Vision Research 167 (2020) 87–99

Contents lists available at ScienceDirect

Vision Research

journal homepage: www.elsevier.com/locate/visres

Can machine learning account for human visual object shape similarity judgments?☆

Joseph Scott German[a], Robert A. Jacobs[b]

[a] Department of Brain and Cognitive Sciences, University of Rochester, Rochester, NY 14627, United States
[b] Department of Brain and Cognitive Sciences, University of Rochester, Rochester, NY 14627, United States

ABSTRACT

We describe and analyze the performance of metric learning systems, including deep neural networks (DNNs), on a new dataset of human visual object shape similarity judgments of naturalistic, part-based objects known as "Fribbles". In contrast to previous studies which asked participants to judge similarity when objects or scenes were rendered from a single viewpoint, we rendered Fribbles from multiple viewpoints and asked participants to judge shape similarity in a viewpoint-invariant manner. Metrics trained using pixel-based or DNN-based representations fail to explain our experimental data, but a metric trained with a viewpoint-invariant, part-based representation produces a good fit. We also find that although neural networks can learn to extract the part-based representation—and therefore should be capable of learning to model our data—networks trained with a "triplet loss" function based on similarity judgments do not perform well. We analyze this failure, providing a mathematical description of the relationship between the metric learning objective function and the triplet loss function. The poor performance of neural networks appears to be due to the nonconvexity of the optimization problem in network weight space. We conclude that viewpoint insensitivity is a critical aspect of human visual shape perception, and that neural network and other machine learning methods will need to learn viewpoint-insensitive representations in order to account for people's visual object shape similarity judgments.

1. Introduction

Judging the similarity of objects is a component of many real-world visual tasks. A person searching for a friend in a crowd uses visual cues to judge which face most closely resembles that of the friend. A mycologist (i.e., an expert in the study of fungi) deciding whether or not to eat a particular mushroom uses visual cues to judge whether the mushroom is more similar to edible or poisonous mushrooms previously encountered. A radiologist examining a suspicious portion of a mammogram uses visual cues to judge whether the portion is similar to previously encountered images of tumors. Clearly, achieving greater insight into the nature of visual similarity judgments will enhance our understanding of the mental processes underlying performance on many important perceptual and cognitive tasks (Edelman, 1998; Edelman & Shahbazi, 2012).

It is not only biological organisms that make visual similarity judgments. Indeed, the acquisition and use of visual similarity metrics is often integral to the processing of computer vision, robotic, and other artificial intelligence systems. Within the field of machine learning (ML), there has been a recent surge of interest in statistical and deep neural network (DNN) approaches to the acquisition of visual similarity metrics (Bellet, Habrard, & Sebban, 2014; Kulis, 2012), as well as the use of metrics for learning hidden or latent representations (Chopra, Hadsell, & LeCun, 2005; Schroff, Kalenichenko, & Philbin, 2015).

This paper reports the results of a behavioral experiment in which human participants were asked to make shape similarity judgments about visual objects in a viewpoint-invariant manner. It also reports the results of our efforts to account for the patterns in participants' responses using a variety of approaches from the ML literature. The research has two related goals. One goal is to better understand people's visual object shape similarity judgments. The other is to better understand the strengths and shortcomings of current ML approaches to visual similarity by asking whether these approaches can successfully account for patterns in our experimental data. Although DNNs and other machine learning approaches have received much fanfare in recent years, our results demonstrate that they often fail to judge similarity as people do. Specifically, the results suggest that they tend to learn representations that are overly sensitive to viewpoint-dependent, 2-D image features, even when they could, in principle, learn representations that are viewpoint-independent and based on 3-D shape properties. In contrast, people judge visual object shape similarity in a more viewpoint-insensitive manner based on 3-D shape features (Erdogan & Jacobs, 2017).

2. Experiment

Previous researchers have conducted behavioral experiments in which human participants judged the similarity of visual and visual-

☆ The first author was supported by an NSF NRT graduate training grant (NRT-1449828). This work was also supported by an NSF research grant (BCS-1824737).

E-mail addresses: jgerman6@ur.rochester.edu (J.S. German), rjacobs@ur.rochester.edu (R.A. Jacobs).

https://doi.org/10.1016/j.visres.2019.12.001
Received 28 May 2019; Received in revised form 22 October 2019; Accepted 12 December 2019

haptic objects on the basis of their shapes. For example, in a set of studies by Wallraven, Bülthoff, and colleagues (Cooke, Jäkel, Wallraven, & Bülthoff, 2007; Gaißert, Bülthoff, & Wallraven, 2011; Gaißert & Wallraven, 2012; Gaißert, Wallraven, & Bülthoff, 2010), participants provided similarity judgments for different sets of objects, both artificial and natural, in visual, haptic, and visual-haptic conditions. Using multidimensional scaling (Cox & Cox, 1994) to analyze the experimental data, it was found that participants' similarity ratings were similar in all three sensory conditions, suggesting that these ratings were based on shared mental representations.

Erdogan and Jacobs (2017) and Erdogan, Yildirim, and Jacobs (2015) asked participants to rate object shape similarity in a visual condition or in visual, haptic, and visual-haptic conditions, respectively. These authors modeled their similarity data using a system combining a symbolic representation of object parts and their possible combinations with a Bayesian inference algorithm to infer 3-D, part-based representations of objects. Erdogan and Jacobs (2017) found that this system accounted for participants' judgments better than DNNs and other methods for representing object shape. The disadvantages of this system include a requirement to specify a set of possible parts and a set of rules for how these parts can be combined, and the fact that it is very computationally expensive.

Conceptually, the procedure of the experiment reported here is similar to that of Erdogan and Jacobs (2017), though the current experiment used different stimuli and collected data from many more participants. The analyses of the experimental data are also quite different.

2.1. Participants

The experimental study was approved by the Research Subjects Review Board at the University of Rochester. Ninety-nine participants took part in the experiment over the world wide web via the Amazon Mechanical Turk (MTurk) crowd-sourcing marketplace. Interfacing with MTurk was facilitated through the use of the psiTurk programming platform (Gureckis et al., 2016). psiTurk was configured so that only individuals based in the United States could participate in the experiment. Participants stated that they were at least 18 years old. It took approximately 20 minutes to complete the experiment, and each participant received \$2.50 for their participation.

2.2. Stimuli

The experiment used novel objects known as "Fribbles" (Barry, Griffith, Rossi, & Hermans, 2014; Hayward & Williams, 2000; Tarr, 2003; Williams, 1997). These are three-dimensional, multipart, naturalistic objects. Novel objects were used so that participants would not have semantic associations with objects, implying that participants' similarity judgments should be based on perceptual features, instead of semantic ones.

All objects used in the experiment came from a single Fribble "species", and thus contained a common part known as its main body. In addition to a main body, each object had four slots or locations, with one of three possible parts attached at each location. Consequently, the experiment used 81 objects (4 locations with 3 possible parts per location for $3^4 = 81$ objects). Objects were visually rendered from 14 viewpoints (see Fig. 1 for sample images; see Fig. 2 for images of all possible object parts). Thus, the experiment used 1134 images (81 objects $\times$ 14 viewpoints $= 1134$ images).

2.3. Procedure

On each experimental trial, participants viewed a display containing three Fribble objects: a "target" object and two "probe" objects. They were asked to select which of the two probes was most similar to the target. Importantly, participants were instructed to base their similarity judgments on 3-D object shape (irrespective of viewpoint). Each participant completed 175 trials, 16 of which were "catch" trials where one of the probes was identical to the target, albeit possibly rendered from a different viewpoint. As described below, these catch trials were used to determine the quality of participants' responses. Each participant encountered the same catch trials at the same, randomly-chosen points in the experiment. In total, 17,325 similarity judgments were collected.

3. Machine learning approaches

We analyzed our experimental data using two prominent ML approaches, metric learning (see also Xu, Zhu, and Rogers (2012)) and DNNs.

3.1. Metric learning

In the machine learning literature, judging the similarity of objects is usually considered a problem of metric learning (Bellet et al., 2014; Kulis, 2012). Metric learning, as its name suggests, attempts to learn a metric or distance function over objects such that similar objects are close together and dissimilar objects are farther apart. Which objects should be considered similar, and thus close, and which ones should be considered dissimilar, and thus far apart, can be based on supervision derived from human judgments.

The most common approach to metric learning is for the learned distance function to take the form of a Mahalanobis distance:

$$d(x_1, x_2) = (x_1 - x_2)^T A (x_1 - x_2) \tag{1}$$

where $d(x_1, x_2)$ is the (square of) the distance between vectors x_1 and x_2, and A, the Mahalanobis matrix, is a positive semidefinite matrix (i.e., $x^T A x \geqslant 0$ for all $x \in \mathbb{R}^n$). Finding matrix A is framed as an optimization problem typically taking the form:

$$\min_{A \in \mathbb{M}_{PSD}} \; r(A) \text{ subject to } \; \forall\, i,\; c_i(X^T A X) \leqslant 0. \tag{2}$$

In this equation, X is a matrix containing the stimuli or data items, $\{c_i(X^T A X)\}$ is a set containing the training constraints on $X^T A X$ by which the supervision is administered, $r(A)$ is a regularizer, and $\mathbb{M}_{PSD}$ is the set of positive semidefinite matrices of appropriate size. The regularizer $r(A)$ is a function of A used to avoid solutions to the optimization problem that are overfit to the training constraints, meaning solutions that are overly based on both the "signal" and the "noise" in the constraints. It is important to avoid such solutions because they are unlikely to generalize well to novel data items or constraints. The positive semidefinite restriction $A \in \mathbb{M}_{PSD}$ both makes intuitive sense (as the distance between two objects should not be negative) and can aid training by making the optimization problem convex.[1] If this restriction makes the optimization problem convex, then the problem can more easily be addressed through semidefinite programming (Vandenberghe & Boyd, 1996).

In this paper, we implemented the training constraints as follows. If a participant in our experiment judged the target stimulus x_1 as more similar to the probe x_2 than to the other probe x_3, then $d(x_1, x_2)$ was constrained to be less than $d(x_1, x_3)$. A "margin" m (set to 1) was also used (Hastie, Tibshirani, & Friedman, 2009), yielding the final constraint $d(x_1, x_2) < d(x_1, x_3) - m$. Because participants in our experiment provided 17,325 similarity judgments, our optimization problem contained 17,325 data items or training constraints.

As explained in Appendix A, one rarely attempts to directly solve the optimization problem in Eq. (2). Instead, the problem is transformed in two ways. First, because it is unlikely that there exists a solution satisfying **all** constraints, it is common to add "slack variables" to

[1] Convex problems are attractive because they have one (global) solution and because they are often relatively easy to solve (Vandenberghe & Boyd, 1996).

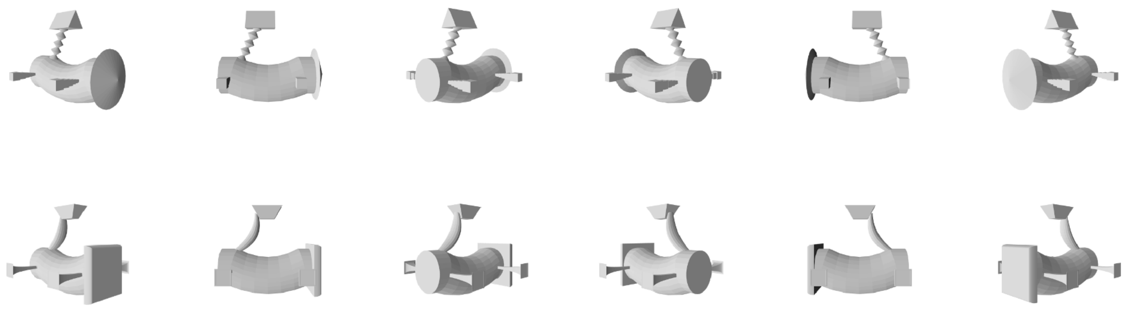


Fig. 1. Examples of Fribble stimuli. Top and bottom rows depict two Fribbles at a variety of viewpoints.

the constraints, and to attempt to minimize those as well (Hastie et al., 2009). Second, it is also common to re-write this constrained optimization problem as an unconstrained problem. In our simulations, optimization was performed by finding a solution to the unconstrained problem using stochastic gradient descent.

In the ML literature, a number of algorithms are often used to solve the unconstrained optimization problem. Four common algorithms were used in our simulations, referred to as the freeform, decomposition, SJ, and LMNN algorithms. The details of these algorithms are provided in Appendix A. It is not the goal of this paper to compare the performance of one algorithm with another, and thus the details of the algorithms are not essential for our purposes. Instead, our goal is to evaluate whether any state-of-the-art metric learning procedure can successfully account for human visual object shape similarity judgments.

3.2. Deep neural networks

DNNs are state-of-the-art artificial intelligence systems that have demonstrated impressive performance in a wide range of domains, including visual perception, speech recognition, text-to-text language translation, and product recommendation. In brief, neural networks consist of interconnected units (LeCun, Bengio, & Hinton, 2015). Some of these units are designated as input units, others are “hidden” units, and still others are output units. A network’s goal is to map patterns of input unit “activations” to target (i.e., desired) patterns of output unit activations. For instance, a network might map patterns representing visual images (e.g., images of vehicles) to patterns representing category labels (e.g., a vehicle might be a car, truck, or bus). Typically, input units connect to one or more layers of hidden units which, in turn, connect to output units.

The power of neural networks is based on the fact that they are capable of learning and generalization. Networks learn by adapting the values of their units’ parameters or weights. Learning is typically supervised, meaning that a “teacher” provides the target output activation pattern for each input activation pattern. During training, a network’s weights are modified to minimize its error, or difference between the target output pattern for each training data item and its actual output pattern. The hope is that following training, the network is capable of generalization to unseen data, meaning that in addition to producing the target output pattern for each input pattern in the training set, it can also produce approximately correct output patterns for novel input patterns that are similar to the training set’s input patterns.

Over the past several decades, researchers in the fields of cognitive science and neuroscience have used neural networks to provide insights into many aspects of human perception and cognition. Despite this history of success, the reasons why these networks often provide useful accounts of mental and neural processing are poorly understood. For instance, there are many types of networks (differing in terms of types of units, patterns of connectivity, training procedures, and many other factors), but researchers do not have a good understanding as to which types provide better versus worse accounts of human behavior. Similarly, many neural networks lack important biological detail, resembling biological neural networks only in seemingly coarse ways. Despite this, several researchers have recently argued that these networks provide useful insights into neural processing, particularly within the visual system (Kriegeskorte, 2015; Wenliang & Seitz, 2018; Yamins & DiCarlo, 2016).

The simulations reported in this paper were conducted using the Keras neural network library for Python (Chollet, 2017). In one set of simulations, we used a pre-trained network, namely VGG-16 (Simonyan & Zisserman, 2015), provided in Keras. In other simulations, we trained and tested our own networks.

4. Direct modeling of human similarity judgments

This section reports the results of our attempts to use metric learning to directly model our experimental participants’ similarity judgments. In these attempts, visual stimuli were coded using either a pixel-based representation, a latent representation acquired by a DNN, or a viewpoint-invariant, part-based representation.

Below we include results from multiple algorithms. However, the primary intent of this paper is not to compare the algorithms to assess which one models our data best. Rather, results from multiple algorithms are reported to confirm that patterns in results are not idiosyncratic properties of a single algorithm. In addition, we evaluated algorithms on multiple subsets of data items, including low-quality, medium-quality, and high-quality data items. This was done to assess algorithms’ performances under the best possible conditions, presumably when trained with high-quality data, and to check that performances are not due to “noise” arising from large individual differences in participants’ responses.

4.1. Pixel-based representation

We began by using a naïve, pixel-based representation of the visual stimuli. We converted the actual 480×480 images used in the experiment to grayscale,[2] and represented each image as a vector of pixel values with 230,400 elements. Use of these vectors would entail learning a Mahalanobis matrix with over 53 billion elements, so for computational feasibility we mapped each vector to a vector with 100 elements via principle component analysis (PCA). This transformation preserved over 95% of the variance in the pixel values. Furthermore, we show later in the paper that all the information required to complete the task remains present in the low-dimensional representation (Fig. 9). Thus, metric learning required us to learn a 100×100 Mahalanobis matrix A.

To train and test metric learning models, we divided the data into training (three-fifths), validation (one-fifth) and test (one-fifth) sets. Models were trained on the training data using a variety of values for

[2] Although the images viewed by the human participants were color, they were essentially monochromatic, and converting them to grayscale did not appreciably alter their appearance.

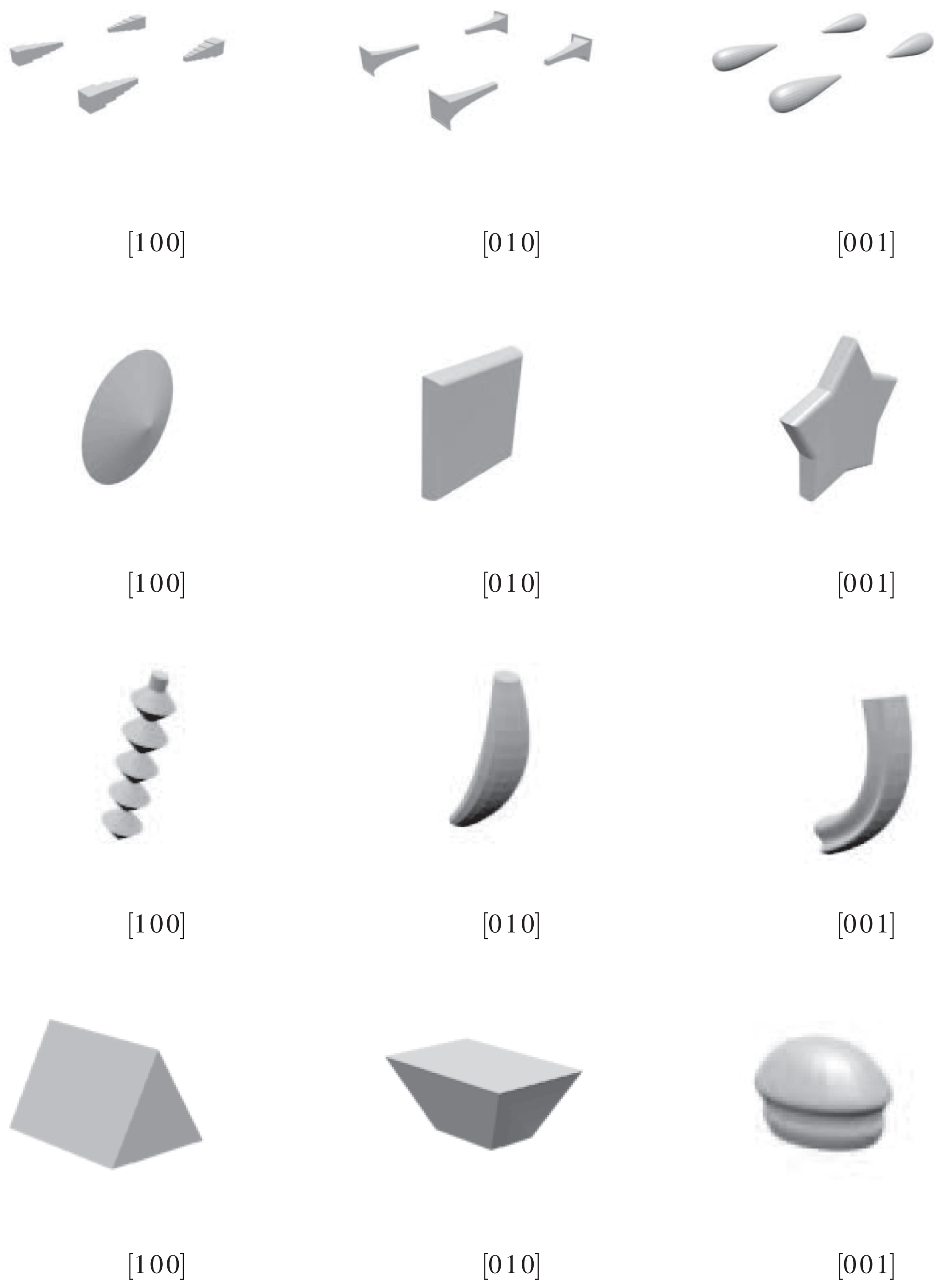


Fig. 2. Fribble parts used in the experiment (except the body which was the identical in all stimuli), along with their corresponding subvectors used in the part-based representation. A Fribble is constructed by combining one part from each row (plus the body), and the part-based representation is constructed by concatenating the associated subvectors, top row to bottom row. Parts are not shown to scale.

the hyperparameter C in Eq. (6) in Appendix A, ranging from 5 to 50 in increments of 5 (C controls the relative importance of minimizing the regularizer versus the "slack" variables).

The resulting models were then tested for their performance on the validation set. The best performing model was then tested on the test set, and the result reported. The results are shown in Fig. 3. In each graph, "accuracy" is plotted on the vertical axis where accuracy is the average proportion of participants' similarity judgments in a test set correctly reproduced by a model. The horizontal axis shows five different models, a baseline model in which the Mahalanobis matrix A was set to a random matrix which was then projected onto the positive semidefinite cone, and four models based on the four metric learning algorithms mentioned above (and described in Appendix A). The four graphs in the figure correspond to four subsets of data items. The top-left graph shows the results using all data items. In addition, we classified the experimental participants based on the number of catch trials on which they correctly responded. Data from participants with 11 or fewer catch trials correct are considered low-quality data (top-right

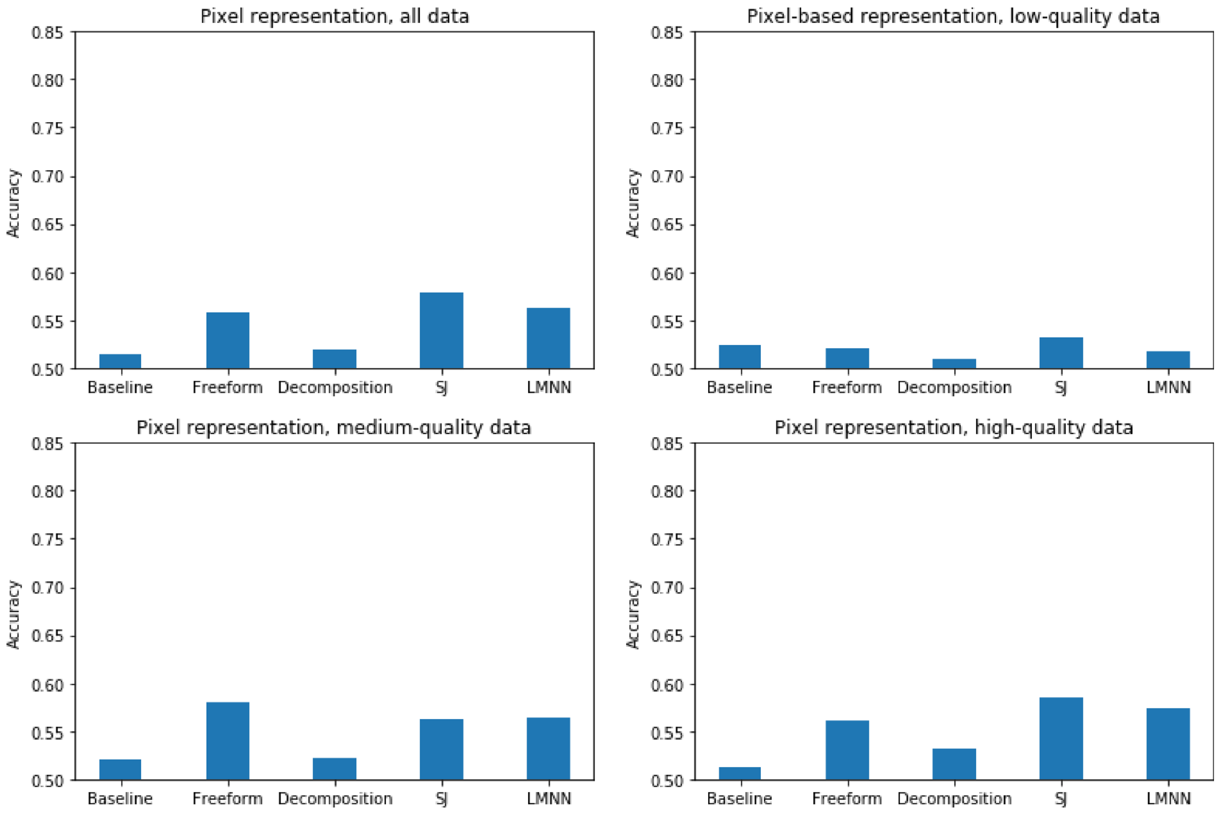


Fig. 3. Model accuracies on test items using pixel-based image representations. Metric learning models were evaluated using all data items (top left), as well as subsets consisting of low quality (top right), medium quality (bottom left), and high quality (bottom right) data items.

graph; 25 participants, 4375 judgments), data from participants with 12–14 catch trials correct are medium-quality data (bottom-left graph; 41 participants, 7175 judgments), and data from participants with 15–16 catch trials correct are high-quality data (bottom-right graph; 33 participants, 5775 judgments). The algorithms iterated over the data at each quality level such that there were roughly the same effective number of examples in each.

In brief, none of the models showed good performance. This result is not surprising considering the naïve nature of the pixel-based representation. The best-performing model was the SJ model, and even with high-quality data it averaged only around 62% accuracy.[3]

The models' poor performances suggest that the pixel-based representation is inadequate for modeling participants' similarity judgments. Next, we evaluate a representation derived from a state-of-the-art DNN.

4.2. DNN-based representation

We next tried using hidden or latent representations acquired from the final feature layer of a state-of-the-art DNN, VGG-16 (Simonyan & Zisserman, 2015), which had been pre-trained on the ImageNet classification task. We used the version of VGG-16 available in the Keras neural network library (Chollet, 2017). We used VGG because it shows good performance, having secured the first and second places in the 2014 ImageNet Challenge localization and classification tracks. In addition, it has been reported that its representations capture important aspects of people's image similarity ratings (albeit not their visual object shape similarity ratings; see Discussion section), often better than alternative DNNs (Peterson, Abbott, & Griffiths, 2018).

Although VGG-16 was not trained to model similarity judgments, we believe that the use of its hidden representations in our application is reasonable. Using (portions of) DNNs trained on one task for feature extraction on another task often yields good results (Cireşan, Meier, & Schmidhuber, 2012; Azizpour, Razavian, Sullivan, Maki, & Carlsson, 2015). Furthermore, we show later in the paper that the VGG-16 representation contains all the information necessary to complete the task (Fig. 9).

We resized the Fribble images to the input resolution of VGG-16 and used them as the network inputs. For each input, the activation values of the hidden units at the last max pooling layer of the network's convolutional base were extracted.[4] The output shape at this layer is $7 \times 7 \times 512$, meaning that there were 25,008 activation values per input. Use of feature vectors with 25,008 elements would entail learning a Mahalanobis matrix with over 600 million elements, so for computational feasibility we mapped each vector to a vector with 100 elements via PCA. This transformation preserved over 95% of the variance in the feature values. Thus, metric learning required us to learn a 100×100 Mahalanobis matrix A.

The results are summarized in Fig. 4. The performance of the learned metrics is only slightly better than the performance of metrics trained on the pixel values. It appears that the representations learned by VGG-16 are also not adequate for modeling participants' similarity judgments.

Before concluding that neural networks are incapable of modeling

[3] Considering that, as a semidefinite programming problem, the loss function is convex for most of these algorithms (the loss function is not convex for the decomposition algorithm, as will be discussed later), one might be surprised that the algorithms did not converge to the same performance. There are a few possible reasons for this. First, although the loss functions were all convex, the algorithms had different loss functions and were each optimizing for different things (see Appendix A). Second, we used stochastic gradient descent, so the true loss function was only approximated at each step.

[4] We also conducted similar tests using the output of the other max pooling layers; the results were not substantially different.

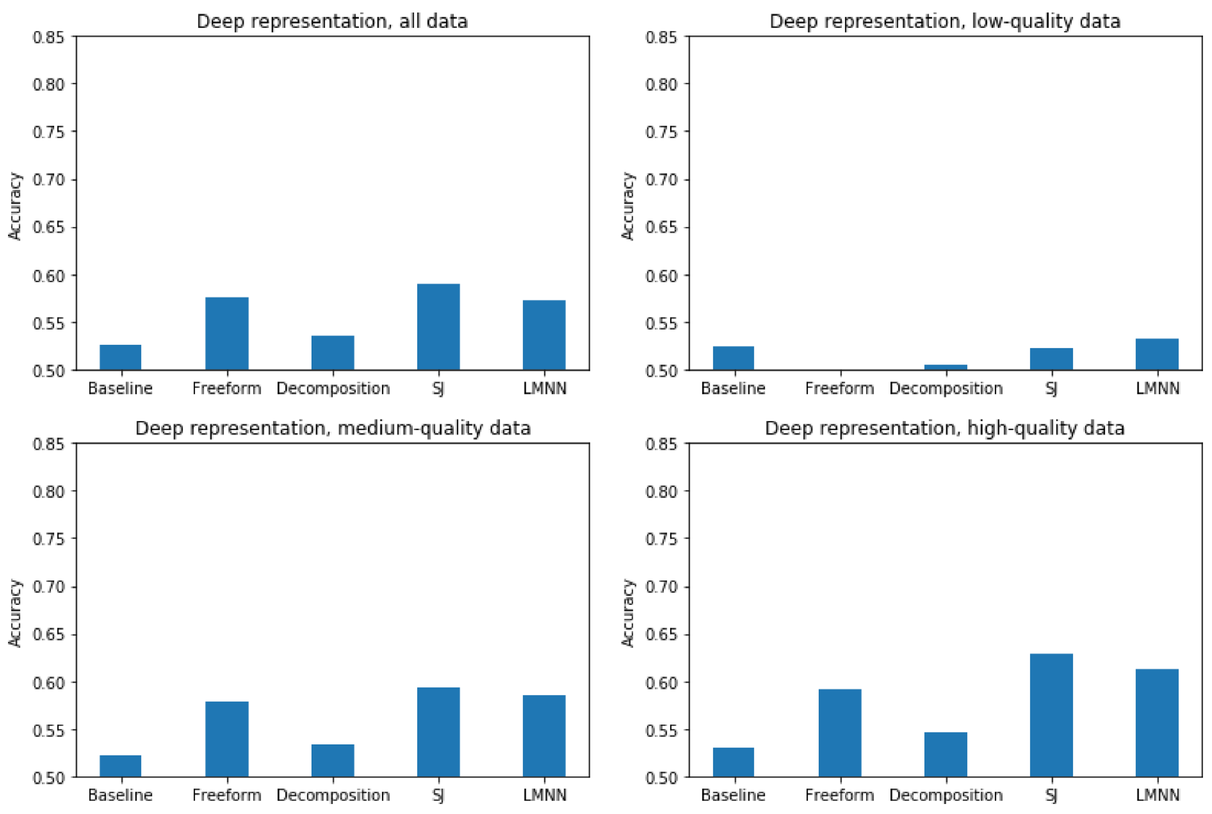


Fig. 4. Model accuracies on test items using DNN-based image representations. Metric learning models were evaluated using all data items (top left), as well as subsets consisting of low quality (top right), medium quality (bottom left), and high quality (bottom right) data items.

human visual object shape similarity data, there are two caveats worth noting. The first is that although the VGG-16 representation is not itself adequate, it may be possible to extract an adequate representation from it. Second, VGG-16 was not trained on visual stimuli like ours, nor was it trained to perform a task like ours. These factors have not prevented other researchers from successfully using VGG-16 and other DNN representations in seemingly unrelated applications, but we must consider the possibility that a DNN could successfully model our similarity judgments given proper training. We will return to these issues later in the paper. For now, we turn our attention to a handcrafted representation that adds viewpoint-invariance and part-sensitivity.

4.3. Part-based representation

We also represented objects in a part-based manner using binary feature vectors. There were three features for each location on a Fribble where a part could appear (ignoring the base part that is common to all Fribbles). These features indicated which of three possible parts was at the corresponding location. Because there were three possible parts per location and there were four locations, a Fribble was represented by a vector with 12 elements (see Fig. 2). Critically, the part-based representation makes a strong assumption: images of Fribbles could be accurately segmented into object parts by our experimental participants and that these parts could be accurately identified. Due to the part-based nature of Fribbles and to the simplicity of the images, it seems very likely that this assumption is valid. If so, a part-based representation allowed participants to represent Fribbles in a viewpoint-insensitive manner.

Because the Mahalanobis matrix A was only 12×12, optimization algorithms ran quickly enough that we could use a cross-validation procedure (Hastie et al., 2009) for tuning hyperparameters (specifically, C in Eq. (6)) making use of training, validation, and test sets. We set aside a fifth of the data (the test data) and performed fourfold cross-validation[5] on the remaining data for a variety of values of C. The models produced with a given value of C that performed best in cross-validation were then tested on the held-out test data, and the average accuracy is reported.

Results are summarized in Fig. 5. When trained and tested on all data items (top-left graph), the freeform, SJ, and LMNN methods performed similarly, achieving results in the range of 71%–72% accuracy. As expected, performances were better with high-quality data (bottom-right graph; accuracies about 78%–79%) and worse with low-quality data (top-right graph; accuracies around 60%). Critically for our purposes, model predictions more closely matched participants' similarity judgments when models used this part-based representation (Fig. 5) than when they used pixel-based or DNN-based representations (Figs. 3 and 4, respectively).

The decomposition method performed worse than other methods. While semidefinite programming problems are convex in the Mahalanobis matrix A, the decomposition method does not directly optimize A, and instead optimizes B where $A = B^T B$. Semidefinite programming problems are not convex in B, and thus it is not surprising that the decomposition method performed poorly. The difficulty of obtaining good performance on nonconvex optimization problems will be important again later in the paper.

Fig. 6a visualizes a Mahalanobis matrix acquired by a part-based model using the LMNN optimization method with all data items. LMNN was chosen because it most clearly demonstrated tendencies present with all algorithms. We initialized these matrices with the identity matrix for clarity: although the optimization problem is convex, it is not strictly convex, so random initializations often produce similar but

[5] In n-fold cross-validation, data are randomly partitioned into n subsets, and a model is trained and validated or evaluated n times. Each time, a different subset serves as the validation set, while the remaining subsets serve as the training set.

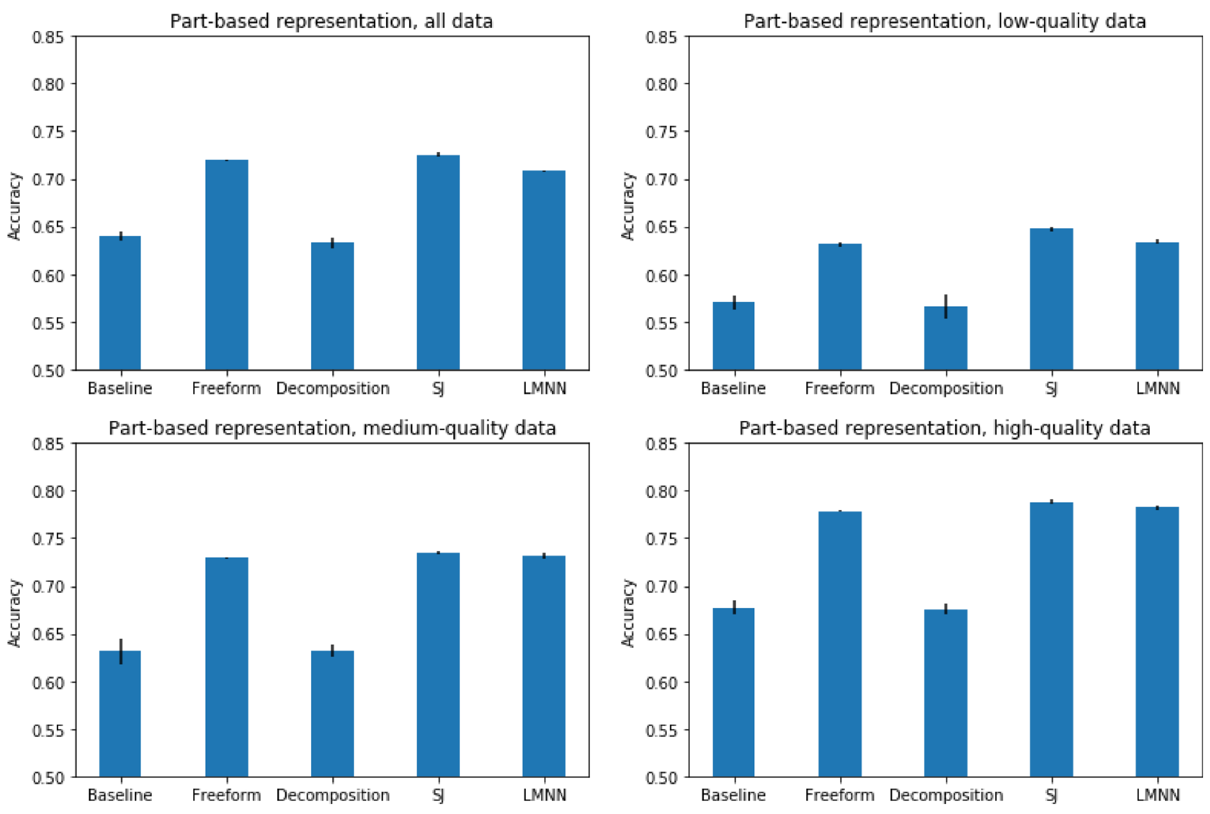


Fig. 5. Model accuracies on test items using part-based object representations. Metric learning models were evaluated using all data items (top left), as well as subsets consisting of low quality (top right), medium quality (bottom left), and high quality (bottom right) data items. Error bars plot the standard errors of the means. The relatively good performance of the Baseline condition (random matrix) seems to be largely due to its projection onto the positive semidefinite cone.

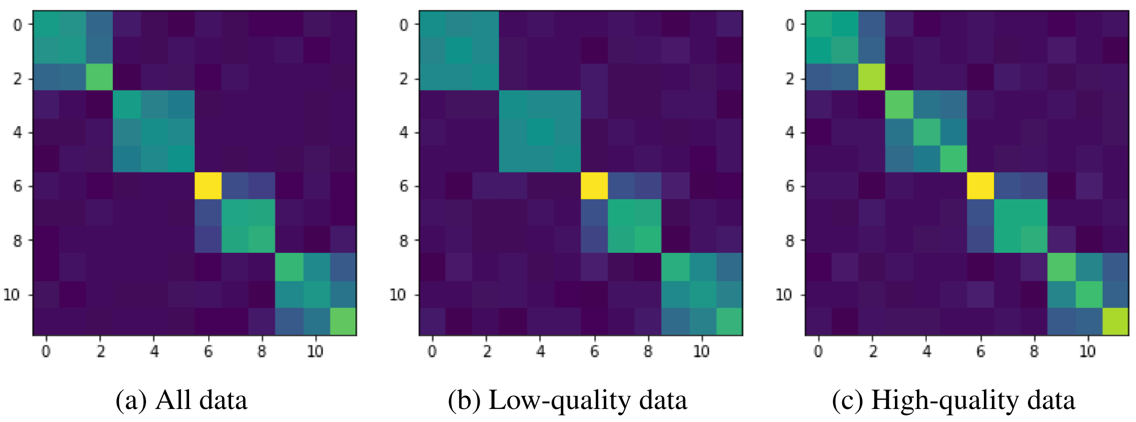
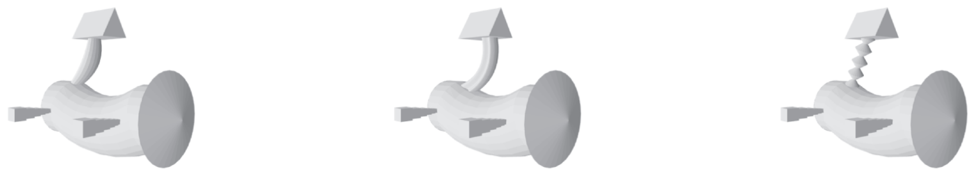


Fig. 6. Representative Mahalanobis matrices acquired by part-based models using the LMNN optimization method with all data items (Panel (a)), and with low-quality (Panel (b)) and high-quality (Panel (c)) data items. Brighter colors represent higher saliences (i.e., greater importance assigned to part differences). Results are similar using other methods.

noisier matrices. Brighter colors represent higher salience (i.e., greater importance assigned to a potential difference in part structures). Notably, the matrix is close to having a block-diagonal form with four blocks (recall that Fribbles have four locations at which parts were attached to the base part) where each block is a 3×3 sub-matrix (recall that one of three possible parts was attached at each location on each Fribble). It appears that the model has learned to predict participants' similarity judgments by comparing parts at corresponding locations (and not comparing parts at non-corresponding locations). Interestingly, the model does not regard all differences in parts as equally important. If a part on a target Fribble and a part at a corresponding location on a probe Fribble have very different 3-D shapes, then this part-difference is weighted heavily. However, if the two parts resemble each other in shape, then the model does not weight this part-difference as heavily.

This point can be illustrated by referring to Fig. 7. Suppose Fig. 7a depicts a target Fribble and Fig. 7b and c depict probe Fribbles. We focus on the Fribble part connecting the base part to the 3-D triangular part at the top of each Fribble (see Fig. 2, third row). This part is different on each Fribble, though this part's shape is more similar on the target and first probe (Fig. 7a and b) than on the target and second probe (Fig. 7a and c). The model has learned to weight the difference in parts less heavily in the former case than the latter. This is true despite the fact that differences in binary part-representations between the target and first probe and the target and second probe are identical (using, for example, Hamming distance). Presumably, this differential



(a) Target: [100100010100] (b) Probe 1: [100100001100] (c) Probe 2: [100100100100]

Fig. 7. Target (Panel (a)) and two probe Fribbles (Panels (b) and (c)), along with their part-based representations. Focusing on the Fribble part connecting the base part to the 3-D triangular part at the top of each Fribble, this part's shape is more similar on the target and first probe than on the target and second probe.

weighting of part-differences reflects the mental operations underlying participants' visual similarity judgments.

Interestingly, a slightly different pattern emerged when we examined Mahalanobis matrices based on low-quality and high-quality data. Low-quality data come from participants who performed poorly on experimental catch-trials, suggesting that these participants were relatively inattentive during the experiment. As illustrated in Fig. 6b, fewer part-differences were salient or noticeable for the least attentive participants, and thus these participants exhibited more extreme differential weighting than other participants (e.g., subtle part-differences were often ignored whereas large part-differences were assigned relatively large weights). These participants seem only to have noticed differences at certain locations with certain parts.

On the other hand, matrices learned using high-quality data (coming from participants who performed very well on experimental catch-trials, suggesting that these participants were especially attentive during the experiment) showed the opposite tendency. As illustrated in Fig. 6c, Mahalanobis matrices acquired on the basis of these participants' data more closely resembled the identity matrix than matrices acquired on the basis of all data or low-quality data. In other words, more part-differences were more salient or noticeable for the most attentive participants, and thus these participants exhibited less extreme differential weighting than other participants (e.g., nearly all part-differences (subtle or obvious) were noticed and weighted).

4.4. *Interim summary*

In summary, the results reported in this section indicate that metric learning with either pixel-based or DNN-based representations fail to account for the experimental participants' similarity judgments, whereas metric learning with a viewpoint-invariant, part-based representation is relatively successful at accounting for these judgments (see Fig. 8 for a summary of the results so far). To us, these results are surprising, particularly the poor performance of the DNN-based representation. After all, DNNs are state-of-the-art artificial intelligence systems whose visual object categorization performance rivals (or sometimes exceeds) that of humans. Moreover, the images used in our experiment and simulations are simple when compared with the complex images often used to train and test DNNs. Given these factors, why does the DNN-based representation perform so poorly? In the remainder of this paper, we explore this question.

5. Learning part-based representations from pixel-based or DNN-based representations

A good account of participants' similarity judgments can readily be provided on the basis of a viewpoint-invariant, part-based representation but not on the basis of pixel-based or DNN-based representations. This raises the following questions: Might it be possible to map from either pixel-based or DNN-based representations (i.e., the relatively unsuccessful representations) to a part-based representation (i.e., the successful representation)? If so, can this mapping be learned by DNNs?

We addressed these questions by training eight DNNs defined by the cross-product of two data sets, two stimulus representations, and two network architectures. The two data sets are referred to as the Random (denoted "R") and Viewpoint-Restricted (denoted "V") data sets. In the former, data items were randomly assigned to either training (four-fifths of data items) or test (one-fifth of data items) sets. Consequently, the training set contained images depicting every possible Fribble part from every possible viewpoint (though, of course, it did not contain images depicting every possible combination of parts (i.e., every object) from every possible viewpoint). In the latter, the training set was restricted so that images depicted Fribbles from 11 of the possible 14 viewpoints. Images depicting Fribbles at the remaining three viewpoints were placed in the test set. Networks trained on the Viewpoint-Restricted data set would perform well on the test set only if they learned viewpoint-invariant representations during training.

The two stimulus representations were the pixel-based (denoted "P") and DNN-based (denoted "D") representations. The two network architectures were a one-layer network (denoted "1") with 100 input units directly connected to 12 output units, and a two-layer network (denoted "2") with 100 input units connected to 100 hidden units which, in turn, were connected to 12 output units. All units used rectified linear (ReLu) activation functions. Networks were trained to take the pixel-based or DNN-based representation of an image as input and to produce the correct part-based representation as output using a mean squared error loss function. Network weights were updated using the Adam algorithm (Kingma et al., 2014) with a batch size of ten data

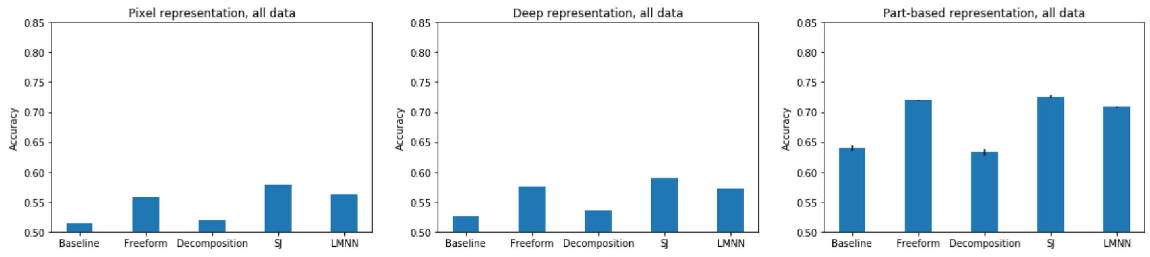


Fig. 8. Summary of the results so far showing the performance of models trained on all data using pixel-based, DNN-based, and part-based representations.

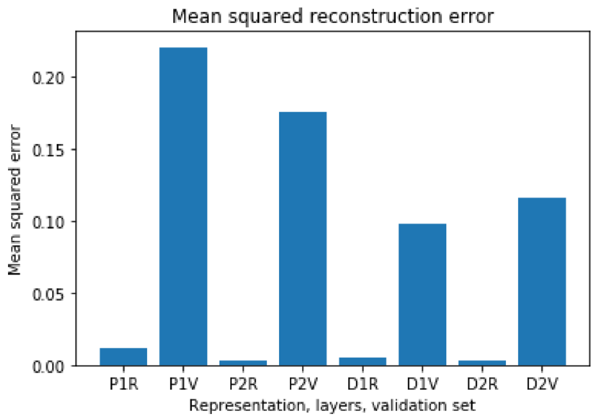


Fig. 9. The horizontal axis labels the eight models. For example, “P1R” denotes a network whose input was coded using the pixel representation, had one layer of units, and that was trained and tested using the Random data set. The vertical axis plots the mean squared error in a network’s estimates of the target part-based representations.

items.

The results are shown in Fig. 9. The horizontal axis labels the eight models. For example, “P1R” denotes a network whose input was coded using the pixel representation, had one layer of units, and that was trained and tested using the Random data set. The vertical axis plots the mean squared error in a network’s estimates of the target part-based representations on the test set. Clearly, test performance was nearly perfect when networks were trained with the Random data set, but performance was relatively poor when networks were trained with the Viewpoint-Restricted data set. This result is not surprising because the training items of the Random data set depicted every possible object part from every possible viewpoint, whereas the training items of the Viewpoint-Restricted data set depicted parts from viewpoints that were different from the viewpoints used in the test images. In other words, since the encoding of individual parts of a Fribble does not impact the encoding of other parts, the training items in the Random data set allowed a network to “generalize” to test stimuli with novel combinations of previously-experienced parts at previously-experienced viewpoints. However, the test stimuli in the Viewpoint-Restricted data set contained previously-experienced parts at novel viewpoints, and thus generalization performance was poor. Lastly, networks using the DNN-based representation outperformed networks using the pixel-based representation.

The results reported here establish that even a simple network can learn to approximate the successful part-based representation as long as its training items include every possible Fribble part at every possible viewpoint. In the next section, we examine whether networks like these, as well as more elaborate ones, are capable of accounting for our experimental data when trained with a loss function designed for this purpose.

6. Networks trained with a triplet loss function

So far, our attempts to use neural networks to model our experimental data have consisted of extracting representations learned in the final convolutional layer of VGG-16, a DNN trained on a task unrelated to our similarity judgment task, and then using these representations for the purposes of metric learning. There is a possibility that a network trained specifically to reproduce our participants’ similarity judgments would be capable of doing so. To investigate this possibility, we adapted the techniques used by Schroff et al. (2015) to train their FaceNet network (Hoffer & Ailon, 2015; Wang et al., 2014). FaceNet, a deep convolutional network, was trained to map high-dimensional face images to low-dimensional embeddings (or latent representations) such that the Euclidean distance between face embeddings corresponded to face similarity:

$$d(x_1, x_2) = \|f(x_1) - f(x_2)\|_2^2 \tag{3}$$

where $\|\cdot\|_2^2$ is the squared ℓ_2 (or Euclidean) norm (equivalent to the Mahalanobis distance using the identity matrix for the Mahalanobis matrix) and $f(\cdot)$ is an embedding produced by the network. The training set consisted of triplets of labeled images, each of which had a probe image depicting a person’s face, a target image depicting the same person’s face as in the probe (possibly at a different viewpoint), and a nontarget image depicting a different face. The loss function, or “triplet loss”, was

$$L = \sum_i \max[0,\ \|f(x_i^t) - f(x_i^p)\|_2^2 - \|f(x_i^t) - f(x_i^n)\|_2^2 + m] \tag{4}$$

where m is a margin, and x_i^p, x_i^t, and x_i^n are the probe, target, and nontarget images in the i^{th} triplet in the training set. Critically, the triplet loss function implemented constraints that are fundamentally identical to the constraints implemented in metric learning (see discussion above and Eq. (6) in Appendix A). FaceNet attempted to minimize the Euclidean distance between the embeddings of images of the same face even when this face was rendered from different viewpoints (i.e., x^t and x^p), and to maximize the distance between embeddings of different faces (i.e., x^t and x^n). This can be seen as the inverse of the metric learning algorithms: instead of learning a Mahalanobis matrix to satisfy the constraints given a stimulus representation, FaceNet learned a stimulus representation satisfying the constraints given a Mahalanobis matrix (in this case, the identity matrix).

For FaceNet, “similar faces” were images of the same person’s face. Our experimental data is slightly different, in that the probe Fribble chosen by a participant to be more similar to a target Fribble is usually not the same Fribble (except on catch trials). Nonetheless, because the underlying principle remains the same, we trained neural networks on our data using the triplet loss function.

We trained a two-layer network with 100 input units, 100 hidden units, and 12 output units, with either the pixel-based or DNN-based representations coding the input. For each input image, the activations of the output units were the network’s embedding of the image. We also trained a deep convolutional network that took the grayscale images, resized to 100×100 pixels, as input to a convolutional layer with 24 channels, each using 7×7 filters with a stride of two, followed by a max pooling layer with 2×2 pools in each channel, a convolutional layer with 12 channels, each using 5×5 filters with a stride of two, another max pooling layer with 2×2 pools in each channel, a convolutional layer with 12 channels, each using 3×3 filters with a stride of two, and then 200-unit, 100-unit, and 12-unit densely connected layers. All layers except the final one used rectified linear activation functions; the final layer used a linear activation function. Network weights were updated using the Adam algorithm (Kingma et al., 2014) with a batch size of 32 data items (for the two-layer networks) or 128 data items (for the convolutional network).[6]

Following training, networks’ embeddings were tested to evaluate how well they satisfied the constraints derived from our experimental data. Training and testing were conducted using fivefold cross-validation (Hastie et al., 2009). Fig. 10 shows the results. The left and middle graphs of Fig. 10 show the results for the two-layer network using the pixel-based and DNN-based representations, respectively, and the right graph shows the results for the convolutional network. The horizontal axis labels the data set (e.g., “LQ” used the data from participants that

[6] We chose this convolutional architecture, as opposed to a larger architecture more representative of modern DNNs from the AI literature, to avoid overfitting on our small dataset of judgments (a risk with large DNNs (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014; Hardt, Recht, & Singer, 2015; Lin, Camoriano, & Rosasco, 2016; Goodfellow, Bengio, & Courville, 2016; Zhang, Vinyals, & Munos, 2018)), and so that the network could be trained fast enough to make cross-validation practical.

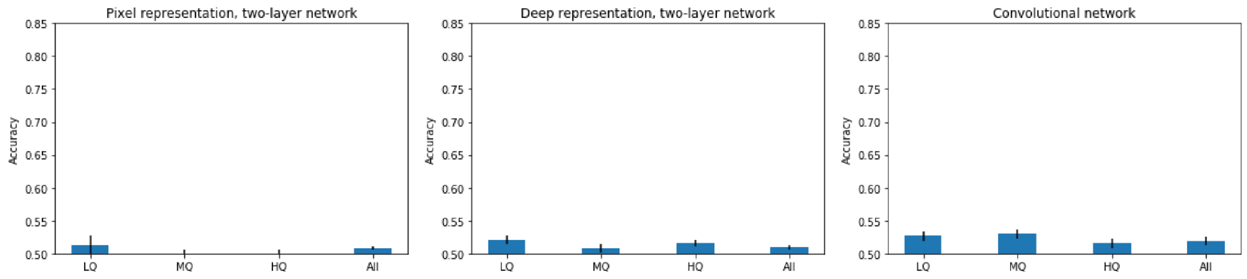


Fig. 10. Mean fivefold cross validation accuracies of networks trained to produce an embedding in which Euclidean distance corresponds to similarity using the triplet loss function with constraints derived from the experimental data. Error bars indicates the standard errors of the means. Results are shown for two-layer networks using pixel-based or DNN-based image representations, as well as for a deep convolutional network trained on grayscale images.

gave low-quality responses, etc.), and the vertical axis shows the mean accuracy (error bars indicate the standard errors of the means). Disappointingly, the networks' embeddings provide significantly worse accounts of our experimental data than metric learning with any of the representations that we tried. We also tried modifying the networks (by adding units, adding layers, and/or adding batch normalization), but this did not have any apparent effect.

Considering our earlier success at training neural networks to learn a part-based representation, and considering our earlier success at training a metric learning system with a part-based representation to provide a good account of our participants' similarity judgments, our failure here is surprising. After all, the networks reported here received training data depicting all Fribble parts from all viewpoints, a condition that was sufficient for successful training of networks to reproduce the part-based representation. So how can we can account for their failure?

We can gain some insight from the value of the loss during training. In general, as networks were trained, the loss oscillated around the value of the margin; any improvement on a given subset of the data only led to worse performance on some other subset. Since the loss function is nonconvex in weight space (as we demonstrate below), there is little reason to expect local gradients of different subsets to point in similar directions. Occasionally, a network would converge towards "collapse", a state in which every input is mapped to exactly the same output. The loss thus became exactly equal to the margin, but obviously this strategy did not lead to good performance at replicating our experimental data.

This result is unsurprising. The triplet loss is notoriously difficult to train, and collapse is a well-known issue in networks using it (Hermans, Beyer, & Leibe, 2017). Triplet selection procedures (i.e., hand-crafted procedures that carefully select the triplets or data items used during training), such as that employed in the original FaceNet paper, are essential to the success of networks attempting to minimize triplet loss. In fact, the triplet selection procedure itself must be delicately crafted so that the resulting training set is neither too easy nor too difficult. Hermans et al. (2017) explain this requirement intuitively: "being told over and over that people with differently colored clothes are different persons does not teach one anything." In other words, if objects in a triplet are similar and dissimilar in trivial ways, the network may simply learn to base its embedding on those trivial features and never learn the importance of task-relevant features that would generalize well. An analogous situation for our dataset would be one in which many of the similar pairs of Fribbles in our dataset are also in similar poses. In this case, a network might learn to rely on irrelevant pose information instead of shape-relevant part-based information. Thus, although networks were able to learn a part-based representation when explicitly trained to do so, the triplets in our dataset may not have been the right ones to force networks to learn shape-relevant features.

In summary, one factor in the difference in performance between our neural networks and part-based metric learning procedures is that the part-based representation gave the metric learning procedures an advantage by isolating the most shape-relevant information "for free". In contrast, neural networks must learn shape-relevant features on their own. However, the neglect of part-based information cannot completely explain our results. While metric learning procedures trained on the pixel-based and DNN-based representations performed significantly worse than those trained on the part-based representation, they still performed better than the DNNs given these same representations. The input was the same, and the metric learning problem and the triplet loss problem are fundamentally the same, so any triplet difficulty-related issues should affect them equally. Yet the metric learning procedures outperformed the triplet loss networks despite being much less powerful. How can we account for this discrepancy?

Our best answer to this question is the following. As noted earlier, the metric learning problem is convex when the parameter to be optimized is the Mahalanobis matrix A, and convex problems are generally easy to solve because they contain a single (global) optimum. In addition, we noted that the constraints embodied in the metric learning problem and those embodied in the triplet loss function are fundamentally identical. Critically, however, the loss function is *nonconvex* in network weight space, and thus it is significantly more difficult to optimize the weights than it is to find a Mahalanobis matrix A that (globally) optimizes the metric learning problem.

To see this, we can imagine building a neural network that performs identically to a metric learning system. Consider a network that maps from either the pixel-based or DNN-based representation of images to a part-based representation in its next-to-last layer. As demonstrated above, we know that this is feasible. The last layer of the network, consisting of linear units, receives part-based representations as inputs and needs to learn a weight matrix such that its activation vectors are image embeddings satisfying the triplet loss function constraints derived from our experimental data. To obtain a network that performs identically to a metric learning system with a part-based representation, it is easy to show that this weight matrix, denoted B, is related to the Mahalanobis matrix A by the following property: $A = B^T B$ (see Appendix B for mathematical details). Crucially, directly finding this weight matrix B that optimizes the triplet loss function is a nonconvex problem.[7] In other words, it is a difficult problem to solve because of local (non-global) optima. This is the same reason the decomposition metric learning algorithm underperformed relative to the other methods (Fig. 5). We conclude that, in principle, the networks reported in this section could have learned image embeddings satisfying the constraints derived from our experimental data. However, the networks failed to do so, presumably due to local (sub-optimal) optima.

Of course, nonconvex loss functions are the norm in the literature on neural networks and, in general, they are not considered a major problem. Indeed, the triplet loss function in FaceNet was likely nonconvex, and yet it achieved good performance. A significant difference between

[7] Of course, one could find it indirectly by first solving the metric learning problem to obtain A, and then performing a Cholesky decomposition of A to obtain B.

FaceNet and our networks (in addition to FaceNet's use of a triplet selection procedure to create a carefully curated training set) is the number of training examples. FaceNet was trained on millions of examples, whereas our networks were trained with only thousands. Having millions of examples may have led to a loss function landscape in which even local optima provided good solutions, especially as FaceNet's training set was carefully constructed, as mentioned earlier, using a special triplet selection procedure that maximized the informativeness of the examples. Thus, although FaceNet may have found only local minima during training, the network performances at these local minima may have been similar to the network performance corresponding to the global minima.

A number of additional factors may have been at play. The first is that, due to computational constraints and a desire to avoid overfitting on our small dataset, our network structures were not as deep or complicated as that used in FaceNet. A more important factor is likely the nature of the stimuli. In the FaceNet simulations, the stimuli were images of faces and, while there was some degree of pose variance, these images did not exhibit the same variety of viewpoints as present in our experiment. In fact, they cannot exhibit the same variety since it is impossible to, for example, present a face "from behind" in the way we did with Fribbles. Thus, it is much less important for FaceNet to be able to extract information about 3-D structure in order to succeed. Furthermore, faces have many intrinsically viewpoint-insensitve features, such as skin tone and hair color. Fribbles have fewer such features, and thus must be distinguished based on an understanding of 3-D properties and part structure. It is possible that these differences make learning to judge the similarity of Fribbles much more difficult for neural networks.

7. Discussion

In summary, we conducted an experiment in which participants were asked to make shape similarity judgments about complex, part-based, 3-D objects in a viewpoint-invariant manner. We attempted to account for participants' judgments using metric learning (Xu et al., 2012), and found that metric learning with pixel-based or DNN-based representations performed poorly whereas metric learning with a viewpoint-invariant, part-based representation provided a relatively good account of our experimental data. Our results indicate that participants made their similarity judgments by comparing the shapes of object parts at corresponding locations on target and probe objects. Notably, this was done in a viewpoint-insensitive manner.

Because we were surprised by the poor performance of metric learning with a DNN-based representation, we examined more carefully the performance of neural networks. We found that neural networks can learn a viewpoint-invariant, part-based representation from pixel-based or DNN-based representations. However, networks using a triplet loss function failed to learn object embeddings satisfying constraints derived from our participants' judgments. Our analyses suggest that, in principle, networks can learn successful object embeddings, but that the presence of poor local minima in the loss function landscape (leading networks to learn task-irrelevant features which generalize poorly) prevented them from doing so. It seems that these networks were overly sensitive to viewpoint-dependent 2-D image features, whereas people judge visual object shape similarity in a more viewpoint-insensitive manner based on 3-D shape features (Erdogan & Jacobs, 2017).

At first glance, the results reported here may seem inconsistent with results previously reported in the scientific literature. Kubilius, Bracci, and de Beeck (2016) reported that DNNs explain human shape judgments for several benchmarks and develop acute sensitivity to minute variations in shape and non-accidental properties. Peterson et al. (2018) found that DNNs (particularly VGG-16; Simonyan & Zisserman, 2015) accounted for human image similarity judgments using images from the Imagenet data set (Russakovsky et al., 2015). Crucially, however, these earlier efforts did not evaluate people's object similarity judgments when objects were rendered from multiple viewpoints. Consequently, they did not evaluate whether DNNs can account for people's viewpoint-insensitive shape similarity judgments. To us, viewpoint insensitivity is a prominent and indispensable property of human visual perception, one that should not be overlooked when considering human visual similarity judgments.

Our findings also underscore recent important results by Geirhos, Rubisch, Michaelis, Bethge, and Wichmann (2018). Analyses by these authors showed that DNNs typically learn to base their judgments on 2-D local image features such as texture information, as opposed to 3-D object shape features. Unlike humans, DNNs do not perform well when tested on images with absent or misleading texture information. Because our experimental stimuli contained useful shape information but lacked useful texture information, it is unsurprising that DNN-based features were unable to account for subjects' judgments, even though simple metric learning systems were able to account for them when provided with viewpoint-invariant part-based representations. Consistent with this finding, Geirhos et al. (2018) found that DNNs trained in a way that biased them to rely more on shape instead of texture information performed almost as well as typical DNNs on standard image classification data sets, and were significantly more robust to image deformations and distortions that people handle easily. An interesting avenue for future work would be to examine the capability of shape-biased networks to explain human similarity judgments like those collected in our experiment.

These results demonstrate that DNNs' texture bias, as opposed to a more human-like shape bias, is not because they lack the expressive power to implement a shape-based solution, but because they lack the inductive biases that would predispose them to learning such a solution. Similarly, as we described earlier and in the appendix, our architectures are provably capable of modeling the experimental similarity data at least as well as the linear methods, so like more complex DNNs and their texture bias, their shortcoming is one of inductive biases, not expressive power.

DNNs have achieved impressive successes on computer vision tasks, and have successfully accounted for important behavioral and neural aspects of biological visual perception. Despite these successes, there is also an awareness in the scientific community that current DNNs have important shortcomings that will need to be addressed in future work. Previous articles have pointed out that people tend to be highly sensitive to 3-D shape features whereas DNNs are more sensitive to 2-D image features, and have noted that people's responses to image distortions or reduced viewing conditions are more robust than those of DNNs, presumably because people are better at using global information such as image context or top-down knowledge (see Jacobs & Bates, 2019, for a review). Here we have highlighted that people tend to perceive visual objects in viewpoint-insensitive (and often part-based) ways. Future DNNs (and other machine learning methods) will need to learn to mimic these properties if they are to provide additional insights into human and machine perception.

Appendix A. Metric learning

This appendix explains how the constrained optimization problem of Eq. (2) was transformed to an unconstrained problem containing "slack variables". It also describes the four algorithms we evaluated for solving the unconstrained problem.

Because it is often unlikely that there exists a matrix A satisfying **all** training constraints in Eq. (2), one can add slack variables to the constraints and attempt to minimize those as well:

$$\begin{aligned}&\min_{\zeta, A\in\mathbb{M}_{PSD}} r(A) + C\sum_i \zeta_i\\ &\text{s.t. } c_i(X^TAX) - \zeta_i \leqslant 0\\ &\qquad\qquad \zeta_i \geqslant 0\end{aligned} \tag{5}$$

where ζ_i is the slack variable corresponding to constraint c_i, and C is a weight that controls the relative importance of minimizing the regularizer versus the slack variables. The optimization problem is now also attempting to minimize the amount of "slack" that must be given to each constraint such that the constraint is satisfied. This new constrained optimization problem can be converted to an unconstrained problem by observing that of the two sets of constraints on ζ_i, only one is "active" for any given training constraint $c_i(X^TAX)$. If $c_i(X^TAX) \geqslant 0$, then $c_i(X^TAX) \leqslant \zeta_i$ is active, and $\zeta_i \geqslant 0$ is redundant; otherwise, $\zeta_i \geqslant 0$ is active and $c_i(X^TAX) \leqslant \zeta_i$ is redundant. Thus one can substitute for the slack variables using the constraints as follows:

$$\min_{A\in\mathbb{M}_{PSD}} r(A) + C\sum_i \max[0, c_i(X^TAX)]. \tag{6}$$

In the ML literature, a number of algorithms are often used to optimize this equation. Here, we describe the four algorithms used in our simulations.

In the "freeform" algorithm, the regularizer in the objective function in Eq. (6) was the Frobenius norm (i.e., the square root of the sum of the squares of the elements of a matrix), a matrix generalization of the Euclidean norm. At each iteration of the algorithm, the modified matrix A was projected onto the cone of positive semidefinite matrices to generate a new estimate.

The "decomposition" algorithm was similar to the freeform algorithm except that G^TG (which is automatically positive semidefinite) replaced A and the algorithm learned matrix G.

In the "SJ" algorithm, a version of the algorithm of Schultz and Joachims (2004), the constraints and regularizer are as above, but A was replaced by $\widetilde{A}D\widetilde{A}^T$ where $\widetilde{A}$ is positive semidefinite (we set $\widetilde{A}$ to the identity matrix) and D is diagonal and non-negative, and the objective function was minimized over D.

Lastly, we also implemented the Large Margin-Nearest Neighbors (LMNN) algorithm (Weinberger et al., 2006), in which the regularizer is $tr(AC)$, where $tr(\cdot)$ denotes the trace operator (i.e., sum of a matrix's diagonal entries) and $C = \sum_{(x_1,x_2)\in S}(x_1 - x_2)(x_1 - x_2)^T$, where S is the set of similar pairs. The positive semidefinite restriction was enforced by projecting onto the cone of positive semidefinite matrices. Inspired by k-nearest neighbors algorithms, LMNN seeks to minimize the distance between similar pairs subject to the constraints.

In the case of the freeform and LMNN algorithms, matrix A was initialized at the start of the optimization procedure by projecting a random matrix onto the cone of positive semidefinite matrices. For the decomposition algorithm, matrix G was initialized to a random matrix. For the SJ algorithm, matrix D was initialized to a random diagonal matrix.

Appendix B. Equivalence of a metric learning system and of a neural network

In the main text, we claimed that it is possible to construct a neural network minimizing the triplet loss function that is equivalent to a metric learning system with a part-based representation. In brief, the network needs to produce a part-based representation in its next-to-last layer, and needs to use a weight matrix B on the connections to the linear output layer such that if A is the Mahalanobis matrix learned by the metric learning system, then B is set such that $A = B^TB$.

Here, we prove this result, though we do so in a general, abstract, and mathematical setting. In the following lemma, x_i is an image of an object (such as a Fribble stimulus), $f(x_i)$ and $g(x_i)$ are representations of image x_i (such as the part-based and deep representations), A corresponds to a Mahalanobis matrix, B to the (square) weight matrix of the final, densely-connected linear layer of a neural network, and $h(\cdot)$ is the embedding of the input by the previous layer of that neural network. Note that the use of x_i and $f(\cdot)$ in this appendix should not be confused with their use in the main body of the paper.

Lemma 1. *Suppose*

A is a real $n \times n$ positive semidefinite matrix
X is a matrix whose i^{th} row is x_i
There is a surjective mapping $f: X \to \mathbb{R}^n$
There is a surjective mapping $g: X \to \mathbb{R}^m$
There is a surjective mapping $h: \mathbb{R}^m \to \mathbb{R}^n$ such that $\forall\ i,\ (h\circ g)(x_i) = f(x_i)$

Then there exists a real $n \times n$ matrixB such that

$$\forall\ i, j\ \ (f(x_i) - f(x_j))^TA(f(x_i) - f(x_j)) = \|B(h\circ g)(x_i) - B(h\circ g)(x_j)\|_2^2$$

Proof. Since $\forall\ i,\ (h\circ g)(x_i) = f(x_i)$, the consequent of the lemma is equivalent to

$$\forall\ i, j\ \ (f(x_i) - f(x_j))^TA(f(x_i) - f(x_j)) = \|Bf(x_i) - Bf(x_j)\|_2^2$$

By definition of the ℓ_2 norm this is equivalent to

$$\forall\ i, j\ \ (f(x_i) - f(x_j))^TA\ \ (f(x_i) - f(x_j)) = (Bf(x_i) - Bf(x_j))^T(Bf(x_i) - Bf(x_j))$$

Simple algebraic manipulation gives us

$$\forall\ i, j\ \ (f(x_i) - f(x_j))^TA(f(x_i) - f(x_j)) = (f(x_i) - f(x_j))^TB^TB(f(x_i) - f(x_j))$$

followed by

$$A = B^TB$$

Thus, for B to exist, it is sufficient for A to have a Cholesky decomposition $A = G^T G$, and in that case, $B = G$. Since A is positive semidefinite, a Cholesky decomposition, and thus B, is guaranteed to exist.

References

Azizpour, H., Razavian, A. S., Sullivan, J., Maki, A., & Carlsson, S. (2015). Factors of transferability for a generic convnet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 38*(9), 1790–1802.

Barry, T. J., Griffith, J. W., Rossi, S. D., & Hermans, D. (2014). Meet the fribbles: Novel stimuli for use within behavioral research. *Frontiers in Psychology, 5*(103), 1–8.

Bellet, A., Habrard, A., & Sebban, M. (2014). A survey on metric learning for feature vectors and structured data. arXiv preprint arXiv: 1306.6709v4.

Chollet, F. (2017). *Deep learning with python.* Shelter Island, NY: Manning.

Chopra, S., Hadsell, R., & LeCun, Y. (2005). Learning a similarity metric discriminatively, with application to face verification. *Proceedings of the ieee computer society conference on computer vision and pattern recognition*.

Cireşan, D. C., Meier, U., & Schmidhuber, J. (2012). Transfer learning for latin and chinese characters with deep neural networks. *The 2012 international joint conference on neural networks (ijcnn)* (pp. 1–6). .

Cooke, T., Jäkel, F., Wallraven, C., & Bülthoff, H. H. (2007). Multimodal similarity and categorization of novel, three-dimensional objects. *Neuropsychologia, 45*, 484–495.

Cox, T. F., & Cox, M. A. A. (1994). *Multidimensional scaling.* London, UK: Chapman & Hall.

Edelman, S. (1998). Representation is representation of similarities. *Behavioral and Brain Sciences, 21*, 449–498.

Edelman, S., & Shahbazi, R. (2012). Renewing the respect for similarity. *Frontiers in Computational Neuroscience, 6*(45), 1–19.

Erdogan, G., & Jacobs, R. A. (2017). Visual shape perception as bayesian inference of 3d object-centered shape representations. *Psychological Review, 124*, 740–761.

Erdogan, G., Yildirim, I., & Jacobs, R. A. (2015). From sensory signals to modality-independent conceptual representations: A probabilistic language of thought approach.PLOS. *Computational Biology, 11* e1004610.

Gaißert, N., Bülthoff, H. H., & Wallraven, C. (2011). Similarity and categorization: From vision to touch. *Acta Psychologica, 138*, 219–230.

Gaißert, N., & Wallraven, C. (2012). Categorizing natural objects: A comparison of the visual and the haptic modalities. *Experimental Brain Research, 216*, 123–134.

Gaißert, N., Wallraven, C., & Bülthoff, H. H. (2010). Visual and haptic perceptual spaces show high similarity in humans. *Journal of Vision, 10*(2), 1–20.

Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A. Brendel, W. (2018). Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. arXiv preprint arXiv:1811.12231.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning.* MIT Press.

Gureckis, T. M., Martin, J., McDonnell, J., Rich, A. S., Markant, D., Coenen, A., Halpern, D., Hamrick, J. B., & Chan, P. (2016). psiturk: An open-source framework for conducting replicable behavioral experiments online. *Behavioral Research Methods, 48*, 829–842.

Hardt, M., Recht, B., Singer, Y. (2015). Train faster, generalize better: Stability of stochastic gradient descent. arXiv preprint arXiv:1509.01240.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (second ed.). Springer.

Hayward, W. G., & Williams, P. (2000). Viewpoint dependence and object discriminability. *Psychological Science, 11*, 7–12.

Hermans, A., Beyer, L., Leibe, B. (2017). In defense of the triplet loss for person reidentification. arXiv preprint arXiv:1703.07737.

Hoffer, E., & Ailon, N. (2015). Deep metric learning using triplet network. *Proceedings of the international conference on learning representations*.

Jacobs, R. A., & Bates, C. J. (2019). Comparing the visual representations and performance of humans and deep neural networks. *Current Directions in Psychological Science, 28*, 34–39.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

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

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) e1004896.

Kulis, B. (2012). Metric learning: A survey. *Foundations and Trends in Machine Learning, 5*(4), 287–364.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*, 436–444.

Lin, J., Camoriano, R., & Rosasco, L. (2016). Generalization properties and implicit regularization for multiple passes sgm. *International conference on machine learning* (pp. 2340–2348). .

Peterson, J. C., Abbott, J. T., & Griffiths, T. L. (2018). Evaluating (and improving) the correspondence between deep neural networks and human representations. *Cognitive Science, 42*, 2648–2669.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L. (2015). ImageNet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV), 115*(3), 211–252.

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 815–823). .

Schultz, M., & Joachims, T. (2004). Learning a distance metric from relative comparisons. In S. Thrun, L. K. Saul, & B. Schölkopf (Eds.). *Advances in neural information processing systems 16* (pp. 41–48). MIT Press.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition.In. *Proceedings of the 2015 international conference on learning representations*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research, 15*(1), 1929–1958.

Tarr, M. J. (2003). Visual object recognition: Can a single mechanism suffice? In M. A. Peterson, & G. Rhodes (Eds.). *Perception of faces, objects, and scenes: Analytic and holistic processes* (pp. 177–211). Oxford University Press.

Vandenberghe, L., & Boyd, S. (1996). Semidefinite programming. *SIAM Review, 38*, 49–95.

Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., & Wu, Y. (2014). Learning fine-grained image similarity with deep ranking. arXiv preprint arXiv:1404.4661v1.

Weinberger, K. Q., Blitzer, J., & Saul, L. K. (2006). Distance metric learning for large margin nearest neighbor classification. In Y. Weiss, B. Schölkopf, & J. C. Platt (Eds.). *Advances in neural information processing systems.* MIT Press.

Wenliang, L. K., & Seitz, A. R. (2018). Deep neural networks for modeling visual perceptual learning. *Journal of Neuroscience, 38*, 6028–6044.

Williams, P. (1997). *Prototypes, exemplars, and object recognition* (Unpublished doctoral dissertation)Department of Psychology, Yale University.

Xu, J.-M., Zhu, X., & Rogers, T. T. (2012). Metric learning for estimating psychological similarities. *ACM Transactions on Intelligent Systems and Technology, 3*(55), 1–19.

Yamins, D. L. K., & DiCarlo, J. J. (2016). Using goal-driven deep learning models to understand sensory cortex. *Nature Neuroscience, 19*, 356–365.

Zhang, C., Vinyals, O., Munos, R. Bengio, S. 2018. A study on overfitting in deep reinforcement learning. arXiv preprint arXiv:1804.06893.