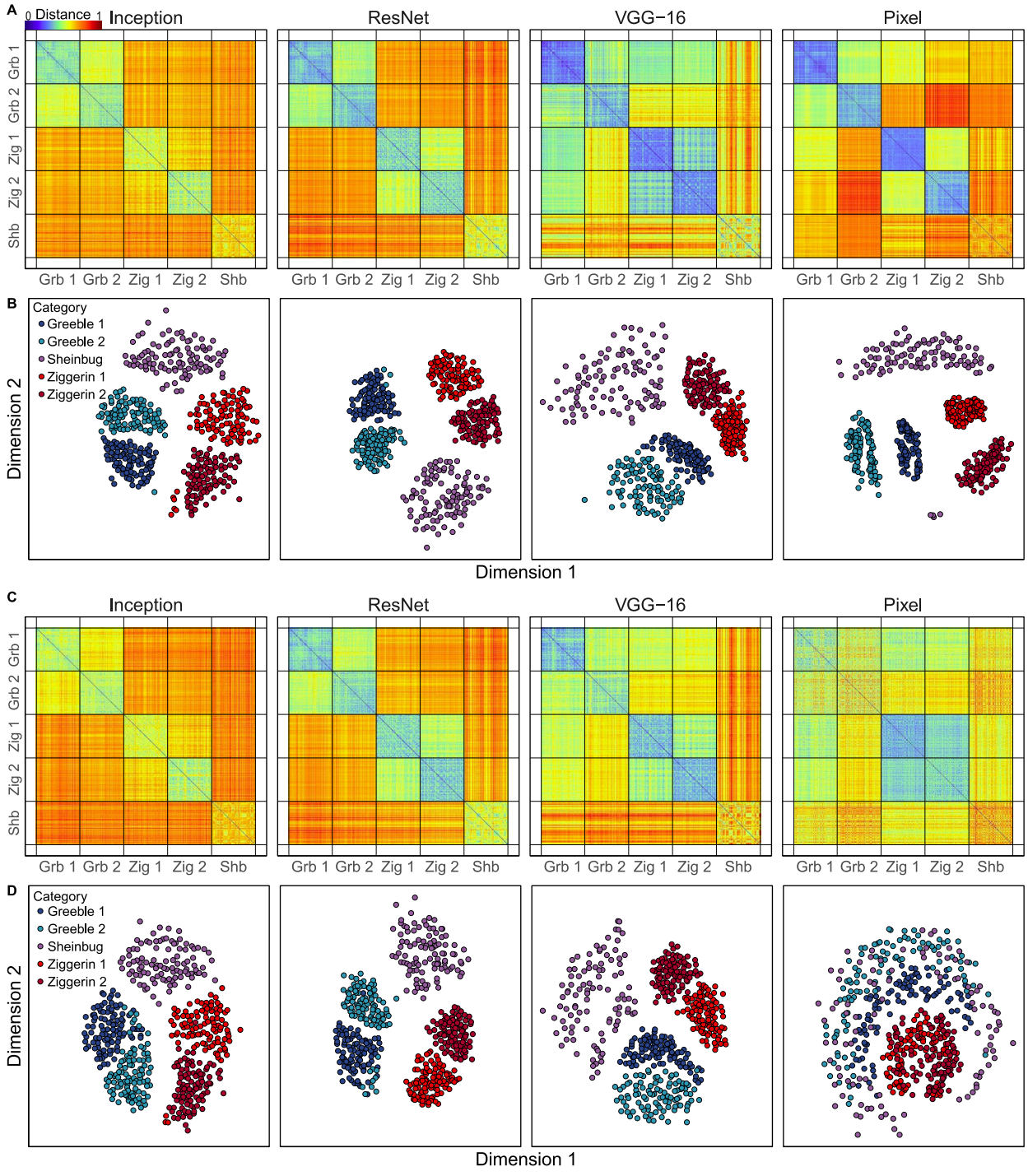

Figure 3. Panel A: Representational Distance Matrices (RDMs) over all novel objects for `Inception-v3`, `ResNet-50`, and `VGG-16`. The rectangular cells demarcate the novel object classes. Panel B: 2-dimensional MDS solution for each of the distance matrices. Panel C and D: RDM and 2-dimensional MDS solutions for randomly scaled and translated images. All results are based on greyscale images.

COMBINING CNNs AND COGNITIVE MODELS

We describe in detail how we combined CNN representations, a cognitive link function, and the LBA to account for behavioral data from Richler et al. (2019). To investigate individual differences in object recognition, these authors used a variety of tasks with novel objects. We used one of their tasks that lends itself particularly well to modeling both errors and response times: a same-different sequential matching task. The bottom panel of Figure 4 shows an example trial of the matching task. On each trial, the subject saw a novel object for 150 ms, followed by a mask for 500 ms, followed by another novel object that was either the same as or different from the first. The first object and second object could appear at one of two possible viewpoints and one of two possible sizes[8]. Subjects were instructed to respond "same" if the two objects had the same identity, regardless of viewpoint or size, and to respond "different" if the two objects had different identities. The two views showed objects rotated about 20 degrees apart along their vertical axis. Each condition (viewpoint, size, same/different identity) was presented an equal number of times and the order of conditions was randomized. The order of the trials was the same for all subjects so as not to confound trial order effects with individual differences. Subjects completed five same-different matching task sessions, each with 180 trials. Each session used a different category of novel objects (two types of Greebles, two types of Ziggerins, and Sheinbugs) and was conducted on different days. Subjects only made same-different decisions on pairs of objects within the same category, not across category. Each category included 50 different objects. An "item" in all subsequent discussions and analyses refers to a

[8] Because Richler et al. (2019) observed little behavioral effects of their size manipulation (2 x 2 vs 1.3 x 1.3 degrees of visual angle), we collapsed across size differences when fitting our models; doing so significantly reduced the complexity of the model fitting.

pair of presented images (that can be the same or different object). A total of 215 subjects completed the same-different tasks.

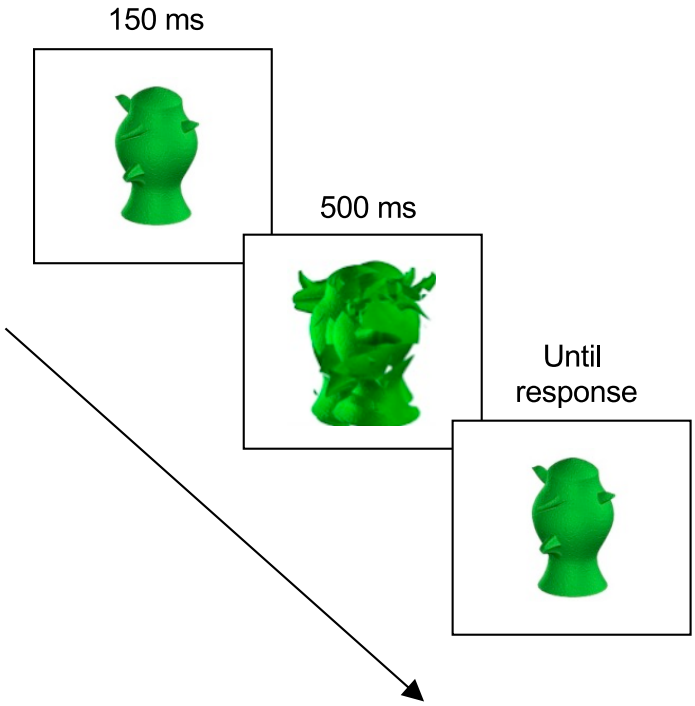

Figure 4. Example trial of same-different matching task.

Figure 5 summarizes the behavioral results. Panels A and B plot subject-level mean accuracy and response times as a function of viewpoint (same or different) and object identity (same or different); panels C and D plot item-level mean accuracy and response times. Subjects were faster and more accurate at deciding that two objects were "same" when they were the same if they were both presented from the same viewpoint. "Different" decisions were less affected by viewpoint.

A prominent feature of the behavioral data is the large amounts of variability observed across individuals and across items. The modeling will attempt to account for this variability in accuracy and RTs at both the subject-level, explaining individual differences, and the item-level, explaining item differences. The strategy is to account for individual differences by differences

in parameters of the cognitive level (Lee & Webb, 2005; Shen & Palmeri, 2016) and to account for item differences by differences in novel object representations created by the CNN. While we acknowledge that individual differences could also arise from differences in the CNN, such explorations are beyond the scope of this article, a point we elaborate on in the General Discussion.

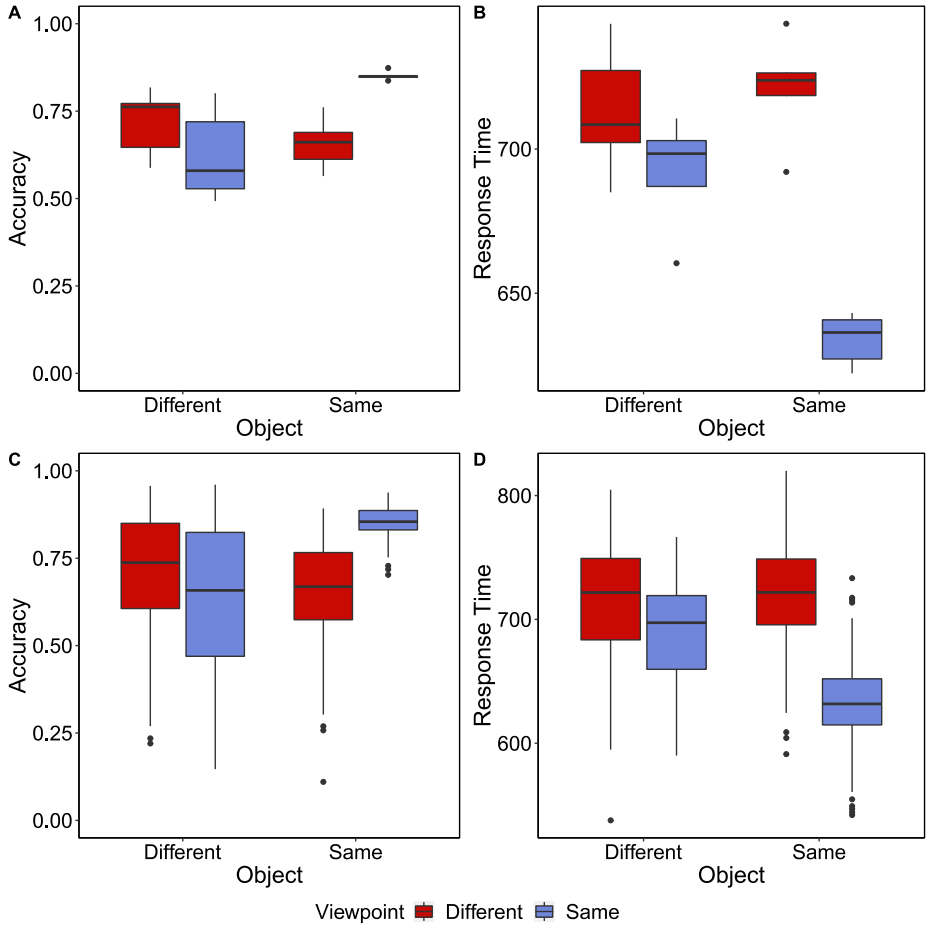

Figure 5. Panel A and B: Subject-level mean accuracy and response time as a function of object identity and viewpoint. Panel C and D: Item-level mean accuracy and response times as a function of object identity and viewpoint. Boxes represent inter-quartile range (IQR). Medians are plotted as lines inside the boxes. The upper whisker extends to the highest value that is within 1.5 x IQR. The lower whisker extends to the lowest value within 1.5 x IQR. Points plotted beyond the whiskers are outliers.

Figure 6 provides a more detailed elaboration of how we combined CNNs with a cognitive link function and a decision process (LBA) to model performance in the same-different matching task. To model performance in the same-different sequential matching task, the image of the first object was passed through the CNN and its representation on the penultimate layer (prior to the classification layer) was stored, then the image of the second object as passed through the CNN and its representation on the penultimate layer was compared to that of the first object.

The comparison was made by computing the Euclidean distance between the first and second object representations, converting that distance to a similarity by an exponential function (Shepard, 1987),[9] and converting the similarity to evidence for "same" relative to a criterion. The drift rate driving the LBA accumulator associated with a "same" response is equal to this evidence; the drift rate driving the accumulator associated with a "different" response was assumed to be simply one minus the evidence for "same". The LBA dynamics (described in detail later) determined the response probabilities and distribution of response times. Parameters of the cognitive link function and the decision process were adjusted to account for individual differences. We first examine how well the raw CNN representations account for item differences and later explore possible transformations of CNN representations to improve the account of item differences.

[9] Computing distances this way and converting them to similarities using an exponential is widely used in cognitive models (Nosofsky, 1992); a convenient property of the exponential is that sufficiently large distances, no matter how large, are all transformed to near zero similarities.

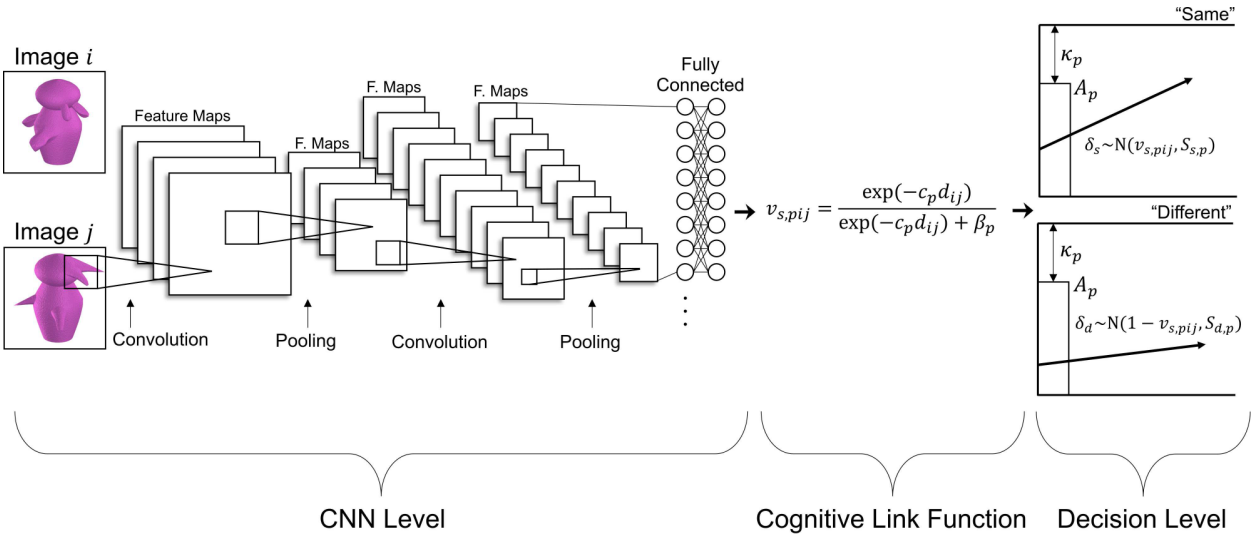

Figure 6. Illustrates how the CNN representation is converted into evidence (drift rate for the LBA) by a cognitive link function that is used to drive a decision level process (the LBA model). Here we illustrate some of the properties of a CNN (convolution operations, pooling, multiple levels of feature maps, and fully-connected layers), acknowledging that most successful CNNs have many layers and more complex architectures than what we illustrate here. The figure illustrates two methods of taking CNN representations and creating the drift rates driving the LBA.

Modeling Methods

CNN Representations

The images, preprocessing steps, and methods of generating representations using `VGG-16`, `ResNet-50`, `Inception-v3`, and the pixel-based method were the same as we used in the CNN representation visualizations reported earlier. The only exception is that for this modeling we used the color images originally used by Richler et al. (2019) rather than the greyscale images used earlier; because each session of the same-different sequential matching task used only objects from a single category and all objects in a category were colored the same way, it did not matter whether the images were colored or greyscale when modeling same-different performance.

To model the same-different matching task, we obtained a representation of the image of the first and second objects. While it would be reasonable to allow for noise in these representations, or some probabilistic failure to encode features of an image, especially the first image, we chose to keep the modeling simple and assume perfect vector representations of the first and second object images during matching. Noise was introduced in the decision process (LBA) rather than in the stages of object representation (CNN) or drift rate calculation (cognitive link function).

Cognitive Link Function

We connect the CNN representations for the first object image and second object image to the decisional process via a cognitive link function. To model the same-different matching task, this function uses the representations of the two-object image to generate the drift rates that drives the decision process. We aimed at a simple transformation from object representations to drift rate as follows: First, the distance between pairs of object representations created by a CNN was computed. Second, the distance was converted into similarity by an exponential function. Third, the similarity was transformed into evidence by a simple function akin to a probabilistic comparison between the similarity and a criterion for judging same versus different. The evidence for same and the evidence for different thus calculated were used to define the drift rate parameters in the LBA.

Calculating Distance from CNN Representations. Formally**,** to model a same-different decision given image *i* and image *j*, we presented a CNN with each image and obtained the

penultimate vector representation of image *i*, $\boldsymbol{x}_{ik}$, and image *j*, $\boldsymbol{x}_{jk}$ (from novel object category *k*). The Euclidean distance between *i* and *j* from category *k*, d_{ijk}, is given by:

$$d_{ijk} = \sqrt{\sum_{l=1}^{D} \left(x_{ilk} - x_{jlk}\right)^2}, \quad (1)$$

where D is the dimensionality of the vectors, l indexes each dimension of the vectors, and x_{ilk} is the value of the CNN representation of image *i* (in category *k*) along dimension *l*. A matrix, $\boldsymbol{d}_k$, forms the RDM for each category *k*[10]. Note, when referring to specific versions of the model in later sections, the distances (*d*) are obviously dependent on the particular CNN used to generate the object representations, which means that any parameters dependent on (*d*) will be dependent on the particular CNN as well; we do not explicitly denote this model-dependence via any further subscript or superscript to avoid symbolic clutter, but this dependence should be kept in mind.

Transforming Distances into Drift Rates. Each distance between the representations of objects *i* and *j* (from category *k*) was then transformed into a similarity, η, for subject *p*:

$$\eta_{pijk} = \exp\left(-c_{pk} d_{ijk}\right), \quad (2)$$

[10] We also fitted a set of models that used the MDS solution of the CNN representations. For these models, the distance matrix, $\boldsymbol{d}_k$, from the CNN is submitted to MDS [we tested 5- and 10-dimensional solutions following the elbow test (e.g., Hout et al., 2013)]. Each dimension of the resulting MDS solution was scaled by an attentional weight parameter resulting in a transformed distance between each image pair, where the attention weight parameters were fit along with the LBA parameters. The models that used the MDS representations provided only slightly better fits (qualitatively) but also contained a larger number of free parameters. This resulted in a hefty penalty by the model selection procedure. Consequently, we do not report these models here.

where c_{pk} is the sensitivity parameter that governs how steeply similarity decreases with distance (Nosofsky, 1992; Shepard, 1987)[11].

We transformed the similarity into the mean drift rate for a "same" response using a function that can be interpreted as a probabilistic comparison of similarity to a criterion (e.g., Maddox & Ashby, 1993). Specifically, the mean drift rate for a "same" response, $v_{s,pijk}$, given image pair *i, j* from category *k* is given by:

$$v_{s,pijk} = \frac{\eta_{pijk}}{\eta_{pijk} + \beta_{pk}}, \quad (3)$$

where β_{pk} is the criterion (for subject *p* with category *k*)[12]. This drift rate has a range between 0 (least evidence for "same") and 1 (greatest evidence for "same"). The drift rate has a value of 0.5 when the similarity η_{pijk} equals the criterion β_{pk}. As with some applications of LBA (Annis & Palmeri, 2019), we assumed that the drift rate for "different" was simply equal to 1 – $v_{s,pijk}$.

Decision Level

The evidence for "same" or "different" determine the trial-by-trial drift rates that drive an accumulation of evidence model of decision making, which in our case was the Linear Ballistic Accumulator (LBA) model (Brown & Heathcote, 2008). The LBA assumes a time to

[11] It may well be that full factorial investigation of different kinds of mappings from CNN representations to evidence driving a decision process might reveal that some mappings work better than others, but that kind of investigation seemed to be a bit outside the scope of the present work.

[12] Mathematically, because the similarities are exponentials, this function is equivalent to a logistic transformation of representational distance. The operations of this function could potentially be absorbed by the LBA decision component, perhaps by allowing different decision boundaries for "same" and "different" responses. One reason for using this function was maintain connection with prior modeling with the accumulator models that also used a criterion term like this (Annis & Palmeri, 2019; Mack & Palmeri, 2010).

perceptually process a stimulus, τ_{pk}[13] (for subject *p* with objects from category *k*). In LBA, evidence for "same" or "different" determine the drift rates that drive accumulators associated with a "same" or "different" decision. In each accumulator, the starting point for evidence accumulation on a given trial (*a*) is a random sample from a uniform distribution, $a \sim U(0, A_{pk})$, where A_{pk} is the upper bound on starting point (for subject *p* with category *k*). The drift rate determines the accumulation, as described in more detailed below. When the accumulation of evidence first reaches the response threshold, b_{pk} in one of the accumulators, the response corresponding to that accumulator is made; thresholds in the "same" and "different" accumulators were set equal to each other. Note, as in many applications of LBA, we parameterize the model using the relative threshold, $\kappa_{pk} = b_{pk} - A_{pk}$.

The cognitive link function determines the drift rate in each accumulator. The drift rate for a "same" response on a given trial for subject *p* given object pair (*i,j*) from category *k*, $\delta_{s,pijk}$, is a random sample from a normal distribution, $\delta_{s,pijk} \sim N(v_{s,pijk}, S_{s,pk})$, where v_s is the mean of the drift rates for the "same" accumulator given by Equation 4, and S_s is the standard deviation of the drift rates for the "same" accumulator. The drift rate for a "different" response on a given trial is a random sample from a normal distribution, $\delta_{d,pijk} \sim N(1 - v_{s,pijk}, S_{d,pk})$. Trial-to-trial variability in starting point (*a*) and drift rates (δ) are responsible for errors and response time variability predicted by LBA.

---

[13] Note that $\tau = \tau_e + \tau_r$, where τ_e is the time to perceptually encode the stimulus and τ_r is the time to execute the motor response. We model the total, τ.

Comparison Models

We compared the models combining CNN representations, cognitive link function, and LBA models with two other model variants. The first control comparison, referred to as the *free model*, is simply the standard LBA. It did not use any object representation to drive evidence accumulation, but simply allowed drift rates to be freely-estimated parameters. Specifically, the free model allowed the mean of the drift rates for the "same" accumulator to freely vary across conditions of object matching (same vs. different) and viewpoint (same vs. different) for each subject. All other parameters in the model were the same as those used in the CNN-LBA model. The free model has many more parameters than the CNN-LBA model because a separate drift rate is estimated for the different conditions of object matching and viewpoint (in the CNN-LBA model, drift rate is not a free parameter but is constrained by CNN representations).

The second control comparison, referred to as the *pixel-based model*, used the raw pixel representations of the images, instead of CNN representations, to determine distance and similarity to drive evidence accumulation in the LBA. Further details of the comparison models can be found in the Appendix.

Fitting Procedure and Model Selection

We cast the models in a Bayesian hierarchical framework (e.g., Annis & Palmeri, 2018a). This allowed us to estimate both individual-level and group-level parameters simultaneously, in a way more robust than other approaches (Katahira, 2016); see the Appendix for a description of the priors we chose to place on model parameters.

We used an advanced MCMC sampler known as Differential Evolution MCMC (DE-MCMC; Ter Braak, 2006; Turner, Sederberg, Brown, & Steyvers, 2013) that is capable of

sampling efficiently from models whose parameters are correlated, which is the case with the LBA. We used two times the number of subject-level parameters as the number of chains and ran the sampler for a total of 3000 iterations discarding the first 1000 as burn-in. Chains were visually inspected for convergence.

Model comparison was performed by computing the Bayes factor, a ratio of the evidence provided by the data in favor one model over another model. We report $\log_{10}$ Bayes factors comparing each model with the free model (the control comparison that simply assumed drift rates as free parameters, not based on CNN representations):

$$\log_{10} BF = \log_{10} \frac{p(\boldsymbol{D}|\mathcal{M})}{p(\boldsymbol{D}|Free\ Model)}. \quad (4)$$

Bayes factors less than 0 indicate evidence in favor of the free model, whereas Bayes factors greater than 0 indicate evidence in favor of the comparison model, $\mathcal{M}$. Bayes factors were estimated using a technique called steppingstone sampling (SS) (Annis, Evans, Miller, & Palmeri, 2019; Xie, Lewis, Fan, Kuo, & Chen, 2011). Briefly, the SS procedure involves drawing samples from a series of posteriors whose likelihood is raised to different powers, called *temperatures*, between 0 and 1. The means of the log likelihoods of the samples from each *power posterior* form points along a one-dimensional curve. The area under this curve turns out to be the log of the marginal likelihood[14] (for a tutorial, see Annis et al., 2019), which can then be used to compute the Bayes factor. SS was implemented by using 30 temperatures and collecting 300

[14] Although the method results in the natural logarithm of the marginal likelihood, we changed the base and used logarithms with base 10 for interpretability relative to the Kass and Raftery (1995) recommendations.

samples per power posterior, which has been shown to result in stable marginal likelihood estimates for the LBA (Annis et al., 2019).

While here we used Bayes factors, it is far more common in the machine learning and computer vision literatures to use out-of-sample methods, such as cross-validation, to assess overfitting and model complexity. In the case of computer vision, model like a CNN is fitted to "data" that corresponds to the thousands or even millions of images in a database like ImageNet, making it relatively easy to pull out large sets of training, validation, and test images. In the case of cognitive modeling, by contrast, the "data" are behavioral results from an experiment, which can often be quite small for experiments testing human subjects; this is especially the case for individual differences research, where it is necessary to test many different subjects, so for practical reasons the sessions are often quite short. In the context of the same-different sequential matching task, subjects may see each pair of objects only once and the number of pairs is on the order of several dozen. The small size of the dataset makes cross-validation infeasible. In fact, especially for small numbers of data points, cross-validation can sometimes prove little better than a coin flip at determining the generating model in a model recovery setting (e.g., Pitt, Myung, & Zhang, 2002). While measures like AIC and BIC sometimes prove little better, Bayesian methods are quite robust at determining the true underlying model.

Results and Discussion

Model Selection

We first report model comparisons and model selection before showing predictions so we can report on the best-fitting model rather than have to devote significant space to the predictions

of all models. Table 1 shows the results of the model selection[15]. The models are plotted in order of the $\log_{10}$ Bayes factor relative to the free model. The two models with Bayes factors exceeding the free model and pixel-based model were those using high-level object representations from `Inception-v3` and `ResNet-50`. The model using representations from `VGG-16` had Bayes factors exceeding the pixel-based model but not the free model. These model selection results suggest that using certain CNN representations to determine drift rates driving the LBA can greatly increase model performance. Next, we investigate the posterior predictions in order to determine why `Inception-v3`-LBA and `ResNet-50`-LBA were favored.

[15] The large scale of the Bayes factors might seem unfamiliar to those who use Bayes factors with simpler statistical models (e.g., Morey & Rouder, 2011). The steppingstone method that we used to compute the Bayes factor has been verified in a comparison study with other methods (Annis et al., 2019) using the LBA with both simulated and experimental data and was also found to produce very large Bayes factors for hierarchical models. It may be more useful to compare the scale of the Bayes factor with the scale of the Bayesian Information Criterion (BIC), which has been used more extensively with cognitive models like the LBA and is a rough approximation to 1/2 of the log marginal likelihood. Indeed, in a later section of the manuscript we will show the BIC reaches extremely high values on the log scale in a similar manner to those produced by the steppingstone method.

Table 1.

Results of the model selection in terms of the log_{10} Bayes factor. The correlations between the observed and predicted accuracy and response time (quantiles) are also shown for both the subject- and item-level.

			Subject Level		Item Level	
Model Name	**# Params**	**log_{10} BF**	**Acc. *r***	**RT *r***	**Acc. *r***	**RT *r***
Inception-v3-LBA	7525	9539	0.75	0.86	0.18	0.92
ResNet-50-LBA	7525	9434	0.73	0.86	0.24	0.93
Free	9675	0	0.95	0.87	0.05	0.93
VGG-16-LBA	7525	-7013	0.72	0.83	0.19	0.92
Pixel-Based	7525	-16360	0.61	0.79	0.08	0.89

Posterior Predictions

Subject-Level Predictions. Figure 7 shows the mean posterior predictions for the free model and all CNN-LBA model variants. Each point in Panel A represents the predicted vs. observed probability of responding "same" for a subject in a given condition. The top and bottom rows of Panel A show the predicted probability of responding "same" in the different-object condition and the same-object condition, respectively. Same and different viewpoint conditions are depicted by blue and red points, respectively. The far left column of Panel A shows the predictions for the free model; the other three columns show the predictions for `Inception-v3`-LBA, `ResNet-50`-LBA, and `VGG-16`-LBA, respectively. Although the free model performed worse in terms of model selection (Bayes factor), it did predict mean accuracy (*r* =

.95) better than `Inception-v3`-LBA (r = .75) and `ResNet-50`-LBA (r = .73). This is especially true in the same-object different-viewpoint condition (free model, r = .95; `Inception-v3`-LBA, r = .57; `ResNet-50`-LBA, r = .53). In this condition, the subject must recognize that the two objects are the same but are shown from different viewpoints. For the free model, this is achieved by simply allowing drift rate to vary across the requisite conditions in whatever manner necessary to achieve a good fit. For `Inception-v3`-LBA, `ResNet-50`-LBA, and `VGG-16`-LBA (as well as the pixel-based model), this requires object representations to be invariant to rotation. We will describe a method that aims to improve performance in the different viewpoint conditions in the next section.

Each point in Panel B shows the predicted vs. observed mean correct-RT quantiles (.1, .3, .5, .7, or .9) for a subject in a given condition (all r > .86). `Inception-v3`-LBA, `ResNet-50`-LBA, `VGG-16`-LBA, and the free model make similar predictions. All models appear to predict subject-level correct RT quantiles adequately across all conditions.

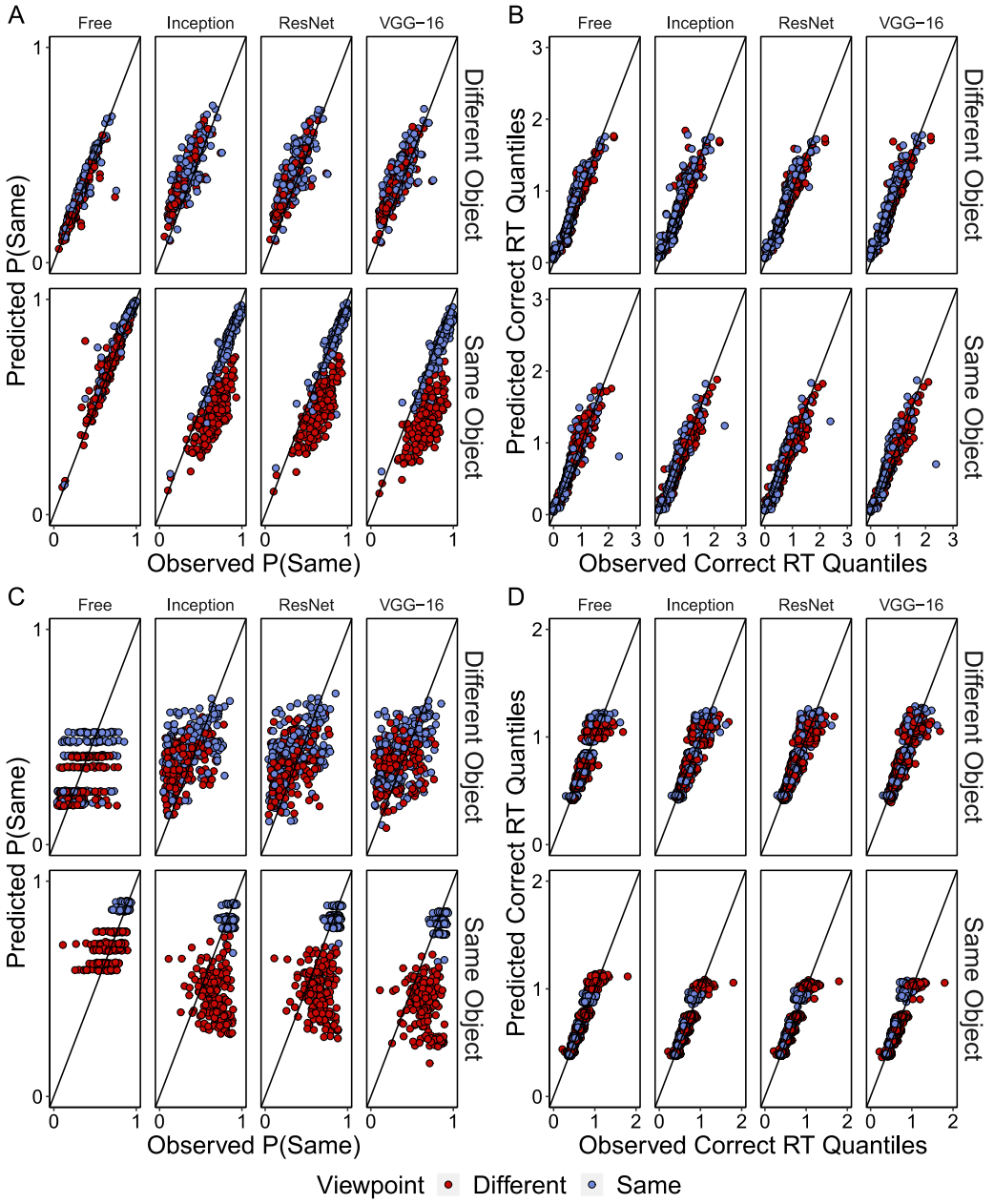

Figure 7. Panel A: Predicted vs. observed subject-level probability of responding "same" for different (top row) vs. same objects (bottom row) for the Free, `Inception-v3`-LBA, `ResNet-50`-LBA, and `VGG-16`-LBA models. Panel B: Predicted vs. observed subject-level correct RT quantiles for each model. Panel C: Predicted vs. observed item-level probability of responding "same." Panel B: Predicted vs. observed item-level correct RT quantiles.

Item-Level Predictions. Panels C and D of Figure 7 show the mean posterior predictions for `Inception-v3`-LBA, `ResNet-50`-LBA, `VGG-16`-LBA, and the free model at the item

level (recall that in the case of a same-different sequential matching task, each *item* is a pair of novel object images on a task trial). Each point in Panel A of Figure 7 represents the predicted vs. observed probability of responding "same" for a given item in a given condition. The free model did not include parameters at the item level (only at the condition level) and therefore was unable to produce variability in item-level predictions within each category. Thus, the free model predicts only the average probability of responding "same" for each novel object category (r = .05); the striations of points produced by the free model are due to collapsing over novel object categories; each row is representative of a different novel object category. The free model was only able to predict average item-level performance. The item-level predictions from `Inception-v3`-LBA, `ResNet-50`-LBA, and `VGG-16`-LBA were better (r = .18, r = .24 and r = .19), although in absolute terms, they were still not very good. As in the subject-level predictions, we especially observed misfits for item-level predictions in the same-object / different-viewpoint condition (all $r < 0$).

Panel B of Figure 5 shows the predicted vs. observed mean correct-RT quantiles (.1, .3, .5, .7, or .9) for each item in a given condition. The qualitative differences between the models are much less apparent than the item-level accuracy fits. All models appear to capture the overall pattern of correct-RT quantiles across items (all $r \geq .92$).

Overall, the posterior predictions indicate that much of the qualitative differences between models can be found in the predicted responses at both the subject and item level. `Inception-v3`-LBA, `ResNet-50`-LBA, and `VGG-16`-LBA made fairly similar predictions. Compared to the free model, the CNN models made qualitatively worse predictions of subject-level responses, but far better predictions of item-level responses. It is important to note that all

model fits were based on estimating subject-level parameters – item-level predictions simply emerged from the distances formed by the representations of the CNNs between novel objects.

IMPROVING MODEL PERFORMANCE BY TRANSFORMING CNN REPRESENTATIONS

We found that the `Inception-v3`-LBA and `ResNet-50`-LBA model were favored by the Bayes factor over the other models we tested. However, in terms of qualitative fits, `Inception-v3`-LBA and `ResNet-50`-LBA had difficulty predicting certain subject- and item-level responses, particularly for the same-object / different-viewpoint condition, and had difficulty predicting item-level accuracy in general (although they were much better than the free model).

Poorer item-level than subject-level fits are partially explained by the fact that the models we tested included subject-level parameters and not item-level parameters. For models of the complexity that we tested, it proved impossible from a practical standpoint to simultaneously estimate parameters for both subject-level and item-level data, which is often only possible for relatively simple psychometric models (e.g., Rouder, Province, Morey, Gomez, & Heathcote, 2015). Here we examined whether it would be possible to perform transformations of the CNN representations to improve item-level predictions, especially in the same-object different-viewpoint condition, while maintaining good subject-level predictions.

Because modeling the same-different task is based on the distance between representations of the first object and the second object, instead of transforming the representations and then calculating distances, we used a technique that performs a

transformation while calculating distances (c.f., Peterson et al., 2016). This approach is known in the machine learning literature as *metric learning* (Bellet, Habrard, & Sebban, 2015).

For our application, using metric learning requires dimensionally-reduced CNN representations. Given the limited number of images we have of the novel objects, we could not estimate a transformation of a full 2048-dimensional representation on the penultimate layer of a CNN (there are far too many parameters in a general transformation matrix to estimate). As will be seen, we explored dimensionality reduction using MDS or PCA and then estimated a transformation of these dimensionality-reduced representations using metric learning. A special case of these metric learning transformations would be to simply weight the dimensions, as in the generalized context model (GCM) (Nosofsky, 1986); metric learning allows for a more general transformation beyond such weighting. Because `ResNet-50`-CNN provided the best account in our earlier model comparisons, we chose to explore metric learning in detail using `ResNet-50` object representations.

Metrics and Their Interpretation

A metric is a function that defines the distance between pairs, in our case pairs of object images as represented by high-dimensional feature vectors through a CNN. One standard metric is the familiar Euclidean metric, here expressed using both vector notation and the more conventional scalar notation:

$$d_{ij} = \sqrt{\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)^{\mathrm{T}}\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)}$$
$$= \sqrt{\sum_{l=1}^{D}\left(x_{il} - x_{jl}\right)^2}. \quad (5)$$

The Euclidean metric is actually a special case of a more general quadratic distance, sometimes referred to as a *Mahalanobis distance*[16] (Bellet et al., 2015) and what we refer to as the *full metric* hereafter:

$$d_{ij}^{M} = \sqrt{\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)^{\mathrm{T}}\mathbf{M}\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)}, \quad (6)$$

where **M** must be positive definite to satisfy the properties of a metric. When **M** is the identity matrix, **I**=diag(1,1, ... 1), the Mahalanobis distance reduces to the Euclidean distance. When **M** is the diagonal matrix, $\mathbf{M} = \mathrm{diag}(w_1, \ldots w_D)$, the distance metric includes dimension weights like those in the GCM (Nosofsky, 1986):

$$d_{ij}^{M} = \sqrt{\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)^{\mathrm{T}}\mathrm{diag}(w_1, \ldots w_D)\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)}$$
$$= \sqrt{\sum_{l=1}^{D} w_l\left(x_{il} - x_{jl}\right)^2}. \quad (7)$$

[16] The Mahalanobis distance originally comes from Mahalanobis (1936) and refers to a distance that involves the correlation among dimensions $d = \sqrt{(\boldsymbol{x_i} - \boldsymbol{x_j})^{\boldsymbol{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x_i} - \boldsymbol{x_j})}$,where $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ are vectors with covariance matrix $\boldsymbol{\Sigma}$. Through an unfortunate choice of terminology, the metric learning literature refers to generalized quadratic distances (Equation 6) as the Mahalanobis distance (Bellet et al., 2015).

Matrix $\mathbf{M}$ can be written as $\mathbf{M} = \mathbf{U}^{\mathrm{T}}\mathbf{U}$. This, in turn, allows us to write Equation 6 as:

$$d_{ij}^{M} = \sqrt{\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}\mathbf{U}\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)} = \sqrt{\left(\mathbf{U}\boldsymbol{x}_i - \mathbf{U}\boldsymbol{x}_j\right)^{\mathrm{T}}\left(\mathbf{U}\boldsymbol{x}_i - \mathbf{U}\boldsymbol{x}_j\right)}. \quad (8)$$

Learning a metric, $\mathbf{M}$, is equivalent to learning the linear transformation, $\mathbf{U}\boldsymbol{x}$. Thus, we can think about metric learning in geometric terms. Because the Euclidean metric assumes the identity matrix in the more general full metric, the Euclidean metric (Equation 8) can be viewed a transformation on $\boldsymbol{x}$ that simply returns $\boldsymbol{x}$. When $\mathbf{M}$ is the diagonal matrix (Equation 7), this results in a rescaling of the dimensions of $\boldsymbol{x}$. With a full metric (Equation 6), any metric-preserving linear transformation is allowed on $\boldsymbol{x}$ such as scaling, rotating, and shearing.

We introduce one final metric that takes the same form as the full metric, except the main diagonal is held constant, such that $\mathbf{M}_{ij} = c\ (i \neq j)$, where c is a constant. This approach allows any metric-preserving linear transformation of $\boldsymbol{x}$ such that scaling along the dimensions of $\boldsymbol{x}$ are constant. We refer to this metric as the *off-diagonal metric*.

So far, we have described four types of metrics, which we refer to as the *Euclidean metric* (Equation 5)*,* the *diagonal metric* (Equation 7), the *off-diagonal metric,* and the *full metric* (Equation 6). Because this approach is extremely flexible, it allows us to gain insight into the complexity of the transformation on CNN representations that one needs to obtain certain levels of predicted performance.

Method

In this section, the goal is to create a new metric that produces a better fit to the data using CNN (`ResNet-50`) representations. The elements of the transformation matrix **M** need to be estimated in much the same way as any other parameter of a model (whether a CNN model or cognitive model). Ideally, the parameters of **M** and the parameters of the cognitive link function and the LBA would be estimated simultaneously. This proved to be impractical for us to implement because the computation of the likelihood for each update proved to be prohibitively slow and we saw no sign of convergence (although we note that convergence should be theoretically guaranteed via alternative sampling schemes that utilize Gibbs sampling). Instead, we used a two-step approach, estimating (learning) the parameters of the metric **M** (which aims to maximize item-level fits) and then estimating the parameters of the cognitive link plus LBA as a second step (which aims to maximize subject-level fits).

Figure 8 illustrates this two-step approach. As before, we first submitted the images to the CNN and obtain their representations on the penultimate layer of the network. Because it would be impossible to estimate M with a 2048-dimensional feature vector (from `ResNet-50`), we performed a dimensionality reduction. We tried both Principle Components Analysis (PCA) and MDS, two standard dimensionality-reduction techniques. The reduced-dimensionality representations were then submitted to one of two possible metric learning algorithms that we tried (described below), which both output an estimated transformation matrix, **M**. After the metric learning was done, we fitted the model using the suitably transformed representations, the cognitive link function, and the LBA to each subject's data.

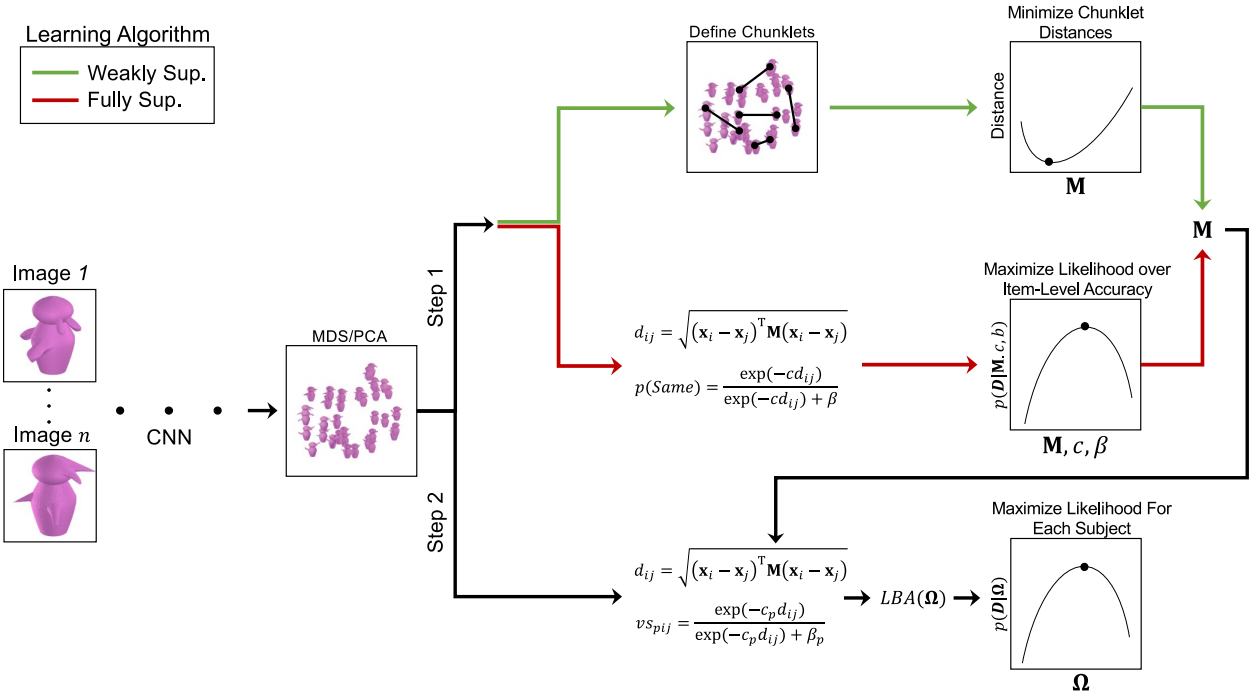

Figure 8. Shows the two-step approach to fitting both the distance function and the LBA. Images are first passed through the CNN. The MDS or PCA representation is then obtained. The distance function is learned via one of two metric learning algorithms. For weakly supervised metric learning, chunks of image pairs are defined. The metric learning algorithm minimizes the distance between chunks by finding the appropriate transformation, **M**. For the fully supervised metric-learning method, the transformation is learned directly from the data; for a given image pair, (*i*,*j*), the similarity is given by $\exp(-cd_{ij})$, where c is the sensitivity parameter and d_{ij} is the distance governed by **M**. The similarity is then used in a logistic function to produce the probability of responding "same" to a given item pair. The best fitting parameters are then obtained using maximum likelihood over the item-level data. After running one of the algorithms in step 1, the matrix, **M**, is then used to fit the LBA over the entire data set in step 2.

Step 1: Metric Learning

In step 1 (Figure 8), metrics were learned using two different types of metric-learning algorithms, which we refer to as *weakly-supervised metric learning* and *fully-supervised metric learning* (Bellet et al., 2015); we adapted these metric learning approaches as commonly used in the machine learning literature. Figure 8 roughly depicts weakly-supervised metric learning with green arrows and fully-supervised metric learning with red arrows. Weakly-supervised metric

learning uses information, called *side-information*, relevant to the task in order to learn the appropriate metric. One type of side-information is *equivalence constraints* in which the user defines a set of objects that are assumed to be part of the same class, called *chunklets*. In the present application, chunklets consisted of pairs of images of the same object from two different viewpoints. The particular metric learning algorithm we used, called *Relevant Component Analysis* (RCA; Bar-hillel & Weinshall, 2005; Shental, Hertz, Weinshall, & Pavel, 2002), minimizes the distance between chunklets (see Appendix for details). After obtaining a set of *D*-dimensional points from either PCA or MDS for each novel object class, *k*, we defined chunklets over the points corresponding to pairs of novel objects that were the same object from different viewpoints. The chunklet constraints along with the PCA or MDS representations were submitted to RCA, resulting in the estimated matrix, $\mathbf{M}_k$.

The fully-supervised metric-learning algorithm learns a metric directly from the data. As was the case for the weakly-supervised approach, we first obtained a set of *D*-dimensional points from either PCA or MDS for each novel object class. The distance between novel object representations in category *k,* $\boldsymbol{x}_{ik}$ and $\boldsymbol{x}_{jk}$, is given by Equation 6. The probability of responding "same" is given by:

$$p(Same)_{ijk} = \frac{\exp\left(-c_k d_{ij}\right)}{\exp\left(-c_k d_{ij}\right) + \beta_k}. \tag{9}$$

The model was fit to item-level response data. We assume the number of "same" responses given to object pair, (*i*,*j*), in category *k* follows the binomial distribution:

$$y_{ijk} \sim Binomial(p(Same)_{ijk}, z_{ijk}), \tag{10}$$

where y_{ijk} is the number of "same" responses across all subjects, and z_{ijk} is the item pair's respective total number of observed responses collapsed across subjects. Because we were only interested in point estimates of $\mathbf{M}_k$ to use in the second step of the model fitting procedure, we performed maximum likelihood estimation to find the best fitting c_k, β_k, and $\mathbf{M}_k$ parameters. Since we fit the model to the item-level data that is collapsed across subjects, the c_k and β_k parameters can be interpreted as averaged sensitivity and criterion noise parameters across subjects.

Step 2: Model Fitting

In step 2, subject-level accuracy and RTs were fit using the metric learned in step 1. Note that $\mathbf{M}_k$ was fixed in step 2, having already been estimated in step 1. Otherwise, the model is exactly the same as the CNN-LBA models described earlier, except that now the distances are derived using the more general metric learning transformation to calculate distances between object representations.

Each subject's data were fit individually using maximum likelihood estimation. Note, we use maximum-likelihood estimation in the second step because it allows us to perform model selection using information about the number of parameters in the first step in which the matrix $\mathbf{M}_k$ was estimated (not possible using the Bayesian methods we used earlier). Using the Bayesian Information Criterion (BIC; Schwarz, 1978), we can easily include the number of parameters in the matrix, $\mathbf{M}_k$, along with the number parameters estimated in step 2. If we were to use Bayes factors as we did in the previous section, it would not be possible to include information about

$\mathbf{M}_k$ in the step 2 model fitting procedure. Thus, while the Bayes factor would not be able to penalize the model for increased complexity in $\mathbf{M}_k$, the BIC allows us to do this[17].

For both the fully-supervised full-metric method and the fully-supervised off-diagonal method, we used three different dimensionalities (determined by the dimensionality reduction step of the MDS/PCA): 5x5, 10x10, and 15x15. For the fully-supervised diagonal-metric method, we used seven different diagonal matrices: $diag(5)$, $diag(10)$, $diag(20)$, $diag(35)$, $diag(50)$, $diag(75)$, where $diag(x) = diag(w_1, \ldots, w_x)$. Note, the maximum overall dimensionality used for the diagonal metric was larger than that of the full metric because we were only estimating the diagonal entries, making the task much more computationally feasible. For the weakly-supervised approach, we used the same sizes of **M** as in the direct full-metric approach. We used PCA and MDS for dimensionality reduction of the CNN's image representations.

Results and Discussion

Table 2 shows the top models from each model class in order of the BIC, where model class is defined by the metric learning type (fully-supervised or weakly-supervised) and the metric type (full, diagonal, or off-diagonal). We also include the `ResNet-50`-LBA model as a comparison. Thus, the top model in Table 2 was the top model overall out of all the models tested. However, because we only list the top model from each class, the rank of a model listed below the top model does not necessarily correspond to its ranking in the full list. The full model

[17] Note, an alternative method might be to regularize the metric via a penalty for deviating from the identity.

selection table can be found in the Appendix. The total number of metric parameters from step 1 were included in the calculation of the BIC and AIC along with the total number of subject-level parameters from step 2 to penalize for the size of the metric.

Table 2.

Top models from each model class in terms of the BIC. Correlations between the observed and predicted accuracy and response time (quantiles) are also shown for both the subject- and item-level. The model whose properties are listed as NA represents the `ResNet-50`*-LBA model in which no metric learning was used.*

						Subject Level		Item Level	
Learning	**Dim.**	**Metric**	**Dim.**	**BIC**	**AIC**	**Acc. *r***	**RT *r***	**Acc. *r***	**RT *r***
Fully Sup.	MDS	off-diag.	15	167741	90985	0.86	0.90	0.79	0.91
Fully Sup.	MDS	diagonal	20	168401	95236	0.86	0.90	0.41	0.91
Fully Sup.	MDS	full	15	176398	98176	0.85	0.90	0.79	0.91
Weakly Sup.	MDS	full	15	187516	115362	0.87	0.90	0.29	0.91
NA	NA	NA	0	212812	140659	0.79	0.89	0.24	0.90

The top performing fully-supervised model (and top performing model overall) used an off-diagonal metric and a 15-dimensional MDS representation, which we refer to as the *FS-Off model*. The top performing fully-supervised model that utilized a diagonal metric was the one that used a 20-dimensional MDS representation, which we refer to as the *FS-Diag model*. The top performing fully-supervised model that utilized a full metric was the one that used a 15-dimensional MDS representation, which we refer to as the *FS-full model*. Lastly, the top

performing weakly-supervised model was the one that utilized a 15-dimensional MDS representation. All top models from each model class were favored over the `ResNet-50`-LBA model, suggesting that the group-metric approach is fairly robust.

Figure 9 shows the top model from each model class we described above. The subject-level predictions comparing the models are shown in Panel A of Figure 9. Overall, the models fared better than `ResNet-50`-LBA (r = .79) in terms of accuracy (all r > .84). Panel A of Figure 8 shows the predictions in the same-object different-viewpoint condition was not problematic (all r > .84) as was the case for `ResNet-50`-LBA (see Figure 7). There was also a qualitative improvement to the different-object condition overall across all models compared to `ResNet-50`-LBA (see Figure 7). Panel B shows response time predictions were very accurate (all r = .90).

Panel C of Figure 9 shows the predictions at the item-level for each model was the area of most improvement. While `ResNet-50`-LBA struggled with predicting item-level accuracy (r = .24), the top model achieved item-level performance that was close to subject-level (r = .79, respectively). The same-object different-viewpoint condition also improved for all models relative to `ResNet-50`-LBA (which predicted r < 0), especially for the FS-full model and the top model (r = .69 and r = .53). Panel D of Figure 9 shows the item-level RT predictions for all models was very good (all r = .91) and were similar to `ResNet-50`-LBA (r = .90).

These results demonstrate that metric learning can be used to transform CNN representations to improve model performance. Metric learning allows for the fine-tuning of the distance function by considering contextual information or can be learned directly from the data. The two-step approach, although not ideal, allowed us to test a large variety of models with high-dimensional metrics. We found that nearly all of the models that incorporated metric learning

outperformed the `ResNet-50`-LBA model suggesting the approach is fairly robust. The model that employed the fully-supervised method with an off-diagonal metric was selected as the top model and achieved item-level performance similar to its subject-level performance.

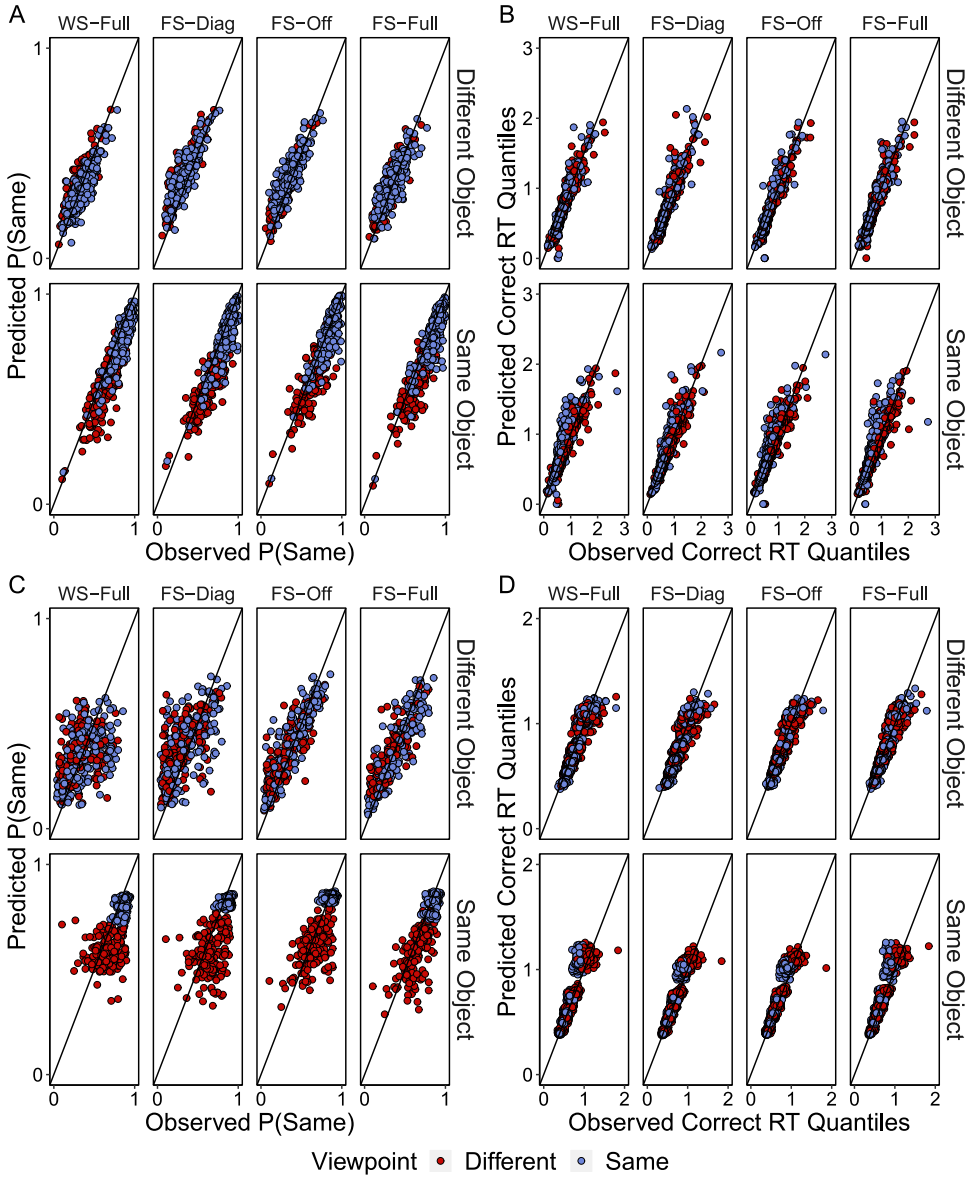

Figure 9. Panel A: Predicted vs. observed subject-level probability of responding "same" for different (top row) vs. same objects (bottom row) for the WS-Full (weakly-supervised full-metric), FS-Diagonal (fully-supervised diagonal-metric), FS-Off-Diagonal (fully-supervised off-diagonal-metric), and FS-Full (fully-supervised full-metric) models. Panel B: Predicted vs. observed subject-level correct RT quantiles for each model. Panel C: Predicted vs. observed

item-level probability of responding "same." Panel B: Predicted vs. observed item-level correct RT quantiles.

GENERAL DISCUSSION

This work presents a unique synergy of convolutional neural network (CNN) models and cognitive models to predict novel object recognition in humans. We used CNN models to create high-level representations of novel objects with item-level variability predicted by object-level differences in CNN representations. We then used those representations to drive evidence for speeded perceptual matching by a cognitive model, allowing differences in parameters of the cognitive model to predict individual differences in human performance. Using CNN representations provided a more parsimonious explanation than using a cognitive model in isolation per a Bayes factor, which weighs model fit against model complexity. Mirroring the way that CNN models have recently provided insights into neural representations of complex objects (e.g., Kriegeskorte, 2015; Yamins et al., 2014), CNN models may provide a useful perceptual front-end for understanding high-level visual cognition at not only the average level but also the individual subject level and the item level.

Other recent work has demonstrated how CNNs may capture aspects of human judgments of object similarities (e.g., Khaligh-Razavi & Kriegeskorte, 2014; Peterson et al., 2016), such work has often used objects similar to those used to train a neural network (e.g., Yu, Liu, Samaras, & Zelinsky, 2019), or used transfer learning to retrain the final layers of a neural network to represent naturalistic objects (Holmes, O'Daniels, & Trueblood, 2019[18]; Sanders &

[18] Note, Holmes et al. (2019) cited an earlier version of the current work that appeared in a conference proceeding, Annis & Palmeri (2018b).

Nosofsky, 2018). We asked whether, and how well, CNNs might capture novel object recognition in human subjects without training the network to recognize those objects.

The use of novel objects that subjects have never seen before[19] is an important tool for understanding visual object recognition, categorization, and memory (Richler & Palmeri, 2014). This may be most obviously the case for understanding category learning and the development of perceptual expertise (Gauthier & Tarr, 1997; Palmeri & Gauthier, 2004), where the goal is to understand how new object representations and new object category knowledge are formed. The use of novel objects is also important for understanding individual differences in visual cognition as a control for past experience not possible using common objects (e.g., Chua et al., 2014; Gauthier et al., 1998; Richler et al., 2017; Wong et al., 2009). Some of this past work has used Greebles (Gauthier & Tarr, 1997), Ziggerins (Wong, Palmeri, & Gauthier, 2009), and Sheinbugs (Richler et al., 2017), the categories we used here (dataset from Richler et al., 2019). By design, these objects are composed of parts (Figure 2) that are unique to each individual instance; they are not created using interchangeable parts, like a Mr. Potato Head or a construction made from Lego pieces. The use of such individually-distinct objects brings with it a cost of not having a clear experimenter-defined coding from which to estimate object similarities. An advantage however, is the ecological similarity to natural object categories (e.g., faces or birds species) in which the corresponding part for two objects may be similar but is never truly identical. We asked whether object representations of novel objects created by CNN models could be used to estimate object similarities and predict performance.

---

[19] As noted earlier, while certain novel object may bear some similarities to previously-experienced objects at a broad, superordinate level (creature-like, tool-like, bug-like), they bear no similarities previously-experienced objects at a basic level.

Despite CNNs never having been trained on Greebles, Ziggerins, and Sheinbugs, we showed that CNN representations clustered by novel object category. We also showed that CNN representations could drive a cognitive model to predict errors and response times in a perceptual matching task using those novel objects to a promising extent, and that CNN representations could be transformed in a way that improved prediction.

The transformations of CNN representations were especially needed to account for same-different performance for pairs of objects differing in viewpoint. While CNNs are trained on images of real objects that vary in viewpoint, rotation is not formally manipulated within the ImageNet database (images were simply sampled from the web), and it is likely that some object images are from a limited number of canonical viewpoints. It is possible that CNNs trained on an image dataset that more systematically manipulates viewpoint might create object representations robust to viewpoint changes. It seems just as likely that CNN representations are not robust to viewpoint changes because these CNNs are trained to classify objects as members of the same category irrespective of viewpoint, not trained to determine whether two objects from different viewpoints are the same thing (Palmeri & Gauthier, 2004; Peissig & Tarr, 2007).

Far more work needs to be done to develop complete models of the visual processing hierarchy that better capture neurophysiological constraints, predicts neural data, and captures detailed aspects of human visual recognition and representation of complex objects and scenes, as well as account for individual differences brain and behavioral measures. Current CNN models allow researchers interested in high-level visual cognition an exciting new approach to modeling visual recognition, visual attention, visual memory, and visual learning, and we anticipate the future CNN models better informed by human and non-human primate vision will provide even deeper insights. For the first time, the same complex images that are presented to

subjects in experiments can be presented to the perceptual front end instantiated by a CNN and its high-level representations can be coupled with alternative models of representation, perceptual and cognitive processing, and decision making to predict visual behavior in ways never before possible. CNNs may provide a way to ground abstractions of psychological representations to the assumptions CNNs make about stimuli, thereby providing a standardized and automatized approach to obtaining representations of complex stimuli.

Just as decision-making models might inform CNN representations, CNNs can also inform decision-making models. Our choice of the LBA was mainly born out of computational ease. However, many different decision models could be used. For example, the LBA assumes a large amount of between-trial variability for drift rate and starting point. While this might be a reasonable explanation of why choice and RT vary between the same pairs of images, a model that only assumes within-trial variability could also be considered. Because the model can predict item-level variability, it poses an opportunity to examine the extent to which within- and between-trial mechanisms explain the variability of decision dynamics. We believe this may be fruitful grounds for future research that has direct implications for decision-making models.

Combining CNN-derived representations with a cognitive link function to drive evidence accumulation in the Linear Ballistic Accumulator (Brown & Heathcote, 2008) allowed us to predict detailed aspects of response times and errors across items and across subjects in a visual same-different sequential matching task with novel objects. Although combined CNN+cognitive models using `Inception-v3` or `ResNet-50` novel object representations were favored by a

Bayes factor[20], which weights model fit against model complexity, over all other models, including a free model with freely-estimated drift rate parameters, subject-level accuracy predictions were quantitatively worse than the free model. Both subject-level and item-level predictions were improved by a metric learning transformation on the CNN representations (see also Peterson et al., 2016). A limitation of this two-step approach is that the learned metric and the cognitive model are not fitted simultaneously, thereby potentially reducing the quality of fit; nevertheless, the approach far exceeded the baseline model that did not employ metric learning and nearly doubled the correlation between the observed and predicted item-level response probabilities. We look forward to next-generation CNN architectures that provide better accounts of human object representations without needing to perform such transformations.

In the present work, we used suitably-transformed CNN representations to predict item-level differences and allowed differences in the parameters of a cognitive model (cognitive link function plus the LBA) to predict subject-level differences. Of course, individual differences in visual cognition could also arise from individual differences in the visual processing hierarchy, not just in the cognitive-level mechanisms associated with task representations, memory, and decision making, perhaps because of individual differences in visual experience, architecture, and other important factors. Exploring such sources is an interesting and important direction for future work but is also a complex undertaking on current computer hardware because it requires

[20] The reason behind `Inception-v3` and ResNet-50 being favored over `VGG-16` would be conjecture on our part. Here, we treated the CNNs as black boxes, however, future work might investigate the underlying bases for why some CNNs are better at forming representations that are human-like than others.

training many convolutional neural networks instantiating different architectures with different parameter tunings on different image training sets.[21]

[21] A single retraining of just the fully-connected layers of a CNN model like `VGG-16` using ImageNet requires several days of processing on a state-of-the-art workstation equipped with multiple state-of-the-art GPU cards.

Acknowledgements

This work was supported by grants from NSF (SMA-1640681 and SBE-1257098), an NEI core grant to the Vanderbilt Vision Research Center (NEI P30-EY008126), and an NEI training grant (T32-EY07135).

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., … Isard, M. (2016). Tensorflow: A system for large-scale machine learning. *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 265–283.

Allaire, J. J., & Chollet, F. (2018). *keras: R Interface to "Keras."* Retrieved from https://keras.rstudio.com

Annis, J., Evans, N. J., Miller, B. J., & Palmeri, T. J. (2019). Thermodynamic integration and steppingstone sampling methods for estimating Bayes factors: A tutorial for psychologists. *Journal of Mathematical Psychology*, *89*, 67–86.

Annis, J., & Palmeri, T. J. (2018a). Bayesian statistical approaches to evaluating cognitive models. *Wiley Interdisciplinary Reviews: Cognitive Science*, *9*(2), e1458. https://doi.org/10.1002/wcs.1458

Annis, J., & Palmeri, T. J. (2018b). Combining convolutional neural networks and cognitive models to predict novel object recognition in humans. *2018 Conference on Cognitive Computational Neuroscience*. https://doi.org/https://doi. org/10.32470/CCN. 2018

Annis, J., & Palmeri, T. J. (2019). Modeling memory dynamics in visual expertise. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *45*(9), 1599–1618.

Bar-hillel, A., & Weinshall, D. (2005). Learning a Mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, *6*(1), 937–965.

Bartlema, A., Lee, M., Wetzels, R., & Vanpaemel, W. (2014). A Bayesian hierarchical mixture approach to individual differences: Case studies in selective attention and representation in category learning. *Journal of Mathematical Psychology*, *59*(1), 132–150. https://doi.org/10.1016/j.jmp.2013.12.002

Battleday, R. M., Peterson, J. C., & Griffiths, T. L. (2019). *Capturing human categorization of natural images at scale by combining deep networks and cognitive models*. Retrieved from http://arxiv.org/abs/1904.12690

Bellet, A., Habrard, A., & Sebban, M. (2015). Metric Learning. In R. J. Brachman, W. W. Cohen, & P. Stone (Eds.), *Synthesis Lectures on Artificial Intelligence and Machine Learning* (pp. 1–151). https://doi.org/https://doi.org/10.2200/S00626ED1V01Y201501AIM030

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning : A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828.

Brown, S. D., & Heathcote, A. (2008). The simplest complete model of choice response time: Linear ballistic accumulation. *Cognitive Psychology*, *57*(3), 153–178. https://doi.org/10.1016/j.cogpsych.2007.12.002

Chua, K.-W., Richler, J. J., & Gauthier, I. (2014). Becoming a Lunari or Taiyo expert: Learned attention to parts drives holistic processing of faces. *Journal of Experimental Psychology: Human Perception and Performance*, *40*(3), 1174–1182.

Dailey, M. N., & Cottrell, G. W. (1999). Organization of face and object recognition in modular neural network models. *Neural Networks*, *12*(7–8), 1053–1074. https://doi.org/10.1016/S0893-6080(99)00050-7

de Leeuw, J., & Mair, P. (2009). Multidimensional Scaling using Majorization: SMACOF in R. *Journal of Statistical Software*, *31*(3), 1–30. Retrieved from http://www.jstatsoft.org/v31/i03/

Donkin, C., Brown, S., & Heathcote, A. (2011). Drawing conclusions from choice response time

models: A tutorial using the linear ballistic accumulator. *Journal of Mathematical Psychology*, *55*(2), 140–151. https://doi.org/10.1016/j.jmp.2010.10.001

Dubey, R., Peterson, J., Khosla, A., Yang, M., & Ghanem, B. (2015). What makes an object memorable. *2015 IEEE International Conference on Computer Vision (ICCV)*, 1089–1097. https://doi.org/10.1109/ICCV.2015.130

Felleman, D. J., & Van Essen, D. C. (1991). Distributed hierarchical processing in the primate cerebral cortex. *Cerebral Cortex*, *1*(1), 1–47.

Gauthier, I., & Tarr, M. J. (1997). Becoming a "Greeble" expert: Exploring mechanisms for face recognition. *Vision Research*, *37*(12), 1673–1682.

Gauthier, I., & Tarr, M. J. (2016). Visual object recognition: Do we (finally) know more now than we did? *Annual Review of Vision Science*, *2*, 377–396.

Gauthier, I., Williams, P., Tarr, M. J., & Tanaka, J. W. (1998). Training greebles'experts: A framework for studying expert object recognition processes. *Vision Research*, *38*(15–16), 2401–2428. Retrieved from http://www.sciencedirect.com/science/article/pii/S0042698997004422

He, K. (2016). Deep residual learning for image recognition. Retrieved from https://github.com/KaimingHe/deep-residual-networks

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *Proceedings of the IEEE International Conference on Computer Vision*, 1026–1034.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*(7), 1527–1554. https://doi.org/10.1162/neco.2006.18.7.1527

Holmes, W. R., O'Daniels, P., & Trueblood, J. S. (2020). A joint deep neural network and evidence accumulation modeling approach to human decision-making with naturalistic images. *Computational Brain & Behavior*, *3*, 1–12. https://doi.org/10.1007/s42113-019-00042-1

Hout, M. C., Papesh, M. H., & Goldinger, S. D. (2013). Multidimensional scaling. *Wiley Interdisciplinary Reviews: Cognitive Science*, *4*(1), 93–103. https://doi.org/10.1002/wcs.1203

Jozwik, K. M., Kriegeskorte, N., Storrs, K. R., & Mur, M. (2017). Deep convolutional neural networks outperform feature-based but not categorical models in explaining object similarity judgments. *Frontiers in Psychology*, *8*, 1726. https://doi.org/10.3389/fpsyg.2017.01726

Kass, R. E., & Raftery, A. E. (1995). Bayes factors. *Journal of the American Statistical Association*, *90*(430), 773–795.

Katahira, K. (2016). How hierarchical models improve point estimates of model parameters at the individual level. *Journal of Mathematical Psychology*, *73*, 37–58. https://doi.org/10.1016/j.jmp.2016.03.007

Khaligh-Razavi, S. M., & Kriegeskorte, N. (2014). Deep supervised, but not unsupervised, models may explain IT cortical representation. *PLoS Computational Biology*, *10*(11). https://doi.org/10.1371/journal.pcbi.1003915

Kriegeskorte, N. (2015). Deep neural networks: A new framework for modeling biological vision and brain information processing. *Annual Review of Vision Science*, *1*(1), 417–446.

https://doi.org/10.1146/annurev-vision-082114-035447

Kriegeskorte, N., & Kievit, R. A. (2013). Representational geometry: Integrating cognition, computation, and the brain. *Trends in Cognitive Sciences*, *17*(8), 401–412.

Krijthe, J. H. (2015). *Rtsne: T-distributed stochastic neighbor embedding using a Barnes-Hut implementation*. Retrieved from https://github.com/jkrijthe/Rtsne

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 1097–1105.

Kubilius, J., Bracci, S., & Op de Beeck, H. P. (2016). Deep neural networks as a computational model for human shape sensitivity. *PLoS Computational Biology*, *12*(4), 1–26. https://doi.org/10.1371/journal.pcbi.1004896

Lake, B. M., Zaremba, W., Fergus, R., & Gureckis, T. M. (2015). Deep neural networks predict category typicality ratings for images. *Proceedings of the 37th Annual Cognitive Science Society*, 1243–1248.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature Methods*, *13*(1), 35. https://doi.org/10.1038/nmeth.3707

Lee, M. D., & Webb, M. R. (2005). Modeling individual differences in cognition. *Psychonomic Bulletin & Review*, *12*(4), 605–621. https://doi.org/10.3758/BF03196751

Mack, M. L., & Palmeri, T. J. (n.d.). Discrimination, recognition, and classification. In M. J. Kahana & A. Wagner (Eds.), *Handbook on Human Memory*. Oxford University Press.

Mack, M. L., & Palmeri, T. J. (2010). Modeling categorization of scenes containing consistent versus inconsistent objects. *Journal of Vision*, *10*(3), 1–11. https://doi.org/10.1167/10.3.11

Maddox, W. T., & Ashby, F. G. (1993). Comparing decision bound and exemplar models of

categorization. *Perception & Psychophysics*, *53*(1), 49–70.

Mahalanobis, P. C. (1936). On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India*, 49–55. India.

Nosofsky, R. M. (1986). Attention, similarity, and the identification-categorization relationship. *Journal of Experimental Psychology*, *115*(1), 39–57.

Nosofsky, R. M. (1992). Similarity scaling and cognitive process models. *Annual Review of Psychology*, *43*(1), 25–53.

Nosofsky, R. M., & Palmeri, T. J. (1997). An exemplar-based random walk model of speeded classification. *Psychological Review*, *104*(2), 266–300. https://doi.org/10.1037/0033-295X.104.2.266

Nosofsky, R. M., & Palmeri, T. J. (2015). An exemplar-based random-walk model of categorization and recognition. In J. R. Busemeyer, Z. Wang, J. T. Townsend, & A. Eidels (Eds.), *Oxford Handbook of Computational and Mathematical Psychology* (pp. 142–164). New York: Oxford University Press.

Nosofsky, R. M., Sanders, C. A., & McDaniel, M. A. (2018). Tests of an exemplar-memory model of classification learning in a high-dimensional natural science category domain. *Journal of Experimental Psychology: General*, *147*(3), 328–353. https://doi.org/http://dx.doi.org/10.1037/xge0000369

Nosofsky, R. M., Sanders, C. A., Meagher, B. J., & Douglas, B. J. (2017). Toward the development of a feature-space representation for a complex natural category domain. *Behavior Research Methods*. https://doi.org/10.3758/s13428-017-0884-8

Ooms, J. (2018). *magick: Advanced graphics and image-processing in R*. Retrieved from https://cran.r-project.org/package=magick

Palmeri, T. J. (1999). Learning hierarchically structured categories: a comparison of category learning models. *Psychonomic Bulletin & Review*, *6*, 495–503.

Palmeri, T. J., & Cottrell, G. W. (2009). Modeling perceptual expertise. In D. N. Bub, M. J. Tarr, & I. Gauthier (Eds.), *Perceptual expertise: Bridging brain and behavior* (pp. 197–245). New York, NY: Oxford University Press.

Palmeri, T. J., & Gauthier, I. (2004). Visual object understanding. *Nature Reviews. Neuroscience*, *5*(4), 291–303. https://doi.org/10.1038/nrn1364

Peissig, J. J., & Tarr, M. J. (2007). Visual object recognition: Do we know more now than we did 20 years ago? *Annu. Rev. Psychol.*, *58*, 75–96.

Peterson, J., Abbott, J., & Griffiths, T. (2016). Adapting deep network features to capture psychological representations. *Proceedings of the 38th Annual Conference of the Cognitive Science Society*, 2363–2368. Retrieved from https://mindmodeling.org/cogsci2016/

Pitt, M. A., Myung, I. J., & Zhang, S. (2002). Toward a method of selecting among computational models of cognition. *Psychological Review*, *109*(3), 472.

Ratcliff, R., & Childers, R. (2015). Individual differences and fitting methods for the two-choice diffusion model of decision making. *Decision*, *2*(4), 237.

Ratcliff, R., & Rouder, J. N. (1998). Modeling response times for two-choice-decisions. *Psychological Science*, *9*(5), 347–356.

Ratcliff, R., & Smith, P. L. (2004). A comparison of sequential sampling models for two-choice reaction time. *Psychological Review*, *111*(2), 333–367. https://doi.org/10.1016/j.pestbp.2011.02.012.Investigations

Richler, J. J., & Palmeri, T. J. (2014). Visual category learning. *Wiley Interdisciplinary Reviews: Cognitive Science*, *5*, 75–94. https://doi.org/10.1002/wcs.1268

Richler, J. J., Tomarken, A. J., Sunday, M. A., Vickery, T. J., Ryan, K. F., Floyd, R. J., … Wong, A. C.-N. (2019). Individual differences in object recognition. *Psychological Review*, *126*(2), 226–251. https://doi.org/http://dx.doi.org/10.1037/rev0000129

Richler, J. J., Wilmer, J. B., & Gauthier, I. (2017). General object recognition is specific: Evidence from novel and familiar objects. *Cognition*, *166*, 42–55. https://doi.org/10.1016/j.cognition.2017.05.019

Riesenhuber, M., & Poggio, T. (1999). Hierarchical models of object recognition in cortex. *Nature Neuroscience*, *2*(11), 1019–1025. https://doi.org/10.1038/14819

Ross, D. A., Deroche, M., & Palmeri, T. J. (2014). *Not just the norm : Exemplar-based models also predict face aftereffects*. 47–70. https://doi.org/10.3758/s13423-013-0449-5

Rouder, J. N., Province, J. M., Morey, R. D., Gomez, P., & Heathcote, A. (2015). The lognormal race: A cognitive-process model of choice and latency with desirable psychometric properties. *Psychometrika*, *80*(2), 491–513. https://doi.org/10.1007/s11336-013-9396-3

Rumelhart, D. E., & McClelland, J. L. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Cambridge, MA: MIT Press.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., … Bernstein, M. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, *115*(3), 211–252. https://doi.org/10.1007/s11263-015-0816-y

Sanders, C. A. (2018). *Using deep learning to automatically extract psychological representations of complex natural stimuli*. Indiana University.

Sanders, C. A., & Nosofsky, R. M. (2018). Using Deep-Learning Representations of Complex Natural Stimuli as Input to Psychological Models of Classification. *Proceedings of the 40th Annual Conference of the Cognitive Science Society*.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, *6*(2), 461–464. https://doi.org/10.1214/aos/1176344136

Shen, J., & Palmeri, T. J. (2016). Modelling individual difference in visual categorization. *Visual Cognition*, *24*(3), 260–283. https://doi.org/10.1080/13506285.2016.1236053

Shental, N., Hertz, T., Weinshall, D., & Pavel, M. (2002). Adjustment learning and relevant component analysis. *Proceedings of the 7th European Conference on Computer Vision-Part IV*, 776–792. Retrieved from http://dl.acm.org/citation.cfm?id=645318.649268

Shepard, R. N. (1987). Toward a universal law of generalization for psychological science. *Science (New York, N.Y.)*, *237*(4820), 1317–1323. https://doi.org/10.1126/science.3629243

Simonyan, K., & Zisserman, A. (2014a). Very deep convolutional networks for large-scale image recognition. *ArXiv Preprint ArXiv:1409.1556*.

Simonyan, K., & Zisserman, A. (2014b). Very deep convolutional networks for large-scale visual recognition. Retrieved March 6, 2019, from http://www.robots.ox.ac.uk/~vgg/research/very_deep/

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2015). Going deeper with convolutions. *Computer Vision and Pattern Recognition*. Retrieved from http://arxiv.org/abs/1409.4842

Tang, Y., Tao, G., & Nan, X. (2015). *dml: Distance metric learning in R*. Retrieved from https://cran.r-project.org/package=dml

Ter Braak, C. J. F. (2006). A Markov Chain Monte Carlo version of the genetic algorithm Differential Evolution: Easy Bayesian computing for real parameter spaces. *Statistics and Computing*, *16*(3), 239–249. https://doi.org/10.1007/s11222-006-8769-1

Turner, B. M., Sederberg, P. B., Brown, S. D., & Steyvers, M. (2013). A method for efficiently

sampling from distributions with correlated dimensions. *Psychological Methods*, *18*(3), 368–384. https://doi.org/10.1037/a0032222

Usher, M., & McClelland, J. L. (2001). The time course of perceptual choice: The leaky, competing accumulator model. *Psychological Review*, *108*(3), 550–592. https://doi.org/10.1037/0033-295X.108.3.550

van der Maaten, L. J. P. (2014). Accelerating t-SNE using tree-based algorithms. *Journal of Machine Learning Research*, *15*, 3221–3245.

van der Maaten, L. J. P., & Hinton, G. E. (2008). Visualizing high-dimensional data using t-SNE. *Journal of Machine Learning Research*, *9*, 2579–2605.

Wong, A. C.-N., Palmeri, T. J., & Gauthier, I. (2009). Conditions for facelike expertise with objects: Becoming a Ziggerin expert—but which type? *Psychological Science*, *20*(9), 1108–1117. https://doi.org/10.1111/j.1467-9280.2009.02430.x

Wong, A. C.-N., Palmeri, T. J., Rogers, B. P., Gore, J. C., & Gauthier, I. (2009). Beyond shape: How you learn about objects affects how they are represented in visual cortex. *PloS One*, *4*(12), e8405.

Xie, W., Lewis, P. O., Fan, Y., Kuo, L., & Chen, M. H. (2011). Improving marginal likelihood estimation for bayesian phylogenetic model selection. *Systematic Biology*, *60*(2), 150–160. https://doi.org/10.1093/sysbio/syq085

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

Yu, C.-P., Liu, H., Samaras, D., & Zelinsky, G. J. (2019). Modelling attention control using a convolutional neural network designed after the ventral visual pathway. *Visual Cognition*,

0(0), 1–19. https://doi.org/10.1080/13506285.2019.1661927

Appendix: Bayesian Implementation

CNN model. For subject *p*, presented with stimulus pair (*i*,*j*) in category *k*, choice response time pairs $\boldsymbol{RT}$, were assumed to be distributed according to the LBA:

$$\boldsymbol{RT}_{pijk} \sim LBA(A_{pk}, k_{pk}, v_{s,pijk}^{raw}, v_{d,pijk}^{raw}, \tau_{pk}, S_{s,pk}, S_{d,pk}),$$

where S_s and S_d are the standard deviations of the drift rates of the "same" and "different" accumulators, respectively, and $v_{s,pijk}^{raw}$ is defined by Equation 4 and $v_{d,pijk}^{raw} = 1 - v_{s,pijk}^{raw}$. The prior on each subject-level parameter followed a truncated normal $TN(a, b, c, d)$ where a is the mean, b is the standard deviation, c is the lower bound and d is the upper bound. For all truncated normal distributions, $c = 0$ and $d = \infty$, so we omit these from the notation for brevity. The subject-level priors for the LBA were the following:

$$A_{pk} \sim TN(\mu_{pk}^{A}, \sigma_{pk}^{A})$$

$$\kappa_{pk} \sim TN(\mu_{pk}^{\kappa}, \sigma_{pk}^{\kappa})$$

$$\tau_{pk} \sim TN(\mu_{pk}^{\tau}, \sigma_{pk}^{\tau})$$

$$S_{s,pk} \sim TN(\mu_{pk}^{S_s}, \sigma_{s,pk}^{S_s})$$

$$S_{d,pk} \sim TN(\mu_{pk}^{S_d}, \sigma_{pk}^{S_d}).$$

Priors on subject-level parameters for the cognitive level, used to transform the CNN distances into mean drift rates for the LBA, also followed truncated normal distributions:

$$c_{pk} \sim TN(\mu_{pk}^{c}, \sigma_{pk}^{c})$$

$$\beta_{pk} \sim TN(\mu_{pk}^{\beta}, \sigma_{pk}^{\beta})$$

Group-level priors on the means and variances were as follows and were loosely based on priors used in prior modeling work with the LBA (e.g., Turner et al., 2013):

$$\mu_{pk}^{\kappa}, \sigma_{pk}^{\kappa} \sim TN(1.4,1.4)$$

$$\mu_{pk}^{\tau}, \sigma_{pk}^{\tau} \sim TN(.3, .3)$$

$$\mu_{pk}^{A}, \sigma_{pk}^{A}, \mu_{pk}^{S_s}, \sigma_{pk}^{S_s}, \mu_{pk}^{S_d}, \sigma_{pk}^{S_d} \sim TN(1,1).$$

The group-level priors on the means and variances of the parameters at the cognitive level also followed truncated normal distributions:

$$\mu_{pk}^{c}, \sigma_{pk}^{c}, \mu_{pk}^{\beta}, \sigma_{pk}^{\beta} \sim TN(1,1).$$

Recall c scales the CNN distance. Because we were uncertain as to the direction or magnitude of the scaling, we chose a truncated normal centered at 1 with a standard deviation of 1 for the group-level priors. This reflects our initial belief to leave the CNN distances relatively unscaled. The group-level priors on criterion noise mean and standard deviations, μ_{pk}^{β} and σ_{pk}^{β}, were based on prior modeling work (Annis & Palmeri, 2019).

Free model. The free model allowed the mean of the drift rates for the "same" accumulator for subject *p* given category *k*, to freely vary across conditions object identity, q (same vs. different), and viewpoint, h (same vs. different), conditions instead of being constrained by CNN- or pixel-based representations. For the free model, choice response time pairs were distributed thusly:

$$\boldsymbol{RT}_{pqhk} \sim LBA\left(A_{pk}, \kappa_{pk}, v_{s,pqhk}, v_{d,pqhk}, \tau_{pk}, S_{s,pk}, S_{d,pk}\right),$$

where $v_{d,pqhk} = 1 - v_{s,pqhk}$. All priors were the same as those described in the CNN-LBA + LBA model with an additional prior placed on v_s:

$$v_{s,pqhk} \sim TN(\mu_{pqhk}^{v_s}, \sigma_{pqhk}^{v_s}),$$

where

$$\mu_{pqhk}^{v_s}, \sigma_{pqhk}^{v_s} \sim TN(3,3).$$

Pixel-Based model. For the pixel-based model, choice response time pairs, $\boldsymbol{RT}$, were assumed to be distributed according to the LBA:

$$\boldsymbol{RT}_{pijk} \sim LBA\left(A_{pk}, k_{pk}, v_{s,pijk}^{pixelRaw}, \tau_{pk}, S_{s_{pk}}, S_{d_{pk}}\right),$$

where $v_{s,pijk}^{pixelRaw}$ is given by Equation 4 and $v_{d,pijk}^{pixelRaw} = 1 - v_{s,pijk}^{pixelRaw}$. All priors are identical to those found in the CNN-LBA model.

Appendix: Relevant Component Analysis

The RCA algorithm (Shental et al., 2002) learns an optimal data transformation that leads to minimizing the total variance within chunklets. The algorithm is straightforward, only requiring the inverse of the chunklet covariance matrix. Formally, given a dataset $X = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\}$, we can define n chunklets, $Y_j = \{\boldsymbol{x}_{j,1}, .. \boldsymbol{x}_{j,n_j}\}$, where $j = \{1, \dots, n\}$. Note that these chunklets are user-defined. In our case, each chunklet contains the two novel objects of the same identity with different viewpoints. For example, chunklet 1 might be: {"Greeble-1 Viewpoint-A", "Greeble-1 Viewpoint-B"}. The within-chunklet covariance matrix is given by the following:

$$\hat{\dot{\mathbf{C}}} = \frac{1}{N}\sum_{j=1}^{n}\sum_{i=1}^{n_j}(\boldsymbol{x}_{ji} - \boldsymbol{m}_j)(\boldsymbol{x}_{ji} - \boldsymbol{m}_j)^{\mathrm{T}}, \quad (1)$$

where $\boldsymbol{m}_j$ is the mean of chunklet C_j. RCA then computes the optimal linear transformation by taking the inverse of $\hat{\mathbf{C}}$, which can then be used in the Mahalanobis distance:

$$d_M(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{x}_i - \mathbf{x}_j)}. \quad (2)$$

This leads to a data representation whose relevant dimensions are weighed more heavily. When novel object pairs that are from different viewpoints were used as chunklets, we found the RCA transformation lead to a space in which those pairs are located more closely. We used the RCA algorithm from the `dml` R package (Tang, Tao, & Nan, 2015).

Appendix: Full Metric Learning Modeling Results

Learning	Dim.	Metric	Dim.	BIC	AIC	Subject Level		Item Level	
						Acc. *r*	RT *r*	Acc. *r*	RT *r*
Fully Sup.	MDS	off-diag.	15	167741	90985	0.86	0.90	0.79	0.91
Fully Sup.	MDS	diagonal	20	168401	95236	0.86	0.90	0.41	0.91
Fully Sup.	PCA	diagonal	75	168535	92589	0.86	0.90	0.56	0.91
Fully Sup.	PCA	diagonal	50	168871	94189	0.85	0.91	0.52	0.91
Fully Sup.	PCA	diagonal	35	169630	95706	0.85	0.90	0.49	0.91
Fully Sup.	MDS	diagonal	35	170558	96635	0.87	0.90	0.40	0.91
Fully Sup.	MDS	diagonal	15	171704	98792	0.85	0.90	0.37	0.91
Fully Sup.	MDS	off-diag.	10	172733	98759	0.86	0.90	0.59	0.91
Fully Sup.	MDS	diagonal	10	173026	100367	0.87	0.91	0.31	0.91
Fully Sup.	MDS	diagonal	50	173704	99022	0.85	0.90	0.39	0.91
Fully Sup.	PCA	off-diag.	15	174303	97547	0.87	0.90	0.74	0.91
Fully Sup.	MDS	off-diag.	5	176250	103793	0.84	0.90	0.29	0.91
Fully Sup.	MDS	full	15	176398	98176	0.85	0.90	0.79	0.91
Fully Sup.	MDS	full	10	176436	101501	0.87	0.90	0.50	0.91
Fully Sup.	PCA	diagonal	20	176809	103645	0.85	0.90	0.43	0.91
Fully Sup.	PCA	off-diag.	10	178870	104896	0.85	0.90	0.55	0.91
Fully Sup.	MDS	full	5	180358	107447	0.87	0.90	0.32	0.91
Fully Sup.	MDS	diagonal	5	180384	107978	0.86	0.91	0.24	0.91
Fully Sup.	MDS	diagonal	75	182026	106080	0.85	0.90	0.40	0.91
Fully Sup.	PCA	diagonal	15	182106	109194	0.85	0.90	0.37	0.91
Fully Sup.	PCA	full	5	186787	113876	0.85	0.90	0.33	0.91
Fully Sup.	PCA	diagonal	10	186864	114205	0.84	0.90	0.28	0.91
Weakly Sup.	MDS	full	15	187516	115362	0.87	0.90	0.29	0.91
Weakly Sup.	PCA	full	10	189034	116881	0.84	0.90	0.25	0.91
Weakly Sup.	MDS	full	10	189967	117814	0.85	0.90	0.28	0.91
Fully Sup.	PCA	diagonal	5	190133	117727	0.84	0.90	0.26	0.91
Weakly Sup.	PCA	full	15	191200	119047	0.83	0.90	0.29	0.90
Weakly Sup.	MDS	full	5	192810	120657	0.84	0.89	0.20	0.91
Weakly Sup.	PCA	full	20	193065	120912	0.83	0.90	0.25	0.90
Fully Sup.	PCA	full	10	194580	119646	0.84	0.90	0.55	0.91
Weakly Sup.	MDS	full	20	195063	122909	0.84	0.89	0.25	0.90
Fully Sup.	PCA	off-diag.	5	198216	125760	0.83	0.90	0.32	0.90
Weakly Sup.	PCA	full	5	200031	127878	0.84	0.90	0.25	0.91
Weakly Sup.	MDS	full	35	202892	130739	0.81	0.89	0.16	0.90
Weakly Sup.	PCA	full	35	205617	133464	0.81	0.89	0.17	0.90
Fully Sup.	PCA	full	15	209054	130831	0.79	0.89	0.62	0.90
Fully Sup.	RAW	NA	0	212812	140659	0.79	0.89	0.24	0.90
Weakly Sup.	PCA	full	50	219681	147528	0.83	0.88	0.14	0.90
Weakly Sup.	MDS	full	50	222688	150535	0.80	0.88	0.17	0.89