5-2018

The Role of Diagnostic Objects in the Temporal Dynamics of Visual Scene Categorization

Julie Stitt Self
Bates College, jself@bates.edu

The Role of Diagnostic Objects in the Temporal Dynamics of Visual Scene Categorization

Julie Self

Honors Thesis

May 3rd, 2018

Department of Neuroscience, Bates College

Bates College Vision Lab

Advisor: Michelle Greene

Lewiston Maine, 04240

Abstract

When we look at the world around us we are able to effortlessly categorize scenes, but it is still unclear what mechanisms we use to do so. Categorization could be driven by objects, low-level features, or a mixture of both. This study investigated the ways in which diagnostic objects (those found nearly exclusively in one scene category) contribute to scene categorization. It paired Electroencephalography (EEG) with machine learning classification to provide detailed temporal information about when categorization occurs. While recording EEG, participants categorized real-world photographs as one of three indoor scene types (bathroom, kitchen, office). They were shown either original images or versions where diagnostic or random objects had been obscured via localized Fourier phase randomization. EEG voltages and the independent components (ICs) of a whole brain independent component analysis (ICA) were used as feature vectors for a linear support vector machine (SVM) classifier to determine time-resolved accuracy. There were no significant differences in decoding accuracy between categories or between diagnostic and random conditions. Poor classifier performance is likely due to a lack of power, or overfitting of the model. It could also reflect unclear EEG-based neural correlates of each scene type due to the inherent similarities in the categories. While the lack of significant decoding makes it difficult to make strong conclusions about the role of diagnostic objects in visual scene categorization, this study addresses important considerations for pairing EEG with decoding techniques and highlights some of the broader difficulties of isolating distinct features of visual scenes.

Introduction

Imagine you are a guest at a party in an unfamiliar house and are trying to find the bathroom. You walk down the hall opening doors, hoping to find the right room. On the way, you encounter a bedroom and an office before successfully reaching the bathroom. This task is deceptively easy. You open a door and in a glance know exactly what category each room belongs to and whether or not it is the room you are looking for. While humans are able to easily and seemingly instantaneously categorize novel visual scenes, there is still little consensus of how we actually achieve this feat. It could be that categorization is driven by specific, informative objects, or by lower-level features such as color, spatial layout and texture that provide a more global representation of the scene as a whole. It could also be that categorization is driven by a mixture of the two.

The first theory is that scene categorization is driven primarily by the presence of objects. Much of the focus on objects in scene categorization research has stemmed from the popularity of 'schemata' and 'frames' in the 1970s and 1980s. Both schemata and frames refer to internal representations of scene categories that integrate semantic information of specific objects with the relationships between objects within a scene. For example, our schema of a particular scene type (such as a kitchen) contains a list of objects that could be found in that scene (i.e stove, sink, fridge) as well as the ways in which those objects interact (blenders are typically found on top of counters and not on the floor) (Biederman 1981; Friedman 1979). While these internal representations are centered around the idea that scene identity is strongly influenced by the semantic identity of the objects it contains, the question still remains what type of information is needed to initially trigger a particular schema or frame. It could be that a single key object is processed first, which activates the schema and facilitates further object identification (Biederman 1981), or it could be that schemata are triggered by low-level coarse representations of the image that bypass object identification (Oliva and Torralba 2001). Even though the field has generally shifted its focus away from schemata and frames there is still significant debate about which type of information (objects vs low-level features) contributes more to catgorization (Malcolm, Groen, and Baker 2016).

In recent years, support for the role of objects in scene categorization has come mainly from the computer vision community and is centered around the success of classifiers trained on object-based models. For example, L.-J. Li, Su, Lim, et al. (2010) ran a series of dozens of classifiers on what they refer to as 'Object Bank' information that outperformed classifiers of traditional low-level properties. The 'Object Bank' classifier uses a series of object detectors to define images based on the identity of high probability objects. This work is further supported by other object-based classifiers that have performed as well as, if not better than classifiers trained on low-level features (Liao et al. 2016). Additionally, an fMRI study found that activation in scene-selective areas of cortex is highly similar when viewing a scene containing a particular object

and when viewing that object in isolation, and that classification performance decreases as key objects are obscured from a scene (MacEvoy and Epstein 2011). These results provide evidence in favor of object-based categorization because they suggest that object information is important for differentiating between scene types.

One criticism of object-based models is that object identification and scene classification occur at about the same time-scale, which implies that scene categorization could only be driven by the identification of a single object (Biederman 1981; Biederman 1987; Potter et al. 2002). While one key object might be sufficient to determine category, how would a person prioritize a single object to fixate? The issues inherent in this timing dilemma are worth consideration and prompted a shift towards focusing on what other features might drive categorization.

The other side of the argument is that low-level features are sufficient for scene categorization. Over the years an extensive body of research has developed documenting the fact that low-level features of a scene such as color (Oliva and Schyns 2000; Goffaux et al. 2004; Castelhano and Henderson 2008), spatial layout and geometry (Henderson, Weeks, and Hollingworth 1999; Sanocki and Epstein 1997; Biederman 1981; Oliva and Torralba 2001; Greene and Oliva 2009), and texture (Renninger and Malik 2004; Groen et al. 2012) are sufficient for successful categorization.

While there is stong evidence in favor of both objects and low-level features, the fact that this debate continues is important unto itself because it calls into question whether emphasizing this false dichotomy is still the most useful way of conducting research. The fact that there is robust evidence supporting both models suggests that each type of feature does in fact contribute to scene categorization. Understanding vision is therefore better served by asking questions about the *ways in which* these features are important rather than solely *which* one drives categorization. Diving more closely into the *ways in which* different features contribute to classifications allows us to ask a broader range of useful questions. For example, could multiple features of a scene (both objects and low-level) interact to assist categorization? How might they do so? Is there a temporal component to this interaction, where some features are important for early stages of processing, while others come online later? By moving away from debating simply *whether* object or low-level based models are more important we begin to be able to develop a more fine-grained understanding of what is inherently a complex process. Previous studies have highlighted the need to look at the independently explained variance of features (Greene et al. 2016; Groen et al. 2018; Lescroart, Stansbury, and Gallant 2015). One of the goals of this paper is therefore to explore questions about the *ways in which* features allow for categorization, specifically in the context of objects.

Before we are able to fully dive into investigating how scene categorization is facilitated by the interaction of multiple features, it is important to acknowledge, that some aspects of a feature can be more informative than others. More specifically when asking about the *ways in which* objects influence scene categorization, the discussion would be incomplete without accounting for the fact not all objects are created equal,

and that some provide more information about scene type then others. For example, when classifying a kitchen, it seems logical that a stove would be more indicative of category type than a potted plant. Even early research generally recognized that some objects are more useful to study than others. Biederman (1981) refers to the *probability* of an object appearing in a given schema, while Friedman (1979) discusses *obligatory* (likely to be found in a scene) and *nonobligatory* (unlikely to be found in a scene) objects. More importantly, subjective a priori ratings of how likely an object is to appear in a scene correlated with initial eye fixations to that object in a novel scene (Friedman 1979). This implies that our expectations about what objects are likely to be in a given scene type influence where, and how we focus our attentaion. Similarly, the knowledge of which objects and scene types co-occur (which objects are likely in a given scene) has been shown to influence perception (Davenport and Potter 2004). These studies demonstrate firstly that there are certain objects that are more probable, and secondly, that the probability of an object influences the ways in which it contributes to categorization.

Another metric for determining the relative utility of a particular object is frequency. There are two ways of thinking about frequency: the *overall object frequency* [p(object) in the world] and the *object frequency* for a given scene [p(object|scene)] (Greene 2013). *Overall object frequency* can be described through Zipf's Law, which states that a very small proportion of all objects make up the majority of instances in scenes (i.e there are generally more chairs than toaster-ovens in the world) (L.-J. Li, Su, Li, et al. 2010; Greene 2013). While useful in describing differences between individual objects, *overall object frequency* does not provide information about the ways in which objects interact with scene category, while *object frequency* does. For example, measures of *object frequency* such as bag-of-words models have been shown to be sufficient for successful categorization (Greene 2013; Bosch, Zisserman, and Muñoz 2006). Additionally, participants are highly consistent in their estimates of how frequently objects occur in a particular scene type, as well as which objects are improbable for a given scene type (Greene 2016). These studies show that we have strong expectations of what objects occur frequently in scenes, which could easily aid categorization.

A third way of assessing the usefulness of an object is its diagnosticity, or the conditional probability that a scene belongs to a specific category given that it contains a particular object [p(scene|object)] (Greene 2013). While the term diagnosticity has been used in the past in different contexts to describe aspects of scenes such as color (Oliva and Schyns 2000) and other features of a scene more generally (Lowe et al. 2016; Delorme, Richard, and Fabre-Thorpe 2010), it has only fleetingly been referenced in the context of objects (Philippe G. Schyns and Aude Oliva 1994) until recently (MacEvoy and Epstein 2011). Diagnostic objects are objects that are indicative of a particular scene type. For example, a stove is diagnostic of a kitchen, while a chair is not. Even though both stoves and chairs occur with high probability and high frequency in kitchens the chair is less diagnostic of a kitchen because it is also commonly found in

offices and dining rooms. This distinction is subtle, but important because it creates a set of criteria for objects that are characteristic of only a single scene type. It is also valuable to note that an object can be diagnostic, yet have low frequency. For example, a sandcastle is highly diagnostic of a beach (occurs with high probability in beaches, and only beaches), yet not every beach contains a sand castle. Therefore, if there is a sandcastle in a scene it is very likely that it is a beach, yet that knowledge might not always be the most useful for scene categorization in general because you would miss successfully categorizing many beaches if you are only searching for sandcastles.

To date, there has been very little research looking explicitly at the role of diagnostic objects in scene categorization, in part due to the lack of a formal definition. However, Greene (2013) provides an empirical measure of diagnosticity based on the frequency statistics of objects that allows for consistency in research surrounding the role of diagnostic objects and expands the types of experimentation that can be done. In this study, we use the concept of diagnosticity to investigate the role that specific objects play in categorizing visual scenes. In doing so, we are diving deeper into the broader question of what role objects play more generally.

While breaking down objects by diagnosticity is one way of looking at the *ways in which* objects contribute to scene identification, it is unlikely that categorization is driven by only a single feature. This is supported by the fact that classifiers trained on a combination of features out-perform ones trained on each component part (L.-J. Li, Su, Li, et al. 2010; Greene et al. 2016). Additionally, fMRI studies of scene selective cortex have reported interactions between multiple types of scene attributes (Lowe et al. 2016; Malcolm, Groen, and Baker 2016; Lescroart, Stansbury, and Gallant 2015) and ERP studies have shown that several features are encoded at multiple stages of visual processing (Groen, Silson, and Baker 2017).

One way that multiple features can contribute is by operating at different time frames. The results of the ERP studies are therefore particularly interesting because they suggest that temporal dynamics might allow for subtle variation in the way various features contribute to scene categorization. There is MEG/EEG-based evidence that shows that certain low-level global features such as color (Goffaux et al. 2004) spatial layout (Cichy et al. 2017) and spatial frequency (Hansen et al. 2011; Hansen, Johnson, and Ellemberg 2012) contribute to categorization at specific time-frames, which implies that this might also be true of objects. As of yet there have been very few studies looking directly at the temporal dynamics of objects in scene categorization. However, of the ERP studies conducted, one demonstrated that it is possible to determine whether or not a scene contains an animal at close to 150 ms after stimulus presentation (Fabre-Thorpe et al. 2001), while a second set of studies have shown that information about the semantic (identity) and syntactic (physical constraints) congruity of an object is not available until around 300-400 and 600 ms respectively (Ganis and Kutas 2003; Võ and Wolfe 2013). However, there is still more work that needs to be done to determine at exactly what time frame objects are used in scene categorization. As a result, this

study aims to better understand when objects contribute by pairing the decoding of EEG signals with a categorization task that directly looks at the effect of diagnostic objects.

Overall, the purpose of this study is two-fold. We not only seek to examine the role of diagnostic objects in scene categorization, but also hope to determine at what timescale objects contribute. In doing so we shift away from asking questions about *whether* or not a specific feature aids scene categorization, and shift towards asking more fine-grained questions about the *ways in which* a feature is used. The basic outline of the experiment will follow that of MacEvoy and Epstein (2011) in which diagnostic objects were obscured from the scene to test which types of objects are important for categorization. However, we will use EEG rather than fMRI in order to better explore the temporal dynamics of classification. As a result, the two studies will be complementary, with one looking at potential spatial interactions of features and the other looking at the temporal patterns. More specifically, there will be three different image conditions: Original, Diagnostic and Random. In the later two conditions, diagnostic and random objects will be selected based on the criterion set out by Greene (2013) and obscured from the image using local Fourier phase scrambling. Objects will be chosen so as to minimalize variation in low-level features to ensure that the only difference in conditions is their diagnosticity score. Therefore, impaired categorization of images where diagnostic objects have been removed would suggest that those objects are important for scene categorization.

Additionally, we will use EEG recording paired with machine learning classification to determine time-resolved decoding accuracy, which will provide us with information about when object information is available from brain signals. We will be able to predict what type of image a participant was viewing by using a support vector machine (SVM) classifier to decode various features of the EEG data. SVM classifiers can be thought of as general pattern detectors that try to predict what category novel data belong to based on supervised learning of a set of training exemplars. More specifically, a SVM kernel classifier finds the hyperplane that most effectively separates the data and then uses those hyperplanes to classify novel data (Mitchell 1997). One of the limitations of previous studies that examined the role of objects in scene categorization is that they rely solely on behavioral accuracy and reaction times (Joubert et al. 2007; Fabre-Thorpe et al. 2001). The value of using EEG coupled with machine learning decoding techniques is that it allows us to look specifically at the time-resolved classification accuracy at the level of miliseconds and thus compare categorization onset, peak and peak latency between conditions. More generally, it allows for a much more fine-grained understanding of the process of visual categorization.

By using a more empiric definition of diagnosticity, and by pairing EEG recording with machine learning decoding techniques this study will expand our understanding of how diagnostic objects contribute to scene categorization and at what time scale they do so. By answering both these questions we will gain insight not only into what

mechanisms are used to classify visual scenes, but also the ways in which different mechanisms might interact with one another on a temporal level.

Methods

Image Selection

Experimental images were selected from the 3,499 images whose objects had been labeled with the LabelMe tool (Russell et al. 2008). For each scene, all object names, locations, and frequency statistics had been computed for every object (Greene 2013). In the database, diagnosticity is quantified as the probability of a scene given of an object ([p(scene|object)]), size is the total number of pixels, and distance is the Euclidean distance from the center of the image. We initially chose to work with only the eight indoor categories (bathroom, bedroom, conference room, corridor, dining room, kitchen, living room and office) because they generally tend to contain more objects, as well as have objects that are more discrete, which minimizes ontological questions about whether items such as "sky" count as objects (Greene 2013). We defined objects with a diagnosticity score greater than 0.9 as diagnostic and less than 0.1 as random. In other words, for a given type of object (i.e stoves), 90% of all instances of that object were found in a single category (i.e kitchens).

Objects were selected that occurred in at least 10 images per category. While objects can be diagnostic without being frequent (the sandcastle on the beach), we wanted to ensure that objects in this study were descriptive of scene category globally rather than being idiosyncratic to a single image. For example, if an infrequent item such as a pair of sunglassess happens to be in one image of an office, we do not want that object to be counted as diagnostic of all offices. We further narrowed down our categories to those that had at least 10 unique diagnostic objects (bathroom, bedroom, kitchen, office). Doing so ensured that participants were not able to anticipate which object to look for in a given scene type and minimized any learning effects.

Once we had a list of diagnostic and random objects, we selected the images in each category containing those objects and generated all possible diagnostic-random pairs within a given image. Because we are interested in comparing diagnostic and random objects and we know that low-level features can contribute to scene categorization (Groen, Silson, and Baker 2017), it was important that none of these factors differ systematically across conditions. We therefore selected object pairs with the minimum squared difference in size and distance from center and ran paired t-tests to ensure that there was no significant difference across size ($t(256) = -0.79$, $p = 0.43$) or distance ($t(256) = -1.92$, $p = 0.056$). Image pairs with high squared differences were excluded until there was no significant difference in either size or distance from center between diagnostic and random objects across all categories. Because beds were inherently much larger than all other objects, this meant that bedrooms were also excluded from analysis. Post-hoc analysis of image saliency using the MATLAB Saliency Toolbox (Itti and Koch 2000) revealed a small significant difference in max saliency between diagnostic and random images ($t(248) = -2.053$ $p = 0.041$). However, since the average saliency score was higher for random ($M = 0.99$, $SD = 1.27$) than for diagnostic ($M = 0.77$, $SD =$

Figure 1: Diagnostic and Random objects from 249 images were obscured using Fourier phase-scrambled noise masks with feathered edges. Objects were selected based on the diagnosticity criteria laid out by Greene (2013) and were matched so that there was no significant difference in size, distance from the center and saliency.

1.19) objects this was deemed acceptable, as having more salient random objects would make it harder, not easier to reject the null hypothesis. Based on the above criteria, we were ultimately left with 249 experimental images across three categories (bathroom (N= 108), kitchen (N=75), office (N=66)) that were resized to be 256x256 pixels. There was an original, diagnostic and random version of each image.[1] For images in the diagnostic and random conditions, objects were obscured using an elliptical Fourier phase-scrambled noise mask with 5.0 pixel feathered edges in order to preserve the global image statistics to the highest extent possible.

Participants

Participants were recruited from undergraduates at Bates College and therefore reflected the general demographics of the institution. They were compensated for their time either monetarily or with course credit. They were asked to identify their age (mean = 19.77, range = 17-22), the gender they identify with (8 female, 9 male) and their handedness (14 right, 3 left). Before proceeding with EEG recording, we tested the participants' visual acuity using an eye chart to make sure it was at least 20/40. We also administered a color-blindness test (Ishihara 1936) to ensure that general visual ability was not a confounding factor. Additionally, recruiting information indicated that participants should not have a history of brain trauma, nor have a

[1]One of the diagnostic images was accidently the phase-scrambled crop of the object rather than the image itself.

```
%Find images with diagnostic/random objects
for  each category
    find all objects with diagnosticity > 0.9/ <0.1
    find all images in that category
    for  those diagnostic objects
         find all objects with instances in more than 10 images
         find the images containing those objects
             get diagnosticity
             get size
             get distance from center
    end
end

%Find best diagnostic/random pairs
for  each category
   for  each image
        find all the diagnostic and random objects it contains
        generate all possible permutations of diagnotic-random pairs
        for each pair
                calculate the squared difference in size and distance
                find the pair with the lowest sum of squared difference in size and distance
         end
   end
end

run t-test on size and difference from center accross all best pairs
remove images with high squared difference scores until t-test is significant
```

Figure 2: Pseudocode outlining the criteria for Diagnostic and Random object selection.

documented neurological illness, as this might influence EEG recordings. Participants read and signed a consent form before participating in testing. We recorded data from 17 participants, but analyzed 16.[2] All data collected were confidential and experimental procedures were approved by the Bates College IRB.

Procedure

Experimental Task

Participants were fitted for an appropriately sized EEG cap by measuring the circumference of the head from nasion to inion. They were instructed to minimize head movements and eye blinks and were given six practice trials to familiarize themselves with the task before the actual experiment. At the start of each trial, participants were asked to fixate on a target square which was displayed for a variable amount of time determined by sampling from N(500 ms, 50 ms). This ensures no lingering effects from the previous trial. Next, the experimental stimulus was presented for 250 ms. For each of 249 images, participants were shown either the original, diagnostic or random version. Conditions were counterbalanced such that across three participants, a full set of data was shown. Image presentation was followed by a response screen with a two-alternative forced choice categorization task containing the correct answer and a foil randomly selected from the remaining two categories. As soon as participants indicated which category they had seen by pressing either the "a" or "l" key a new trial began. After every 20 trials, participants were able to take break for as long as they needed to stretch and rest their eyes. In total, participants completed 249 trials each. Stimulus presentation was controlled using the MATLAB Psychophysics Toolbox (Brainard, n.d.; Pelli 1997) and EEG data were saved using Pycorder open source acquisition software (BrainVision, Germany). A photocell attached to the computer monitor ensured that the recording was time-locked to stimulus presentation by detecting a white rectangle flashed at the bottom corner of the screen at the same time as the stimulus.

EEG Recording

EEG data were recorded continuously from a 64-channel sensor net with Ag/AgCl elctrodes using the ActiChamp acquisition system (BrainProducts, Germany). EEG signals were amplified and digitized at 1000Hz by a 24-bit analog-to-digital converter. Impedances were kept below values optimized for the system (15 kOhm). DC offset and 60Hz line noise were removed by band-pass filtering of EEG signals from 0.1 Hz to 40 Hz. The vertex (Cz electrode) was used as an online reference, and two electrodes placed at the bottom and outer canthi of the right eye detected eye movements and blinks. Eye electrodes were linked to a right mastoid reference.

[2]The photocell during EEG recording did not work properly for one participant making it impossible to pre-process and epoch and so their data were excluded from analysis.

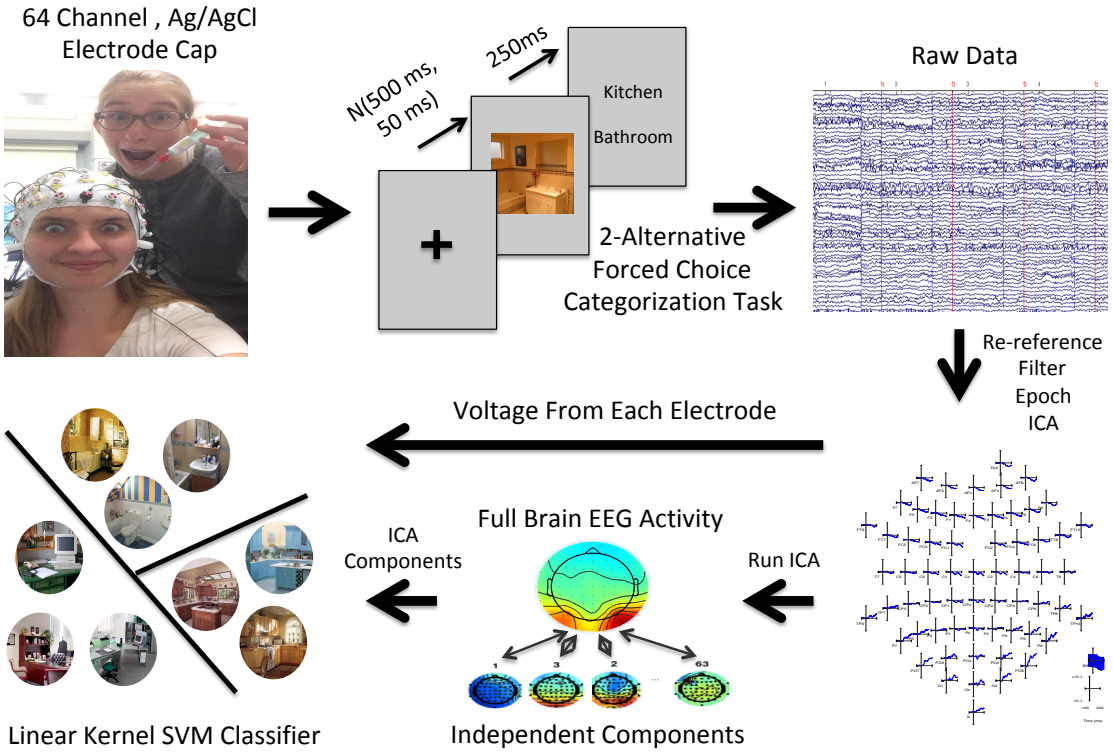

Figure 3: Outline of experimental procedure. EEG data were recorded while participants engaged in a categorization task. The data were re-referenced, filtered, epoched and eye artifacts were removed. EEG voltages were either run directly though a support vector machine (SVM) classifier, or first run through independent component analysis (ICA) before being decoded.

EEG Pre-processing

Raw EEG data were re-referenced to the average of all electrodes and high-pass filtered at 1Hz. The data were epoched into 350 ms sections such that 100 ms pre and 250 ms post stimulus onset was preserved. Independent component analysis (ICA) was run, and visually identified eye movement artifacts were removed from the data.

Analysis

Behavioral

Categorization accuracy was defined as the percentage of correct responses in the categorization task.

Decoding

While behavioral accuracy gives us a general sense of the extent to which removing diagnostic objects impairs scene categorization, it does not provide a more detailed understanding of what is actually going on. Instead of having only a single value to define each condition, we can look at the difference in categorization accuracy millisecond by millisecond by defining key features such as onset of significant decoding, the peak of decoding accuracy and the latency at which peak decoding occurs. In doing so we are able to see exactly the ways in which processing scenes lacking diagnostic objects differs from processing scenes without random objects.

All decoding was run through a linear support vector machine (SVM) classifier (implemented as LIBSVM, Chang and Lin (2011)) using the original data as a training set and then testing on the diagnostic and random data which allows us to asses how each of the two object-removed conditions differ from the original. Upper and lower bounds of chance-level decoding accuracy were calculated using 1000 bootstrap samples of baseline decoding, defined as the decoding accuracy 100 ms before the image appeared on the screen. The onset of significant decoding was defined as the first time point greater than the upper bound of the bootstrapped baseline that was followed by at least five subsequent values also above baseline. The onset for trials where no time point met the conditions outlined above was set to 350 ms (the length of the trial). Peak decoding accuracy was the maximum decoding accuracy obtained after stimulus onset, and peak latency was the time point at which maximum decoding occurred.

EEG Voltage Decoding

The first classifier was run on the normalized voltage values of each of 63 electrodes using a 40 ms sliding window to determine time-resolved decoding accuracy. Each electrode was assigned pre-defined clusters based on anatomical region (frontal,

temporal, occipital), which allowed us to examine differences in activity across brain areas.

ICA Whole Brain Decoding

The voltage recorded by each electrode in the brain represents a mixture of information of varying strengths from multiple different regions. As a consequence, each electrode has access to a mixture of information, which could decrease decoding ability. Previous studies have shown that breaking the EEG signal down into its component parts before running it through a classifier can increase decoding accuracy because each feature given to the classifier represents one type of information rather than a mixture (Stewart, Nuthmann, and Sanguinetti 2014). Independent component analysis (ICA) extracts each of these components and when analyzed together provides a clearer picture of whole brain activity (Hyvärinen, Hurri, and Hoyer 2009). As mentioned above, ICA was run during the pre-processing phase to remove artifacts of eye blinks. The activations of each of 63 independent components (ICs) from a second round of ICA were then used as inputs to the classifier.

Results

Behavioral Results

Participants' categorization accuracies for all three conditions were all above 90%. While the mean accuracy for diagnostic images (M= 0.92, SEM= 0.049) was numerically lower than for original (M= 0.94 , SEM = 0.50) or random images (M= 0.94, SEM= 0.034), a one-way ANOVA revealed no significant difference in classification accuracy across condition $F(2,45) < 1$ (Figure 4).

EEG Voltage Decoding Results

The time-resolved decoding accuracies of a SVM classifier run on normalized EEG voltages were noisy and showed no clear trends (Figure 5). Decoding accuracy was within around 6% of chance and there were no obvious difference in decoding accuracy pre- and post-stimulus onset. The data revealed no visible onset, peak, or peak latency of decoding accuracy due to high levels of noise. However, each was determined quantitatively (see methods) and the means are reported in Table 1. We ran separate repeated measures ANOVAs with two factors on onset, peak and peak latency. The first factor was Condition with two levels (diagnostic, random) and the second was Region with three levels (frontal, temporal, occipital). There was no main effect of Region for onset $F(2,30) < 1$, peak $F(2,30) < 1$, or peak latency $F(2,30) = 1.57$, $p > 0.05$, nor was there a main effect of Condition for onset, peak, or peak latency $F(1,15) < 1$. There was also no significant interaction between Region and Condition for onset $F(2,30) < 1$, peak $F(2,30) = 1.094$, $p > 0.05$, or peak latency $F(2,30) < 1$. Since the data were highly variable, we ran a jackknife analysis to bring out any trends that were obscured by general noise. Jackknifing is a well-established technique where each data point is defined as the average of all other data points and reduces noise by allowing for statistical measures to be run on the grand mean of the data (Luck 2014). However, our analysis revealed no change in the statistical significance of decoding accuracy.

ICA Whole Brain Decoding Results

We ran independent component analysis (ICA) on pre-processed EEG data and trained a classifier using all 63 normalized independent components (ICs) as features, but saw no increase in SVM classifier performance (Figure 6). Decoding accuracy remained within 6% of chance and there was no clear difference between baseline and experimental accuracies. Mean onset, peak and peak latency were quantified and are reported in Table 2. A paired t-test revealed no significant difference in onset ($t(15) = -0.88$, $p = 0.40$), peak decoding accuracy ($t(15) = 1.013$, $p = 0.33$) or peak latency ($t(15) = -0.88$, $p = 0.39$) between Diagnostic and Random conditions. Similarly, conducting a jackknife analysis did not improve significance.

Post-Hoc Analyses

The theoretical upper bound for decoding accuracy was determined by running a SVM classifier on only the original images using 10-fold cross validation (Figure 7). Doing so provided a baseline for the maximum decoding that should be expected across experimental conditions. The decoding accuracy for this classifier was also around 6% above chance and showed no clear difference in pre-and post stimulus decoding accuracy. The decoding accuuracy of the previous EEG voltage classifier (Figure 5) was therefore around max decoding threshold.

Additionally, we modeled what decoding performance might have looked like with a larger number of trials by bootstrapping the whole brain independent components before running them through the classifier. Bootstrapping is a technique that uses sampling with replacement to artificially inflate the sample and allows us to predict what decoding accuracy would be if we had originally had more images. Maximum decoding accuracy reached around 15% above chance, and there was no clear difference in pre-and post stimulus decoding accuracy. Mean onset, peak and peak latency were quantified and are reported in Table 3. A paired t-test revealed a significant difference in onset ($t(15) = -2.60$, $p = 0.020$), but no significant difference in peak decoding accuracy ($t(15) = 0.40$, $p = 0.70$) or peak latency ($t(15) = -0.71$, $p = 0.49$) between Diagnostic and Random conditions. However, it is unwise to lean too strongly on the significant difference between Diagnostic and Random onset for a number of reasons. First, the high level of noise in the data makes it almost visually impossible to define a clear onset, and secondly, it is established that defining an onset in the way that we did (see methods) with noisy data sets is less effective (Luck 2014).

		Diagnostic	+/-	Random	+/-
Onset					
	Frontal	39.46	4.6	43.91	5.188
	Temporal	40.35	5.22	43.47	5.62
	Occipital	48.92	7.66	47.37	4.47
Peak					
	Frontal	48.73	0.24	48.40	0.30
	Temporal	48.54	0.25	48.46	0.31
	Occipital	48.87	0.40	48.39	0.33
Peak Latency					
	Frontal	121.10	6.52	115.45	5.96
	Temporal	116.63	5.08	120.72	7.01
	Occipital	129.80	7.18	112.88	7.65

Table 1: Mean values (+/- SEM) of onset, peak and peak latency of decoding accuracy reported by brain region for a SVM classifier run using normalized EEG voltages as features.

	Diagnostic	+/-	Random	+/-
Onset	148.19	34.88	49.25	28.10
Peak	113.38	0.6422	107.81	0.57
Peak Latency	49.92	15.10	130.13	18.59

Table 2: Mean values (+/- SEM) of onset, peak and peak latency of decoding accuracy for a SVM classifier run using whole brain independent components (ICs) as features.

	Diagnostic	+/-	Random	+/-
Onset	23.56	6.11	73.87	22.23
Peak	62.80	0.93	62.35	1.04
Peak Latency	123.44	17.53	141.94	18.31

Table 3: Mean values (+/- SEM) of onset, peak and peak latency of decoding accuracy for a SVM classifier run using bootstrapped whole brain independent components (ICs) as features.

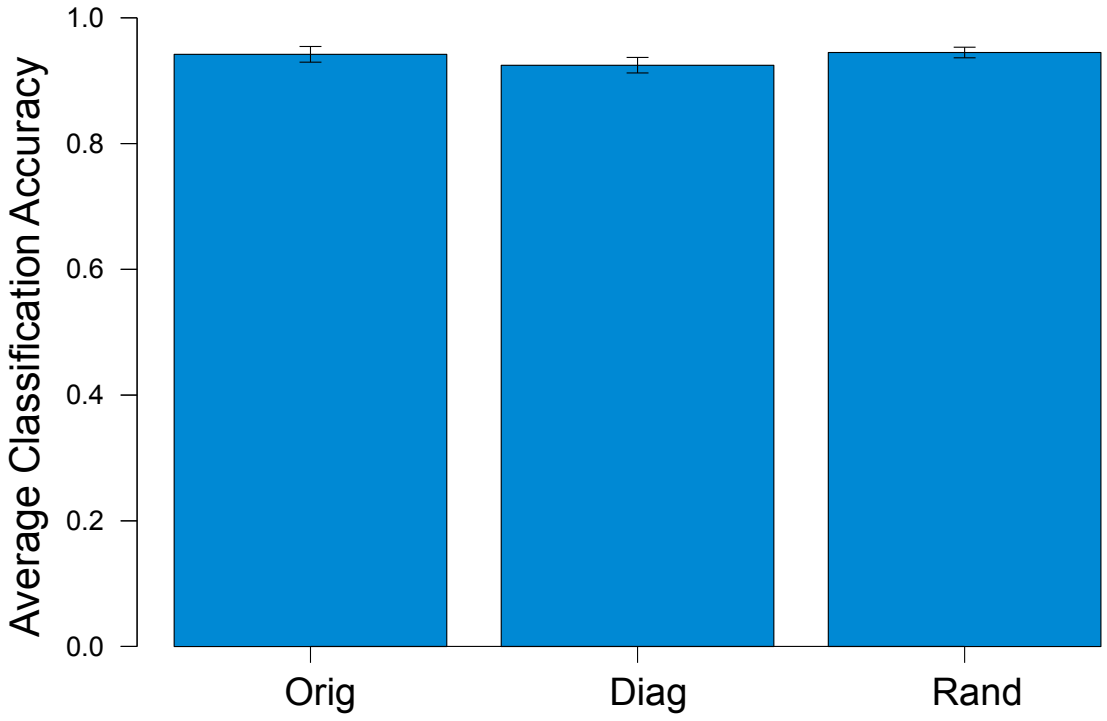

Figure 4: Mean behavioral accuracy on a two-alternative forced choice categorization task. One-way ANOVA revealed no significant difference in classification accuracy across condition. Error bars represent +/- one standard error of the mean (SEM).

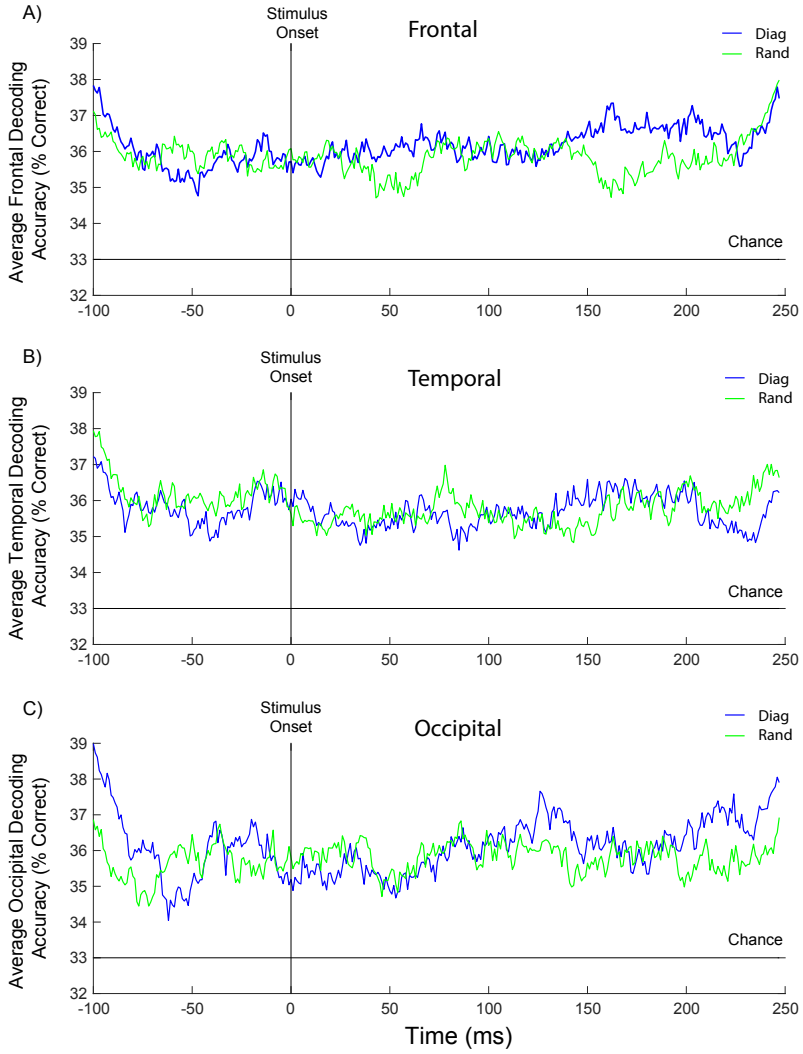

Figure 5: Mean time resolved decoding accuracy of a linear SVM classifier using EEG voltages as features for both Diagnostic and Random conditions across anatomically pre-determined regions of the brain: (a) frontal, (b) temporal, (c) occipital. Repeated measures ANOVA revealed no significant main effect of condition or region, and no significant interaction between the two for decoding onset, peak decoding accuracy and latency of peak decoding accuracy.

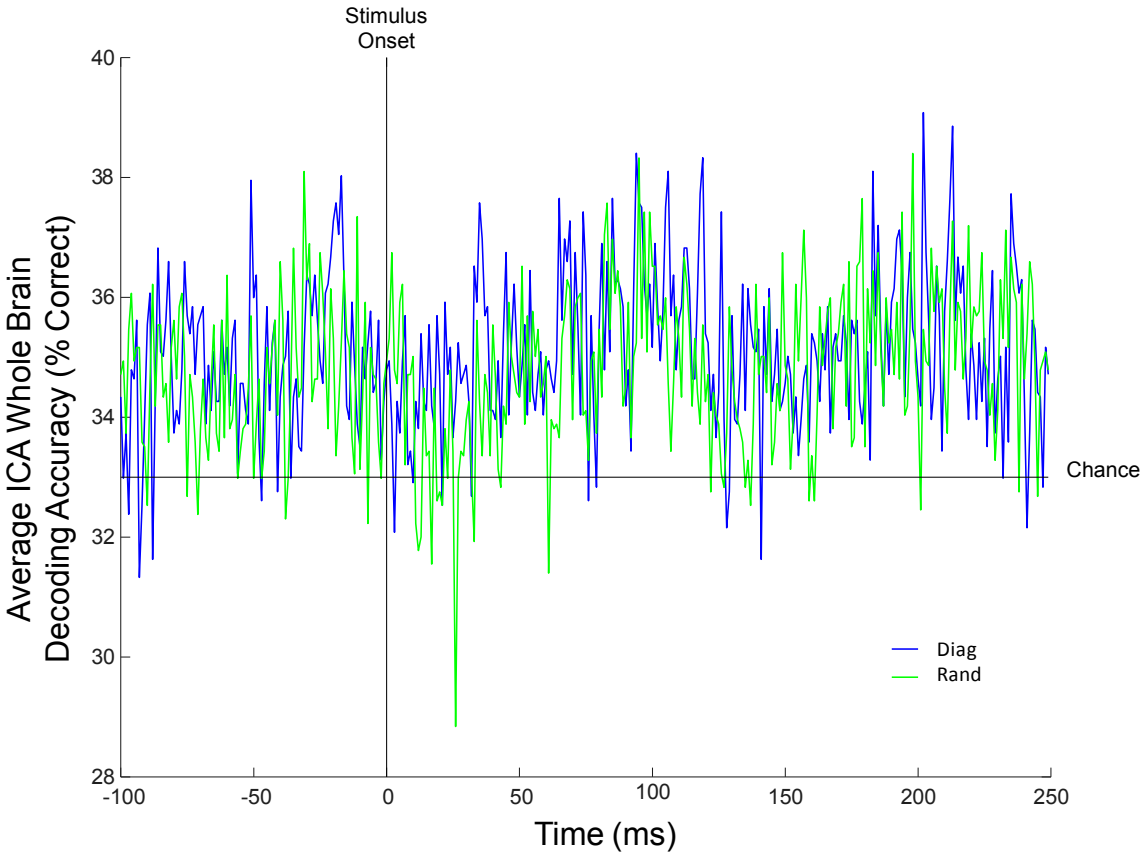

Figure 6: Mean time resolved decoding accuracy of a linear SVM classifier using whole brain independent components (ICs) as features. Paired t-test revealed no significant difference in decoding accuracy between Diagnostic and Random conditions.

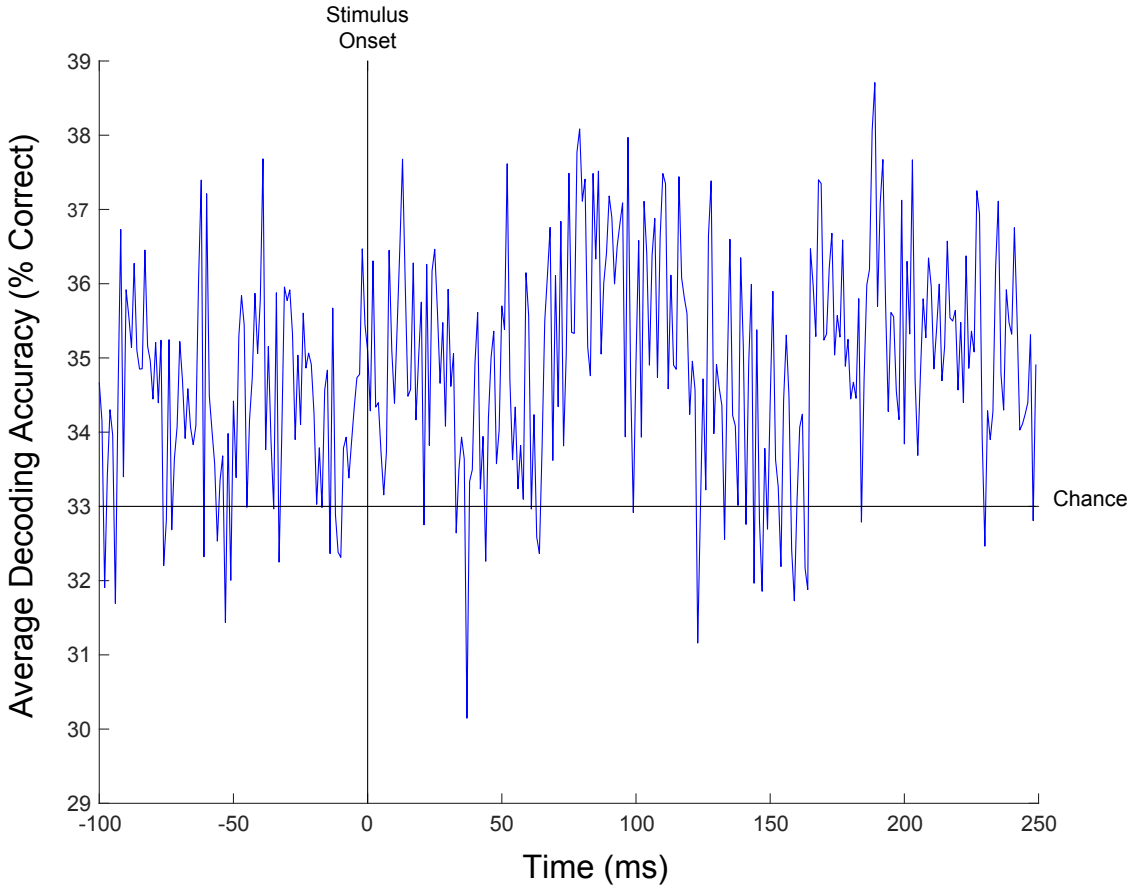

Figure 7: Mean time resolved decoding accuracy of a linear SVM classifier run on normalized EEG voltages from only the Original condition using 10-fold cross validation.

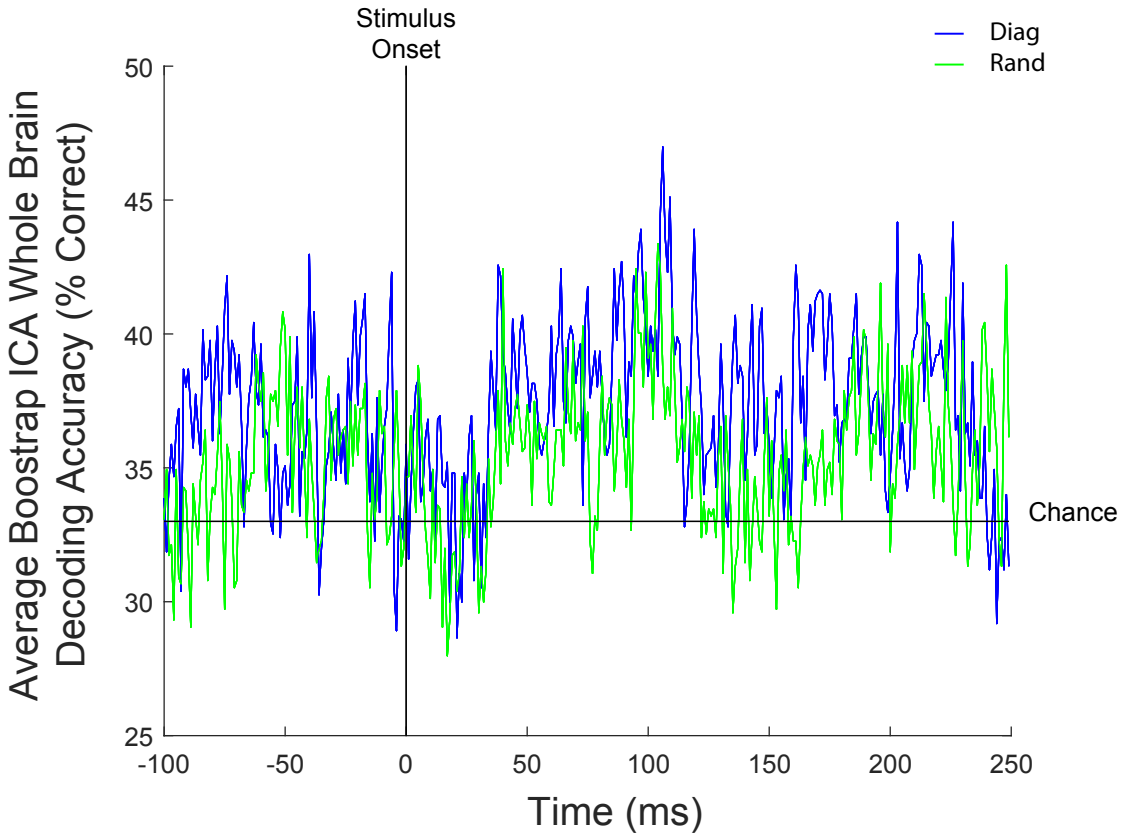

Figure 8: Mean time resolved decoding accuracy of a linear SVM classifier using bootstrapped whole brain independent components (ICs) as features. Paired t-test revealed significant difference in onset of meaningful decoding accuracy t(15) = -2.60, p = 0.020.

Discussion

Overall, our experiment produced very little interpretable data. Any differences between diagnostic and random conditions were obscured by the fact that we observed poor overall classification when looking only at the original images. The fact that there was no increase in decoding accuracy above baseline suggests one of two things: either there are technical limitations to our pairing of EEG and machine classification that are hindering our ability to see clear trends in the data, or there are no exploitable differences in EEG neural activity when viewing kitchens, bathrooms and offices. In either case, conclusions about the role of diagnostic objects become entirely speculative. However, there are a number of interesting and informative things to discuss about why it might be that we observed no significant difference in ability to classify between categories.

The first, and most likely explanation for our null-results is that our machine classifier is not preforming optimally. This is almost definitely due to a lack of power. We only had 249 trials per participant, which is miniscule in the world of machine learning more globaly where data sets can be in the trillions (Halevy, Norvig, and Pereira 2009) and small for EEG-based SVM classification which can rely on up to as many as 15,000 samples (Al-Taei 2017). Even vision-based classifiers using MEG/EEG data tend to operate at a range closer to 500-1000+ trials (Cichy and Pantazis 2017; Seeliger et al. 2017; Saeedi and Arbabi 2017). Without enough samples, the classifier is unable to learn generalizable patterns between categories and thus struggles to differentiate between them.

Knowing that machine learning is a data-hungry tool, it is important to discuss why it was not possible to have a larger sample size to begin with. One of the most critical criteria in our image selection process was minimizing the variation in low-level features across all image conditions. Maintaining similar low-level properties allowed us to ask pointed questions about the role of objects without worrying that we were instead seeing the contribution of low-level features. As discussed in the introduction, differentiating between the role of low-level features and objects was one of the key goals of this study because it allows us to examine the ways in which specific features contribute independently to scene categorization (Groen, Silson, and Baker 2017). The downside is that by maintaining this control, we were also limited in which images could be used and which categories were included in the study. For example, all bedrooms were excluded because of the large size of beds compared to objects in other categories. One way to minimize this reduction of categories in the future would be to make sure that there is no statistical difference between diagnostic and random objects within a given category, rather than across all categories. While this runs the risk of making categories with larger objects obscured more difficult to classify (MacEvoy and Epstein 2011), the trends between diagnostic and random objects would remain the same.

A separate measure of raw decoding accuracy between categories could be generated

using cross-validation to train and test on just the original images. This would allow us to expand the number of categories used and increase our power, while still minimizing variance in low-level features between diagnostic and random conditions. Running t-tests on the objects in each category individually rather than across all categories would also help increase our classification accuracy because by selecting only images and categories where there was no significant between all objects, we also inherently ended up trying to classify the categories with the most similar objects. This reduction of potential categories to only the most similar is an important point and will be addressed later.

While changing how we define our control is one way of increasing power in future experiments, it does not address the small sample size in this present study. Fortunately, we were able to simulate what having more trials might look like using bootstrapping, which artificially increases power by sampling existing trials with replacement to generate a data set with a larger number of trials. Other studies have successfully used this method to increase their sample size and improve categorization (Cichy and Pantazis 2017; Clarke et al. 2015), so we anticipated that bootstrapping our data would increase our decoding accuracy. However, even after re-processing the data we found no clear improvement in classifier performance.

Since artificially increasing our power did not seem to alter classification ability, it could be that our poor decoding performance also stems from massive overfitting of the training set. Overfitting occurs when a classifier divides a training set of data using hyperplanes that are too specific and do not generalize well to a novel testing set. It is characterized by having high training accuracies but poor testing accuracies and by having a high number of support vectors (Mitchell 1997; Noble 2006). These trends can be seen in our classifier, with training accuracies of 100%, and around half the data points acting as support vectors. Overfitting would explain why there appears to be no differences between categories, why our decoding accuracies at baseline are above chance and why there is no significance in decoding accuracy pre- and post-stimulus onset. In the case of baseline decoding being above chance, the model is picking up on absolutely any difference in the feature space, thus resulting in an apparent ability to predict category based on only a gray screen.

There are a few different techniques that we can use in future analysis to minimize the chance of overfitting. The first is to soften the margin of our classifier, which expands the number of support vectors that are allowed in the margins. Doing so increases generalizability because it better discounts outliers in the data (Noble 2006). Secondly, it is possible that our data are currently not linearly separable. We could therefore use a nonlinear Gaussian or radial basis function (RBF) SVM classifier in an attempt to better fit the data (Chang and Lin 2001). To prevent overfitting one of these models, we could first use cross-validation and grid search to optimize our parameters for our given data set (Chang and Lin 2001). However, partitioning off a chunk of our already small training set might further decrease our power, and so might

only be advisable on the bootstrapped data set.

It is also plausible that using principal component analysis (PCA) to determine a key set of features could reduce overfitting. PCA is a method for dimensionality reduction that isolates the features of the data that explain the highest levels of variance (Subasi and Ismail Gursoy 2010). For example, Stewart, Nuthmann, and Sanguinetti (2014) found that classifiers trained using whole brain features showed no significant decoding activity while using the same classifier parameters increased classification accuracy to 0.7 area under the curve (a measure of accuracy) when trained on a subset of highly informative features and attributed this difference to overfitting. It appears that independent component analysis (ICA) differentiates between sources of noise and neural activity by separating them into subsets of independent components (ICs). This means that taking only the most informative ICs eliminates artifacts and noise, while all of that unnecessary information is retained when a classifier is given all possible ICs.

While technical issues with our classifier are the most likely source of our non-significant results, it could also be that it is simply not possible to differentiate between bathrooms, offices and kitchens based on EEG data alone. The main argument against this is that a number of studies using a range of techniques such as multi-unit neuron cell recordings (Hung et al. 2005), fMRI (MacEvoy and Epstein 2011; Walther et al. 2009; Diana, Yonelinas, and Ranganath 2008) and most importantly MEG/EEG (Simanova et al. 2010; Cichy, Pantazis, and Oliva 2014) have successfully classified natural scenes and objects, which implies that there are inherent detectable neural differences between categories.

However, there are two important things to take into consideration here. The first is that EEG recordings only reflect the aggregate activity of large populations of neurons in the cortical gyri (Fisch 1999). Consequently, it is highly probable that we are unable to record some, or even all of the most important neural signals. In the case of the fMRI studies this is particularly true since many of the most classifiable regions (such as the para-hippocampal place area (PPA)) are found deeper than the cortex, so activity there would not reach EEG electrodes (Epstein et al. 1999; Fisch 1999). These limitations of EEG helps explain why we can have ceiling level behavioral categorization accuracy, but non-significant decoding results.

The second point is that many of the previously mentioned categorization tasks involved scenes and objects from relatively distinct categories. More importantly, classification tasks often pull examples from different superordinate categories (Walther et al. 2011; Simanova et al. 2010). Superordinate categories are defined by broad groupings of scenes or objects that all fall under a generic umbrella label (i.e urban/natural, indoor/outdoor environments) (Kadar and Ben-Shahar 2012). In contrast to the studies mentioned above, our image categories belong to the same superordinate category (indoor environments), which means that they likely share many common features. In fact, one of the reasons that we chose to look only at indoor categories was because they had a larger number of objects than outdoor scenes (Greene 2013). Additionally,

our categories were selected to minimize low-level differences, making them even more perceptually similar than just all being indoor scenes. The similarity of images within the same superordinate category is highlighted by a study by Kadar and Ben-Shahar (2012) that presented participants with two images and asked if they belonged to the same category or not. The most common classification errors occurred with pairs from the same superordinate category. Significantly, the highest error out of any reported pair was for kitchens and bedrooms, which are both indoor environments. A follow up experiment revealed that this trend also extended to offices. This implies that scenes within indoor environments are perceived to be highly similar. These findings suggest that it is plausible that our lack of decoding ability could stem from similarities in the stimuli themselves.

On the other hand, the study by MacEvoy and Epstein (2011) mentioned above successfully categorized between bathrooms and kitchens using fMRI multivariate pattern analysis, which demonstrates that the similarity of the category does not necessarily exclude successful categorization. However, the same limitation discussed earlier applies here too, that the success of fMRI classifiers relies on activity in deeper brain structures that EEG does not have access to.

In addition to the difficulties in identifying the best parameters for our classifier and the high similarity in categories, there are a few other, more global limitations that are worth addressing. The first is that there are multiple diagnostic and random objects in each image. By obscuring only a single object we decrease the amount of diagnostic information available in an image, but do not remove it entirely. Therefore, if we had observed any differences in decoding accuracy between diagnostic and random conditions it would have been difficult to definitely attribute the results entirely to diagnosticity. Future work could therefore remove a larger number of objects to see if obscuring more diagnostic information decreased performance. However, removing multiple objects is not as simple as it seems. As discussed above, it is already difficult enough to find a sufficiently large set of images with no significant difference in low-level features between a single pair of objects, let alone multiple. Removing a larger number of objects would therefore reduce the sample size further and make the divisions between the categories even less clear. That said, preserving low-level similarities between conditions by removing a set percentage of total pixels could theoretically do the opposite and increase the number of usable images. Additionally, at some point there would also no longer be any useful visual information left and participants' classification ability would plummet. To illustrate this, one study found that behavioral classification decreased as the number of obscured objects in an image increased and that performance correlated with the number of pixels removed (MacEvoy and Epstein 2011). However, there is likely a sweet spot, that maximizes the number of objects removed while preserving some level of categorization ability.

The second limitation is related to this potential loss of information and highlights other issues inherent to studying only a single component of a visual scene: objects

themselves have distinctive low-level properties. When viewing an object as a whole we often think about it in terms of its semantic name (i.e that is a "stove"). However each object is also associated with a specific set of lower-level characteristics such as size and position that contribute to its identity (i.e refrigerators are often larger than chairs and blenders are found on counters, not on the floor) (Biederman 1981). Obscuring multiple objects thus disrupts low-level information (particularly spatial layout) in addition to removing object information. This is true of removing a single object as well, but we tried to minimize the information lost by using Fourier phase scrambled masks to preserve as many low-level features as we could. However, understanding that objects and low-level features are correlated is an important consideration when designing visual categorization studies because to manipulate one is to manipulate the other. The inherent correlation between the two is likely one of the reasons that there has been so much debate in the literature about which is more important, and why ultimately it seems that it is really a mixture of both that contribute (Groen, Silson, and Baker 2017; Lowe et al. 2016; Al-Taei 2017; Greene et al. 2016). The reality is that we would need to find a way to de-correlate features before each can truly be studied independently (Lescroart, Stansbury, and Gallant 2015; Greene et al. 2016; Groen et al. 2018).

Over all this study sought to provide a clearer understanding of the mechanisms used in visual scene categorization. Our goal was to expand the debate about whether objects or low-level features contribute more to categorization by asking pointed questions about the specific role of diagnostic objects and the temporal dynamics of object-based processing. We were unable to draw any strong conclusions about the role of diagnostic objects due to poor performance of our classifier, but were able to gain insight into the types of things to consider when pairing EEG with machine learning classification. We discuss ways of maximizing power, as well as minimizing overfitting. Additionally our results highlight the difference between decoding categories with high levels of similarity verses those that are much more easily distinguishable. It appears that trying to differentiate between indoor environments might be a particularly hard task. More generally, this study also emphasizes the inherent difficulty in studying either objects, or low-level features in isolation. However, finding ways of doing so allows us to ask questions about *the ways in which* a specific feature contributes to scene classification and moves the field away from the dichotomy of simply asking *whether* or not it does.

Acknowledgments

I would like to thank my advisor Michelle Greene for her unending support, and numerous hours sacrificed helping to complete this thesis. My fellow lab mates Katie Hartnett, Hanna De Bruyn and Priyanka Takle were also instrumental in providing help when I got stuck and providing feedback at various stages of the process. I would also like to thank the Bates Neuroscience Program for making this thesis possible, and all of the students who participated in the study.

References

Al-Taei, Ali. 2017. "Ensemble Classifier for Eye State Classification Using EEG Signals." *arXiv:1709.08590 [Cs]*, September. http://arxiv.org/abs/1709.08590.

Biederman, I. 1981. *On the Semantics of a Glance at a Scene*.

Biederman, Irving. 1987. "Recognition-by-Components: A Theory of Human Image Understanding." *Psychological Review* 94: 115–47.

Bosch, Anna, Andrew Zisserman, and Xavier Muñoz. 2006. "Scene Classification via pLSA." In *Computer Vision – ECCV 2006*, 517–30. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg. doi:10.1007/11744085_40.

Brainard, David H. n.d. "Psychophysics Toolbox," 7.

Castelhano, Monica S., and John M. Henderson. 2008. "The Influence of Color on the Perception of Scene Gist." *Journal of Experimental Psychology. Human Perception and Performance* 34 (3): 660–75. doi:10.1037/0096-1523.34.3.660.

Chang, Chih-Chung, and Chih-Jen Lin. 2001. "LIBSVM: A Library for Support Vector Machines." *ACM Transactions on Intelligent Systems and Technology* 2 (3): 1–27. doi:10.1145/1961189.1961199.

———. 2011. "LIBSVM: A Library for Support Vector Machines." *ACM Trans. Intell. Syst. Technol.* 2 (3): 27:1–27:27. doi:10.1145/1961189.1961199.

Cichy, Radoslaw Martin, and Dimitrios Pantazis. 2017. "Multivariate Pattern Analysis of MEG and EEG: A Comparison of Representational Structure in Time and Space." *NeuroImage* 158 (September): 441–54. doi:10.1016/j.neuroimage.2017.07.023.

Cichy, Radoslaw Martin, Aditya Khosla, Dimitrios Pantazis, and Aude Oliva. 2017. "Dynamics of Scene Representations in the Human Brain Revealed by Magnetoencephalography and Deep Neural Networks." *NeuroImage* 153 (June): 346–58. doi:10.1016/j.neuroimage.2016.03.063.

Cichy, Radoslaw Martin, Dimitrios Pantazis, and Aude Oliva. 2014. "Resolving Human Object Recognition in Space and Time." *Nature Neuroscience* 17 (3): 455–62. doi:10.1038/nn.3635.

Clarke, Alex, Barry J. Devereux, Billi Randall, and Lorraine K. Tyler. 2015. "Predicting the Time Course of Individual Objects with MEG." *Cerebral Cortex* 25 (10): 3602–12. doi:10.1093/cercor/bhu203.

Davenport, Jodi L., and Mary C. Potter. 2004. "Scene Consistency in Object and Background Perception." *Psychological Science* 15 (8): 559–64. doi:10.1111/j.0956-7976.2004.00719.x.

Delorme, Arnaud, Ghislaine Richard, and Michèle Fabre-Thorpe. 2010. "Key Visual

Features for Rapid Categorization of Animals in Natural Scenes." *Frontiers in Psychology* 1. doi:10.3389/fpsyg.2010.00021.

Diana, Rachel A., Andrew P. Yonelinas, and Charan Ranganath. 2008. "The Effects of Unitization on Familiarity-Based Source Memory: Testing a Behavioral Prediction Derived from Neuroimaging Data." *Journal of Experimental Psychology. Learning, Memory, and Cognition* 34 (4): 730–40. doi:10.1037/0278-7393.34.4.730.

Epstein, Russell, Alison Harris, Damian Stanley, and Nancy Kanwisher. 1999. "The Parahippocampal Place Area: Recognition, Navigation, or Encoding?" *Neuron* 23 (1): 115–25. doi:10.1016/S0896-6273(00)80758-8.

Fabre-Thorpe, Michèle, Arnaud Delorme, Catherine Marlot, and Simon Thorpe. 2001. "A Limit to the Speed of Processing in Ultra-Rapid Visual Categorization of Novel Natural Scenes." *Journal of Cognitive Neuroscience* 13 (2): 171–80. doi:10.1162/089892901564234.

Fisch, Bruce. 1999. *Fisch and Spehlmann's EEG Primer: Basic Principles of Digital and Analog EEG, 3e.* 3 edition. Amsterdam: Elsevier.

Friedman, A. 1979. "Framing Pictures: The Role of Knowledge in Automatized Encoding and Memory for Gist." *Journal of Experimental Psychology. General* 108 (3): 316–55.

Ganis, Giorgio, and Marta Kutas. 2003. "An Electrophysiological Study of Scene Effects on Object Identification." *Cognitive Brain Research* 16 (2): 123–44. doi:10.1016/S0926-6410(02)00244-6.

Goffaux, Valerie, Corentin Jacques, Andre Mouraux, Aude Oliva, Philippe G. Schyns, and Bruno Rossion. 2004. "Diagnostic Colors Contribute to the Early Stages of Scene Categorization: Behavioral and Neurophysiological Evidence." *Journal of Vision* 4 (8): 873–73. doi:10.1167/4.8.873.

Greene, Michelle R. 2013. "Statistics of High-Level Scene Context." *Frontiers in Psychology* 4 (October). doi:10.3389/fpsyg.2013.00777.

———. 2016. "Estimations of Object Frequency Are Frequently Overestimated." *Cognition* 149 (April): 6–10. doi:10.1016/j.cognition.2015.12.011.

Greene, Michelle R., and Aude Oliva. 2009. "Recognition of Natural Scenes from Global Properties: Seeing the Forest Without Representing the Trees." *Cognitive Psychology* 58 (2): 137–76. doi:10.1016/j.cogpsych.2008.06.001.

Greene, Michelle R., Christopher Baldassano, Andre Esteva, Diane M. Beck, and Li Fei-Fei. 2016. "Visual Scenes Are Categorized by Function." *Journal of Experimental Psychology. General* 145 (1): 82–94. doi:10.1037/xge0000129.

Groen, Iris I. A., Sennay Ghebreab, Victor A. F. Lamme, and H. Steven Scholte. 2012. "Low-Level Contrast Statistics Are Diagnostic of Invariance of Natural Textures."

Frontiers in Computational Neuroscience 6 (June). doi:10.3389/fncom.2012.00034.

Groen, Iris I. A., Edward H. Silson, and Chris I. Baker. 2017. "Contributions of Low- and High-Level Properties to Neural Processing of Visual Scenes in the Human Brain." *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences* 372 (1714). doi:10.1098/rstb.2016.0102.

Groen, Iris IA, Michelle R Greene, Christopher Baldassano, Li Fei-Fei, Diane M Beck, and Chris I Baker. 2018. "Distinct Contributions of Functional and Deep Neural Network Features to Representational Similarity of Scenes in Human Brain and Behavior." *eLife* 7. Accessed March 26. doi:10.7554/eLife.32962.

Halevy, A., P. Norvig, and F. Pereira. 2009. "The Unreasonable Effectiveness of Data." *IEEE Intelligent Systems* 24 (2): 8–12. doi:10.1109/MIS.2009.36.

Hansen, Bruce C., Theodore Jacques, Aaron P. Johnson, and Dave Ellemberg. 2011. "From Spatial Frequency Contrast to Edge Preponderance: The Differential Modulation of Early Visual Evoked Potentials by Natural Scene Stimuli." *Visual Neuroscience* 28 (3): 221–37. doi:10.1017/S095252381100006X.

Hansen, Bruce C., Aaron P. Johnson, and Dave Ellemberg. 2012. "Different Spatial Frequency Bands Selectively Signal for Natural Image Statistics in the Early Visual System." *Journal of Neurophysiology* 108 (8): 2160–72. doi:10.1152/jn.00288.2012.

Henderson, John M., Phillip A. Weeks, and Andrew Hollingworth. 1999. "The Effects of Semantic Consistency on Eye Movements During Complex Scene Viewing." *Journal of Experimental Psychology: Human Perception and Performance* 25 (1): 210–28. doi:10.1037/0096-1523.25.1.210.

Hung, Chou P., Gabriel Kreiman, Tomaso Poggio, and James J. DiCarlo. 2005. "Fast Readout of Object Identity from Macaque Inferior Temporal Cortex." *Science (New York, N.Y.)* 310 (5749): 863–66. doi:10.1126/science.1117593.

Hyvärinen, Aapo, Jarmo Hurri, and Patrik O. Hoyer. 2009. "Independent Component Analysis." In *Natural Image Statistics*, 151–75. Computational Imaging and Vision. Springer, London. doi:10.1007/978-1-84882-491-1_7.

Ishihara, S. (Satoshi). 1936. "Series of Plates Designed as Tests for Colour-Blindness." http://agris.fao.org/agris-search/search.do?recordID=US201300265800.

Itti, Laurent, and Christof Koch. 2000. "A Saliency-Based Search Mechanism for Overt and Covert Shifts of Visual Attention." *Vision Research* 40 (10-12): 1489–1506. doi:10.1016/S0042-6989(99)00163-7.

Joubert, Olivier R., Guillaume A. Rousselet, Denis Fize, and Michèle Fabre-Thorpe. 2007. "Processing Scene Context: Fast Categorization and Object Interference." *Vision Research* 47 (26): 3286–97. doi:10.1016/j.visres.2007.09.013.

Kadar, Ilan, and Ohad Ben-Shahar. 2012. "A Perceptual Paradigm and Psychophysical

Evidence for Hierarchy in Scene Gist Processing." *Journal of Vision* 12 (13): 16–16. doi:10.1167/12.13.16.

Lescroart, Mark D., Dustin E. Stansbury, and Jack L. Gallant. 2015. "Fourier Power, Subjective Distance, and Object Categories All Provide Plausible Models of BOLD Responses in Scene-Selective Visual Areas." *Frontiers in Computational Neuroscience* 9 (November). doi:10.3389/fncom.2015.00135.

Li, Li-Jia, Hao Su, Fei-Fei Li, and Eric Xing. 2010. "Object Bank: A High-Level Image Representation for Scene Classification & Semantic Feature Sparsification." *Advances in Neural Information Processing Systems*, December. http://repository.cmu.edu/machine_learning/228.

Li, Li-Jia, Hao Su, Yongwhan Lim, and Li Fei-Fei. 2010. "Objects as Attributes for Scene Classification." In *Trends and Topics in Computer Vision*, 57–69. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg. doi:10.1007/978-3-642-35749-7_5.

Liao, Y., S. Kodagoda, Y. Wang, L. Shi, and Y. Liu. 2016. "Understand Scene Categories by Objects: A Semantic Regularized Scene Classifier Using Convolutional Neural Networks." In, 2016-June:2318–25. doi:10.1109/ICRA.2016.7487381.

Lowe, M.X., J.P. Gallivan, S. Ferber, and J.S. Cant. 2016. "Feature Diagnosticity and Task Context Shape Activity in Human Scene-Selective Cortex." *NeuroImage* 125: 681–92. doi:10.1016/j.neuroimage.2015.10.089.

Luck, Steven J. 2014. *An Introduction to the Event-Related Potential Technique*. MIT Press.

MacEvoy, Sean P., and Russell A. Epstein. 2011. "Constructing Scenes from Objects in Human Occipitotemporal Cortex." *Nature Neuroscience* 14 (10): 1323. doi:10.1038/nn.2903.

Malcolm, George L., Iris I. A. Groen, and Chris I. Baker. 2016. "Making Sense of Real-World Scenes." *Trends in Cognitive Sciences* 20 (11): 843–56. doi:10.1016/j.tics.2016.09.003.

Mitchell, Tom M. 1997. *Machine Learning*. McGraw-Hill Series in Computer Science. New York: McGraw-Hill.

Noble, William S. 2006. "What Is a Support Vector Machine?" *Nature Biotechnology* 24 (12): 1565–7. doi:10.1038/nbt1206-1565.

Oliva, Aude, and Philippe G. Schyns. 2000. "Diagnostic Colors Mediate Scene Recognition." *Cognitive Psychology* 41 (2): 176–210. doi:10.1006/cogp.1999.0728.

Oliva, Aude, and Antonio Torralba. 2001. "Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope." *International Journal of Computer Vision* 42

(3): 145–75. doi:10.1023/A:1011139631724.

Pelli, Denis G. 1997. "The VideoToolbox Software for Visual Psychophysics: Transforming Numbers into Movies." *Spatial Vision* 10 (4): 437–42. doi:10.1163/156856897X00366.

Philippe G. Schyns, and Aude Oliva. 1994. "From Blobs to Boundary Edges: Evidence for Time- and Spatial-Scale-Dependent Scene Recognition." *Psychological Science* 5 (4): 195–200. doi:10.1111/j.1467-9280.1994.tb00500.x.

Potter, Mary C., Adrian Staub, Janina Rado, and Daniel H. O'Connor. 2002. "Recognition Memory for Briefly Presented Pictures: The Time Course of Rapid Forgetting." *Journal of Experimental Psychology. Human Perception and Performance* 28 (5): 1163–75.

Renninger, L, and J Malik. 2004. "When Is Scene Identification Just Texture Recognition?" *Vision Research* 44 (19): 2301–11. doi:10.1016/S0042-6989(04)00191-9.

Russell, Bryan C., Antonio Torralba, Kevin P. Murphy, and William T. Freeman. 2008. "LabelMe: A Database and Web-Based Tool for Image Annotation." *International Journal of Computer Vision* 77 (1-3): 157–73. doi:10.1007/s11263-007-0090-8.

Saeedi, Ali, and Ehsan Arbabi. 2017. "Effects of Images with Different Levels of Familiarity on EEG." *arXiv:1710.04462 [Q-Bio, Stat]*, October. http://arxiv.org/abs/1710.04462.

Sanocki, Thomas, and William Epstein. 1997. "Priming Spatial Layout of Scenes." *Psychological Science* 8 (5): 374–78. doi:10.1111/j.1467-9280.1997.tb00428.x.

Seeliger, K., M. Fritsche, U. Güçlü, S. Schoenmakers, J. -M. Schoffelen, S. E. Bosch, and M. A. J. van Gerven. 2017. "Convolutional Neural Network-Based Encoding and Decoding of Visual Object Recognition in Space and Time." *NeuroImage*, July. doi:10.1016/j.neuroimage.2017.07.018.

Simanova, Irina, Marcel van Gerven, Robert Oostenveld, and Peter Hagoort. 2010. "Identifying Object Categories from Event-Related EEG: Toward Decoding of Conceptual Representations." *PLOS ONE* 5 (12): e14465. doi:10.1371/journal.pone.0014465.

Stewart, Andrew X., Antje Nuthmann, and Guido Sanguinetti. 2014. "Single-Trial Classification of EEG in a Visual Object Task Using ICA and Machine Learning." *Journal of Neuroscience Methods* 228 (May): 1–14. doi:10.1016/j.jneumeth.2014.02.014.

Subasi, Abdulhamit, and M. Ismail Gursoy. 2010. "EEG Signal Classification Using PCA, ICA, LDA and Support Vector Machines." *Expert Systems with Applications* 37 (12): 8659–66. doi:10.1016/j.eswa.2010.06.065.

Võ, Melissa L.-H., and Jeremy M. Wolfe. 2013. "Differential Electrophysiological Signatures of Semantic and Syntactic Scene Processing." *Psychological Science* 24 (9):

1816–23. doi:10.1177/0956797613476955.

Walther, Dirk B., Eamon Caddigan, Li Fei-Fei, and Diane M. Beck. 2009. "Natural Scene Categories Revealed in Distributed Patterns of Activity in the Human Brain." *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience* 29 (34): 10573–81. doi:10.1523/JNEUROSCI.0559-09.2009.

Walther, Dirk B., Barry Chai, Eamon Caddigan, Diane M. Beck, and Li Fei-Fei. 2011. "Simple Line Drawings Suffice for Functional MRI Decoding of Natural Scene Categories." *Proceedings of the National Academy of Sciences* 108 (23): 9661–6. doi:10.1073/pnas.1015666108.

Appendix

Experimental Set-Up Code

```
%FileName: PotentialDiagObjectsFinal.m
%Inputs: basicStats.mat
%Outputs: PotentialDiagImages.csv
%Summary: Find all diagnostic objects with diagnosticity greater than
%0.9, more than 10 object instances, more than 10 images with those
%objects. Same code can be used to find random objects -
%set Diagnosticity to < 0.1 and change the variable names.
%(PotentialDiagImages.csv produces a list where some of the objects
%are wrong. Delete those images and save new csv file as
%"PotentialDiagImagesForCodeFinal.csv")

load basicStats;

for c = 1:8 %loops through all categories
    sceneDiagsObj = find(diagnosticity(:,c)>.9); %Find objects in
      %each scene category with diagnosticity > 0.9
    logicalMatrix = logical(instancecounts);
    sceneCatImages = find(C==c);%Find all individual images in a scene
      %category

    myCount = 0; % find image instances above 10 for those diag objs
    for n = 1:length(sceneDiagsObj)
        DiagImageInst = find(logicalMatrix(sceneDiagsObj(n),:));
        if length(DiagImageInst) >= 10
            myCount = myCount + 1;
            goodSceneDiags(myCount) = sceneDiagsObj(n); %all obj w/
              %diag > .9 AND 10+ image instances
        end
    end

    for i = 1:length(goodSceneDiags) %loop through all objects w/
      %diag > .9 AND 10+ image instances
        scenesWithObj = find(logicalMatrix(goodSceneDiags(i),:));
            % images with our diagnostic object
        finalImageID = intersect(scenesWithObj,sceneCatImages); % scene
            %indices in category c with diagnostic object
        for j = 1:length(finalImageID)
            Diag = diagnosticity(goodSceneDiags(i), c); % gets the
```

```
    %diagnosticity

targetObjectName = objectnames(goodSceneDiags(i));
found = 0;
thisAnnotation = D(finalImageID(j)).annotation;
info = imfinfo(fullfile(localFolderImages,
    thisAnnotation.folder, thisAnnotation.filename));
nrows=info.Height;
ncols=info.Width; % gets scene dimensions as these are
    %not uniform
mask = zeros(nrows,ncols); % proxy for whole scene
[x,y]=meshgrid(1:ncols,1:nrows);
for k = 1:length(thisAnnotation.object) % loops through
    %objects
    thisObject = thisAnnotation.object(k).name;
    if strcmp(thisObject, targetObjectName) % are the two
        %names the same?
        objectIndex = k;
        break
    end
end
[X,Y]=getLMpolygon...
     (thisAnnotation.object(objectIndex).polygon);
mask=logical(inpolygon(x,y,X,Y)); % binary in which pixels
    %inside border are 1 and all others 0
objectSize = sum(mask(:)/(nrows*ncols)); % this is the
    %object's size

center=[mean(X)/info.Width mean(Y)/info.Height];
distance=sqrt((center(1)-.5)^2+(center(2)-.5)^2); % this is
  the distance

dataFileName = 'Test1.csv';%'PotentialDiagImages.csv';
    if ~exist(dataFileName, 'file')
        fileID = fopen(dataFileName, 'a+'); % 'a+' = Open
            %or create new file for reading and writing.
            %Append data to the end of the file.
        %fprintf(fileID, '%s \n',['Category, Image, Object,
        ...Diagnosticity, Size, Distance from Center']);
        fprintf(fileID, '%s \n',...
      ['Category, Image, Object,'...
      'Diagnosticity, Size,Distance from Center,c, i,j,k']);
```

```
                    fclose(fileID);
                end
                dataFormatString = '%s, %s, %s, %6.3f, %6.3f, %6.3f,'...
                    '%d,%d, %d, %d \n';

                %save data to that file
                dataFile = fopen(dataFileName, 'a'); %'a' = Open or
                    %create new file for writing. Append data to
                    %the end of the file.
                fprintf(dataFile, dataFormatString, sceneCategories{c},
                    D(finalImageID(j)).annotation.filename,
                    D(finalImageID(j)).annotation.object(k).name, Diag,
                    objectSize, distance, c, i, j,k);
                fclose('all');
        end
    end
end
```

```
%FileName: mergeDataCateforyFinal.m
%Inputs:PotentialDiagImagesForCodeFinal.csv and
  %PotentialRandImagesForCodeFinal.csv
%Outputs: mergeDataWithCategoryFinal.csv
%Summary: produce csv with all possible pairings of diag
  %and rand objects for each image

%Define inputs
diagData = importdata('PotentialDiagImagesForCodeFinal.csv');
randData = importdata('PotentialRandImagesForCodeFinal.csv');
randImages = randData.textdata(:,2);
diagCats = diagData.textdata(:,1);
randCats = randData.textdata(:,1);

%initialize output csv file
fid = fopen('mergeDataWithCategoryFinal.csv','a+');

%define categories (only 4 b/c not enough images in other categories)
cats = {'bathroom','bedroom','kitchen','office'};

%For each category
for i = 1:length(cats)
    thisCat = cats{i};
    catIndDiag = strmatch(thisCat, diagCats); %find indices of all
```

```
        %diagnostic objects
    catIndRand = strmatch(thisCat, randCats); %find indices of all
        %random objects

    %redefine the dataset to only include the diag objects in that
    %category
    smallDiag.textdata = diagData.textdata(catIndDiag,:);
    smallDiag.data = diagData.data(catIndDiag,:);

    %redefine the dataset to only include the rand objects in that
        %category
    smallRand.textdata = randData.textdata(catIndRand,:);
    smallRand.data = randData.data(catIndRand,:);

    %Find Image name for each diag/rand object
    smallDiagNames = smallDiag.textdata(:,2);
    smallRandNames = smallRand.textdata(:,2);

    for j = 1:length(smallDiag.data) %for each diagnostic object
        thisImName = smallDiag.textdata{j,2}; %find the image name
        ind = strmatch(thisImName, smallRandNames); %find the
            %indicies of the random objects that are also in that
            %image

        %save data to csv file
        for k = 1:length(ind)
            fprintf(fid, '%s, %s, %s, %s, %6.3f, %6.3f,...
                %6.3f, %6.3f, %d, %d \n', ...
                thisCat, thisImName, smallDiag.textdata{j,3}, ...
                smallRand.textdata{ind(k),3}, smallDiag.data(j,1), ...
                smallRand.data(ind(k),2), smallDiag.data(j,3),...
                smallRand.data(ind(k),3)); smallRand.data(ind(k),1),...
                smallDiag.data(j,2), ...)
        end
    end
end

fclose(fid)
```

```
%FileName: dataForTTestFinal.m
%Inputs: mergeDataWithCategoryFinal.csv
%Outputs: bestPairsFinal.csv
```

```
%Summary: Determine the best pair of diag-rand objects for each
%image based on diff size and distance from center.
%(run through T-test (using R), remove pairs with high diff
%by hand until T-test is sig. save results as ImageListFinal.csv)

allData = importdata('mergeDataWithCategoryFinal.csv');
fid = fopen('bestPairsFinal.csv','a+'); %opening a file where we will
    %write the best data

myCatNames = allData.textdata(:,1);
myUniqueCat = unique(myCatNames);

clear myCatBin
for i = 4:length(myUniqueCat) %for each category
    catCellList = strfind(myCatNames,myUniqueCat(i)); %mark all IDs of
        %that category with a 1, and all IDs not in that category
        %with 0 (as a cell)
    for m = 1:length(catCellList) %for all IDs (both category and not)
        if isempty(catCellList{m}) %If that ID is not in the category
            %then put a 0 into myCatBin (isempty returns 1 if x
            %is empty)
            myCatBin(m) = 0;
        else %If that ID is in the category then put a 1 into myCatBin
            myCatBin(m) = 1;
        end
    end
    catID = find(myCatBin); %stopped indexing by i because want
        %whole vector
    catTextData = allData.textdata(catID,:); % just text data for cat.
    catNumData = allData.data(catID,:); % just numerical data for cat

    %loop through images w/in category
    myImageNames = catTextData(:,2);
    myUniqueImages = unique(myImageNames);

    clear myImageBin
    for j = 1:length(myUniqueImages) %for each image
        imageID = strmatch(myUniqueImages(j),myImageNames); %mark all
            %IDs of that iamge with a 1, and all IDs that aren't that
            %image with 0 (as a cell)

        clear diffSize diffDistance
```

```
        for k = 1:length(imageID)
            %get the two sizes and squared siff. Store in Vector
            diffSize(k) = (catNumData(imageID(k),1) - catNumData...
              (imageID(k),2))^2;
            %get the two distances and squared siff. Store in vector
            diffDistance(k) = (catNumData(imageID(k),3) - ...
              catNumData(imageID(k),4))^2;
        end

        sumVector = diffSize +diffDistance; %find sum
        bestPair = find(sumVector == min(sumVector)); %find ID of
            %best pair
        bestPair = bestPair(1);

        % Print Data to the file
        fprintf(fid,'%s, %s, %s, %s, %6.3f, %6.3f, %6.3f, %6.3f,'...
        '%6.3f, %6.3f, %d, %d \n', myUniqueCat{i}, ...
        myUniqueImages{j}, catTextData{imageID(bestPair),3},...
        catTextData{imageID(bestPair),4},...
        catNumData(imageID(bestPair),1),...
        catNumData(imageID(bestPair),2),diffSize(bestPair),...
        catNumData(imageID(bestPair),3),catNumData(imageID(bestPair),4),
        diffDistance(bestPair),catNumData(imageID(bestPair),5),...
        catNumData(imageID(bestPair),6));
    end
end
```

```
%FileName: saveObjects.m
%Inputs: ImageListFinal.csv and LabelMeImages folder
%Outputs: "cropDiagOut" and "cropRandOut" folders
%Summary: crop out the diag and rand object for each image.

data = importdata('ImageListFinal.csv');

load basicStats;
outfolderDiag = '/Users/julieself/Dropbox/MATLAB/LabelMeDataset/'
    ...'cropDiagOut/';
outfolderRand = '/Users/julieself/Dropbox/MATLAB/LabelMeDataset/'
    ...'cropRandOut/';

localFolderImages = '/Users/julieself/Dropbox/MATLAB/...
```

```
    LabelMeDataset/LabelMeImages/';

for i = 1:length(data.data)
    thisCategory = data.textdata{i,1};
    catInd = strmatch(thisCategory, sceneCategories);
    theseCats = find(C==catInd);
    thisImage = data.textdata{i,2};

    % find database index of relevant image
    imNum = 0;
    for j = 1:length(theseCats)
        if strcmp(D(theseCats(j)).annotation.filename, thisImage)
            imNum = theseCats(j);
            break
        end
    end

    % read in image
    thisAnnotation = D(imNum).annotation;
    im = imread(fullfile(localFolderImages, thisAnnotation.filename));
    imSize = size(im);

    % get the diagnostic object and save it
    clear minX minY maxX maxY X Y
    diagNum = data.data(i,7);
    [X,Y] = getLMpolygon(D(imNum).annotation.object(diagNum).polygon);
    minX = max(1,min(X)); maxX = max(X);
        if (maxX + (.1*(maxX-minX))) <= imSize(2)
            maxX = maxX + (.1*(maxX-minX));
        end
        if (minX - (.1*(maxX-minX))) >= 1
            minX = minX - (.1*(maxX-minX));
        end
    minY = max(1,min(Y)); maxY = max(Y);
        if (maxY + (.1*(maxY-minY))) <= imSize(1)
            maxY = maxY + (.1*(maxY-minY));
        end
        if (minY - (.1*(maxY-minY))) >= 1
            minY = minY - (.1*(maxY-minY));
        end
    smallDiagIm = im(minY:maxY, minX:maxX, :);
    diagImName = [thisCategory '-diag-' thisAnnotation.filename];
```

```
    imwrite(smallDiagIm,[outfolderDiag diagImName],'jpeg','quality',100);

    % get the random object and save it
    clear minX minY maxX maxY X Y
    randNum = data.data(i,8);
    [X,Y] = getLMpolygon(D(imNum).annotation.object(randNum).polygon);
    minX = max(1,min(X)); maxX = max(X);
        if (maxX + (.1*(maxX-minX))) <= imSize(2)
            maxX = maxX + (.1*(maxX-minX));
        end
        if (minX - (.1*(maxX-minX))) > 1
            minX = minX - (.1*(maxX-minX));
        end
    minY = max(1,min(Y)); maxY = max(Y);
        if (maxY + (.1*(maxY-minY))) <= imSize(1)
            maxY = maxY + (.1*(maxY-minY));
        end
        if (minY - (.1*(maxY-minY))) > 1
            minY = minY - (.1*(maxY-minY));
        end
    smallRandIm = im(minY:maxY, minX:maxX, :);
    randImName = [thisCategory '-rand-' thisAnnotation.filename];
    imwrite(smallRandIm,[outfolderRand randImName],'jpeg','quality',100);
    i

end
```

```
%FileName: Colorphase2.m
%Inputs:"cropDiagOut" and "cropRandOut"
%Outputs:"cropDiagPhase" and "cropRandPhase"
%Summary: phase scramble the cropped objects

folder = '/Users/julieself/Dropbox/MATLAB/LabelMeDataset/cropRandOut/';
outfolder='/Users/julieself/Dropbox/MATLAB/LabelMeDataset/'
    ...'CropRandPhase/';

files= dir([folder '*.jpg']);
Nimages = length(files)

for i = 1:Nimages
    img = mat2grayBCVL(double(imread([folder files(i).name])));
    imSize = size(img);
```

```
    nameima = strcat('ps',files(i).name);

    randomPhase = angle(fft2(rand(imSize(1), imSize(2))));

    clear imFourier
    clear amp
    clear phase
    clear imScrambled
    for layer = 1:imSize(3)
        imFourier(:,:,layer) = fft2(img(:,:,layer));
        amp(:,:,layer) = abs(imFourier(:,:,layer));
        phase(:,:,layer) = angle(imFourier(:,:,layer));
        phase(:,:,layer) = phase(:,:,layer)+randomPhase;
        imScrambled(:,:,layer) = ifft2(amp(:,:,layer).*exp...
            (sqrt(-1)*(phase(:,:,layer))));

        % normalize
        iS = imScrambled(:,:,layer);
        iS=iS(:);

        iS = (iS - min(iS(:)));
        iS = 255 * iS / max(iS(:));

        meangray=mean(iS);

        for n=1:size(iS,1)
            iS(n,:) = (iS(n,:)-(meangray-128));
        end

        %ima=reshape(ima,256,256);

        j = find(iS<0); iS(j) = 0;
        j = find(iS>255); iS(j) = 255;

        imScrambled(:,:,layer) = reshape(iS,[imSize(1) imSize(2)]);
    end

    imScrambled = real(imScrambled);
    imwrite(uint8(imScrambled),[outfolder nameima],'jpeg','quality',100);
    i
end
```

```
%FileName: Make3ExperimentalLists.m
%Inputs: ExperimentalOriginal, ExperimentalDiag,
    %ExperimentalRand
%Outputs:ExperimentalLists.mat
%Summary: Make three experimental lists saved as a structure
%that include image name, category, non-categories, condition
%(orig, diag, rand).

folderOrig = '/Users/julieself/Dropbox/MATLAB/LabelMeDataset/'
    ...'ExperimentalOriginal/';
folderDiag = '/Users/julieself/Dropbox/MATLAB/LabelMeDataset/'
    ...'ExperimentalDiag/'
folderRand = '/Users/julieself/Dropbox/MATLAB/LabelMeDataset/'
    ...'ExperimentalRand/'

ExperimentalOrig = dir([folderOrig '*.jpg'])
ExperimentalDiag = dir([folderDiag '*.jpg'])
ExperimentalRand = dir([folderRand '*.jpg'])

%Make 3 experimental Lists (cell arrays) with an even mixture
%of Orig,Diag,Rand
%Make IDs for each category
for i = 1:3:249
OrigID(i) = 1;
OrigID(i+1) = 2;
OrigID(i+2) = 3;
end
clear i

for i = 1:3:249
DiagID(i) = 2;
DiagID(i+1) = 3;
DiagID(i+2) = 1;
end
clear i

for i = 1:3:249
RandID(i) = 3;
RandID(i+1) = 1;
RandID(i+2) = 2;
end
clear i
```

```
%Find those IDs within ExperiemtnalOrig
for j = find(OrigID == 1)
    ListA{j} = ExperimentalOrig(j).name;
end
clear j
for j = find(OrigID == 2)
    ListB{j} = ExperimentalOrig(j).name;
end
clear j
for j = find(OrigID == 3)
    ListC{j} = ExperimentalOrig(j).name;
end

%Find those IDs within ExperiemtnalDiag
for k = find(DiagID == 1)
    ListA{k} = ExperimentalDiag(k).name;
end
clear k
for k = find(DiagID == 2)
    ListB{k} = ExperimentalDiag(k).name;
end
clear k
for k = find(DiagID == 3)
    ListC{k} = ExperimentalDiag(k).name;
end

%Find those IDs within ExperiemtnalRand
for l = find(RandID == 1)
    ListA{l} = ExperimentalRand(l).name;
end
clear l
for l = find(RandID == 2)
    ListB{l} = ExperimentalRand(l).name;
end
clear l
for l = find(RandID == 3)
    ListC{l} = ExperimentalRand(l).name;
end

save ListA ListB ListC
```

```
%save each List into a structure that also includes ImageName,
%Category, NonCategory, Condition

%Save structure including Category, NonCategory
for h = 1:108
finalListA(h).Category = 'bathroom';
finalListA(h).NonCategory = {'kitchen','office'}
finalListB(h).Category = 'bathroom';
finalListB(h).NonCategory = {'kitchen','office'}
finalListC(h).Category = 'bathroom';
finalListC(h).NonCategory = {'kitchen','office'}
end
clear h
for h = 109:183
finalListA(h).Category = 'kitchen';
finalListA(h).NonCategory = {'bathroom','office'}
finalListB(h).Category = 'kitchen';
finalListB(h).NonCategory = {'bathroom','office'}
finalListC(h).Category = 'kitchen';
finalListC(h).NonCategory = {'bathroom','office'}
end
clear h
for h = 184:249
finalListA(h).Category = 'office';
finalListA(h).NonCategory = {'kitchen','bathroom'}
finalListB(h).Category = 'office';
finalListB(h).NonCategory = {'kitchen','bathroom'}
finalListC(h).Category = 'office';
finalListC(h).NonCategory = {'kitchen','bathroom'}
end

%save structure including imagename

for m = 1:249
  finalListA(m).Name = ListA(m);
  finalListB(m).Name = ListB(m);
  finalListC(m).Name = ListC(m);
end

for n = 1:3:249
finalListA(n).Condition = 'original' ;
finalListA(n+1).Condition = 'random' ;
```

```
finalListA(n+2).Condition = 'diagnostic' ;

finalListB(n).Condition = 'diagnostic' ;
finalListB(n+1).Condition = 'original' ;
finalListB(n+2).Condition = 'random' ;

finalListC(n).Condition = 'random' ;
finalListC(n+1).Condition = 'diagnostic' ;
finalListC(n+2).Condition = 'original' ;
end

clearvars -except ListA ListB ListC finalListA finalListB finalListC
save ExperimentalLists
```

Data Processing Code

```
%FileName: conditionOrderList.m
%Inputs: decodeDiagnosticObjectsToMakeList.csv
%Outputs:CatCndOrderLists_1-17
%Summary:Create a numerically coded vector for the order in which
%the different conditions were presented to each participant.
%(0 = orig, 1 = diag, 2 = rand)

%Create Vector of order that "condition" was presented in experiment
    %read in all data
    allData = importdata('decodeDiagnosticObjectsToMakeList_1-17.csv');
    allDataLabels = allData.textdata(:,8);

    %Code all labels as numbers
    %(0 = Original, 1 = Diagnostic, 2 = Random )
        orig = strmatch('orig', allDataLabels);
        diag = strmatch('diag', allDataLabels);
        rand = strmatch('rand', allDataLabels);

        vectorData = zeros(length(allDataLabels),1);
        vectorData(orig) = 0;
        vectorData(diag) = 1;
        vectorData(rand) = 2;

    %create vector for each participant with condition name

    sbj1cnd = vectorData(1:249);
    sbj2cnd = vectorData(250:498);
    sbj3cnd = vectorData(499:747);
    sbj4cnd = vectorData(748:996);
    sbj5cnd = vectorData(997:1245);
    sbj6cnd = vectorData(1246:1494);
    sbj7cnd = vectorData(1495:1743);
    sbj8cnd = vectorData(1744:1992);
    sbj9cnd = vectorData(1993:2241);
    sbj10cnd = vectorData(2242:2490);
    sbj11cnd = vectorData(2491:2739);
    sbj12cnd = vectorData(2740:2988);
    sbj13cnd = vectorData(2989:3237);
    sbj14cnd = vectorData(3238:3486);
    sbj15cnd = vectorData(3487:3735);
    sbj16cnd = vectorData(3736:3984);
```

```
    sbj17cnd = vectorData(3985:4233);

    %Save Lists
    clear 'allData' 'allDataLabels' 'diag' 'rand' 'orig' 'vectorData'
    save 'conditionOrderLists_1-17.mat',
    clear all

%Create Vector of order that "category" was presented in experiment
    %read in all data
    allData = importdata('decodeDiagnosticObjectsToMakeList_1-17.csv');
    allDataLabels = allData.textdata(:,7);

    %Code all labels as numbers
    %(0 = kitchen, 1 = bathroom, 2 = office )
        kitchen = strmatch('kitchen', allDataLabels);
        bathroom = strmatch('bathroom', allDataLabels);
        office = strmatch('office', allDataLabels);

        vectorData = zeros(length(allDataLabels),1);
        vectorData(kitchen) = 0;
        vectorData(bathroom) = 1;
        vectorData(office) = 2;

    %create vector for each participant with category name
    sbj1cat = vectorData(1:249);
    sbj2cat = vectorData(250:498);
    sbj3cat = vectorData(499:747);
    sbj4cat = vectorData(748:996);
    sbj5cat = vectorData(997:1245);
    sbj6cat = vectorData(1246:1494);
    sbj7cat = vectorData(1495:1743);
    sbj8cat = vectorData(1744:1992);
    sbj9cat = vectorData(1993:2241);
    sbj10cat = vectorData(2242:2490);
    sbj11cat = vectorData(2491:2739);
    sbj12cat = vectorData(2740:2988);
    sbj13cat = vectorData(2989:3237);
    sbj14cat = vectorData(3238:3486);
    sbj15cat = vectorData(3487:3735);
    sbj16cat = vectorData(3736:3984);
    sbj17cat = vectorData(3985:4233);
```

```
    %Save Lists

    clear 'allData' 'allDataLabels' 'bathroom' 'kitchen'...
        'office' 'vectorData'
    save 'categoryOrderLists_1-17.mat',
    clear all
%Create matrix for each participant where first column is category and
%second column is condition

load categoryOrderLists_1-17.mat
load conditionOrderLists_1-17.mat

sbj1 = [sbj1cat,sbj1cnd];
sbj2 = [sbj2cat,sbj2cnd];
sbj3 = [sbj3cat,sbj3cnd];
sbj4 = [sbj4cat,sbj4cnd];
sbj5 = [sbj5cat,sbj5cnd];
sbj6 = [sbj6cat,sbj6cnd];
sbj7 = [sbj7cat,sbj7cnd];
sbj8 = [sbj8cat,sbj8cnd];
sbj9 = [sbj9cat,sbj9cnd];
sbj10 = [sbj10cat,sbj10cnd];
sbj11 = [sbj11cat,sbj11cnd];
sbj12 = [sbj12cat,sbj12cnd];
sbj13 = [sbj13cat,sbj13cnd];
sbj14 = [sbj14cat,sbj14cnd];
sbj15 = [sbj15cat,sbj15cnd];
sbj16 = [sbj16cat,sbj16cnd];
sbj17 = [sbj17cat,sbj17cnd];

save ('CatCndOrderLists_1-17.mat', 'sbj1', 'sbj2', 'sbj3', 'sbj4', ...
      'sbj5', 'sbj6', 'sbj7', 'sbj8', 'sbj9', 'sbj10', 'sbj11', ...
      'sbj12', 'sbj13', 'sbj14', 'sbj15', 'sbj16', 'sbj17');
```

```
%FileName: MakePhotoCell.m
%Inputs:.eeg file of re-referenced and filtered EEG data
%Outputs:.eeg file where channel 64 is a binary photocell
    %indicating the start location of each trial
%Summary: Generate a binary photocell to indicate the exact time
%of image onset for each trial. Allows us to epoch the data using
%EEGlab
```

```
% define photocell channel
photocellRaw = EEG.data(64,:);
plot(photocellRaw)

% Define start and threshold
start = input('Please enter start point: ');
thresh = input('Please enter thresh point: '); %(usually around 50000)

% make binary photocell
photocellRaw(1:start) = 0;
a = find(photocellRaw > thresh);
photocell = zeros(size(photocellRaw));
photocell(a) = 1;

%replace orginal channel data with binary photocell
EEG.data(64,:) = photocell;
```

```
%FileName: makeRecForSaliency.m
%Inputs:Experimental Originals
%Outputs:thisDiagRect1_249.mat, thisRandRect1_249.mat,
    %mySaliencyCellArray.mat
%Summary: Find coordinates of diagnostic and random objects to put
%into the Saliency Toolbox. The code runs a loop, that opens one
%image at a time and allows you to draw a box by hand around the
%diagnostic/random object. The coordinates of that box are then
%saved (repeat for all 249 images for diagnostic and random
%separately). Generates a list containing the names of images in
%the order they were presented (MATLAB calls them differently than
%the order in the original file)

%find coordinates of diagnostic object to put into Saliency Toolbox
%have path open to Experimental Originals

ims = dir('*.jpg');
for i = 1:length(ims)
    imagesc(imread(ims(i).name));
    rect = imrect;
    thisDiagRect(i,:) = rect.getPosition();
end

%find coordinates of random object to put into Saliency Toolbox
%have path open to Experimental Originals
```

```
ims = dir('*.jpg');
for i = 199:length(ims)
    imagesc(imread(ims(i).name));
    rect = imrect;
    thisRandRect(i,:) = rect.getPosition();
end

%make cell array with image names in correct order

ims = dir('*.jpg');
for i = 1:length(ims)
    thisImg = ims(i).name;
    mySaliencyCellArray{i} = thisImg;
end
```

```
%FileName: MakeSaliencyMap.m
%Inputs:mySaliencyCellArray.mat, thisDiagRect1_249.mat,
    %thisRandRect1_249.mat
%Outputs:SaliencyDataForTTest.csv
%Summary:Takes the coordinates of diag/rand objects, finds the
%saliency mac/mean for each and records it in a .csv file that
%will later be used to run paired t-tests. v

%Julie Self
%3/12/18
%TThis Code takes the coordinates of diag/rand objects, finds the
%saliency max/mean for each and records it in
%'SaliencyDataForTTest.csv' which will be run as a t-test in R

load mySaliencyCellArray
load thisDiagRect1_249
load thisRandRect1_249

saliencyIm = batchSaliency(mySaliencyCellArray); %generate compressed
    %saliency maps -will need access to "ExperimentalOriginal" folder

%Diagnostic Images
for n = 1:length(thisDiagRect1_249) %for all Diag Images
    thisDiagObj = round(thisDiagRect1_249(n,:)); %Get vector with
        %coordinates of the object
    thisImage = importdata(mySaliencyCellArray{n}); %get information
        %about original image
```

```
%define coordinates
X1 = thisDiagObj(1);
Y1 = thisDiagObj(2);
X2 = thisDiagObj(1) + thisDiagObj(3);
Y2 = thisDiagObj(2) + thisDiagObj(4);

%fix where imrect boxes were dragged outside boundary of image
if X2 > (size(thisImage,2))
    X2 = size(thisImage,2);
end
if Y2 > (size(thisImage,1))
    Y2 = size(thisImage,1);
end
if (X1 < 1)
    X1 = 1;
end
if (Y1 < 1)
    Y1 = 1;
end

%create full size Saliency Map
img = initializeImage(saliencyIm(n).origImage.filename);
fullSizeSalMap = imresize(saliencyIm(n).data,img.size(1:2));
    %(adjust map to be same dimension as original image)


%calculate saliency stats of object
diagSaliencyIm = fullSizeSalMap(Y1:Y2,X1:X2); %make crop of
    %saliency map w/ diag object
%meanDiagSaliency = squeeze(mean(diagSaliencyIm,1));
meanDiagSaliency(n) = mean(diagSaliencyIm(:)); %get mean
    %saliency value for diag object
%maxColmnValue = max(diagSaliencyIm);
maxDiagSaliency(n) = max(diagSaliencyIm(:)); %get max
    %Saliency Value for Diag object


%plot Saliency Map w/ crop and Original w/ crop to sanity check
%everything
     figure (1)
    imagesc(fullSizeSalMap)
     colorbar
```

```
          figure(2)
          imagesc(diagSaliencyIm)
          colorbar
          figure(3)
          imagesc(thisImage)
          figure(4)
          imagesc(thisImage(Y1:Y2,X1:X2,:))
end



%Random Images
for n = 1:length(thisRandRect1_249) %for all Rand Images
    thisRandObj = round(thisRandRect1_249(n,:)); %Get vector with
        %coordinates of the object
    thisImage = importdata(mySaliencyCellArray{n}); %get information
        %about original image

    %define coordinates
    X1 = thisRandObj(1);
    Y1 = thisRandObj(2);
    X2 = thisRandObj(1) + thisRandObj(3);
    Y2 = thisRandObj(2) + thisRandObj(4);

    %fix where imrect boxes were dragged outside boundary of image
    if X2 > (size(thisImage,2))
        X2 = size(thisImage,2);
    end
    if Y2 > (size(thisImage,1))
        Y2 = size(thisImage,1);
    end
    if (X1 < 1)
        X1 = 1;
    end
    if (Y1 < 1)
        Y1 = 1;
    end

    %create full size Saliency Map
    img = initializeImage(saliencyIm(n).origImage.filename);
    fullSizeSalMap = imresize(saliencyIm(n).data,img.size(1:2));
        %(adjust map to be same dimension as original image)
```

```
    %calculate saliency stats of object
    randSaliencyIm = fullSizeSalMap(Y1:Y2,X1:X2); %make crop of
        %saliency map w/ rand object
    meanRandSaliency(n) = mean(randSaliencyIm(:));  %get mean saliency
        %value for Rand object
    maxRandSaliency(n) = max(randSaliencyIm(:)); %get max Saliency Value
        %for Rand object


    %plot Saliency Map w/ crop and Original w/ crop to sanity
    %check everything
    figure (1)
     imagesc(fullSizeSalMap)
     colorbar
     figure(2)
     imagesc(randSaliencyIm)
     colorbar
     figure(3)
     imagesc(thisImage)
     figure(4)
     imagesc(thisImage(Y1:Y2,X1:X2,:))
end

SaliencyDataForTTest = [maxDiagSaliency; maxRandSaliency;...
    meanDiagSaliency; meanRandSaliency];
save SaliencyDataForTTest SaliencyDataForTTest

 %Save Data to CSV file
csvwrite('SaliencyDataForTTest.csv', SaliencyDataForTTest');
```

```
%FileName: WholeBrainICA_SVM.m
%Inputs:CatCNDOrderLists_1-17, normalizedICAFeaturesXXX.mat
%Outputs: decodeDiagXXX_ICAwholeBrain.mat
%Summary: Decode whole brain ICA features to find time resolved
%decoding Accuracy

clear all; close all;

% step 1: make sure LIBSVM is in your path
% ex: addpath('/data/libsvm-3.22/matlab');
```

```
% step 2: load the dataset of interest
% needs: 64 channel x N time points x M trials
% .      vector of length M that indicates trial type
% note: in this demo code, I am calling the former "dataMat" and the
%latter "trialVector"

% dataMat = EEG.data;


load CatCndOrderLists_1-17;
%load normalizeICAFeatures004.mat;
trialVector = sbj17(:,1);
thisCond = sbj17(:,2);
trainInds = find(thisCond==0);
test1Inds = find(thisCond==1);
test2Inds = find(thisCond==2);
dataMat = icaDataNorm; %Load in "normalizeICAFeaturesXXX.mat"

% step 3: define cross validation partitions (here, using 10-fold)
%indices = crossvalind('Kfold',trialVector,10);

% step 4: loop through the different partitions, one electrode at a
%time in a sliding window of 40 ms.

for timePoint = 1:350; %fill this in: should be end of epoch - 2
    thisWindow = squeeze(dataMat(:, timePoint, :));
    thisWindow = double(thisWindow);

    % complete test 1
    trainData = thisWindow(:,trainInds);
    testData = thisWindow(:,test1Inds);
    model = svmtrain(trialVector(trainInds),trainData','-t 0');
    [predLabel,accuracy,dv] = svmpredict(trialVector(test1Inds),...
        testData', model);
    acTotal = accuracy(1);
    decodeDiag(timePoint) = acTotal;

    % complete test 2
    testData = thisWindow(:,test2Inds);
    model = svmtrain(trialVector(trainInds),trainData','-t 0');
    [predLabel,accuracy,dv] = svmpredict(trialVector(test2Inds),
        testData', model);
```

```
    acTotal = accuracy(1);
    decodeRand(timePoint) = acTotal;

save decodeDiag017_ICAwholeBrain decodeDiag decodeRand
```

```
%FileName: WholeBrianICA_findOnsetPeakPeakLat.m
%Inputs: decodeDiagXXX_ICAwholeBrain
%Outputs: decodeAccuracyWholeBrainICA.csv
%Summary: Find the onset, peak and peak latency of
%decoding accuracy.

myData = dir('*.mat'); %load in dataset
for j = 1:length(myData)
    thisData = importdata(myData(j).name);
    diagWholeBrainICAMat(j,:) = thisData.decodeDiag;
    randWholeBrainICAMat(j,:) = thisData.decodeRand;
end


% Get Diag Onset,Peak,PeakLat
for i = 1:16 %for each participant

    %load/defineData
    data = squeeze(diagWholeBrainICAMat(i,:));
    preImageData = data(1:100);

    % find bounds
        thisData = preImageData;
        % compute 1000 bootstrap samples
        sampData = bootstrp(1000,@mean,thisData);
        % get empirical CI
        bounds = prctile(sampData,[2.5 97.5]);
        upperBound = bounds(2);


    % identify onset of statistically significant activity
        thisData = data(101:end);
        a = find(thisData > upperBound);
        adiff = diff(a);
        peak(i) = max(thisData); %find peak
        findPeakLat = find(thisData == max(thisData));
        peakLatency(i) = findPeakLat(1); %find peak latency
```

```
        %find onset: (defined as first time point at which decode
        %accuracy for that time point and next five time points
        %are all above upper bound)
        for k = 1:length(adiff) %if there aren't 5 in a row, set time
        %point equal to zero
            if k+4 > length(adiff)
                onset(i) = 0;
                break;
            else
                if sum(adiff(k:k+2)) == 3
                    onset(i) = a(k+1);
                    break
                end
            end
        end
end

diagAccuracyWholeBrainICA = [onset', peak', peakLatency'];
clearvars -except diagAccuracyWholeBrainICA diagWholeBrainICAMat
    randWholeBrainICAMat


%Get Rand Onset,Peak,PeakLat
for i = 1:16 %for each participant

    %load/defineData
    data = squeeze(randWholeBrainICAMat(i,:));
    preImageData = data(1:100);

    % find bounds
        thisData = preImageData;
        % compute 1000 bootstrap samples
        sampData = bootstrp(1000,@mean,thisData);
        % get empirical CI
        bounds = prctile(sampData,[2.5 97.5]);
        upperBound = bounds(2);


    % identify onset of statistically significant activity
        thisData = data(101:end);
        a = find(thisData > upperBound);
        adiff = diff(a);
```

```
        peak(i) = max(thisData); %find peak
        findPeakLat = find(thisData == max(thisData));
        peakLatency(i) = findPeakLat(1); %find peak latency

        %find onset: (defined as first time point at which decode
        %accuracy for that time point and next five time points
        %are all above upper bound)
        for k = 1:length(adiff) %if there aren't 5 in a row, set time
        %point equal to zero
            if k+4 > length(adiff)
                onset(i) = 0;
                break;
            else
                if sum(adiff(k:k+2)) == 5
                    onset(i) = a(k+1);
                    break
                end
            end
        end
end

randAccuracyWholeBrainICA = [onset', peak', peakLatency'];


AccuracyWholeBrainICA = [diagAccuracyWholeBrainICA,
    ...randAccuracyWholeBrainICA];
csvwrite('decodeAccuracyWholeBrainICA.csv', AccuracyWholeBrainICA');
```

```
%FileName: NormEEGVoltage_findOnsetPeakPeakLat.m
%Inputs: decodeDiagXXXNorm.mat
%Outputs: rand/diag NormDecodedForANOVA.csv
%Summary: Find the onset, peak, peak lat of decoding accuracy
%for frontal, temporal and occiptial electrode clusters
%(Run once for diag, once for rand. change labels appropriatly)

  Data = dir('*.mat'); %load in dataset ("decodeDiagXXXNorm.mat")
for j = 1:length(myData)
    thisData = importdata(myData(j).name);
    decodeNormDiagMat(j,:,:) = thisData.decodeDiag;
    decodeNormRandMat(j,:,:) = thisData.decodeRand;
end
```

```
for i = 1:16 %for each participant

    %load/defineData
    data = squeeze(decodeNormRandMat(i,:,:));
    preImageData = data(:,1:100);

    % find bounds for each electrode
    for electrode = 1:63
        thisData = preImageData(electrode,:);
        % compute 1000 bootstrap samples
        sampData = bootstrp(1000,@mean,thisData);
        % get empirical CI
        bounds = prctile(sampData,[2.5 97.5]);
        upperBound(electrode) = bounds(2);
    end

    % identify onset of statistically significant activity
    for electrode = 1:63
        thisData = data(electrode,101:end);
        a = find(thisData > upperBound(electrode));
        adiff = diff(a);
        peak(electrode,i) = max(thisData); %find peak
        findPeakLat = find(thisData == max(thisData));
        peakLatency(electrode,i) = findPeakLat(1); %find peak latency

        %find onset: (defined as first time point at which decode
        %accuracy for that time point and next five time points
        %are all above upper bound)
        for k = 1:length(adiff) %if there aren't 5 in a row,
            %set time point equal to zero
            if k+4 > length(adiff)
                onset(electrode,i) = 0;
                break;
            else
                if sum(adiff(k:k+4)) == 5
                    onset(electrode,i) = a(k+1);
                    break
                end
            end
        end

    end
```

```
end

%define electrode cluster
 frontal=[1 2 3 29 30 31 32 33 34 35 36 59 60 61];
 temporal=[4 8 9 20 25 26 37 41 54];
 occipital=[15 16 17 45 46 47 48 49 63];

 %Find Diag/Rand onset, peak, peak lat for all participants
 randFntOnset = onset(frontal,:);
 randTmpOnset = onset(temporal,:);
 randOccOnset = onset(occipital,:);
 randFntPeak = peak(frontal,:);
 randTmpPeak = peak(temporal,:);
 randOccPeak = peak(occipital,:);
 randFntPeakLat = peakLatency(frontal,:);
 randTmpPeakLat = peakLatency(temporal,:);
 randOccPeakLat = peakLatency(occipital,:);

 %Find mean Diag/Rand onset, peak, peak lat.
 randFntOnset = mean(randFntOnset,1);
 randTmpOnset = mean(randTmpOnset,1);
 randOccOnset = mean(randOccOnset,1);
 randFntPeak = mean(randFntPeak,1);
 randTmpPeak = mean(randTmpPeak,1);
 randOccPeak = mean(randOccPeak,1);
 randFntPeakLat = mean(randFntPeakLat,1);
 randTmpPeakLat = mean(randTmpPeakLat,1);
 randOccPeakLat = mean(randOccPeakLat,1);

 randNormDecodedForANOVA = [randFntOnset; randTmpOnset; randOccOnset;
    ...randFntPeak; randTmpPeak; randOccPeak; randFntPeakLat;
    ...randTmpPeakLat; randOccPeakLat]

csvwrite('randNormDecodedForANOVA.csv', randNormDecodedForANOVA');
```