Reading Minds: Predicting Image Categorization from fMRI Data

Alexander Carlisle, Roger Song, Morgan Tanasijevich
waba,rsong17,mtanasij

1 Introduction

Our perceptive abilities are truly incredible. When we see a house, regardless of its color, size or shape, we know it's a house. As obvious as this seems, the vast majority of us will take these abilities for granted. For instance, what if we couldn't distinguish cars and buildings? What if we couldn't even recognize friends and family, and mistook them for strangers? This condition is actually termed agnosia, and it's easy enough to imagine how different our lives would be if we couldn't perceive properly.

With how remarkable our brains are, it's a wonder that we know so little about neuroscience. However in recent years, much more focus has placed on this field, with object recognition at the forefront. While major theories have developed about how the brain processes information, nothing has been officially proven. There is evidence that suggests object recognition is attributed to modularization in the ventral temporal area (meaning certain areas of the brain are responsible for recognizing different stimuli), but there are also theories for it being a result of distributed systems (meaning all areas of ventral temporal cortex contribute in object recognition).

For this project, we decided to implement machine learning algorithms, in an attempt to see if we could find correlations between specific areas of the brain and object recognition processes, or to at least gather more concrete evidence as to how our cerebral functions are carried out. Specifically, we wanted to see if we could predict the visual stimuli a subject was looking at given their fMRI scans while viewing the image. fMRI scans measure blood flow across the brain, which correspond to brain activity/stimulation. If we are correctly able to predict visual stimuli, our results could prove or at least further support the modularity of our brain in object recognition, as well as help us better understand how we as humans function (which we believe is a goal we should always strive for). Creating algorithms to achieve these results could also have far more practical effects as well, such as enabling or bettering communication with the physically impaired. If someone's in a coma, fMRI scans could give us crucial information about the state of thoughts of that person. What we're trying to achieve is essentially mind-reading (to a much lesser degree), and we'll leave it to the reader to imagine the many other endless possibilities.

2 Data

Our data was given to us by graduate student Lior Bugatus of the Stanford Psychology department. Due to fMRIs being a time consuming and expensive process we expected and acknowledge a scarcity of data.

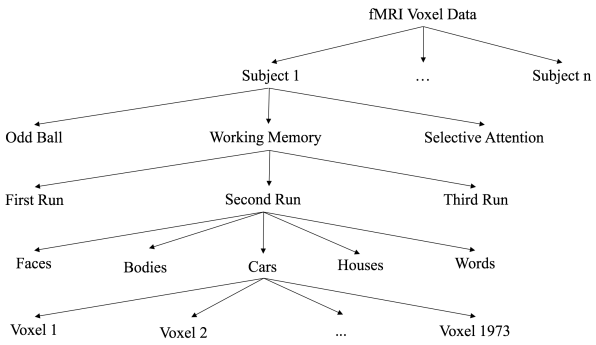


Figure 1. Tree representation of the data

The examples were split across four subjects. Each subject was given one of five visual stimulus categories (faces, bodies, houses, words, cars) while performing one of three tasks (Odd Ball, Selective Attention, Working Memory - the details of which are not important to our project) and asked to perform three trials, for a total of 180 examples.

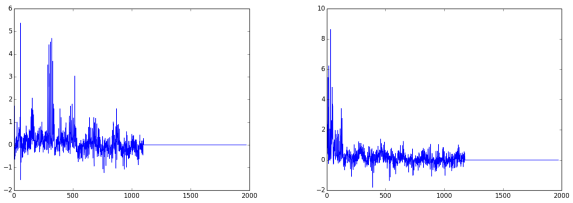


Chart 1. Voxel amplitudes of the original data for two random examples.

For each example, the fMRI data we were given was formatted as a 1973 element vector, each representing a voxel (specific area of the brain) and the corresponding amplitude of blood flow to that voxel (brain activity).

As a part of processing the data and as a way to allow for equal weighting of each voxel number, we normalized the mean and variance of each voxel, as follows:

1. $\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$
2. Replace each $x^{(i)}$ with $x^{(i)} - \mu$
3. Let $\sigma_j^2 = \sum_i \left(x_j^{(i)}\right)^2$
4. Replace each $x_j^{(i)}$ with $x_j^{(i)}/\sigma_j$

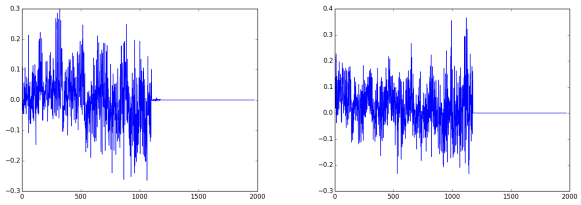


Chart 2. Voxel amplitudes of the normalized data for the same two random examples.

However, after normalizing the data, we noticed that the normalized data actually added more variance and noise to each individual example. Therefore, we wanted to eliminate this unwanted noise through data smoothing. Because, our voxels are presumably dependent on their neighboring voxels, we modified the amplitude of a specific voxel to be a weighted average of the closest C voxels by index. More specifically,

For every voxel v_i in a data example x:

$$v_i := \frac{\sum_{j=0}^{C} \frac{v_{(i+j)} + v_{(i-j)}}{j+1}}{2C}$$

Our data after our hybrid data smoothing algorithm are presented below (both before and after normalizing).

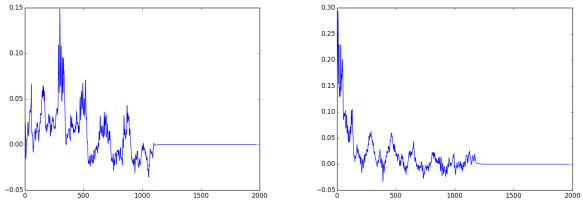


Chart 3. Voxel amplitudes of the smoothed data for the same two random examples.

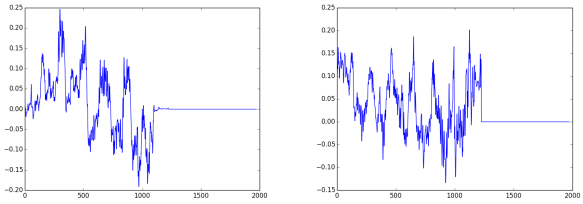


Chart 4. Voxel amplitudes of the smoothed data after normalization for the same two random examples.

The noise reduction technique was able to vastly reduce the amount of deviation in the data, which we hoped would help prevent or limit the magnitude of overfitting for our different algorithms.

Our data also had a large number of features (1973), especially when compared to the number of training examples. We ran principal component analysis in an attempt to reduce the feature space. However, it did not improve results for any number of principal components. The likely cause of this is the amount of noise produced by the process of fMRI scanning.

3 Algorithms

In full disclosure, we used *k-means*, *Gaussian Naive Bayes*, and *k-nearest neighbors* for CS 229, however our implementation of logistic regression, neural network, and correlation classifications was used for just this project.

Categories	NB	1-NN	K-NN	Neural
Faces, Bodies	71%	79%	56%	67%
Faces, Cars	75%	76%	60%	78%
Faces, Houses	96%	89%	65%	78%
Faces, Words	100%	88%	71%	72%
Bodies, Cars	83%	83%	57%	83%
Bodies, Houses	79%	82%	63%	50%
Bodies, Words	96%	83%	69%	83%
Cars, Houses	75%	88%	63%	39%
Cars, Words	92%	86%	65%	39%
Houses, Words	71%	56%	61%	50%
Average	84%	81%	63%	64%

Table 1. Binary classification accuracies.
NB - *Gaussian Naives Bayes*
1-NN - *Nearest Neigbhor*
K-NN - *K-Nearest Neigbhor*
Neural - *Backpropogation Neural Network*

3.1 K-Means Clustering

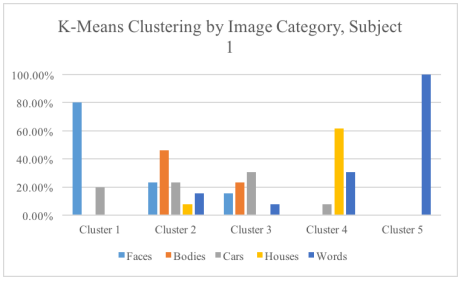


Chart 5. *K-means* clustering by image category.

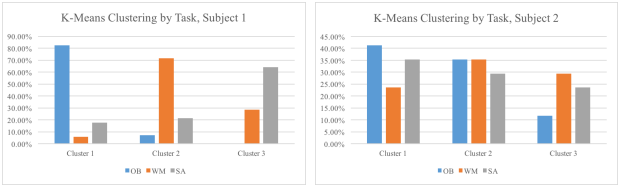


Chart 6. *K-means* clustering by task.

K-means attempts to find natural clusters of data by:

1. Initializing **centroids** $\mu_1, \mu_2, ..., \mu_k$
2. Repeating until convergence: {

   For every i, set $c^{(i)} := arg\min_j ||x^{(i)} - \mu_j||^2$

   For each j, set $\mu_j := \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$

   }

We ran *k-means* with 5 centroids and 3 centroids in an attempt to see if our data could be clustered by task and by image. Without going into too much detail, we found that for some subjects the data clustered nicely by tasks, but for others it did not. We also found that the data was still able to be clustered by image category across multiple subjects, but clustered into better categories when we just ran kmeans on one subject, and clustered almost completely by image category if we only included data from one subject and one task alone. We attribute this to the (somewhat trivial) notion that subjects can't be expected to perform identically in terms of focus and memory capabilities, and that noise due to these functionality differences is also inevitable. Our clusters are more precise in determining separations as the amount of noise across tasks and subjects is removed. As we can see in the charts below, we cluster better by training and testing on a single task and even better when conditioning on a single task and subject, rather than across all tasks and subjects.

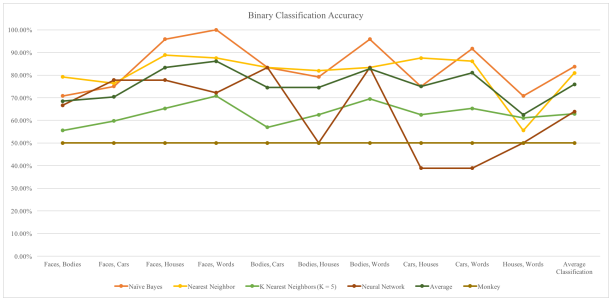


Chart 7. Binary classification accuracies for all image category pairs.

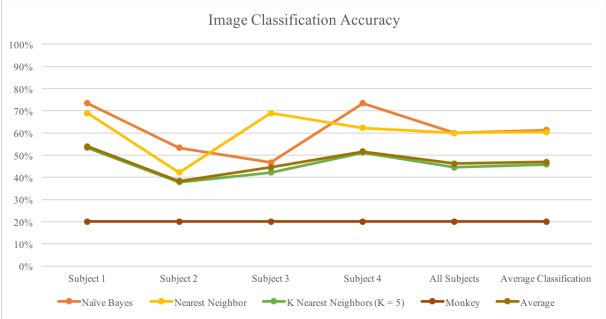


Chart 8. General image multi-classification accuracies.

3.2 Correlation Clustering and Classification

We wondered if we could classify examples based on correlations (between categories) alone. The idea is to find (if they exist) certain pairs of categories, that might be more correlated and are thus recognized better by similar groups of voxels. We first calculate the average correlation between all possible pairs of categories and tasks (for instance the average correlation between all face-word pair examples), and then plotted these on a heat map (where lighter colors signify high correlation and darker colors signify low correlation).

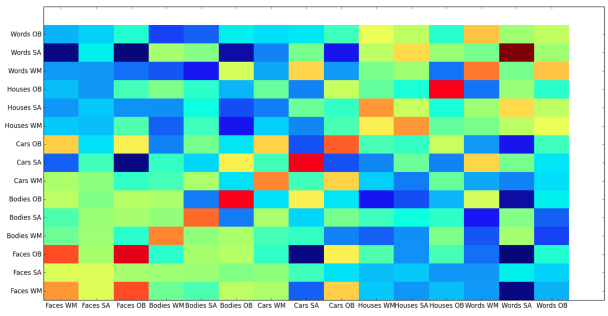


Chart 9. Correlations across categories and tasks.

Above is the a heat map across all categories and tasks. We can see that while face examples obviously correlate well with other face examples, some category pairings are also inherently better correlated than others. We can see that face and body examples are similar as shown by the light colors, whereas face and word examples are very unrelated, as indicated by the darker blue colors. We also found that correlations, even for different categories, were better within task (such as Houses OB and Faces OB). This is likely a result of reducing noise across tasks.

Because of the subject and task noise that we just mentioned and also identified with K-Means clustering, we decided to fix the subject and task and even see how well we could classify examples. We first calculated the correlations between our test example (unknown category) and all image categories in our training set. Then, we compared those correlations to each of the correlation rows in our heat map. Finally, we classified our test example as the category which minimizes the sum of the distances between our test correlations and its correlation row.

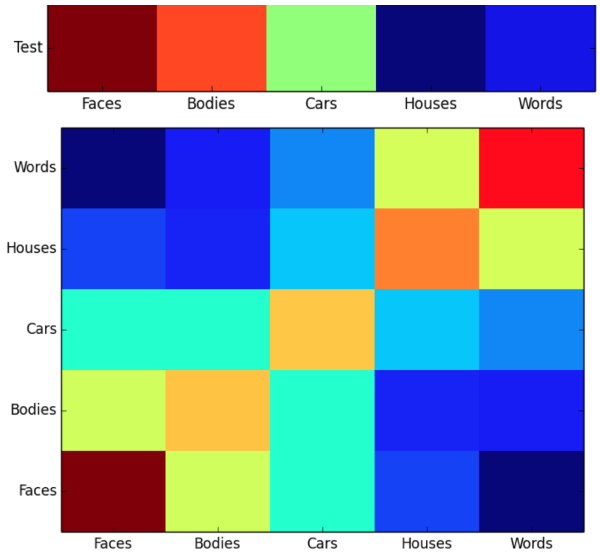


Chart 10. Classifying test examples.

In the heat image/map above for example, the image category that minimizes this "loss" would be the row that is closest in color (heat) to our test row. In this case, it would be the faces row, so we would categorize our test point as a face example.

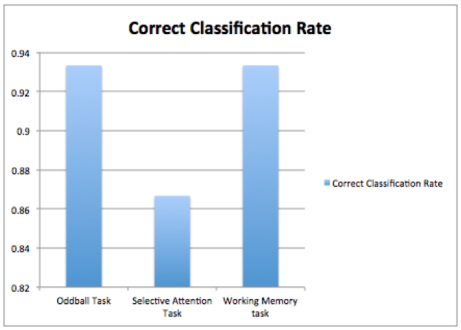


Chart 11. Correlation Algorithm classification accuracies.

This graph shows that given a fixed subject (subject four in this case) and task (which represents the majority of the noise in our data), and matching up correlations as we described above, we can achieve high multi-classification rate through looking at these correlation scores. Unfortunately, across all subjects and all tasks, the excessive noise caused this classification algorithm to perform poorly enough to be infeasible.

3.3 Logistic Regression

We begun by running *logistic regression* to provide a good baseline in categorizing our fMRI data for future algorithms. Because we knew we would implement a neural network later, we implemented *logistic regression* as a neural network without any hidden layers, so that we were just optimizing one synapse (or weight vector θ of the features) with a sigmoid function $\sigma(z)$ where:

$$\sigma(z) = \frac{1}{1 + e^{(-z)}}$$

Then, to compute the optimal weight vector, we ran stochastic gradient descent until convergence by:

$$\theta_j := \theta_j + \alpha(y^{(i)} - \sigma(\theta^T x^{(i)}))x_j^{(i)}$$

To classify the testing data, the assignment is simply equal to the $sign(\theta^T x)$.

We tested this algorithm in a number of ways. First we isolated subjects. For a given subject, we trained our weights across all examples (all five categories and all three tasks) corresponding to the first two runs and tested on examples corresponding to the third run. It performed well, classifying nearly every test example correctly.

Then, we ran our algorithm within all subjects. So we trained again on the first two runs, but now incorporated all subjects, and then tested on the third run for our examples. We correctly classified about 80% of our testing examples, which is still well above the random baseline of 50% obtained when guessing. We attributed this dip in performance to noise across subjects due to differences in focus or memory that are beyond our control. However, because the three runs for a given task and category are likely to produce extremely similar results for a given subject, and we classified based on data that's already been seen before essentially, we considered this a good indicator of our training error.

We then decided to run our algorithm across subjects. We trained on all subjects except for one, and then tested on the one we left out. Because of subject noise and the fact that we we didn't train on previous runs for that subject, we expected this approach to perform worse. However, it performed much worse than expected, achieving a 50% classification rate, which is no better than guessing. Because we are trained on an unseen subject, we considered this a good indicator of our testing error. The 30% between our training and testing error suggested that we were overfitting the data, likely because the data is not linearly separable and our small dataset.

3.4 Neural Network

To combat overfitting, we decided to implement a three layer *neural network* (input, hidden layer, output) through backpropagation training, compared to *logistic regression* which is essentially a two layer network (input, output). To test how well our regression fit the data, we considered pairs of voxels as features, which would be more appropriate for non-linear data. This also makes sense on a broader level because areas close together in the brain are likely to have similar functionality, if we assume the brain is at least somewhat modularly structured.

At each intermediate level h_j and corresponding weight vector v_j:

$$h_j = \sigma(v_j^T x)$$

Then, to produce a final score:

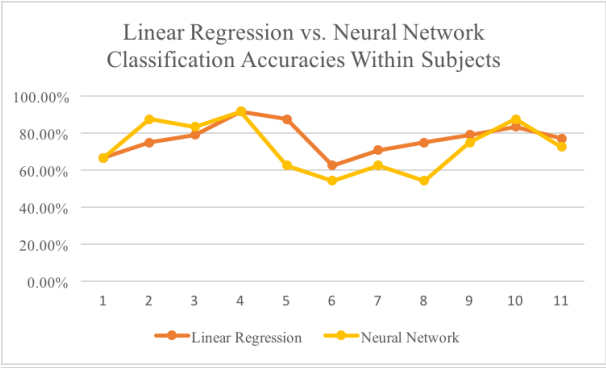
$$score = w^T h$$

The assignment of the testing example x given the final weight vector w is equal to:

$$y^{(i)} = sgn(\sigma(w^T h))$$

We performed the same three tests as we did with *logistic regression*. Training on an isolated subject's run one and two examples and testing on the third performed similarly - near perfection.

When we expanded our training and test sets across all subjects, we performed worse, with an a 72% classification rate. Our training error increased by about 8%. When we ran our *neural network* algorithm across subjects by testing on the one subject that was not used for training (leave-one-out), we achieved much better results. We correctly classified approximately 65% of the testing data, which represents a 15% decrease in the testing error. The difference in our errors is now 7%, which overfits much less. The graphs below compare these error rates and visually show how a *neural network* approach provides more consistent error rates. With only one additional layer, even though voxel readings will vary across subjects, we see that the involvement of certain pairs of voxels in category recognition is much more uniform and consistent across subjects (cancels noise) and that our data is likely not linearly separable. Because of these results, we decide to learn more about how groups of voxels are correlated with category recognition.



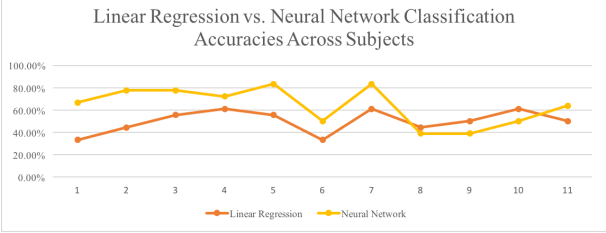


Chart 12. *Linear Regression* vs *Neural Network* accuracies

3.5 K-Nearest Neighbors

Again we will briefly mention another algorithm we used in our 229 version of this project. For K Nearest Neighbors, given an unclassified example, we search for the most similar example in the dataset and classify the test example as its "nearest neighbor". When determining the nearest neighbor, we used the Euclidean distance to measure this "closeness" where $d(i,j) = ||x^{(i)} - x^{(j)}||_2^2$.

We extend this to look at the k-closest neighbors and choose the image category that appeared most frequently within the k-closest neighbors for a range of k-values. More specifically, we selected the top k-closest neighbors to the current example and then selected the image category of one of the examples randomly with probability weights equal to $\frac{1}{d(i,j)}$. On the entire data set, our algorithm was 60% correct in classifying each image category (compared to a baseline of 20%) and over 80% correct on average for classifying pairs (compared to a baseline of 50%).

We used the leave-one-out training and testing approach for different examples, which worked quite well when we left out one example at a time. We suspected this might be susceptible to overfitting because for any given example, there are two other runs/trials for the same subject, task and category. This meant our algorithm would likely choose the nearest of those two runs. This in turn implies that if we were to test on an unseen subject, (which we did and it was just as bad as guessing), we would only be comparing an example for that subject with examples from different subjects, which we already determined was noisy.

3.6 Gaussian Naive Bayes

Our implementation of the *Gaussian Naive Bayes* algorithm attempts to categorize a new vector of fMRI data by choosing the image category that had the highest probability given the newly observed training data. One strong assumption *Naive Bayes* makes is that for a given

image category, it assumes each feature is independent from every other feature, thus making the likelihood of the data:

$$p(y) = \prod_{x \in X} p(x, y)$$

Furthermore, since $p(x, y) = p(x|y)p(y)$ and in our case each image category was equally likely to occur thus making $p(y)$ a constant, and $p(x)$ is unchanging, we can simplify $p(y|x) = \frac{p(x|y)p(y)}{p(x)}$ to $p(y|x) = p(x|y)$ for all voxels. Now for the probability of a voxel's amplitude given a category, we assumed the voxel's amplitudes were normally distributed within an image category and therefore the calculated probability of voxel x having amplitude μ as:

$$p(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}.$$

or the *probability density function*.

Seeing that knowing the amplitude of one voxel quite likely informs upon the amplitude of the neighboring voxels because of the modularity association we discovered earlier, *Naive Bayes*' crucial assumption that voxel amplitudes are independent of each other is a little dubious.

Consequently, we were surprised to see that our implementation of *Naive Bayes* produced quite promising results. For binary classification when dividing our splitting our dataset by trial run, we observe testing errors of less than 20% overall, and error rates of less than 10% when also conditioning an individual subject. For general image classification, the result was a testing error less than 40%.

4 Conclusion

Our feature set contained nearly 2,000 different voxels, however, many of these voxels we determined early on were similar and therefore can be grouped together as more dense features. We ran *PCA* initially to try and remove this variability across similar voxels, but there was extra noise due to task and subject variability, and so we could not identify these groupings. So while we were able to identify that voxel groupings were better features when we used a *neural network*, we were not able to identify which specific voxels were most correlated to each other.

In a similar effort, we calculated correlations between categories. One of our best findings was that face and body stimuli voxel readings were highly correlated, especially when constrained to a certain task or subject. This made us look at our k-means clusters again, and we realized that many of our clusters grouped these two category examples together. What we initially thought were incorrect groupings turned out to not be so bad. In one cluster for example, we had 6 face examples and 3 body examples. Even though the cluster didn't comprise of all face examples or all body examples, it still showed that face and body processing are similar in terms of brain activations.

Like many other fMRI studies, our data has shown to be highly subject dependent as all our classification algorithms worked better when we had training data and test data from the same subject. We attribute the differences across subjects to two primary causes. First, human brains are shaped slightly differently and so what is voxel 700 on subject 1 may correspond to a different voxel number on subject two. Secondly, if different subjects paid differing levels of attention to the tasks it would be responsible for adding in extra noise.

The success rates we achieved are on par with many of the current classification algorithms in the field, but we were limited by the amount of data we had. While other papers utilized data with multiple time readings to significantly increase the size of the dataset, we were only given one snapshot or time series in our data. We planned on getting fMRI scans on ourselves to provide additional data, but we could not coordinate this with a psychology professor within the time constraints of this project. We are confident that more data could have pushed this classification rate up or at least, in the case of our neural network, establish a more concrete and confident bound on what classification rates can be achieved by better fitting the data.

5 fMRI Machine Learning Literature

Our project is not the first time that machine learning fMRIs has attempted to classify object recognition in the brain. Many of these other studies have time series of voxel readings, but we were only given a single snapshot as I mentioned earlier. In addition, *Support Vector Machines* have shown a lot of success in classifying and separating brain fMRI data in other papers [1]; however, our implementation of a *SVM* did not perform as well as our other algorithms initially, so we decided not to pursue it further in this project [2]. A pattern–correlation classifier was also used by Haxby [3] and was instrumental in providing evidence for the object form topography theory of object recognition. We implemented a similar pattern-correlation classifier and found it was very successful when testing on an isolated subject, but was performed barely better than guessing categories for examples. *Gaussian Naive Bayes* has also been used as a classifier for fMRI data [4] and was one of the most successful algorithms we ran as shown in our results section. *K-Nearest Neighbors* was used by Mitchell in 2004.

References

[1] Cox DD, Savoy RL. *Functional Magnetic Resonance Imaging (fMRI) "Brain Reading": Detecting and Classifying Distributed Patterns of fMRI Activity in Human Visual Cortex*. NeuroImage, 2003.

[2] LaConte S, Strother S, Cherkassky V, Anderson J, Hu X. *Support vector machines for temporal classification of block design fMRI data*. NeuroImage. 2005

[3] Haxby JV, Gobbini MI, Furey ML, Ishai A, Schouten JL, Pietrini P. *Distributed and overlapping representations of faces and objects in ventral temporal cortex*. Science. 2001

[4] Mitchell TM, Hutchinson R, Niculescu RS, Pereira F, Wang X, Just M, Newman S. *Learning to Decode Cognitive States from Brain Images*. Machine Learning. 2004

[5] Trask Andrew. *A Neural Network in 11 lines of Python (Part 1)* `http://iamtrask.github.io/2015/07/12/basic-python-network/`

[6] Liang Percy. *CS 221 Course Material*

[7] Ng Andrew. *CS 229 Course Material*