Thought Recognition: Predicting and Decoding Brain Activity Using the Zero-Shot Learning Model

Mark M. Palatucci

CMU-RI-TR-11-05

April 25, 2011

Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Tom M. Mitchell, Chair
Dean Pomerleau
J. Andrew Bagnell
Andrew Ng, Stanford University

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Keywords: functional magnetic resonance imaging, machine learning, zero-shot learning, thought recognition

To my grandmother, who always defended my right to have ice cream.

Abstract

Machine learning algorithms have been successfully applied to learning classifiers in many domains such as computer vision, fraud detection, and brain image analysis. Typically, classifiers are trained to predict a class value given a set of labeled training data that includes all possible class values, and sometimes additional unlabeled training data.

Little research has been performed where the possible values for the class variable include values that have been *omitted from the training examples*. This is an important problem setting, especially in domains where the class value can take on many values, and the cost of obtaining labeled examples for all values is high.

We show that the key to addressing this problem is not predicting the held-out classes directly, but rather by recognizing the semantic properties of the classes such as their physical or functional attributes. We formalize this method as *zero-shot learning* and show that by utilizing semantic knowledge mined from large text corpora and crowd-sourced humans, we can discriminate classes without explicitly collecting examples of those classes for a training set.

As a case study, we consider this problem in the context of thought recognition, where the goal is to classify the pattern of brain activity observed from a non-invasive neural recording device. Specifically, we train classifiers to predict a specific concrete noun that a person is thinking about based on an observed image of that person's neural activity.

We show that by predicting the semantic properties of the nouns such as *"is it heavy?"* and *"is it edible?"*, we can discriminate concrete nouns that people are thinking about, even without explicitly collecting examples of those nouns for a training set. Further, this allows discrimination of certain nouns that are within the same category with significantly higher accuracies than previous work.

In addition to being an important step forward for neural imaging and brain-computer-interfaces, we show that the zero-shot learning model has important implications for the broader machine learning community by providing a means for learning algorithms to extrapolate beyond their explicit training set.

Funding Sources

Financial support for this work was provided by the National Science Foundation, W.M. Keck Foundation, and Intel Corporation. Mark Palatucci is supported by graduate fellowships from the National Science Foundation and Intel Corporation.

Acknowledgments

Whoever said *the journey is the reward* was right. Despite being proud of the results in this thesis, the real reward of graduate school was the experience itself. The ability to think really deeply about a topic for so long fundamentally changes the way a person perceives the world. The experience has changed me considerably, and I can honestly say the hard choice of committing to so many years of study was the best choice I ever made.

Of course there were bumps in the road: the occasional self doubt as to whether I really belonged here, code bugs that turned groundbreaking results into mush, and delusions of grandeur shattered by the reality that most of humanity doesn't care about the work we do or why it's important. At least for now.

Despite the occasional setback, my experience at Carnegie Mellon has been nothing short of incredible. This is largely due to the talented group of people that I've been able to interact with, many of whom deserve an enormous debt of gratitude.

First, it should be noted the people that led me to Carnegie Mellon in the first place. From my undergraduate days at the University of Pennsylvania, Norm Badler gave me my first taste of research while Max Mintz taught me the beauty of number theory, proof by intimidation, and countless other things. Tom Cassel taught me about the most important word in English, *trust*. All three supported me immensely, both during my undergraduate days and also during the stressful graduate application process.

My journey into robotics, machine learning, and ultimately Carnegie Mellon really started thanks to Sebastian Thrun, Mike Montemerlo, and David Stavens, who graciously let me be a part of the Grand Challenge team at Stanford University. This was despite the fact that I didn't even know what a multivariate Gaussian was at the time. Sebastian gave me several of the best pieces of advice I ever received, and made me comfortable with the fact the life is not supposed to take a linear path from point A to point B.

I also need to thank Jessica Hodgins, who saved my CMU application from the garbage heap and realized I belonged at the Robotics Institute. She was right.

The Robotics Institute is an incredible place, probably the closest real world equivalent of Harry Potter's Hogwarts. It's filled with crazy-smart, good-natured people whose work would be called magical by many. I'll probably never be so lucky to work with such a diverse and interesting set of characters again.

I can't imagine any RI thesis acknowledgement not mentioning Suzanne Muth, who never misses a single detail and who singlehandedly provides the glue that ties the universe together. Yes, we all sing songs about you. And to Matt Mason, who could make red tape disappear with emails containing the two magic words: *sounds good*.

To my RI crew: Geoff Hollinger, Tomasz Malisiewicz, Boris Sofman, and Hanns Tappeiner, your friendship has been, by far, the most important part of my life for the last five years. May we spend many more days pondering the infinite on the trails of La P'tit Bonheur and Mount Qua-Qua. And to Jonathan Huang, the first person who really made me understand *expectation-maximization*, you break the laws of nature by being so smart *and* so kind.

I also need to thank Geoff Hinton for his incredible wit, humor, and for teaching me not to fear non-convex loss functions. He welcomed me into his laboratory during my extended visit to the University of Toronto during the Summer of 2008, and much of the work in this thesis resulted from our collaborations there. Many thanks to the other UofT folks who also made my visit a great experience: Rich Zemel, Graham Taylor, Laurens van der Maaten, Arnold Binas, Ilya Sutskever, and Dustin Lang.

Many thanks to the super smart individuals whom I collaborated with, both in my lab group and across CMU: Francisco Pereira, Rebecca Hutchinson, Indra Rustandi, Kai-min Chang, Marcel Just, Gus Sudre, Alona Fyshe, Leila Wehbe, Andy Carlson, Han Liu, Ajit Singh, Geoff Gordon, Burr Settles, and Khalid El-Arini.

I can honestly say that this work would not have been possible without the resources and support of Intel Labs Pittsburgh. Their support came in many ways: cloud computing infrastructure that let me run experiments on hundreds of cores, a Ph.D Fellowship that freed me to focus on the most important questions, and an extremely comfortable space to work in. Thanks to Dave O'Hallaron, Limor Fix, Jason Campbell, Rahul Sukthankar, and especially to Richard Gass and Mike Ryan for their help navigating the Intel cluster.

And of course, to Dean Pomerleau, to whom I owe so very much. Dean brings such enormous energy and creativity to whatever he touches, collaborating with him has been enormous fun. His advice, both on a personal and professional level has been invaluable, and in many ways made my graduate experience the best it possibly could have been. Our results would not have been nearly as exciting had he not magically appeared two years ago with his insights. He has been, in many ways, my *spirit guide*.

My advisor, Tom Mitchell, never ceases to amaze me with his enthusiasm and insight. He always let me work on the *big questions* and somehow kept the faith even when experiments failed. And somehow things always seemed to work out in the end. Being able to work with someone of such good nature, patience, and complete lack of cynicism was one of the greatest gifts of my life. It would be impossible to enumerate all the things he has taught me.

I also would like thank the rest of my thesis committee: Drew Bagnell, for the many impromptu discussions and whiteboard sessions, and to Andrew Ng, whose vision for artificial intelligence is a constant inspiration. I have been extremely lucky to have interacted with such world class scientists.

Lastly, I need to thank my family for their incredible support throughout all my crazy endeavors, especially my parents for giving me a love of learning and for teaching me the importance of education. My brother always offered a well needed laugh, something I particularly needed to keep sane while thesis writing during the freezing Pittsburgh winter. Katie helped me stay on task by guilting me off Facebook. And my mother, who has sacrificed much for my brother and me, has been an unwavering source of support.

Contents

1 Introduction **1**
- 1.1 Overview of Thought Recognition . . . 2
- 1.2 Thesis Statement and Contributions . . . 4
- 1.3 Overview of Functional Magnetic Resonance Imaging (fMRI) . . . 6
- 1.4 Overview of Other Brain Scanning Technologies . . . 9
- 1.5 Related Work . . . 10
  - 1.5.1 Thought Recognition and Cognitive State Classification . . . 10
  - 1.5.2 Zero-Shot and Attribute Based Learning . . . 11
  - 1.5.3 Feature Selection, Sparsity, and Multi-task Learning . . . 12
  - 1.5.4 Self-Taught Learning and Latent Variable Models . . . 12
- 1.6 Chapter Summaries . . . 13

2 Thought Recognition using Zero-Shot Learning **15**
- 2.1 Background . . . 15
- 2.2 The Zero-Shot Learning Model . . . 17
  - 2.2.1 Overview . . . 17
  - 2.2.2 Additional Related Work . . . 18
  - 2.2.3 Formalism . . . 19
  - 2.2.4 Theoretical Analysis . . . 20
- 2.3 Case Study: Classifying Concrete Nouns using fMRI . . . 26
  - 2.3.1 fMRI Imaging Paradigm and Task . . . 26
  - 2.3.2 fMRI Data Collection Procedures . . . 27
  - 2.3.3 Computing Cluster . . . 27
  - 2.3.4 Acquiring Semantic Knowledge Bases . . . 28
  - 2.3.5 Data Normalization and Preprocessing . . . 28
  - 2.3.6 Regression Model . . . 29
  - 2.3.7 Experimental Results . . . 30
- 2.4 Future Work . . . 40
- 2.5 Conclusion . . . 41
- 2.6 Additional Chapter Remarks . . . 42

3 Semantic Feature Selection using Support Vector Ranking **43**
- 3.1 Background . . . 44
- 3.2 Support Vector Ranking . . . 45

3.2.1 Model Formalism . . . 45
3.3 Experimental Results . . . 46
3.4 Relationship to Other Feature Selection Methods . . . 48
3.5 Theoretical Considerations . . . 49
3.6 Future Work . . . 54
3.7 Conclusion . . . 54
3.8 Additional Chapter Remarks . . . 55

4 Predicting Neural Activity using Multi-Task Learning 57
4.1 Introduction to the Neural Activity Prediction Problem . . . 58
4.2 Background on Coordinate Descent Optimization . . . 61
4.3 The Multi-task Lasso . . . 62
4.4 Blockwise Coordinate Descent . . . 63
4.5 Case Study: Neural Activity Prediction . . . 65
4.5.1 Datasets . . . 65
4.5.2 Experimental Results . . . 66
4.6 Visualizing Semantic Features for Individual Regions of Interest (ROIs) . . . 71
4.7 Discussion and Future Work . . . 72
4.8 Conclusion . . . 74
4.9 Additional Chapter Remarks . . . 74

5 Combining Data from Multiple Scanning Modalities 79
5.1 Introduction . . . 80
5.2 Canonical Correlation Analysis (CCA) . . . 81
5.3 Case Study: Combining Data from fMRI and MEG/EEG . . . 84
5.3.1 Paradigm and Task . . . 84
5.3.2 MEG/EEG Data Collection Procedures . . . 85
5.3.3 Data Norm and Preprocessing . . . 85
5.3.4 Model . . . 85
5.4 Experimental Results . . . 86
5.4.1 MEG and EEG Baseline Experiments . . . 86
5.4.2 kCCA Experiments with MEG/EEG and fMRI . . . 91
5.5 Future Work . . . 94
5.6 Conclusion . . . 94
5.7 Additional Chapter Remarks . . . 95

6 Conclusion 97
6.1 Summary of Contributions . . . 97
6.1.1 Thought Recognition using Zero-Shot Learning . . . 97
6.1.2 Semantic Feature Selection using Support Vector Ranking . . . 98
6.1.3 Predicting Neural Activity using Multi-Task Learning . . . 98
6.1.4 Combining Data from Multiple Scanning Modalities . . . 99
6.2 Software Contributions . . . 99
6.3 Future Work . . . 99

6.3.1 Thought Recognition using Zero-Shot Learning 100
6.3.2 Semantic Feature Selection using Support Vector Ranking 101
6.3.3 Predicting Neural Activity using Multi-Task Learning 102
6.3.4 Combining Data from Multiple Scanning Modalities 103

A Derivation of Update Rule for Support Vector Ranking **105**

B Correctness Proof of the Blockwise Coordinate Descent Algorithm for Solving the Multi-task Lasso **109**

C The `Human218` Semantic Features **115**

Bibliography **119**

List of Figures

1.1 Thought recognition is the application of pattern recognition and machine learning methods to discriminate patterns of brain activity. Like other applications of pattern recognition like speech and handwriting recognition, neural signals observed using scanners like fMRI contain enough structure that statistical methods can be used for analysis and classification. Image Credits: character recognition image from Neural Network (OCR) application by Andrew Kirillov, handwriting recognition image from the DioPen application by DioTek, speech image from Dan Jurafsky, Stanford University, and fMRI image from Tom Mitchell. 4

1.2 An image of the fMRI facility located at the Scientific Imaging and Brain Research Center, Carnegie Mellon University. Image courtesy of SIBRC. 6

1.3 A structural or anatomical image of the brain using MRI. Anatomical images display just the static physical structure of matter, and do not indicate areas of changing neural activity. 7

1.4 The plot on the left shows the hemodynamic response curve for fMRI. There is a 4-5 second lag between the stimulus onset when the strongest activation signal is recorded. The image on the right shows the BOLD response plotted on an anatomical MRI of the brain. Red colors indicate areas of stronger activity. . . . 7

1.5 A plot of the fMRI timeseries observed for each voxel in the brain. The image here represents one 2-D slice of the observed 3-D volume of activity. Image courtesy of Tom Mitchell. 8

1.6 An image of the *actiCap* EEG electrode cap from Brain Products GmbH (left) and a MEG scanner located at the University of Pittsburgh Medical Center (right). While EEG and MEG have temporal resolution nearly 1000 times that of fMRI, the spatial resolution is significantly worse, especially for EEG. The form factor and cost of EEG, however, is a small fraction of fMRI or MEG. As a result, EEG remains popular in cognitive neuroscience and brain-computer-interface research. EEG image courtesy of Brain Products GmbH and MEG image courtesy of Gus Sudre. 9

2.1 The semantic output code classifier (SOCC) is a two stage classifier (shown on right) compared to a typical single stage classifier (shown on left). The SOCC uses a layer of intermediate semantic features in between the input features and the class label. These semantic features represent attributes of the class labels. . . 17

2.2 This figure shows the relationships between input features X^d, the classes Y, and the intermediate semantic features F^p. For the purposes of this analysis, we assume that we can observe the output of the deterministic functions $\mathcal{H}^{-1}(\cdot)$ and $\mathcal{L}^{-1}(\cdot)$ for a given class drawn from an arbitrary distribution $\mathcal{Y}$. This provides samples from the input distribution $\mathcal{X}$ and also from the semantic feature distribution $\mathcal{F}$. In our PAC-based model, we learn the function $\mathcal{H}(\cdot)$ indirectly through $\mathcal{S}(\cdot)$ and $\mathcal{L}(\cdot)$. We analyze the performance based on the number of samples N drawn from the input distribution $\mathcal{X}$ and also from the number of examples M drawn from semantic feature distribution $\mathcal{F}$. 21

2.3 The event related paradigm. Line drawings were presented as white lines on black background (shown reversed here for clarity). 26

2.4 Percent accuracies for leave-two-out-cross-validation for 9 fMRI participants (labeled P1-P9). The values represent classifier percentage accuracy over 1,770 trials when discriminating between two fMRI images, both of which were omitted from the training set. The chance accuracy level is shown as a solid black line and the significance threshold (at the 5% level) is shown as the dotted black line. 31

2.5 Average accuracies when discriminating words in the same category in the *leave-two-out* prediction task. Results are averaged by category across nine fMRI participants. This result shows that using intermediate semantic features often allows words to be discriminated even when they are in the same category of object. 32

2.6 Average distances between words in the same category using the `human218` semantic features. Features for an individual word were normalized to unit length before comparisons were made. The categories *clothing, foods* and *tools* had the smallest semantic feature distances on average, and also the lowest within-category predictive accuracy. The categories *buildings* and *furniture*, however, had relatively small semantic feature distances between words, but these two categories had the highest within-category prediction accuracy, suggesting that features that discriminate words within these categories can be predicted more accurately than the features that discriminate other categories such as *clothing*. . 33

2.7 An image reproduced from the Online Supporting Material of Mitchell et al. [2008] that shows the cosine similarity between fMRI images for the 60 stimuli words. This is the same dataset used in our work. We see high similarity for images in the *buildings* category and also for the *tools* category. Interestingly, the *buildings* category had the highest predictive performance using our `human218` semantic features, while *tools* had the lowest. This is most likely the result of small distances between semantic features describing *tools* as well as higher predictive accuracy of semantic features that discriminate *buildings*. . . 35

2.8 Ten semantic features from the human218 knowledge base for the words *bear* and *dog*. The true encoding is shown along with the predicted encoding when fMRI images for bear and dog were left out of the training set. The Y-axis represents the value of the semantic features after scaling normalizations. 36

2.9 The mean and median rank accuracies across nine participants for two different semantic feature sets. Both the original 60 fMRI words and a set of 940 nouns were considered. 37

2.10 A 2-D visualization of the semantic feature space for the 60 fMRI stimulus words, along with the predicted feature vector for the word *bear* shown in red. The prediction for the word bear was closer to the true encoding for bear than for any other word. This two-dimensional plot was generated by applying the dimensionality reduction method *t-SNE* to the original semantic space of 218 dimensions. Circles were added to show the category information. 39

3.1 Rank accuracies for the top N weighted semantic features (solid line) and bottom N weighted semantic features (dashed line). The red lines are for the median rank accuracy while the blue lines are for the mean rank accuracy. Note that eventually the sets contain the same features which is why the solid and dashed red lines converge, and also why the solid and dashed blue lines converge. The semantic features were learned using support vector ranking on our most accurate fMRI subject. The accuracy results reported are then averaged over the remaining 8 fMRI subjects. 47

3.2 A visualization of a simple semantic space. Points (representing classes) that are densely packed together require greater accuracy when predicting individual semantic features, particularly when the nearest neighbor algorithm is used to recover the true class from the predicted semantic features. 50

3.3 A visualization of Equation 3.2 that shows the effect on error tolerance as a function of the number of dimensions used in nearest neighbor classifier. The gamma bounds guarantees with the specified probability that the nearest neighbor is no farther than the indicated distance. As the number of dimensions increases, the error tolerance increases quickly and then the rate of increase declines quickly as more than 50 dimensions are used. In this equation, we use $M = 1000$, and show two probability bounds, $\gamma = 0.95$ and also $\gamma = 0.05$. We also assume the semantic feature points lie in a unit-hypercube with uniform distribution. 51

3.4 This figure shows the average ℓ_∞ distance between the 60 stimuli words and their nearest neighbor as a function of the number of dimensions used in the the semantic feature set. We computed this for sets ranging from 1 to 218 dimensions using the `human218` dataset. The plot shows the results averaged by repeating the experiment 25 times using random permutations of the features to eliminate any artifacts due to a specific feature ordering. Even using real data, we see the plot follows the same progression that is predicted by the theory in Figure 3.3. Specifically, we see the average distance increases quickly as dimensions are added, and then tapers off as more than 50 dimensions are used. 53

4.1 This image describes the neural prediction problem from Mitchell et al. [2008]. Words are represented as a vector of statistical co-occurrences with 25 sensory-action verbs such as `eat` and `ride` in a large text corpora. A model is trained to predict a neural image given the co-occurrence statistics for a given word. The image is the expected neural activity while a human is thinking about the particular word. 59
4.2 We formulate the neural image prediction problem as a multi-task learning problem. In this image, each voxel of neural activity is regressed onto a large set of possible semantic features for the word `apple`. The multi-task learning model learns all tasks simultaneously, and selects a common subset of features that are useful *as a basis* for all the tasks. Note that each task will weight each feature as necessary to predict its output, and tasks may have different weights on the same feature. 60
4.3 (Top) The regularization constraint penalty for the single-task Lasso and its effect on the solution vector. Only the most relevant variables have non-zero coefficients in the solution vector. (Bottom) The regularization constraint penalty for the multi-task Lasso and its effect on the solution matrix. Like the single-task Lasso, the multi-task Lasso encourages sparsity in the model. However, it attempts to learn a solution that is row-sparse, meaning that it picks a small number of input variables that are useful for predicting all the tasks. 63
4.4 The algorithm for the multi-task Lasso. 64
4.5 The leave-two-out-cross-validation protocols 67
4.6 Accuracies for 9 fMRI Participants. Features were selected *per-subject*. The chance accuracy level is shown as a solid black line and the significant threshold (at the 5% level) is shown as the dotted black line. This significance threshold is computed in Mitchell et al. [2008] by computing the empirical distribution of accuracies by randomly choosing semantic features from the 500-5000 most frequent words and applying the same leave-two-out train and test procedure. . . 68
4.7 Accuracies for 9 fMRI Participants. Features were selected from *combined participant data*. The chance accuracy level is shown as a solid black line and the significant threshold (at the 5% level) is shown as the dotted black line. This significance threshold is computed in Mitchell et al. [2008] by computing the empirical distribution of accuracies by randomly choosing semantic features from the 500-5000 most frequent words and applying the same leave-two-out train and test procedure. 69
4.8 Regression weights to each voxel for word Tools. We see strong activation (red) in the superior temporal sulcus which is believed to be associated with the perception of biological motion [Vaina et al., 2001]. We also see strong activation in the postcentral gyrus which is believed to be associated with premotor planning [Pelphrey et al., 2005]. 75
4.9 The 10 most highly weighted semantic features for the brain region Right-Medial-Frontal. 76
4.10 The 10 most highly weighted semantic features for the brain region Left-Post-Central. 76

4.11 The 10 most highly weighted semantic features for the brain region Right-Post-Central. 77

4.12 The 10 most highly weighted semantic features for the brain region Left-Triangularis. 77

4.13 The 10 most highly weighted semantic features for the brain region Right-Inferior-Extrastriate. 78

4.14 The 10 most highly weighted semantic features for the brain region Calcarine-Sulcus. 78

5.1 The standard zero-shot learning classifier shown on left, compared to the model augmented with intermediate kCCA features. The features were generated by finding the kCCA loadings that maximize the correlation between projections of the MEG view and fMRI view of the data. After training, the MEG or EEG data can be transformed in the kCCA feature space using its corresponding loading. The features can then be used as normal in the zero-shot learning classifier. Note that fMRI data is not needed when applying the classifier at test time. 82

5.2 The event related paradigm. Line drawings were presented as white lines on black background (shown reversed here for clarity). 84

5.3 Percent accuracies of the 9 MEG participants for the leave-two-out task using both source and sensor data. The chance accuracy is shown as a solid black line, the mean accuracy for all participants (sensor) is shown as a solid blue line, and the mean accuracy for all participants (source) is shown as a solid red line. We also show the 5% significance level (i.e. the threshold for which the p-value would be found significant if $p = 0.05$) for individual participant accuracy as a dashed "-" line, and the 5% significance level for the mean accuracy shown as a dash-dot "- ." line. We found accuracies of sensor data to be significantly better than source data, with eight of nine participants significant at the 5% level. Both sensor and source means were found significant at the 5% level as well. We computed these significance values by computing the empirical distribution of accuracies using random data in place of the real MEG/EEG data and repeating the experiment over a thousand times. 86

5.4 Percent rank accuracies of the 9 MEG participants for the harder leave-one-out task using both source and sensor data. The chance accuracy is shown as a solid black line, the mean accuracy for all participants (sensor) is shown as a solid blue line, and the mean accuracy for all participants (source) is shown as a solid red line. We also show the 5% significance level for individual participant accuracy as a dashed "-" line, and the 5% significance level for the mean accuracy shown as a dash-dot "- ." line. We found accuracies of sensor data to be significantly better than source data in this experiment as well. In this harder task, few participants were found significant at the 5% level, although the mean of sensor data was found significant at the 5% level, and the source mean just under the 5% level. 87

5.5 Percent accuracies of the 3 available EEG participants (P1-P2-P4) for the leave-two-out task using sensor data. The chance accuracy is shown as a solid black line, the mean accuracy for the three participants is shown as a solid blue line. We also show the 5% significance level for individual subject accuracy as a dashed "-" line, and the 5% significance level for the mean accuracy shown as a dash-dot "- ." line. We found two of three participants significant at the 5% level, and mean was found significant at the 5% level as well. 88
5.6 Percent rank accuracies of the 3 available EEG participants (P1-P2-P4) for the harder leave-one-out task using sensor data. The chance accuracy is shown as a solid black line, the mean accuracy for the three participants is shown as a solid blue line. We also show the 5% significance level for individual subject accuracy as a dashed "-" line, and the 5% significance level for the mean accuracy shown as a dash-dot "- ." line. This is the hardest task, with the noisiest data, and neither the individual subject accuracies nor mean were found significant at the 5% level. The mean accuracy was found significant at the 10% level (shown just under the dotted "..." line). 88
5.7 MEG accuracies using sensor data for the leave-two-out task averaged over all nine human subjects. Results are reported for each possible time window between 100-1900ms using 100ms increments. The Y-axis indicates the start time of the window in seconds after stimulus onset. The X-axis indicates the end time of the window in seconds after stimulus onset. 90
5.8 Average effect over nine human subjects on MEG and EEG test performance by training the kCCA model using fMRI data. Both leave-one-out and leave-two-out experiments were performed. Results are also reported when the model was trained using random Gaussian data instead of real fMRI data. Jackknife 95% confidence intervals are reported for the mean estimator. 91
5.9 Comparison of the MEG leave-one-out task using the standard regression model, and the kCCA model that includes fMRI data during training. Shown are percent rank accuracies of the 9 MEG participants for the harder leave-one-out task. The chance accuracy is shown as a solid black line, the mean accuracy for all participants (standard regression model) is shown as a solid blue line, and the mean accuracy for the kCCA model that includes fMRI data is shown as a solid red line. We also show the 5% significance level for individual participant accuracy as a dashed "-" line, and the 5% significance level for the mean accuracy shown as a dash-dot "- ." line. We found that kCCA only decreased accuracy in one participant, while occasionally causing an increase of 8-15%. While most subjects were not individually significant at the 5% level, both means were found significant at the 5% level. 92
5.10 Plots of the four most dominant kCCA loadings when plotted onto anatomical brains using MNE software. These plots show neural activities measured in fMRI and MEG for similar neural concepts. Although kCCA does not consider geometry in its learning algorithm, several spatial correlations can be seen. Several activities, however, only appear in one scanning modality. 93

List of Tables

2.1 The sixty stimulus words with five exemplars from twelve different semantic categories. 27
2.2 The top five predicted words for a novel fMRI image taken for the word in bold (all fMRI images taken from the top performing participant). The number in the parentheses contains the rank of the correct word selected from 941 concrete nouns in English. Eight words were selected to show the semantic similarities of the predicted words for both accurate and inaccurate predictions. 38

3.1 The 25 highest weighted features (highest to lowest) and the 25 lowest weighted features (lowest to highest) as discovered by the Support Vector Ranking algorithm when applied to Subject 1. 46

4.1 The semantic basis used in Mitchell et al. (2008) 61
4.2 An example of 25 learned semantic basis words. 71

Chapter 1

Introduction

Imagine being able to control any device simply by sending a command via radio waves. It's the future, Watson.

— Sherlock Holmes to Dr. Watson Story by Lionel Wigram

MACHINE learning algorithms have been successfully applied to learning classifiers in many domains such as computer vision, fraud detection, and brain image analysis. Typically, classifiers are trained to approximate a target function $f : X \rightarrow Y$, given a set of labeled training data that includes all possible values for Y, and sometimes additional unlabeled training data.

Little research has been performed on *zero-shot learning*, where the possible values for the class variable Y include values that have been *omitted from the training examples*. This is an important problem setting, especially in domains where Y can take on many values, and the cost of obtaining labeled examples for all values is high. One obvious example is computer vision, where there are tens of thousands of objects which we might want a computer to recognize.

Another example is automatic speech recognition, where it is desirable to recognize words without explicitly including them during classifier training. To achieve vocabulary independence, speech recognition systems typically employ a phoneme-based recognition strategy. Phonemes are the *component parts* which can be combined to construct the words of a language. Speech recognition systems succeed by leveraging a relatively small set of phoneme recognizers in conjunction with a large knowledge base representing words as combinations of phonemes.

In the case of speech recognition, phonemes are an obvious way to represent words as nearly all words can be expressed using a small set of phonemes. Further, all these phonemes can easily be enumerated by a linguistics or speech expect.

In other domains, it may not be obvious what component parts should be used to describe the classes. This is certainly true for computer vision, where it is not clear what component parts are shared between the thousands of objects that might appear in an image.

For such domains, this leads to several general questions: *What component parts are shared between the thousands of possible classes? Where would we obtain such information? Would it be possible to automatically learn these component parts from large quantities of data? Perhaps human knowledge of the class properties might help? Given a description of the component*

parts, could we make any guarantees with respect to classifier performance?

In this thesis, we address each of these questions and provide methodologies for zero-shot learning in domains where it is not obvious what component parts are useful for describing the classes. As a case study, we consider the domain of *thought recognition*, a term we use to describe the collection of work that uses machine learning methodologies to classify patterns of brain activity observed using neural scanners.

We begin with a brief overview of this field and neural scanning technologies, and then show how the problem of predicting a concrete noun a person is thinking about can be formulated as a zero-shot learning problem.

1.1 Overview of Thought Recognition

In the last several years, there has been considerable progress developing methodologies to discriminate patterns of brain activity based on neural images. This progress is the result of two primary developments. The first is the advancement of functional magnetic resonance imaging (fMRI), a brain scanning technology that can detect brain activity at the millimeter level. Although still far away from measuring the firing of individual neurons, the technology yields unprecedented ability to study the relationship between brain activity and cognitive function.

The second development that has enabled thought recognition is advancement in statistical pattern recognition and machine learning algorithms. In many respects, interpreting speech signals recorded using a microphone is similar to interpreting brain activity signals recorded using fMRI. Both signals contain enough structure that statistical methods can be used for analysis and classification. Novel algorithms specific for fMRI data, along with advancements made in other types of pattern recognition such has handwriting and speech (Figure 1.1), have benefitted the field of thought recognition.

Although fMRI is the most popular brain scanner due to its high spatial resolution, advances in other scanning technologies such as magnetoencephalography (MEG) and electroencephalography (EEG) have impacted the field of thought recognition as well (Figure 1.6). Although MEG and EEG have significantly lower spatial resolution than fMRI, the *temporal* resolution is significantly higher than fMRI, allowing brain signals to be recorded at millisecond resolution, nearly 1000 times the temporal resolution of fMRI. Together, these three neural recording devices provide new ways to study the spatial-temporal patterns of brain activity associated with a given *cognitive state* or physiological condition.

Cognitive state is a loosely defined term used in the research literature to describe some property of the brain activity observed from a scanner. For example, the cognitive state might refer to the task a person is performing such as *viewing a picture*, *reading a sentence*, or *thinking about a particular category of object*. In addition to cognitive state, certain patterns of brain activity may be the result of a physiological condition or psychological disorder such as schizophrenia or Alzheimer's disease.

Understanding these different conditions allows us to give a broad definition of thought recognition:

Thought recognition is the application of pattern recognition and machine learning methods

to discriminate patterns of brain activity associated with a particular cognitive state or physiological condition.

Thought recognition has many applications, and in the last five years a large and growing body of research has emerged across many fields, ranging from psychology and neuroscience to machine learning and statistics. Although the work is called many different names across these fields including *cognitive state classification, mind reading, thought identification, and neural activity decoding*, all this work relies on statistical pattern recognition and machine learning methods. As a result, we feel the term *thought recognition* broadly includes all these works, and is easily understandable by a non-technical audience, similar to the terms *speech recognition* or *handwriting recognition*.

Much of the work in thought recognition is concerned with analyzing brain function. For example, a researcher might want to classify from a brain scan if a person is viewing a picture or listening to an audio clip. A more difficult task might be to discriminate between a picture of a house or a tool. If a machine learning classifier can be trained to discriminate between cognitive states, then the trained classifier can provide virtual sensors of the information encoded by neural activity at specific cortical locations at specific times.

Similarly, thought recognition is used to study pathology of the brain. For example, classifiers have been trained to diagnose schizophrenia and Alzheimer's diseases [Caprihan et al., 2008, Hayasaka et al., 2006]. Interpretation of the learned classifiers has given insight into how various neural disorders affect the brain.

Another application of thought recognition is *brain-computer-interfaces* (BCI) [Vaadia and Birbaumer, 2009, Sajda et al., 2008, Lin et al., 2008]. The goal of BCI is to use the signals recorded by a brain scanner for control and communication. This application has huge potential benefits for disabled persons and could greatly enhance quality of life [Schalk, 2008]. For example, a paralyzed person might be able to control a wheelchair just by thinking of a desired direction of movement [Choi and Cichocki, 2008].

Thought recognition is also of interest to quantitative machine learning and statistics researchers because the domain pushes the limits of modern classification methods. The datasets produced by brain scanners such as fMRI and MEG are incredibly high dimensional, often with hundreds of thousands or even millions of raw features. The data are also noisy, meaning that if the same experiment is repeated, a different brain image is obtained (sometimes dramatically different). This "noise" is caused by both the physical sensing process as well as the contextual influences on the human research subject.

Typically, human subjects are kept in a fMRI scanner between 30-60 minutes to minimize discomfort as they must remain extremely still. Brain scanning also involves a large financial cost, as both the scanner and subject's time must be paid for. In addition, since fMRI measures the slow hemodynamic (blood) response, a sample is typically collected only once per 10 seconds, and multiple samples of a given state are typically collected to mitigate the effects of noise. As a result, the number of collected samples for a given task is usually very small (i.e. approximately 100-300 samples).

With so few samples, it may not be possible to collect training examples for each class (cognitive state) that we may want to discriminate. For example, if we consider the cognitive state to be *thinking about a particular concrete noun*, there are potentially thousands of nouns to consider.

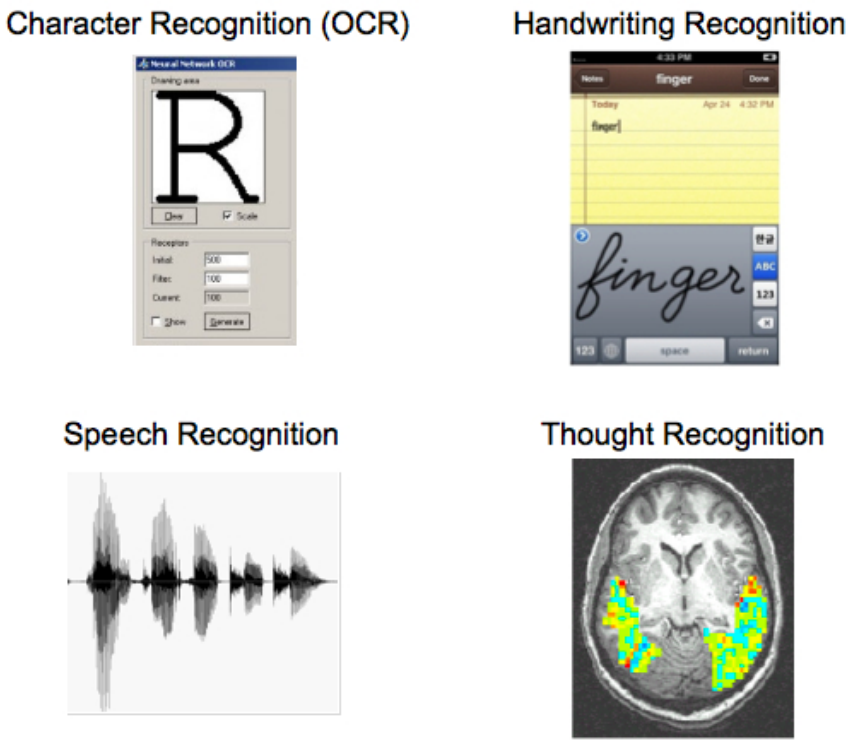


Figure 1.1: Thought recognition is the application of pattern recognition and machine learning methods to discriminate patterns of brain activity. Like other applications of pattern recognition like speech and handwriting recognition, neural signals observed using scanners like fMRI contain enough structure that statistical methods can be used for analysis and classification. Image Credits: character recognition image from Neural Network (OCR) application by Andrew Kirillov, handwriting recognition image from the DioPen application by DioTek, speech image from Dan Jurafsky, Stanford University, and fMRI image from Tom Mitchell.

It would be extremely difficult to collect samples for each of these nouns.

Thus, the main quantitative problem that must be addressed is:

How can we classify large numbers of cognitive states, even when examples for some cognitive states we wish to discriminate are omitted from the training set?

In this thesis, we consider cognitive states induced by *thinking about particular concrete nouns*. We will show that the key to answering this question is in utilizing semantic knowledge, specifically the physical and functional properties of the concrete nouns.

1.2 Thesis Statement and Contributions

The central thesis of this work is that:

It is possible for a machine learning classifier to discriminate classes that were not explicitly included in a training set by leveraging semantic knowledge of the classes such as their physical or functional attributes. Further, it is possible for the classifier to automatically select the semantic attributes or features that are most useful for a particular task, minimizing the effort required by a human to precisely define the optimal set of semantic attributes in advance.

When applied to neural imaging data, we found the semantic attributes of specific concrete nouns are predictable from the brain activity observed while a person is thinking about those nouns. This makes it possible to train a machine learning classifier to discriminate concrete nouns that people are thinking about, even without explicitly collecting examples of those nouns for a training set. Further, this allows discrimination of certain nouns that are within the same category at much higher accuracies than previous work.

Contributions

This thesis makes contributions to both core machine learning methodology, as well as applied neuro-imaging and brain-computer-interfaces. The following summarizes the primary contributions of this thesis:

1. **General framework for zero-shot learning with semantic knowledge** - We develop a formalism for classifiers that must discriminate classes that did not explicitly appear in a training set. The framework uses a knowledge base of semantic information that contains features (i.e. attributes) of the classes. By learning an intermediate layer of semantic features rather than the class labels directly, a classifier can extrapolate and discriminate novel classes that were omitted from a training set. This enables decoding of a much larger number of classes than were collected during training. We also study this problem theoretically, and establish certain conditions under which the model can be guaranteed to predict a novel class.
2. **Case study of zero-shot learning for decoding neural activity** - We apply the zero-shot learning framework to the problem of predicting concrete nouns from neural imaging data collected using fMRI. We also explore different sets of semantic features based on large corpus statistics as well as human labeling. This allows us to train a machine learning classifier to discriminate concrete nouns that people are thinking about, even without explicitly collecting examples of those nouns for a training set. Further, this allows discrimination of certain nouns that are within the same category with higher accuracy than previous work.
3. **Methodology for automatically selecting semantic features for classification using the zero-shot learning model** - We develop a methodology based on support vector ranking to choose the semantic features that are most useful for classification using the zero-shot learning model when *semantic features are predicted from neural imaging data*. This eases the burden of selecting the most useful semantic features prior to classifier training.
4. **Methodology for automatically selecting semantic features for predicting neural activity** - We also consider the reverse problem of zero-shot classification: to predict neural activity using *semantic features as inputs to the model*, rather than using neural activity to predict semantic features as outputs. Although this model can also be used for classification as well by comparing predicted neural images, the primary objective is to obtain a

Figure 1.2: An image of the fMRI facility located at the Scientific Imaging and Brain Research Center, Carnegie Mellon University. Image courtesy of SIBRC.

generative model to predict the expected neural image for a particular stimulus. As a result of the different modeling assumptions, the semantic features necessary for this generative model might be different that those most useful purely for classification (i.e. when the semantics features are predicted from neural activity). We develop a methodology based on the multi-task Lasso to automatically select the semantic features for this prediction problem. Further, we apply this methodology and report the semantic features that are most useful for predicting neural activity in specific regions of interest (ROIs) in the brain.

5. **Case study of concrete noun decoding using MEG/EEG** - We explore applications of the zero-shot learning model to other types of input data. Specifically, we test other neural scanners that have significantly better temporal, but lower spatial resolution than fMRI. We explore different time windows of the data to discover *when* semantic features are most easily predictable.
6. **Methodology to combine data from fMRI and MEG/EEG** - We develop a latent variable model based on CCA to combine data from fMRI and MEG/EEG. We use these latent factors as a filter to improve signal quality of the lower spatial resolution MEG/EEG data. By leveraging high-resolution fMRI data *at training time*, the classifier can improve its performance decoding lower-resolution MEG/EEG data *at test time*.

1.3 Overview of Functional Magnetic Resonance Imaging (fMRI)

The majority of the experimental work in this thesis analyzes neural activity data recorded using functional magnetic resonance imaging (fMRI) (Figure 1.2). We provide here a brief overview of fMRI and refer the reader to Huettel et al. [2004] for more detail.

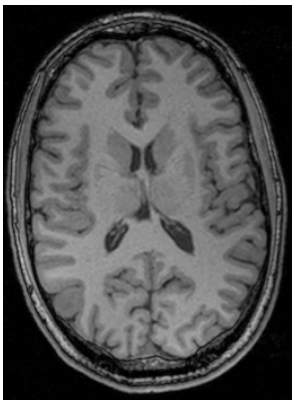
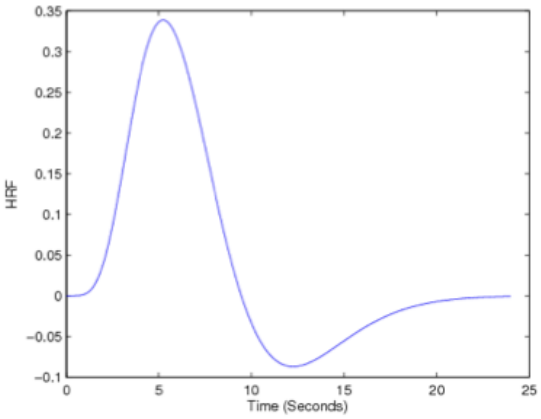


Figure 1.3: A structural or anatomical image of the brain using MRI. Anatomical images display just the static physical structure of matter, and do not indicate areas of changing neural activity.



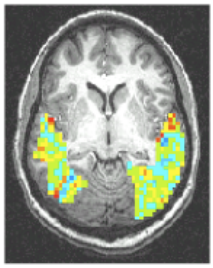


Figure 1.4: The plot on the left shows the hemodynamic response curve for fMRI. There is a 4-5 second lag between the stimulus onset when the strongest activation signal is recorded. The image on the right shows the BOLD response plotted on an anatomical MRI of the brain. Red colors indicate areas of stronger activity.

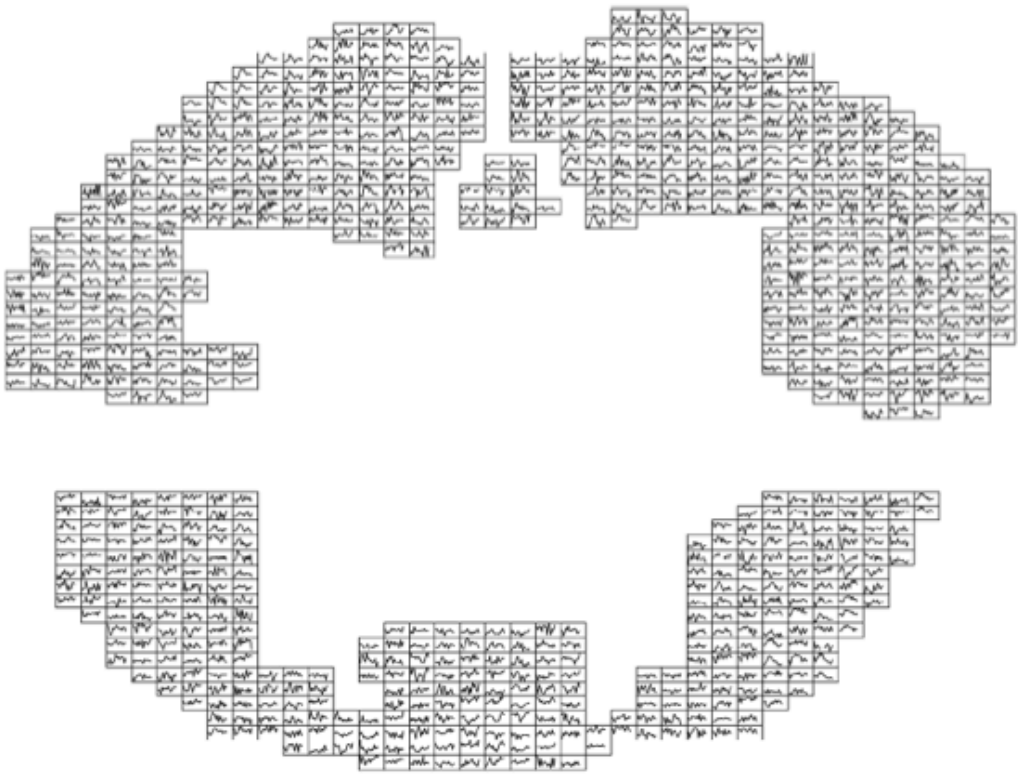


Figure 1.5: A plot of the fMRI timeseries observed for each voxel in the brain. The image here represents one 2-D slice of the observed 3-D volume of activity. Image courtesy of Tom Mitchell.

The reader is likely already familiar with structural or anatomical MRI as it is commonly used in clinical medical applications to diagnose injuries and disease. See Figure 1.3. Whereas structural MRI only detects static physical matter, functional MRI (fMRI) can provide an indication of the *functional neural activity* that occurs and changes over time.

Modern fMRI scanners can measure neural activity with millimeter spatial resolution and can create a 3-D volumetric image of that activity containing over 20,000 voxels (volume-elements). It takes approximately one second to record an image, which is significantly slower than the firing rate of neurons, which can be nearly 200 times per second.

The slow recording capability of fMRI is due to the fact that fMRI does not measure neural activity directly. Rather, fMRI measures changes in the level of oxygen concentration in the blood. When neurons fire, they consume glucose and blood flow increases in this area to replace the consumed glucose. This process results in changes to the ratio of oxyhemoglobin and deoxyhemoglobin, which can be thought of level of oxygen in the hemoglobin of the blood. This process is typically called the BOLD (blood-oxygen-level-dependent) or hemodynamic response, and takes 1-5 seconds to occur. See Figure 1.4 and 1.5.

Although the BOLD response leads to fairly accurate estimations of neural activity, the large delay makes it difficult to study precise timing of neural firing. As a result, other neural scanners such as magnetoencephalography (MEG) and electroencephalography (EEG) are more useful for

Electroencephalography (EEG)

Magnetoencephalography (MEG)

Figure 1.6: An image of the *actiCap* EEG electrode cap from Brain Products GmbH (left) and a MEG scanner located at the University of Pittsburgh Medical Center (right). While EEG and MEG have temporal resolution nearly 1000 times that of fMRI, the spatial resolution is significantly worse, especially for EEG. The form factor and cost of EEG, however, is a small fraction of fMRI or MEG. As a result, EEG remains popular in cognitive neuroscience and brain-computer-interface research. EEG image courtesy of Brain Products GmbH and MEG image courtesy of Gus Sudre.

studying the temporal processes of neural activity.

1.4 Overview of Other Brain Scanning Technologies

In this thesis, we also consider several datasets that were collected using magnetoencephalography (MEG) and electroencephalography (EEG), two other brain imaging devices that have very different properties than fMRI. See Figure 1.6.

Unlike fMRI, MEG measures brain activity directly by using extremely sensitive devices called SQUIDS to measure very tiny changes in magnetic field when neurons fire. Useful overviews are provided by Hämäläinen et al. [1993], Baillet et al. [2001], Hansen et al. [2010]. Although the spatial resolution of MEG is not as high as fMRI, the temporal resolution is far superior, with sampling rates up to 1000 Hz, compared to 0.5-2 Hz sampling rates for fMRI. Further, because MEG observes neural activity directly and is not subject to the hemodynamic delay, it is possible to detect much more precisely *when* neural activity occurs in response to a given stimulus. Although, MEG is also an expensive scanner like fMRI, the form factor is a little smaller, and more flexible experimental paradigms are possible because it is easier to correct for extensive head movements [Imada et al., 2006].

EEG is one of the oldest and simplest brain scanning technologies that measures voltage potentials across the scalp when neurons fire. See Swartz and Goldensohn [1998] for a historial overview. The primary benefit of EEG is its simplicity, form factor, and cost. While a research quality EEG system may cost $50,000, consumer versions have become recently available for less than $100. EEG also has a practical form factor, and its electrodes can be easily integrated into a wearable hat (Figure 1.6). As a result, EEG remains very popular in the cognitive neuroscience and brain-computer-interface communities [Guger et al., 2011]. While the temporal resolution of EEG can be quite high, with sampling rates of 1000 Hz, the spatial resolution is

very poor compared to fMRI and even MEG, which severely hampers potential applications.

1.5 Related Work

This thesis makes contributions at the intersection of machine learning, neuro-imaging, and brain-computer-interfaces. Here we provide an overview of the subsets of these fields most relevant to this thesis.

1.5.1 Thought Recognition and Cognitive State Classification

Machine learning classifiers have made a large impact on the field of cognitive neuroscience by showing that a person's cognitive state can be discriminated based on an image of that person's neural activity. Much of the early work in this area analyzed patterns of neural activity recorded using fMRI while human subjects perceived different objects [Cox and Savoy, 2003, Mitchell et al., 2004, 2003]. Later work showed that other cognitive states such as political affiliation [Kaplan et al., 2007], drug addition [Zhang et al., 2005], and even truthfulness [Davatzikos et al., 2005] could be classified. Excellent overviews of this line of research are available in Haynes and Rees [2006] and Norman et al. [2006].

Classification techniques have also been applied to the study of disease pathology. Studies have shown that it is possible to discriminate between schizophrenics and healthy controls [Caprihan et al., 2008, Skelly et al., 2007]. Other work has investigated Alzheimer's disease [Hayasaka et al., 2006] as well as stroke recovery [Schmah et al., 2008]. One particularly interesting study classified degrees of cognitive impairment in vegetative patients [Owen et al., 2006].

There has also been a large body of related work in the field of non-invasive brain computer interfaces (BCI), where the goal is to use neural signals for control and communication. Most of this work has focused on electroencephalography (EEG) because of its more practical form factor. Overviews of this field are available in Vaadia and Birbaumer [2009], Sajda et al. [2008], Lin et al. [2008]. Studies have also investigated BCI tasks in other neural scanners such as magnetoencephalography (MEG) [Birbaumer and Cohen, 2007] and even fMRI [Weiskopf et al., 2004]. Current state-of-the-art results have been recently reported in Guger et al. [2011].

There has been recent work to build *generative models* of neural activity. Rather than focus on purely discriminative (i.e classification) tasks, this work tries to build models that can predict patterns of neural activity in response to novel stimuli. The work of Kay et al. [2008] demonstrates a model that can predict neural activity in response to novel visual scenes, while the work of Mitchell et al. [2008] shows that it is possible to predict neural activity for concrete nouns in English.

Much of the related work to date has focused on problems with moderate amounts of training data and small numbers of classes (e.g. binary). While a few works have considered classification with very limited training data [Palatucci and Mitchell, 2007, Palatucci and Carlson, 2008], this thesis focuses on a more difficult learning scenario: when there are potentially thousands of classes to discriminate and several of the classes of interest are not included in the training set.

This is an important problem setting and especially relevant to neural imaging, largely due to the difficulty of collecting large amounts of labeled training data from neural scanners.

1.5.2 Zero-Shot and Attribute Based Learning

The goal of zero-shot learning is to learn a classifier $f : X \rightarrow Y$ that must predict novel values of Y that were omitted from a training set. To our knowledge, one of the first works to identify this problem setting and its importance to the broader machine learning community was Larochelle et al. [2008]. In their work, they show the ability to predict novel classes of handwritten digits that were omitted from a training set. They accomplish this by learning the relationship between observed handwritten digits and a character font bitmap which is used as an *intermediate class description* to describe both the observed classes in the training data and also the unobserved classes. One limitation of this work, however, is that it does not discuss where to obtain intermediate class descriptions for non-trivial problems such as object recognition or thought recognition, where the intermediate class descriptions are non-obvious.

To address the problem of non-obvious class descriptions Mitchell et al. [2008] suggest using co-occurrence statistics from large text corpora while our own previous work [Palatucci et al., 2009] considers both co-occurrence statistics as well as crowd-sourcing human semantic knowledge.

In parallel, researchers in the computer vision community have known the importance of sharing feature representations or class descriptions for some time [Torralba and Murphy, 2007, Bart and Ullman, 2005]. Important work within this community to describe classes by their semantic or visual attributes was performed by Farhadi et al. [2009] who showed an ability to learn object classifiers with extremely small numbers of training examples. Another very important work was that of Lampert et al. [2009] who showed that attributes obtained from human labeling could be used to perform zero-shot learning of object classes. Since the near simultaneous release of all these works within one year, zero-shot learning has become an increasing important topic in computer vision with many recent papers [Farhadi et al., 2010, Wang et al., 2010, Parikh and Grauman, 2011].

In addition to computer vision, zero-shot learning is an important problem setting for thought recognition, as well as any domain where it may be impossible to collect training data for all classes that a classifier must discriminate. At a high level, this problem setting is similar to the challenges of automatic speech recognition, where it is desirable to recognize words without explicitly including them during classifier training.

To achieve vocabulary independence, speech recognition systems typically employ a phoneme-based recognition strategy [Waibel, 1989]. Phonemes are the component parts which can be combined to construct the words of a language. Speech recognition systems succeed by leveraging a relatively small set of phoneme recognizers in conjunction with a large knowledge base representing words as combinations of phonemes.

To apply a similar approach to thought recognition, we must discover the "phonemes of thought". In other words, we must learn how to infer the component parts of a words meaning from neural activity. While there is no clear consensus as to how the brain encodes semantic information [Plaut, 2002], there are several proposed representations that might serve as a knowledge base of neural activity.

In this thesis, we develop a formalism for zero-shot learning, study it theoretically, and propose an algorithm that uses the proposed representations of neural encoding to build a classifier that can discriminate concrete nouns from neural images, even when those words are omitted from a training set.

1.5.3 Feature Selection, Sparsity, and Multi-task Learning

The problem of selecting useful features for prediction is central to machine learning research. Feature selection is one of the oldest topics of research in this field, and a very comprehensive survey is available in Guyon and Elisseeff [2003]. One increasingly popular way to select features is to use regularized methods that induce sparsity, meaning that only a small number of available features are used to build a predictive model. This removes the burden of preselecting the most appropriate features, and allows the model to automatically ignore irrelevant features and choose the features most useful for the prediction task. The most popular sparse, linear model is the Lasso [Tibshirani, 1996]. Other models that achieve sparsity through regularization include additive non-linear models [Ravikumar et al., 2008] as well as those that perform structure learning of graphs [Friedman et al., 2007a].

Sparsity constraints have also been used in multi-task learning problems. In a multi-task learning setting, a learner is required to predict several tasks simultaneously, and large body of work has shown that learning *related* tasks together can improve performance over single-task learning, especially in situations with limited training data [Caruana, 1997, Thrun, 1996a]. In a similar fashion to single-task learning, sparse methods have been used to select predictive features across multiple tasks simultaneously. This question has been addressed in several fields. In machine learning the problem is known as multi-task feature selection [Ando and Zhang, 2005, Obozinski et al., 2006, Argyriou et al., 2008, Bradley, 2010], in statistics as the simultaneous or multi-task Lasso [Turlach et al., 2005], and in signal processing as simultaneous sparse approximation [Tropp, 2006].

In this thesis, we present the first large-scale case study of the multi-task Lasso and use a recently developed blockwise coordinate descent algorithm to perform joint feature selection with thousands of features and tasks. We apply this method to a neural activity prediction task and show how we can automatically learn which semantic properties, i.e. which physical and functional attributes are useful for predicting neural activity in a specific region of the brain.

1.5.4 Self-Taught Learning and Latent Variable Models

As mentioned earlier, one of the central problems of machine learning is selecting features from data that are useful for a predictive task. Often, the problem of feature selection is the most difficult part of building a prediction algorithm, and much research in fields from natural language processing to computer vision is dedicated to selecting features or constructing novel features from raw streams of data (e.g. text corpora or video streams). As a result, one common criticism of machine learning methods is that algorithms are useless without humans investing large amounts of time constructing useful features.

While this criticism is valid for some domains, it ignores recent progress in automated feature construction and *self-taught learning* [Raina et al., 2007]. Self-taught learning is an emerging

body of research that deals with algorithms than can construct useful, novel features automatically from data (rather than just selecting the most relevant features like the previously described Lasso). Often learning is performed in a two-stage process, with the first stage dedicated to constructing novel features in an unsupervised manner, and the second stage dedicated to the supervised task of learning the relationship between the novel features and some output variable. These methods work particularly well when a large pool of data is available, and results have shown that useful features can be constructed even when data in the pool is from tasks that are different from the current learning task. Thus, self-taught learning is one way to transfer knowledge from previously learned tasks to a new task [Lee et al., 2008, 2009].

A similar idea is that of *deep-belief networks* which works by stacking models such as the Restricted Bolzmann Machine (RBM) to construct a hierarchy of novel features from a large pool of data [Hinton et al., 2006, Bengio et al., 2007]. This is related to components analysis, which tries to find lower-dimensional representations of data [Bishop, 2006].

The common theme amongst all these methods is that useful features can be automatically constructed by learning latent, generative models of the data, where the latent factors become the novel features. A key benefit is that tasks that are weakly statistically related in the raw input space may share an underlying set of latent features. Thus, data from related tasks can be used to improve performance of a given learning task, especially when data is limited for the current task. One example of such a benefit applied to the fMRI domain was given in Rustandi et al. [2009], which shows how *canonical correlation analysis* (CCA) can be used to combine data from multiple participants a fMRI study to improve a neural decoding task.

In this thesis, we extend latent variable models based on CCA to discover latent structure between fMRI and MEG/EEG. We use these latent factors as a filter to improve signal quality of the lower spatial resolution MEG/EEG data. By leveraging high-resolution fMRI data *at training time*, the classifier can improve its performance decoding lower-resolution MEG/EEG data *at test time*. To our knowledge, this is the first demonstration that classification performance in a particular scanning modality can be improved using another human's data observed in a different scanning modality.

1.6 Chapter Summaries

In Chapter 2, we introduce the zero-shot learning model, present its formalism, and also study some of its theoretical properties. We apply this model to a fMRI classification task, and show we can discriminate between concrete nouns even when neither noun appeared in the training set. Using this model, we also show we can classify words within the same category with accuracies much higher than previous work. Further, we study classification using semantic feature sets collected automatically by mining corpus statistics, as well as through crowd sourced human labeling. We found the human labeled semantic features to be much more useful, and using these we show the model can classify correctly even when it must make its prediction from a large set of 1000 candidate words.

In Chapter 3, we study the question of how to select the best semantic features for the zero-shot learning model from a large set of possible features. We develop an algorithm based on Support Vector Ranking that can learn useful weightings of the semantic features, thereby de-

creasing the impact of the irrelevant features in the model. We found this method is useful for choosing semantic features when the desired feature set is small (i.e. the semantic representation of a class is less than 50 features. This is useful for minimizing computation needed in the zero-shot model). Surprisingly, the impact of irrelevant features becomes less important as the size of the original feature set increases (e.g. more than a few hundred). We investigate one theoretical argument to explain this phenomenon and show how the error tolerance for semantic feature predictions grows with added dimension, thereby mitigating the impact of poorly predicted semantic features. As a case study, we apply this method to fMRI data and report the top 25 and lowest 25 weighted features from our human labeled set.

In Chapter 4, we study the inverse of the zero-shot classification model. Specifically, we build a model to predict brain activity from the semantic features, instead of predicting semantic features from brain activity. This model is generative in the sense that given the semantic features for a novel word, it can often predict the brain activity that would be observed in fMRI images collected from a human subject thinking about the word. Similar to the zero-shot classification model, we have the problem of choosing the best semantic features to input into the model. Although the previous Support Vector Ranking approach does not apply to the generative model, we show an alternative method that can be used when predicting brain activity from semantic features. Specifically, we show how to formulate this semantic feature selection problem using the multi-task Lasso objective function. In collaboration with statisticians working to develop the first scalable algorithm to solve this objective, we present the first case study of a large scale multi-task Lasso problem, and show how we can automatically learn a useful set of semantic features that perform as well or better than the handcrafted set of features reported in Mitchell et al. [2008]. Further, we apply the multi-task Lasso to different regions of interest (ROIs) of the brain, and report the set of semantic features that are most useful for predicting activity in that region.

In Chapter 5, we return to the zero-shot classification model, but now apply the method to data collected from MEG and EEG. These two scanners have superior temporal resolution and form factor compared with fMRI, but significantly lower spatial resolution. We test the same experimental paradigm described in Chapter 2, and although we found classification performance much lower than fMRI, we found prediction accuracies of the semantic features were sufficient to discriminate classes that never appeared in the training set. In this chapter we also study a latent variable model based on CCA to combine data from fMRI and MEG/EEG. We use these latent factors as a filter to improve signal quality of the lower spatial resolution MEG/EEG data. Our results show it is possible to leverage high-resolution fMRI data *at training time* to improve classification accuracy of lower-resolution MEG/EEG data *at test time*. This is an important development for neural engineering and brain-computer-interfaces, as it suggests that the lower quality signal of a more practical (i.e. lower cost, better form factor) device can be improved by learning an *a priori* filter using a less practical but higher resolution device.

In Chapter 6, we present our conclusions and implication for the fields of neural imaging and machine learning. We also discuss several directions for future research.

In the Appendix, we provide a derivation of the support vector ranking algorithm for selecting semantic features.

Chapter 2

Thought Recognition using Zero-Shot Learning

In this chapter, we consider the problem of predicting a specific concrete noun that a person is thinking about from a functional magnetic resonance (fMRI) image of that person's neural activity. By introducing a novel machine learning framework called *zero-shot learning*, we show that it is possible to predict concrete nouns that people are thinking about, even without explicitly collecting examples of those nouns for a training set. Further, this allows discrimination of certain nouns that are within the same category with significantly higher accuracies than previously possible.

Chapter Notes for Neural Imaging Researchers
For the neural imaging researcher, the experimental results described above will be of greatest interest. While a basic understanding of the zero-shot learning paradigm is useful, the theoretical analysis of the model can be safely skipped.

Chapter Notes for Machine Learning Researchers
For the machine learning researcher, the experimental results provide an interesting case study of the zero-shot learning paradigm. We believe this paradigm has broad impacts across many applications, as it shows how to combine abstract semantic knowledge with raw high-dimensional signals, thereby allowing classes that were omitted from the training set to be classified. Similar models are already appearing in the computer vision community [Lampert et al., 2009, Farhadi et al., 2009], and we feel this is a very ripe area for additional research. In addition to the model formalism, we provide first the analysis of the theoretical properties of this model.

2.1 Background

Machine learning classifiers and multi-voxel pattern analysis (MVPA) algorithms have made a large impact on the field of neural imaging in recent years. While traditional analysis of neural imaging data focused primarily on functional brain mapping using univariate statistical tech-

niques, multivariate machine learning classifiers allow for the recognition and discrimination of subtle patterns of neural activity, even when that activity is distributed across large regions of the brain.

Consequently, a large and growing literature has emerged showing that it is possible to discriminate a large variety of *cognitive states*, ranging from disease diagnosis in patients with schizophrenia or Alzheimer's diseases [Caprihan et al., 2008, Hayasaka et al., 2006] to detecting drug addicted patients from controls [Zhang et al., 2005]. Others have shown it is possible to discriminate concrete objects [Shinkareva et al., 2008], political affiliation [Kaplan et al., 2007], and even lies from truth [Davatzikos et al., 2005]. Overviews of this line of research can be found in Haynes and Rees [2006], Norman et al. [2006] and also Pereira et al. [2009].

Most of this prior work focuses on binary classification for tasks with very few classes to discriminate. Recent work from Chan et al. [2010] makes a step towards a many-class classifier by using a hierarchical tree of classifiers to discriminate ten classes, but virtually no work has focused on discriminating among potentially thousands of possible classes.

Building a classifier algorithm for neural imaging tasks with thousands of classes is an extremely difficult pattern recognition problem as the data are usually very high dimensional (i.e. $10^4 - 10^5$), and the number of training examples is typically small (i.e. $10 - 10^2$). Further, due to the expense and time required to gather data from human subjects, merely collecting data for a large number of classes (i.e. cognitive states) is a difficult challenge. In some tasks, it may not be possible to collect data for each class that we want to discriminate.

In this chapter, we address these two challenges by posing and answering the following question:

How can we classify large numbers of cognitive states, even when examples for some cognitive states we wish to discriminate are omitted from the training set?

We consider this question in the context of *concrete noun recognition* where the goal is to predict a specific concrete noun that a person is thinking about from a large set of possible nouns. We analyze data collected from functional magnetic resonance imaging (fMRI). In subsequent chapters we consider magnetoencephalography (MEG) and electroencephalography (EEG) as well. We introduce a novel machine learning methodology called *zero-shot learning*, which can classify specific words even when examples for those words are omitted from the training set. We also show that for fMRI, the classifier can discriminate far above the chance level even when choosing from nearly 1,000 possible words.

Using the zero-shot learning methodology, classes (e.g. words) are not represented by using simple class labels, but rather by using an intermediate semantic representation that includes attributes and features of the underlying classes. For example, words could be represented as a vector of features that describe the size, shape, weight, or function of the word. This is similar in spirit to the work of Kay et al. [2008], Mitchell et al. [2008], Just et al. [2010], who utilize semantic representations in a *generative* fashion to predict neural images for novel stimuli. Like these works, the success of using an intermediate semantic representation depends largely on the quality of the features. In this chapter, we evaluate the effectiveness of two semantic representations: one constructed from a large text corpora and another from human labeling.

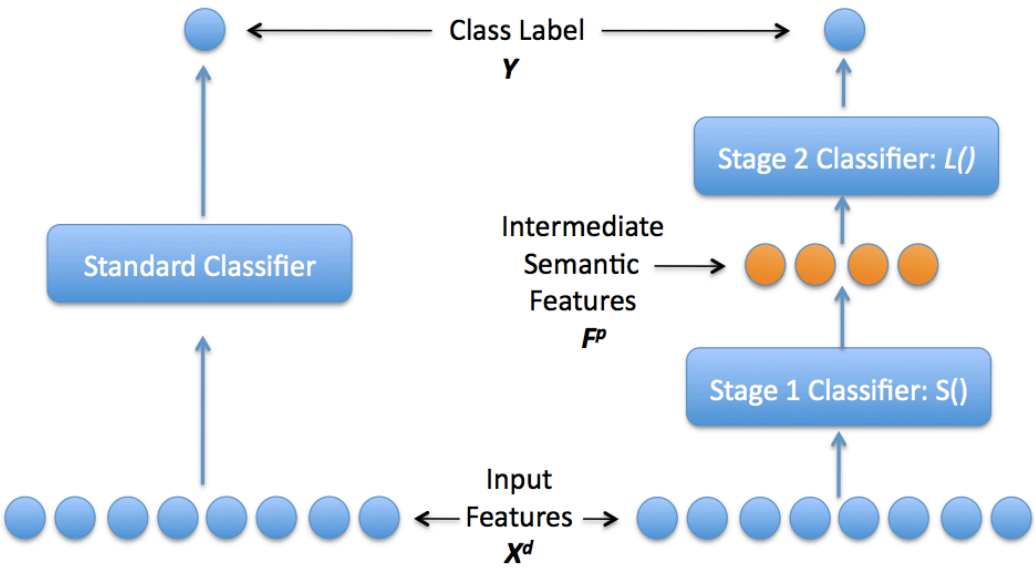


Figure 2.1: The semantic output code classifier (SOCC) is a two stage classifier (shown on right) compared to a typical single stage classifier (shown on left). The SOCC uses a layer of intermediate semantic features in between the input features and the class label. These semantic features represent attributes of the class labels.

2.2 The Zero-Shot Learning Model

2.2.1 Overview

Standard machine learning classifiers such as support vector machines or logistic regression learn a function $h : X \rightarrow Y$, that maps a vector of input features X to a class label Y. For example, the predictive features might be voxel activities from fMRI averaged over a particular time window, and the class label might be a cognitive state such as whether the human subject is viewing a picture or a sentence. The classifier is typically trained using a set of examples, each of which includes values for the features X and their corresponding class labels Y. After training, the classifier is used to predict the class label for a novel test example.

While this classifier approach works well for many applications, there are several drawbacks when applied to neural imaging data. A primary concern is that most neural imaging classification problems are inherently high dimensional, meaning that there are typically many thousands of features in the input X to the classifier. In general, the higher the dimensionality of X, the more training examples are required, but most neural image data sets have only a few hundred training samples. Without applying feature selection, regularization, or dimensionality reduction methods (e.g. PCA or LDA), it is very difficult to train a classifier than can perform better than random chance. Further, the neural data are *noisy*, meaning that the predictive features for a specific class label can vary a large amount as examples are collected, making statistical correlation between predictive features and a given class label very weak.

As a result, classification of neural imaging data with binary or even small numbers of classes is a difficult problem. The problem scales in difficulty as more classes are considered and there-

fore, little work has addressed problems on the scale of hundreds or even thousands of classes. Further, collecting neural imaging data from human subjects is difficult due to the time and expense involved, making the collection of data containing thousands of classes an impractical task. This is especially true for the task of concrete noun recognition, as it would be impossible to collect training examples for the ten of thousands of nouns we may wish to classify.

To address the challenge of classifying many classes with little data, we developed *zero-shot learning*, a novel machine learning classification methodology that does not require training examples for each class that the classifier must discriminate. This is similar to previous work in one-shot learning [Fei-Fei et al., 2006] that uses very few training examples, although we describe a more difficult learning scenario in which the classifier much discriminate classes that do not appear in the training set. Further, by using zero-shot learning the difficult problem of creating a complex multi-class classifier is ameliorated by splitting the complex task into simpler subtasks which can be often solved using binary classification or simple regression.

We focus on one particular implementation of a zero-shot learning algorithm known as the *semantic output code classifier* which we introduced in Palatucci et al. [2009] and expand upon in this chapter. Intuitively, our approach is to treat each class as a vector of *semantic features* characterizing a large number of possible classes. Our models learn the relationship between input data and the semantic features. This learned relationship is then used in a two step prediction procedure to recover the class for novel input data (see Figure 2.1). Given new input data, the models predict a set of semantic features corresponding to that input, and then find the class in a semantic knowledge base that best matches that set of predicted features. Significantly, this procedure works even if no input space example is available for the class, as long as a semantic feature encoding of it exists in a semantic knowledge base.

2.2.2 Additional Related Work

The problem of discriminating classes that were omitted from a training set is similar to the challenges of automatic speech recognition, where it is desirable to recognize words without explicitly including them during classifier training. To achieve vocabulary independence, speech recognition systems typically employ a phoneme-based recognition strategy [Waibel, 1989]. Phonemes are the component parts which can be combined to construct the words of a language. Speech recognition systems succeed by leveraging a relatively small set of phoneme recognizers in conjunction with a large knowledge base representing words as combinations of phonemes.

To apply a similar approach to neural activity decoding, we must discover how to infer the component parts of a words meaning from neural activity. While there is no clear consensus as to how the brain encodes semantic information [Plaut, 2002], there are several proposed representations that might serve as a knowledge base of neural activity, thus enabling a neural decoder to recognize a large set of possible words, even when those words are omitted from a training set.

In other domains, where the semantic representation of the classes is not obvious, little work has been done. Some work by Larochelle et al. [2008] on *zero-data learning* has shown the ability to predict novel classes of digits that were omitted from a training set by learning a mapping between the digits and a canonical font. In computer vision, techniques for sharing features across object classes have been investigated [Torralba and Murphy, 2007, Bart and Ullman, 2005]

but relatively little work has focused on recognizing entirely novel classes, with the exception of Lampert et al. [2009] predicting visual properties of new objects and Farhadi et al. [2009] using visual property predictions for object recognition. In all of these works, the primary difficulty is finding a useful intermediate representation that can be shared between all the classes, including those that were omitted from a training set. In addition, none of these works consider the problem from a theoretical standpoint to establish conditions for which prediction of a novel class is possible. For more information and additional references and discussion of this literature, see Section 1.5.2.

2.2.3 Formalism

We now present a formalism of the zero-shot learning model, and start with a few definitions and simple examples for clarity.

Definition 1. *Semantic Feature Space*
A semantic feature space of p dimensions is a metric space in which each of the p dimensions encodes the value of a semantic property. These properties may be categorical in nature or may contain real-valued data.

As an example, consider a semantic space for describing high-level properties of animals. In this example, we will consider a small space with only $p = 5$ dimensions. Each dimension encodes a binary feature: *is it furry? does it have a tail? can it breathe underwater? is it carnivorous? is it slow moving?* In this semantic feature space, the prototypical concept of *dog* might be represented as the point $\{1, 1, 0, 1, 0\}$.

Definition 2. *Semantic Knowledge Base*
A semantic knowledge base $\mathcal{K}$ relating a p dimensional semantic feature space F^p to a set Y of M labels is a collection of pairs $\{f, y\}_{1:M}$ such that $f \in F^p$ and $y \in Y$. We assume a one-to-one encoding between class labels and points in the semantic feature space.

A *knowledge base* of animals would contain the semantic encoding and label for many animals.

Definition 3. *Semantic Output Code Classifier*
A semantic output code classifier $\mathcal{H} : X^d \to Y$ maps points from some d dimensional raw-input space X^d to a label from a set Y such that $\mathcal{H}$ is the composition of two other functions, $\mathcal{S}$ and $\mathcal{L}$, such that:

$$\begin{aligned} \mathcal{H} &= \mathcal{L}(\mathcal{S}(\cdot)) \\ \mathcal{S} &: X^d \to F^p \\ \mathcal{L} &: F^p \to Y \end{aligned}$$

This model of a *zero-shot classifier* first maps from a d dimensional raw-input space X^d into a semantic space of p dimensions F^p, and then maps this semantic encoding to a class label. For example, we may imagine voxel activity features from a neural image collected while a human subject is thinking of the word *dog*. These features are first mapped into the semantic encoding

of a dog described earlier, which is then mapped to the class label *dog*. As a result, our class labels can be thought of as a *semantic output code*, similar in spirit to the error-correcting output codes in the multi-class classifiers of Dietterich and Bakiri [1995].

As part of its training input, this classifier is given a set of N examples $\mathcal{D}$ that consists of pairs $\{x, y\}_{1:N}$ such that $x \in X^d$ and $y \in Y$. The classifier is also given a knowledge base $\mathcal{K}$ of M examples that is a collection of pairs $\{f, y\}_{1:M}$ such that $f \in F^p$ and $y \in Y$. Typically, $M >> N$, meaning that data in semantic space is available for many more class labels than in the raw-input space. Thus,

A semantic output code classifier can be useful when the knowledge base $\mathcal{K}$ covers more of the possible values for Y than are covered by the input data $\mathcal{D}$.

To learn the mapping $\mathcal{S}$, the classifier first builds a new set of N examples $\{x, f\}_{1:N}$ by replacing each y with the respective semantic encoding f according to its knowledge base $\mathcal{K}$. In the simplest case, $\mathcal{S}$ is trained as a classifier independently for each semantic feature, and $\mathcal{L}$ can simply be a lookup from the knowledge base $\mathcal{K}$ and does not require a separate training step. More elaborate implementations of $\mathcal{S}$ might learn a joint model of the features using a neural network, and $\mathcal{L}$ could also be trained as a classifier to learn a conditional distribution of the class label Y from the semantic features.

The intuition behind using this two-stage process is that the classifier may be able to learn the relationship between the raw-input space and the individual dimensions of the semantic feature space from a relatively small number of training examples in the input space. When a new example is presented, the classifier will make a prediction about its semantic encoding using the learned $\mathcal{S}$ map. Even when a new example belongs to a class that did not appear in the training set $\mathcal{D}$, if the prediction produced by the $\mathcal{S}$ map is close to the true encoding of that class, then the $\mathcal{L}$ map will have a reasonable chance of recovering the correct label. As a concrete example, if the model can predict that the object has fur and a tail, it would have a good chance of recovering the class label *dog*, even without having seen images of dogs during training. In short:

By leveraging a rich semantic encoding of the classes, the classifier can extrapolate and recognize novel classes.

2.2.4 Theoretical Analysis

In this section we consider theoretical properties of a *semantic output code classifier* that determine its ability to recognize instances of novel classes. In other words, we will address the question:

Under what conditions will the semantic output code classifier recognize examples from classes omitted from its training set?

In answering this question, our goal is to obtain a PAC-style bound: we want to know how much error can be tolerated in the prediction of the semantic properties while still recovering the novel class with high probability. We will then use this error bound to obtain a bound on the number of examples necessary to achieve that level of error in the first stage of the classifier. The idea is that if the first stage $\mathcal{S}(\cdot)$ of the classifier can predict the semantic properties well, then the second stage $\mathcal{L}(\cdot)$ will have a good chance of recovering the correct label for instances from novel classes.

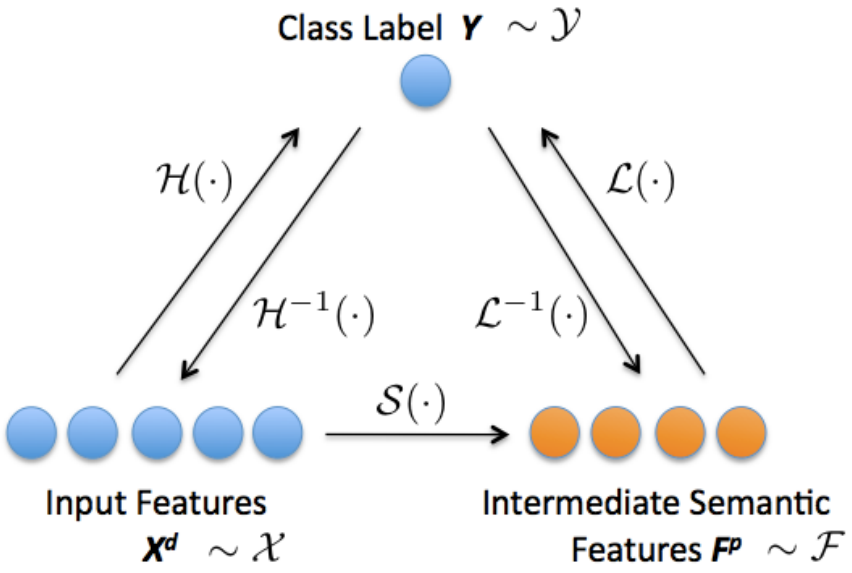


Figure 2.2: This figure shows the relationships between input features X^d, the classes Y, and the intermediate semantic features F^p. For the purposes of this analysis, we assume that we can observe the output of the deterministic functions $\mathcal{H}^{-1}(\cdot)$ and $\mathcal{L}^{-1}(\cdot)$ for a given class drawn from an arbitrary distribution $\mathcal{Y}$. This provides samples from the input distribution $\mathcal{X}$ and also from the semantic feature distribution $\mathcal{F}$. In our PAC-based model, we learn the function $\mathcal{H}(\cdot)$ indirectly through $\mathcal{S}(\cdot)$ and $\mathcal{L}(\cdot)$. We analyze the performance based on the number of samples N drawn from the input distribution $\mathcal{X}$ and also from the number of examples M drawn from semantic feature distribution $\mathcal{F}$.

As a first step towards a general theory of zero-shot learning, we will consider one instantiation of a semantic output code classifier. We will assume that semantic features are *binary labels*, the first stage $\mathcal{S}(\cdot)$ is a collection of PAC-learnable linear classifiers (one classifier per feature), and the second stage $\mathcal{L}(\cdot)$ is a 1-nearest neighbor classifier using the *Hamming distance metric*. By making these assumptions, we can leverage existing PAC theory for linear classifiers as well as theory for approximate nearest neighbor search. Much of our nearest-neighbor analysis parallels the work of Ciaccia and Patella [2000].

In Figure 2.2 we show the relationships between input features X^d, the classes Y, and the intermediate semantic features F^p. For the purposes of our analysis, we assume that we can observe the output of the deterministic functions $\mathcal{H}^{-1}(\cdot)$ and $\mathcal{L}^{-1}(\cdot)$ for a given class drawn from an arbitrary distribution $\mathcal{Y}$. This provides samples from the input distribution $\mathcal{X}$ and also from the semantic feature distribution $\mathcal{F}$. In our PAC-based model, we learn the function $\mathcal{H}(\cdot)$ indirectly through $\mathcal{S}(\cdot)$ and $\mathcal{L}(\cdot)$. We analyze the performance based on the number of samples N drawn from the input distribution $\mathcal{X}$ and also from the number of examples M drawn from semantic feature distribution $\mathcal{F}$.

In our analysis, we consider that M samples are drawn from our semantic feature distribution $\mathcal{F}$. We call this specific set of samples our knowledge base $\mathcal{K}$. The probabilistic bounds we compute will be specific to this particular sample set, and will change if another sample set is

used.

Further, we assume that there exists a point (class) q in this knowledge base that we would like to predict. Depending on the set of classes in the knowledge base and also the particular choice of semantic features, it should be obvious that points in semantic space may not be equidistant from each other. A point might be far from others, which would allow more room for error in the prediction of semantic features for this point, while maintaining the ability to recover its unique identity (label). Conversely, a point close to others in semantic space will have lower tolerance of error. In short, *the tolerance for error in prediction of the semantic features F^p is relative to a particular point in relation to other points in semantic space.*

Let $d(q, q')$ be the distance between the class of interest q and some other sample q' representing another class in our sampled knowledge base. We define the relative distribution $R_{q,\mathcal{K}}$ for a point q and sampled knowledge base $\mathcal{K}$ as the probability that the distance from q to another point q' is less than some distance z:

$$R_{q,\mathcal{K}}(z) = \mathbb{P}\left(d(q, q') \leq z\right) \qquad \forall q' \in \mathcal{K}, q' \neq q$$

This empirical distribution is just the fraction of points in $\mathcal{K}$ that are less than some distance z away from q:

$$R_{q,\mathcal{K}}(z) = \mathbb{P}\left(d(q, q') \leq z\right) = \frac{\#\text{points } q' \neq q \text{ in } \mathcal{K} \text{ with distance } d(q, q') <= z}{M}$$

Using this distribution, we can also define a distribution on the distance to the nearest neighbor of q in $\mathcal{K}$, defined as η_q:

$$G_{q,\mathcal{K}}(z) = \mathbb{P}\left(\eta_q \leq z\right)$$

which is the probability that *at least* one point is $\leq z$ distance from q. This is given in Ciaccia and Patella [2000] as:

$$G_{q,\mathcal{K}}(z) = 1 - (1 - R_{q,\mathcal{K}}(z))^M \tag{2.1}$$

where M is the sample size of $\mathcal{K}$. Now suppose that we define τ_q to be the distance from the class q to a semantic feature prediction $\widehat{q}$ generated by applying $\mathcal{S}(\cdot)$ to an input example x whose true class label is q, i.e. $\mathcal{H}(x) = q$.

Intuitively, we would like the nearest neighbor to $\widehat{q}$ in the knowledge base $\mathcal{K}$ to be the true class q. However, we do not know until after training and prediction where exactly $\widehat{q}$ will lie in the semantic space, and what feature predictions resulted in errors. As a result, we will consider instead the nearest neighbor to the true class q, relative to its neighbors in $\mathcal{K}$ in order to determine how many training examples N are needed from $\mathcal{X}$ to ensure the semantic features are sufficiently predicted such that $\widehat{q}$ is the nearest neighbor to q. This is a slight subtlety, but if the distance is sufficiently small, both q and $\widehat{q}$ will be nearest neighbors of each other.

Recall that τ_q is the distance from our true class q to the prediction $\widehat{q}$. Intuitively, we would like τ_q to be less than the distance from q to any other point in $\mathcal{K}$. In other words, we would like a small probability that the nearest neighbor to q from the other points in $\mathcal{K}$ is less than or equal to τ_q. We'll call this probability γ:

$$\mathbb{P}_{q,\mathcal{K}}(\eta_q \leq \tau_q) \leq \gamma$$

Note that this is the same probability distribution $G_{q,\mathcal{K}}$ defined in Equation 2.1. Substituting τ_q for z:

$$\begin{aligned} \mathbb{P}_{q,\mathcal{K}}(\eta_q \leq \tau_q) &\leq \gamma \\ G_{q,\mathcal{K}}(\tau_q) &\leq \gamma \end{aligned}$$

If $G_{q,\mathcal{K}}$ were invertible, we could immediately recover the value τ_q for a desired γ. But we cannot guarantee that it will be invertible for all distributions. However, since $G_{q,\mathcal{K}}$ is a distribution function, it will always monotonically increase from 0 to 1. As a result, we can define a function $G^{-1}_{q,\mathcal{K}}$ such that:

$$G^{-1}_{q,\mathcal{K}}(\gamma) = \arg\max_{\tau_q} \Big[G_{q,\mathcal{K}}(\tau_q) \leq \gamma \Big]$$

Therefore, if our prediction error $\tau_q \leq G^{-1}_{q,\mathcal{K}}(\gamma)$, then the chance of another point in $\mathcal{K}$ being closer to q is no more than γ. As a result, we also have at least $(1-\gamma)$ probability the prediction $\widehat{q}$ is closest to q.

To ensure that we achieve this error bound, we need to make sure the total error of $\mathcal{S}(\cdot)$ is less than or equal to $G^{-1}_{q,\mathcal{K}}(\gamma)$ which we define as τ_q^{max} . We assume in this analysis that we have p binary semantic features and a Hamming distance metric, so τ_q^{max} defines the total number of mistakes we can make predicting the binary features. Note with our assumptions, each semantic feature is PAC-learnable using a linear classifier from a d dimensional input space X^d. To simplify the analysis, we will treat each of the p semantic features as independently learned. By the PAC assumption, the true error (i.e. probability of the classifier making a mistake) of each of the p learned hypotheses is ϵ, then the expected number of mistakes over the p semantic features will be τ_q^{max} if we set $\epsilon = \tau_q^{max}/p$. Further, the probability of making τ_q^{max} mistakes or fewer is given by the binomial distribution: BinoCDF$(\tau_q^{max}; p, \tau_q^{max}/p)$

We can obtain the desired error rate for each hypothesis by utilizing the standard PAC bound for VC-dimension[1] [Mitchell, 1997]. To obtain a hypothesis with $(1-\delta)$ probability that has true error at most $\epsilon = \tau_q^{max}/p = G^{-1}_{q,\mathcal{K}}(\gamma)/p$, then the classifier requires a number of examples $N_{q,\mathcal{K},\delta}$ to be drawn from the input distribution $\mathcal{X}$:

$$N_{q,\mathcal{K},\delta} \geq \frac{p}{\tau_q^{max}} \Big[4\log(2/\delta) + 8(d+1)\log(13p/\tau_q^{max}) \Big] \tag{2.2}$$

If each of the p classifiers (feature predictors) is learned with this many examples, then with probability $(1-\delta)^p$, *all* feature predictors will achieve the desired error rate. But note that this is only the probability of achieving p hypotheses with the desired true error rate. The binomial CDF yields the probability of making τ_q^{max} mistakes or fewer, and the $(1-\gamma)$ term above specifies the probability that $\widehat{q}$ is closest to q if a maximum of this many mistakes were made. Therefore, there are three probabilistic events that determine the probability that the semantic output code classifier will make a prediction $\widehat{q}$ that is closer to q than all the other points in $\mathcal{K}$. The total (joint) probability of these events is the product of:

$\mathbb{P}$ (there are p feature predictors with true error $\leq \tau_q^{max}/p$) $\cdot$

$\mathbb{P}$ (at most τ_q^{max} mistakes made | there are p feature predictors with true error $\leq \tau_q^{max}/p$) $\cdot$

$\mathbb{P}$ (the prediction $\widehat{q}$ is the nearest neighbor to true class q | at most τ_q^{max} mistakes made)

[1]The VC dimension of linear classifiers in d dimensions is $d+1$

and since $\tau_q^{max} = G_{q,\mathcal{K}}^{-1}(\gamma)$, the total probability is given by:

$$(1-\delta)^p \cdot \text{BinoCDF}(\tau_q^{max}; p, \tau_q^{max}/p) \cdot (1-\gamma) \tag{2.3}$$

In summary, given desired error parameters $(1-\gamma)$ and $(1-\delta)$ for the two classifier stages, Equation 2.3 provides the total probability that the prediction $\widehat{q}$ will be the nearest neighbor of q. Given the value for γ we can compute the ϵ necessary for each feature predictor. We are guaranteed to obtain the total probability if the feature predictors were trained with $N_{q,\mathcal{K},\delta}$ input examples sampled from $\mathcal{X}$ as specified in Equation 2.2.

To our knowledge, Equations 2.2 and 2.3 specify the first formal guarantee that provides conditions under which a classifier is likely to predict novel classes that were omitted from the input training set.

To help understanding, we'll now provide a concrete example of using the above equations to compute the probability of recovering a novel class using zero-shot learning.

Example 4. *Computing the probability of recovering a novel class using zero-shot learning*

In this example, we'll assume there are $d = 5$ dimensions in the input space X, and also $p = 20$ binary semantic features in F. We'll draw a training set $\mathcal{K}$ of $M = 10,000$ samples from our distribution of semantic points $\mathcal{F}$. We obtain samples from this distribution by sampling y from any arbitrary distribution $\mathcal{Y}$ over the classes Y, then applying a deterministic semantic labeling function $\mathcal{L}^{-1}(\cdot)$. In this example, we'll assume this function just assigns a random point in the semantic space to this class. Specifically, we'll assume that each point in the semantic space is generated by using a fair Bernoulli trial for each of the $p = 20$ dimensions. In other words, we would run the Bernoulli trial 20 times to obtain a point such as: {0,1,1,0,1,1,1,0,0,1,0,1,1,0,0,1,1,1,0,1}. We repeat this process $M = 10,000$ times to obtain our sample set $\mathcal{K}$.

For this example, we assume the true class we want to recover is the point with all zeros, i.e. $q = \{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0\}$. Given this true class q, we can compute the Hamming distance $d(q,\cdot)$ from q to each of the 10,000 points in our sampled set to obtain the relative distribution:

$$R_{q,\mathcal{K}}(z) = \mathbb{P}\left(d(q,\cdot) \leq z\right) = \frac{\#\text{points in } \mathcal{K} \text{ with Hamming distance} <= z \text{ from q}}{10,000}$$

Obtaining this relative distribution allows us to immediately obtain the nearest neighbor distribution from Equation 2.1 where $M = 10,000$:

$$G_{q,\mathcal{K}}(z) = 1 - (1 - R_{q,\mathcal{K}}(z))^{10,000}$$

We computed $G_{q,\mathcal{K}}(z)$ for all values $z = 1 \ldots 20$. The values for $z = 1 \ldots 5$ are: {0, 0.63214, 0.99988 , 1, 1}. If we set $\gamma = 0.635$, then $G_{q,\mathcal{K}}^{-1}(\gamma) = 2$. This means with probability $\gamma = 0.635$, the nearest neighbor is less than or equal to distance 2 from q. We'll return to this point in a moment.

Now suppose we want to PAC-learn the first stage $\mathcal{S}(\cdot)$ of the zero-shot learner. We assume that each semantic feature (i.e. dimension) is PAC-learnable from X using a linear classifier. We'll draw a training set $\mathcal{D}$ of N samples from our distribution of input points $\mathcal{X}$. We'll obtain these samples by sampling y from any arbitrary distribution $\mathcal{Y}$ over the classes Y, then applying a deterministic input labeling function $\mathcal{H}^{-1}(\cdot)$ to obtain the the input example $x \in X$. For each x, we also obtain the semantic labeling $f \in F$. This provides the training set to PAC learn a collection of $p = 20$ functions, one for each binary semantic feature:

$$S_i : X \rightarrow \{0, 1\} \qquad i = 1 \ldots 20$$

Now, since we're assuming each feature is PAC-learnable using a linear classifier of d dimensions, the standard VC bound applies [Mitchell, 1997]:

$$N \;\; \geq \;\; \frac{1}{\epsilon}\Big[4\log_2(2/\delta) + 8(d+1)\log_2(13/\epsilon)\Big] \tag{2.4}$$

If we use at least N training examples, then with probability $(1-\delta)$, we will obtain a classifier with true error at most ϵ. If we choose an error rate $\epsilon = 0.05$ and $\delta = 0.001$, with $d = 5$ we obtain $N = 8,579$.

Assuming we use $N = 8,579$ training examples, we can compute the total probability of recovering the class by calculating:

1. *The hypothesis probability:* The probability of obtaining p classifiers, each with error rate at most ϵ, is obtained by computing $(1-\delta)^p$ since we assume each is independently learned. With $\delta = 0.001$ and $p = 20$, we obtain 0.98019.
2. *The max error probability:* Making z errors when predicting the semantic features is the same as a Hamming distance of z because z bits are incorrect. Since each of the p classifiers has error rate at most ϵ, the probability of incurring z errors or less is given by the Binomial CDF, binocdf (z; p, ϵ). If the max number of errors is $z = 2$, with $p = 20$ and $\epsilon = 0.05$, we obtain 0.92452.
3. *The recovery probability:* The probability of the nearest neighbor of q being less than z is $\mathbb{P}(\eta_q \leq z) = G_{q,\mathcal{K}}(z) \leq \gamma$. If the error of the prediction $\widehat{q}$ is z, and the nearest neighbor to q in $\mathcal{K}$ is less than z, then the class was most likely not predicted correctly since some point in $\mathcal{K}$ is closer to q than $\widehat{q}$. Since the probability of making this mistake is at most γ, the probability of $\widehat{q}$ being closest to the true class q is at least $(1-\gamma)$. Given our specific sample of points $\mathcal{K}$ along with the true class we want to recover q, we can compute z using $G^{-1}_{q,\mathcal{K}}(\gamma)$. In our example, we choose $\gamma = 0.635$ to obtain $z = 2$. Therefore, the probability of $\widehat{q}$ being closest to q given we make at most $z = 2$ errors is at least $(1 - 0.635) = 0.365$.

To obtain the total (joint) probability, we multiple these three probabilistic events together. Thus, using $M = 10,000$ samples from our semantic space of $p = 20$ dimension, and $N = 8,579$ samples from our input space of $d = 5$ dimensions, the total probability of recovering the point $q = \{0, \ldots, 0\}$ is: 0.33076 using PAC-learnable linear classifiers.

In this particular example the bound is not tight, primarily because of the choice of true class q, along with the random distribution of points in the semantic space. We believe that

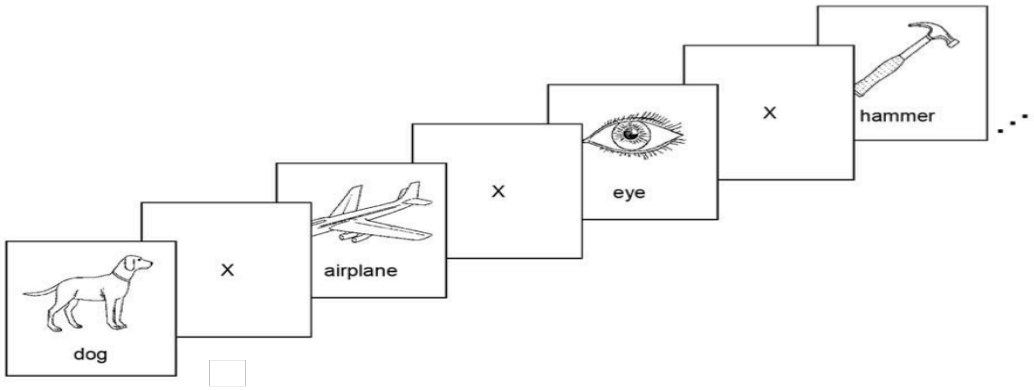


Figure 2.3: The event related paradigm. Line drawings were presented as white lines on black background (shown reversed here for clarity).

given a different distribution as well as true class q that is reasonably far from its neighbors, the bound would be significantly tighter. This is despite the fact that our model follows a worst-case analysis that is typical in the PAC framework.

2.3 Case Study: Classifying Concrete Nouns using fMRI

In this section we empirically evaluate a semantic output code classifier on a neural decoding task. The objective is to decode novel concrete nouns a person is thinking about from fMRI images of the person's neural activity, without including example fMRI images of those words during training.

2.3.1 fMRI Imaging Paradigm and Task

The fMRI datasets we used were the same as those in Mitchell et al. [2008]. Participants were presented with 60 stimuli, each being a picture drawing and corresponding word labeling of a concrete noun. There were five exemplars from twelve different semantic categories (Table 2.1). The line drawings were taken from or modeled after those in Snodgrass and Vanderwart [1980]. White lines and white text were shown over a black background.

The following event-related paradigm was used in fMRI: the stimulus was presented for 3s, followed by 7s of a fixation condition where an X was shown in the center of the screen (Figure 2.3). The set of 60 stimuli were shown six times for fMRI with a random permutation of the set used for each presentation.

The participants were asked to think about properties of the stimulus words during presentation. To encourage a repeatable set of imagined properties for each word, the participants were asked before entering the scanner to create a list of properties for each of the 60 words. Each participant was allowed to create his/her own set of properties and were not limited to a particular set. Further, each participant had no knowledge of the properties selected by the other participants.

Category	Exemplar 1	Exemplar 2	Exemplar 3	Exemplar 4	Exemplar 5
animals	bear	cat	cow	dog	horse
body parts	arm	eye	foot	hand	leg
buildings	apartment	barn	church	house	igloo
building parts	arch	chimney	closet	door	window
clothing	coat	dress	pants	shirt	skirt
furniture	bed	chair	desk	dresser	table
insects	ant	bee	beetle	butterfly	fly
kitchen utensils	bottle	cup	glass	knife	spoon
man-made objects	bell	key	refrigerator	telephone	watch
tools	chisel	hammer	pliers	saw	screwdriver
vegetables	carrot	celery	corn	lettuce	tomato
vehicles	airplane	bicycle	car	train	truck

Table 2.1: The sixty stimulus words with five exemplars from twelve different semantic categories.

2.3.2 fMRI Data Collection Procedures

Eleven human participants were drawn from the Carnegie Mellon community and gave written informed consent approved by the University of Pittsburgh and Carnegie Mellon Institutional Review Boards. Two participants were excluded due to excessive head motion and the remaining nine were right-handed (5 female). The images were collected with a Siemens Allegra 3.0T scanner using a gradient echo EPI pulse sequence with TR = 1000ms, TE = 30 ms and a 60° flip angle. Seventeen 5-mm thick oblique-axial slices were imaged with a gap of 1-mm between slices, and the acquisition matrix was 64 x 64 with 3.125-mm x 3.125-mm x 5-mm voxels. Raw images contained approximately 20,000 voxels. The data were processed using the Statistical Parametric Mapping software (SPM2, Wellcome Department of Cognitive Neurology, London, UK) for slice timing, motion correction, linear trend, and were also normalized into MNI space. A percent change relative to a fixation condition was computed at each voxel for each presentation. A single fMRI mean image was created for each of the 360 item presentations by taking the mean of the four images collected around the peak hemodynamic response (4s-7s post stimulus onset).

2.3.3 Computing Cluster

The experiments described in this chapter were performed using the Open Cirrus [Avetisyan et al., 2010] computing cluster at Intel Labs Pittsburgh. The experiments were implemented using the Distributed Computing Engine (DCE) and Parallel Computing Toolbox (PCT) of MATLAB and were performed with 32 processing cores using the virtual machine manager Tashi.

2.3.4 Acquiring Semantic Knowledge Bases

The success of the zero-shot learning model depends on choosing an appropriate semantic space for the classes. Specifically, the features need to be reasonably predictable from the raw input data. Further, they need to be sufficiently different between classes, so even if they are imperfectly predicted from the first stage of the classifier, the second stage will still have a chance to predict the correct class label.

We collected and tested two semantic knowledge bases for 1,000 concrete nouns, including the 60 stimuli words. In the first semantic knowledge base, `corpus5000`, each word is represented as a co-occurrence vector with the 5000 most frequent words from the Google Trillion Word Corpus, which is a database of statistical co-occurrences of words computed over several billion webpages [23]. Additional parsing and formatting of the co-occurrence statistics were performed by Andy Carlson and the dataset can be downloaded from:
`http://www.thoughtrec.com`.

The second semantic knowledge base, `human218`, was created using the Mechanical Turk human computation service from Amazon.com. Humans were asked 218 questions about the semantic properties of the 1,000 words. The questions were inspired by the game *20 Questions* and were selected to reflect psychological conjectures about neural activity encoding for concrete nouns. For example, the questions were related to size, shape, surface properties, and typical usage. These were selected based on conjectures that neural activity for concrete objects is correlated to the sensory and motor properties of the objects. Example questions include *is it shiny?* and *can you hold it?*. Users of the Mechanical Turk service answered these questions for each word on a scale of 1 to 5 (definitely not to definitely yes). At least three humans scored each question and the median score was used in the final dataset. The dataset was collected by Dean Pomerleau and can be downloaded from: `http://www.thoughtrec.com`.

2.3.5 Data Normalization and Preprocessing

After collecting the fMRI and semantic datasets described above, we applied several unsupervised preprocessing steps before applying the model described below. For fMRI, there were six presentations of each of the 60 stimuli words. We normalized each of the 360 images (i.e. each row) to zero mean and unit variance, and then averaged the six presentations together to obtain a single image for each word. The means over each voxel (i.e. column) were then subtracted to yield zero-mean columns. In the language of the semantic output code classifier, this dataset represents the collection $\mathcal{D}$, with 60 raw-input space examples. In this data matrix, the row index indicates the word, and the column index indicates the voxel.

Also, for both semantic feature sets, the rows were normalized to have unit length. Each row index indicates the word and the column index indicates the semantic feature. This step is especially important for the `corpus5000` text features, to normalize for the different frequencies between words.

[2] Vectors are normalized to unit length and do not include 100 stop words like *a, the, is*.

[3] The Google Trillion Word Corpus is available for download at: http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html

2.3.6 Regression Model

In our experiments, we use *multiple output linear regression* to learn the $\mathcal{S}(\cdot)$ map of the semantic output code classifier. Let $\mathbf{X} \in \Re^{N*d}$ be a training set of examples where each row is the image for a particular word and d is the number of dimensions of the neural activity pattern. During training for fMRI, we apply the *voxel-stability* preprocessing step described in Mitchell et al. [2008] to reduce d from about 20,000 voxels to 500. This is a dimensionality reduction process which selects the voxels with the most consistent response from trial to trial. Let $\mathbf{F} \in \Re^{N*p}$ be a *matrix* of semantic features for those N words (obtained from the knowledge base $\mathcal{K}$) where p is the number of semantic features for that word (e.g. 218 for the `human218` knowledge base). We learn a matrix of weights $\widehat{\mathbf{W}} \in \Re^{d*p}$ which maps from neural activity to a set of semantic features. In this model, each output is treated independently, so we can solve all of the resulting least squares solutions quickly in one matrix operation (even with thousands of semantic features):

$$\widehat{\mathbf{W}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_d)^{-1}\mathbf{X}^T\mathbf{F} \tag{2.5}$$

where $\mathbf{I}_d$ is the identity matrix with dimension $d * d$ and λ is a regularization parameter chosen automatically using the cross-validation scoring function [Hastie et al., 2001, page 216][4]. One disadvantage of Equation (2.5) is that it requires an inversion of a d by d matrix, which is computationally slow (or even intractable) for any moderate number of input features. With this form, it would be impossible to compute the model for fMRI without first reducing the voxels to a smaller number using some method such as the voxel-stability step described earlier.

However, a simple computational technique can overcome this problem by rewriting Equation (2.5) in its *dual* form, also known as its *kernel form*. Following the procedure described in [Jordan and Latham, 2004] we obtain:

$$\widehat{\mathbf{W}} \;=\; \mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_N)^{-1}\mathbf{F} \tag{2.6}$$

This equation is known as *kernel ridge regression* and only requires inversion of an $N * N$ matrix. This is highly useful for neural imaging tasks where N is the number of examples which is typically small, while the number of features d can be very large. Further, another computational technique from [Guyon, 2005] shows that with a little pre-computation, it is possible to obtain the inverse for any additional regularization parameter λ in time $O(N)$, much faster than the time required for a full inverse $O(N^3)$. Combined with the cross-validation scoring function from Hastie et al. [2001, page 216], the end result is an extremely fast method for solving the resulting regression even with thousands of input and semantic features, while automatically selecting the best regularization parameter λ from a large grid of possible parameter choices. Sample code for this procedure is available at:
`http://www.thoughtrec.com`

[4]We compute the cross-validation score for each output (i.e. prediction of a particular semantic feature), and choose the parameter that minimizes the average loss across all outputs.

Using this form, it is possible to quickly obtain the weight matrix $\widehat{\mathbf{W}}$. Then, given a novel neural image $\mathbf{x} \in \Re^{1*d}$, we can obtain a prediction $\widehat{\mathbf{f}} \in \Re^{1*p}$ of the semantic features for this image by multiplying the image by the weights:

$$\widehat{\mathbf{f}} = \mathbf{x} \cdot \widehat{\mathbf{W}} \tag{2.7}$$

For the second stage of the semantic output code classifier, $\mathcal{L}(\cdot)$, we simply use a 1-nearest neighbor classifier. In other words, $\mathcal{L}(\widehat{\mathbf{f}})$ will take the prediction of features and return the closest point in a given knowledge base according the specified distance metric. Since our semantic datasets used real valued numbers, we will use the Euclidean distance (L_2) metric.

2.3.7 Experimental Results

Using the model and datasets described above, we now pose and answer several questions.

1. Can we build a classifier to discriminate between two classes, where neither class appeared in the training set?

To answer this question, we performed a *leave-two-out-cross-validation*. Specifically, we trained the model in Equation 2.6 to learn the mapping between 58 fMRI images and the semantic features for their respective words, leaving out the other two words from the 60 available. For both held out images, we applied the learned weight matrix $\widehat{\mathbf{W}}$ to obtain a prediction of the semantic features, and then we used a Euclidean distance metric that equally weights features to compare the vector of predictions to the true semantic encodings of the *two held-out words*. The labels were chosen by computing the sum of the distances in the two labeling configurations (i.e. the words with their true labels or the reverse labeling) and choosing the labeling that results in the smallest total distance. For example, if *dist()* is the Euclidean distance, and p_1 and p_2 are the two predictions for the held out words, and s_1 and s_2 are the true semantic encodings, then the labeling was correct if:

$$dist(p_1, s_1) + dist(p_2, s_2) < dist(p_1, s_2) + dist(p_2, s_1)$$

This process was repeated for all $\binom{60}{2} = 1,770$ possible leave-two-out combinations.

Figure 2.4 shows the results for the two different semantic feature knowledge bases described earlier. We see that the `human218` semantic features significantly outperformed the `corpus5000` features, with mean accuracies over the nine participants of 88.9% and 76.3% respectively, while chance accuracy is only 50%.

We believe that the lower performance of the `corpus5000` features is due to the overloaded meaning of individual words that results from using co-occurrence statistics of web corpora. For example, the word *bee* (the insect) most frequency occurs with the word *gees*, obviously because of the rock band *The Bee Gees*. As another example, we noticed the word *saw* has co-occurrences based on its many uses, both as a noun and verb. Clearly, the context cannot be captured by using simple co-occurrence statistics. In the `human218` features, context is not an issue as it was specified in advance and given as part of the labeling instructions for a given word.

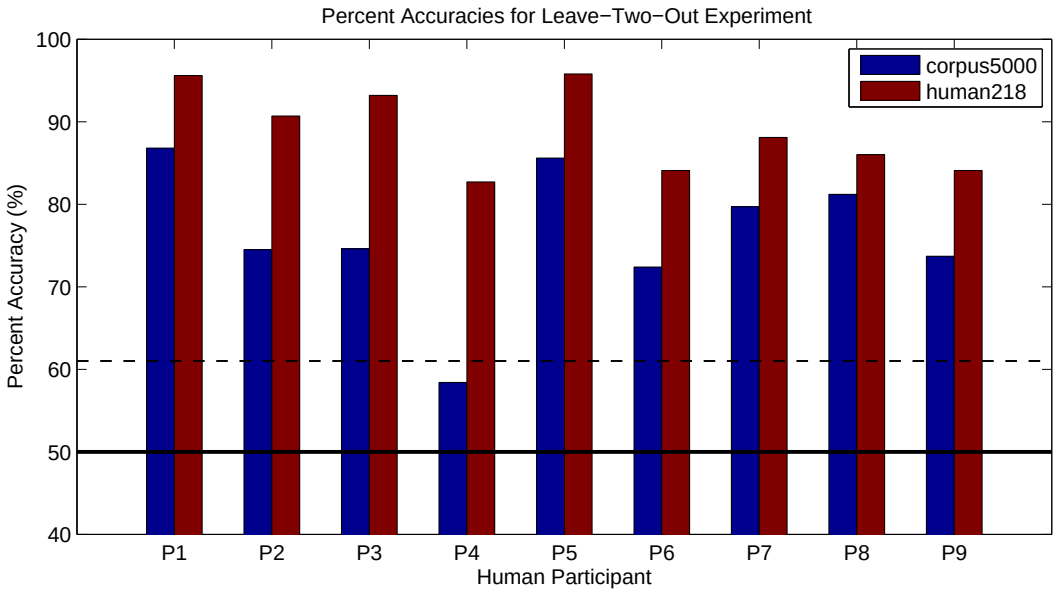


Figure 2.4: Percent accuracies for leave-two-out-cross-validation for 9 fMRI participants (labeled P1-P9). The values represent classifier percentage accuracy over 1,770 trials when discriminating between two fMRI images, both of which were omitted from the training set. The chance accuracy level is shown as a solid black line and the significance threshold (at the 5% level) is shown as the dotted black line.

To determine statistical significance of both accuracies, we estimated the empirical distribution of chance accuracies using simulation. Specifically, we repeated the experiment 1,000 times, replacing the true fMRI data for each subject with random data generated from a normal distribution. We used the same preprocessing pipeline on the random data. Using the resulting accuracies of these random experiments, we determined the significance threshold at the 5% level to be 61%. At this level, eight of nine subjects were found to be significant for the `corpus5000` features, and all nine subjects were found to be significant for the `human218` features.

For both feature sets, we see that:

For each of the nine participants, it is possible to discriminate between two novel classes, even though neither appeared in the training set.

This is an important result for neural imaging studies, as it is not absolutely necessary to collect training data for each class we wish to discriminate. It is only necessary that the training data share semantic similarities with the held out classes. This is similar in spirit to our previous results in Mitchell et al. [2008] that shows we can *generate* fMRI images for a novel word. *However, for the purpose of concrete noun recognition, where the goal is to decode the specific noun, a model that predicts the semantic features from the fMRI far outperforms a model that predicts the fMRI image from the semantic features.* The increased accuracy is not just from using a better feature set, as using the same generative model described in Mitchell et al. [2008] with the `human218` features described here yields accuracy of 82%, compared to the accuracy

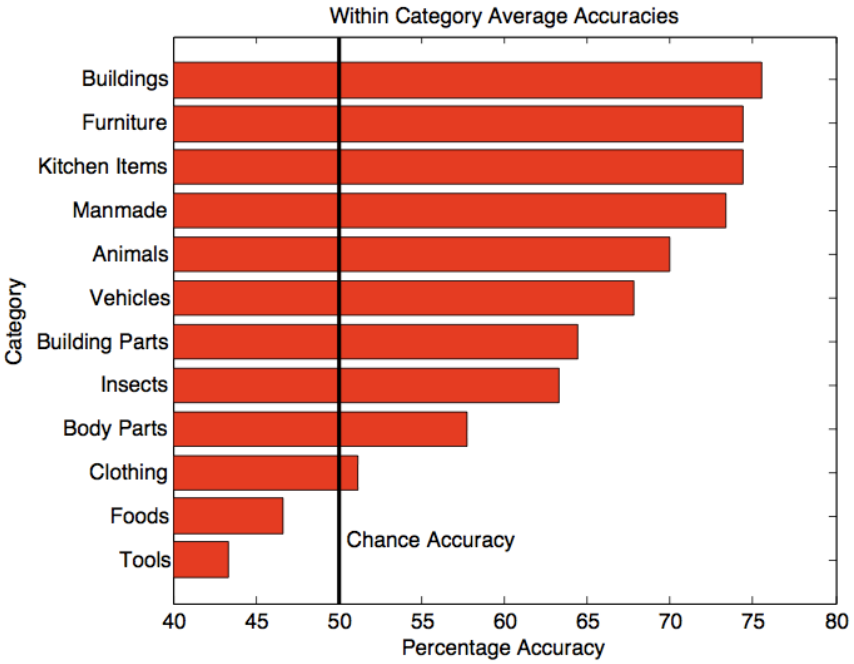


Figure 2.5: Average accuracies when discriminating words in the same category in the *leave-two-out* prediction task. Results are averaged by category across nine fMRI participants. This result shows that using intermediate semantic features often allows words to be discriminated even when they are in the same category of object.

using the zero-shot learning model of 88.9%.

We conjecture that even using voxel selection, there are only a few relevant voxels useful for predicting semantic features. For a given semantic feature, we believe most voxels will be irrelevant and will be disregarded because of the model regularization if voxels are used as the model's input. If voxels are used as outputs, however, we suspect many will not be learnable from the semantic features. This task is also harder, since the voxels values are real-valued compared to the more categorial nature of the semantic features, at least in the case of the `human218` features. As another point, there is significant redundancy in the `human218` features as many are highly correlated. For example *is it alive?* and *is it manmade?* are nearly perfectly correlated. We believe this redundancy leads to more accurate predictions in the second stage of the zero-shot learner, similar to the effects observed in multi-class classifiers that utilize error-correcting output codes [Dietterich and Bakiri, 1995].

2. What if the two classes are from the same category?

We also computed the accuracy of the leave-two-out experiment when only considering test pairs of held-out words that *are in the same category*. The goal was to determine whether the classifier could distinguish words at a granularity higher than the category of object. Figure (2.5) shows that for ten of twelve categories, it is possible to discriminate between words in the that category above the chance level. When averaged over all nine fMRI subjects and twelve cate-

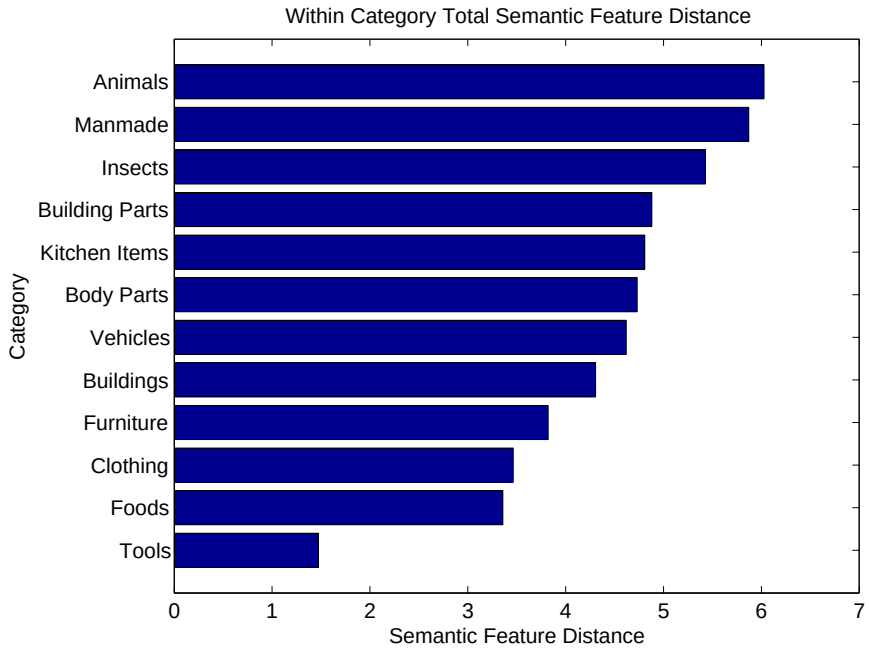


Figure 2.6: Average distances between words in the same category using the `human218` semantic features. Features for an individual word were normalized to unit length before comparisons were made. The categories *clothing, foods* and *tools* had the smallest semantic feature distances on average, and also the lowest within-category predictive accuracy. The categories *buildings* and *furniture*, however, had relatively small semantic feature distances between words, but these two categories had the highest within-category prediction accuracy, suggesting that features that discriminate words within these categories can be predicted more accurately than the features that discriminate other categories such as *clothing*.

gories, 686 of 1080 possible tests were correct, yielding nearly 64% accuracy. For comparison, the accuracy when word pairs were not in the same category was 90.7%.

To investigate why certain categories had higher within-category accuracy, we plotted the average distances between words in the same category using the `human218` semantic features in Figure 2.6. The categories *clothing, foods* and *tools* had the smallest semantic feature distances on average, and also the lowest within-category predictive accuracy. The categories *buildings* and *furniture*, however, had relatively small semantic feature distances between words, but these two categories had the highest within-category prediction accuracy, suggesting that features that discriminate words within these categories can be predicted more accurately than the features that discriminate other categories such as *clothing*.

We also reproduced an image from the Online Supporting Material of Mitchell et al. [2008] that shows the cosine similarity between the actual fMRI images for the 60 stimuli words used in our experiments. The Figure 2.7 was generated using the same dataset used in our work. We see high similarity for images in the *buildings* category and also for the *tools* category. Interestingly, the *buildings* category had the highest within-category predictive performance using our `human218` semantic features, while *tools* had the lowest. This is most likely the result of small

distances between semantic features describing *tools* as well as higher predictive accuracy of semantic features that discriminate *buildings*.

This leads us to conjecture that within-category discrimination is possible when enough semantic features differ between objects in the same category, and these features are also predictable from the neural activity observed. We explore this conjecture in the next section.

3. How is the classifier able to discriminate between closely related novel classes?

Figure 2.8 shows ten semantic questions (features) from the `human218` dataset. The graph shows the true values along with the predicted feature values for both *bear* and *dog* when trained on the other 58 words. We see the model is able to learn to predict many of the key features that bears and dogs have in common such as *is it an animal?* as well as those that differentiate between the two, such as *do you see it daily?* and *can you hold it?* For both of these novel words, the features predicted from the neural data were closest to the true word.

4. Can we decode the word from a large set of possible words?

Given the success of the semantic output code classifier at discriminating between the brain images for two novel words, we now consider the much harder problem of discriminating a novel word from a large set of candidate words. To test this ability, we performed a *leave-one-out-cross-validation*, where we trained using Equation 2.6 on images and semantic features for 59 words. We then predicted the features for the held-out image of the 60th word, and then performed a 1-nearest neighbor classification in a set of candidate words.

We tested two different candidate word sets. The first was `mri60` which is the collection of all 60 concrete nouns for which we collected fMRI data, including the 59 training words and the single held out word (these were the *only* words for which we collected fMRI data). The second set was `noun940`, a collection of 940 English nouns with high familiarity, concreteness and imaginability, compiled from Wilson [1988] and Snodgrass and Vanderwart [1980][5]. For this set of words, we added the true held-out word to the set of 940 on each cross-validation iteration. We performed this experiment using both the `corpus5000` and `human218` feature sets. The rank accuracy results (over 60 cross-validation iterations) of the four experiments are shown in Figure 2.9.

The rank accuracy shown measures the rank or likeliness of the true class against all the other class predictions. It is useful to gauge classifier performance when the classifier does not rank the true class as the most likely. For example, if there are 100 possible classes and the true class was considered more likely than 90 other classes (rank of 10), the rank accuracy would be 0.909 or 90.9%. Rank accuracy can be computed by:

$$\text{Rank accuracy} = 1 - \frac{\text{Rank of true class rank among predictions} - 1}{\#\text{ of classes} - 1}$$

Using this rank accuracy measure in Figure 2.9, we see the `human218` features again significantly outperform `corpus5000` on both mean and median rank accuracy measures. While

[5]The list of words does not include the 60 words for which we collected fMRI data. This list is available online at: `http://www.thoughtrec.com`

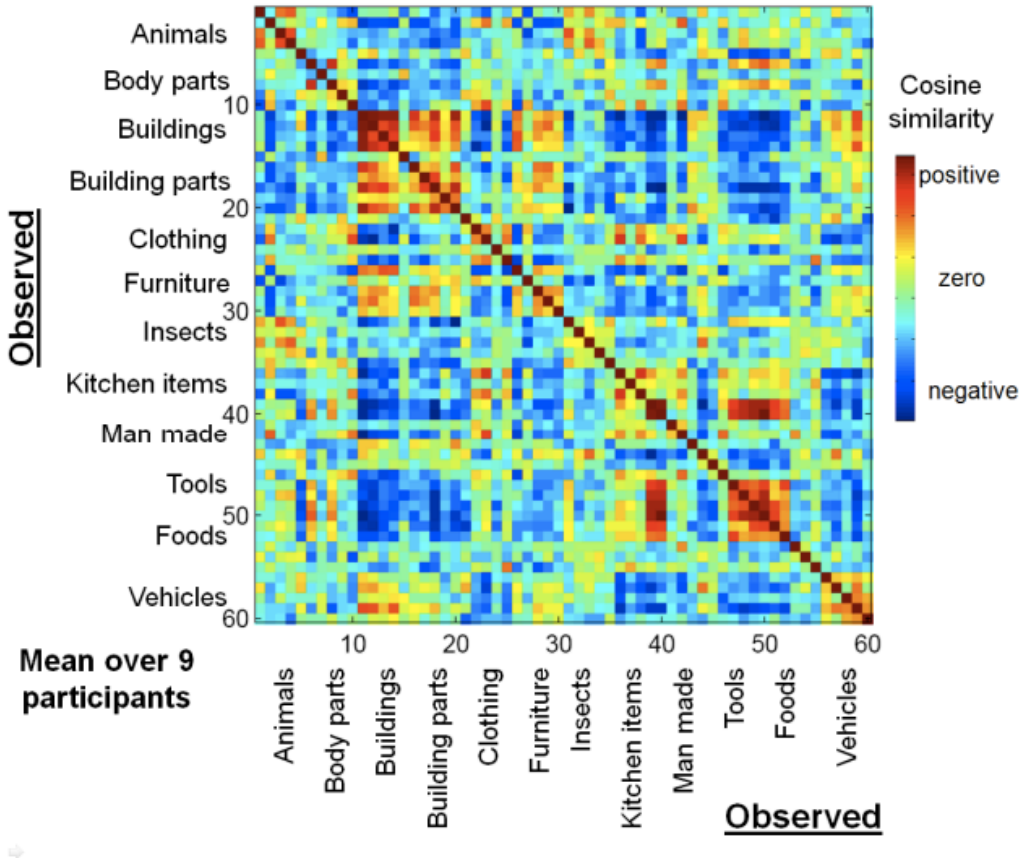


Figure 2.7: An image reproduced from the Online Supporting Material of Mitchell et al. [2008] that shows the cosine similarity between fMRI images for the 60 stimuli words. This is the same dataset used in our work. We see high similarity for images in the *buildings* category and also for the *tools* category. Interestingly, the *buildings* category had the highest predictive performance using our `human218` semantic features, while *tools* had the lowest. This is most likely the result of small distances between semantic features describing *tools* as well as higher predictive accuracy of semantic features that discriminate *buildings*.

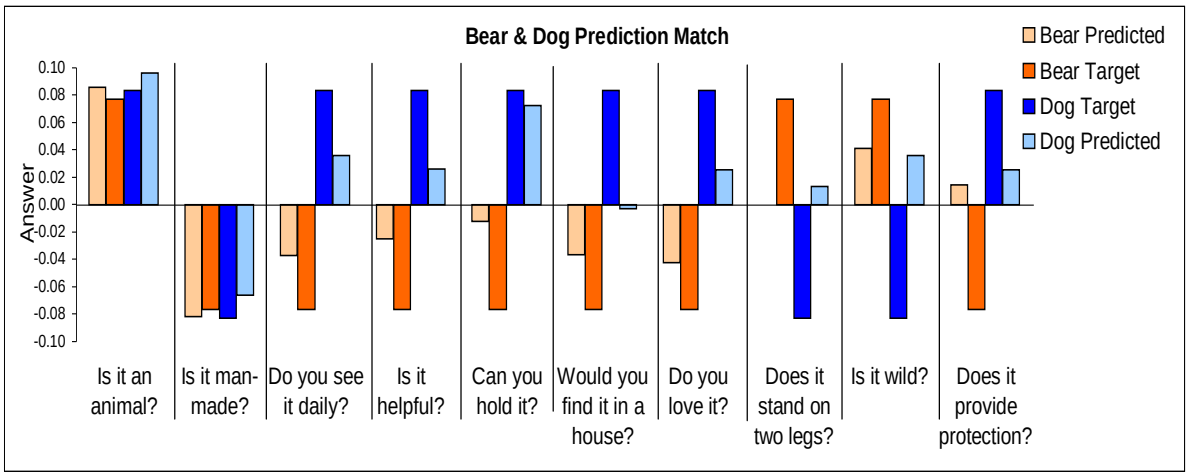


Figure 2.8: Ten semantic features from the human218 knowledge base for the words *bear* and *dog*. The true encoding is shown along with the predicted encoding when fMRI images for bear and dog were left out of the training set. The Y-axis represents the value of the semantic features after scaling normalizations.

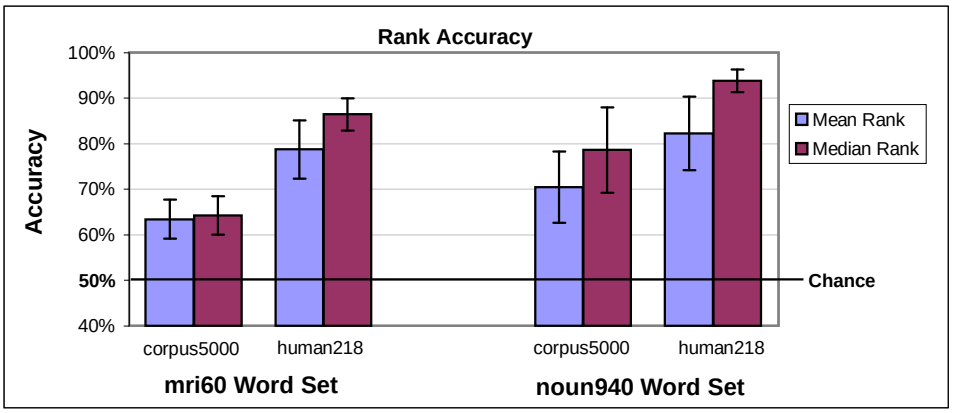


Figure 2.9: The mean and median rank accuracies across nine participants for two different semantic feature sets. Both the original 60 fMRI words and a set of 940 nouns were considered.

both feature sets perform well above chance, we believe once again that the `human218` features performed much better because they do not suffer the multiple meanings and overloaded context that are endemic in the `corpus5000` features. For example, in one context on the web the word *saw* might correspond to the physical tool, but in another context it might represent the past tense of the verb *to see*. Further, the `human218` features were also collected while humans were actually thinking about the attributes of the given words, similar to what the human subjects were asked to do in the neural imaging experiment.

On 12 of 540 total presentations of the `mri60` words (60 presentations for each of nine participants), the `human218` features *predicted the single held-out word above all 59 other words in its training set*. With 60 possible choices, the chance accuracy is 1.7%. While just a bit above expected number of chance correct predictions (9 of 540), the fact that the model *ever* chooses the held-out word over all the training words is noteworthy since the model is likely to be biased towards predicting feature values similar to the words on which it was trained. On the `noun940` words, the model predicted the correct word from the set of 941 alternatives *a total of 26 times* (4.8%) for the `human218` features and 22 times (4.1%) for the `corpus5000` features. For the most accurate subject, the model correctly picked the right word from the set of 941 nearly 12% of the time. With 941 possible choices, the chance accuracy of predicting a word correctly is only 0.1%, meaning we would expect less than one correct prediction across all 540 presentations.

As Figure 2.9 shows, the median rank accuracies are often significantly higher than the mean rank accuracies. Using the `human218` features on the `noun940` words, the median rank accuracy is above 90% for each participant while the mean is typically about 10% lower. Using simulation, we determined the accuracy at the 5% statistical significance level to be 63.8%, yielding p-values near zero for all participants for both mean and median results.

We believe the mean to be worse than the median due to the fact that several words are consistently predicted poorly. The prediction of words in the categories *animals, body parts, foods, tools,* and *vehicles* typically perform well, while the words in the categories *furniture, man-made items,* and *insects* often perform poorly.

Bear	**Foot**	**Screwdriver**	**Train**	**Truck**	**Celery**	**House**	**Pants**
(1)	(1)	(1)	(1)	(2)	(5)	(6)	(21)
bear	*foot*	*screwdriver*	*train*	jeep	beet	supermarket	clothing
fox	feet	pin	jet	*truck*	artichoke	hotel	vest
wolf	ankle	nail	jail	minivan	grape	theater	t-shirt
yak	knee	wrench	factory	bus	cabbage	school	clothes
gorilla	face	dagger	bus	sedan	*celery*	factory	panties

Table 2.2: The top five predicted words for a novel fMRI image taken for the word in bold (all fMRI images taken from the top performing participant). The number in the parentheses contains the rank of the correct word selected from 941 concrete nouns in English. Eight words were selected to show the semantic similarities of the predicted words for both accurate and inaccurate predictions.

To gain intuition as to how the model is able predict held out words from such a large set, we selected the word `bear` and plotted the predicted semantic features for `bear` along with the semantic feature vectors for all 60 fMRI stimulus words (from the `human218` dataset). To visualize the classifier performance, we created a 2-D embedding by performing a dimensionality reduction on the combined data using *t-weighted stochastic neighborhood embedding*(t-SNE) [van der Maaten and Hinton, 2008]. Figure 2.10 shows the result. We see the predicted semantic features for the word `bear` in red, along with its true encoding shown in black. In both the original space as well as this lower dimensional space, we can see that the prediction for `bear` is closer to its true encoding than any other word. As a result, this prediction was made correctly.

Even when the correct word is not the closest match, however, the words that best match the predicted features are often very similar to the held-out word. Table 2.2 shows the top five predicted words for eight different held-out fMRI images for participant P1 (i.e. the 5 closest words in the set of 941 to the predicted vector of semantic features). For example, while `truck` was not the closest word to the model's prediction, it was second behind `jeep`, and all five of the closest words to the prediction were vehicles. These results show that:

By using an intermediate semantic feature representation, it is often possible to discriminate words that are within the same category, even when choosing from a large possible set of words.

Negative Results

Computational speed was a large factor in choosing the kernel ridge regression model for the first stage of the classifier. A common question we receive is why not use a more modern method like Support Vector Machines or Support Vector Regression. Besides being significantly computationally slower, our tests found no performance advantage of these more complicated algorithms over the simpler kernel ridge regression model.

Support Vector Regression offered no noticeable advantage and was roughly equivalent in performance, while Support Vector Machines offered a substantial decrease in performance. We attribute this to the information loss incurred when modifying the semantic features to binary. It does appear from our experiments that there is substantial value to having semantic features that

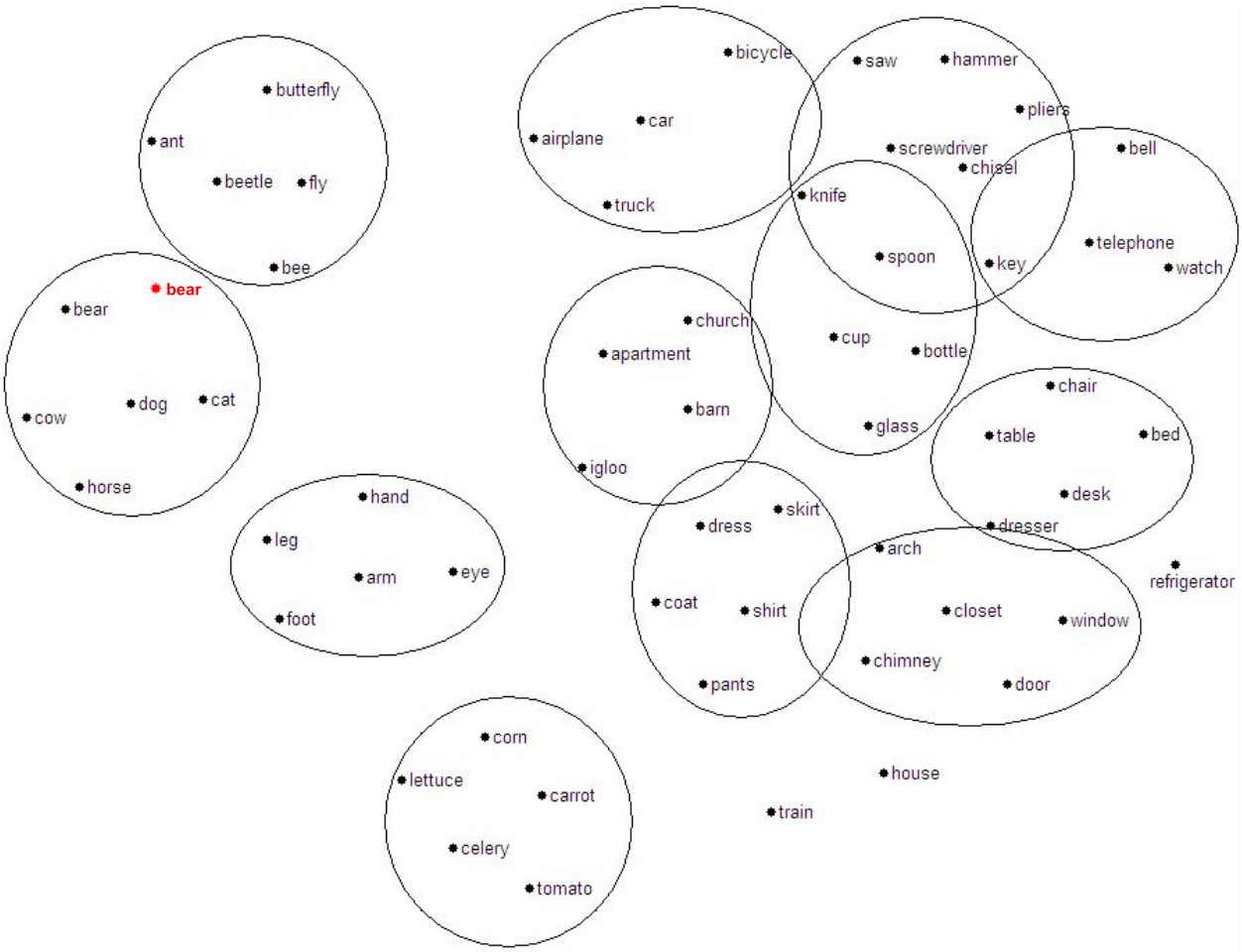


Figure 2.10: A 2-D visualization of the semantic feature space for the 60 fMRI stimulus words, along with the predicted feature vector for the word *bear* shown in red. The prediction for the word bear was closer to the true encoding for bear than for any other word. This two-dimensional plot was generated by applying the dimensionality reduction method *t-SNE* to the original semantic space of 218 dimensions. Circles were added to show the category information.

are real valued, or at least have several levels of gradation.

We also experimented with Markov-Random-Field (MRF) models to leverage the geometric relationships between the voxels. Again, performance was slightly lower than our simpler heavily regularized linear models.

We also explored Restricted Boltzmann Machines (RBMs) to learn better input features to the linear model. Although the RBMs learned a layer of features that appeared to be highly localized neural activity patterns, the performance was not better than the standard regularized linear model.

Similarly, supervised neural networks offered little to no performance advantage over the standard regularized linear model.

Regularization did matter significantly, and improper tuning of the lambda penalty could lead to poor performance. Given the very noisy data, with many irrelevant voxel features, and

so few examples, we are not optimistic about the performance potential of more sophisticated algorithms.

Using the kernelized version of the ridge regression algorithm we explored models that utilized all 20,000 voxels across the entire brain rather than just the 500 selected by the unsupervised preprocessing *voxel-stability* method. While the models worked, performance was fairly lower.

In our *leave-two-out* experiment, we explored numerous methods for selecting better subsets of semantic features per holdout. Intuitively, we believed we should only focus on the semantic features that allowed us to most reliably discriminate the held-out pair. While we were convinced this would lead to better performance, we were unable to generate a method that performed universally better than just using all the available features. We explore this in more depth in the following chapter, and report several important, and slightly counter-intuitive discoveries about the zero-shot learning model.

2.4 Future Work

We found in our zero-shot learning experiments that the largest improvement in classifier accuracy resulted from using the human labeled semantic features (`human218`) rather than those generated from corpus statistics (`corpus5000`). While the `human218` semantic feature set performed reasonably well, additional work should explore what other semantic features could be classified accurately. The features we considered focused mostly of physical properties such as size and shape. Other semantic features, such as those related to function or purpose should be explored as well.

Future work should also explore classifying words that are not concrete nouns, such as adjectives, verbs, and also more abstract nouns like *democracy*. If different parts of speech can be classified, an obvious next goal would be to experiment with classifying phrases or short sentences. This challenging goal will require models that fully utilize the temporal dimension of neural activity, rather than the simpler single time point features described in this chapter. It is likely that scanners such as MEG and EEG that better measure the temporal dimension of neural activity will be required.

Another interesting direction would be to combine the methodologies described here with other brain-computer-interface techniques like P300 [Donchin et al., 2000]. P300 is a very consistent pattern of neural activation that occurs 300 milliseconds after a thought or perception and is commonly used as a triggering signal in brain-computer-interfaces [Guger et al., 2011]. To combine with P300, the zero-shot learning model could be used to narrow word selections down to a small set of semantically similar choices, while the P300 signal could be used to accurately make a selection from the small set. We believe the combination of these two techniques may eventually lead to non-invasive *vocal prosthetics*. While it may not be possible to decode full sentences, recognition of even small vocabularies combined with speech synthesis capabilities may lead to useful applications for the disabled.

Other researchers have explored vocal prosthetics using invasive techniques that measure firings of small numbers of neurons directly [Brumberg et al., 2010]. Rather than considering word semantics, these researchers focus on predicting specific phonemes of the desired words. To our knowledge, exploring *invasive* techniques for classification of word semantics have not

been explored, and we believe this would be an interesting area for further research. However, in our own work and also that of Mitchell et al. [2008], we've found the neural activity patterns of semantic processing to be highly distributed, making it difficult to study invasively using the tiny microarrays commonly used in invasive neural prosthetics [Schwartz, 2004].

For more machine learning oriented researchers, the zero-shot learning paradigm is a rich area for further study. We believe there are many applications of this model including computer vision and analogy solving. For example, in computer vision, it has recently been shown that it is possible to classify certain attributes of objects from images [Lampert et al., 2009, Farhadi et al., 2009]. Similar to our own work, these papers show that it is possible to classify objects with far fewer training examples.

With regards to analogy solving, we believe one way to solve analogies, such as those used in standardized tests like the SAT, is to model the relationships between words using crowd-sourcing tools like Mechanical Turk to generate a semantic feature space that contains both physical and functional attributes of words. Solving an analogy might be reduced to computing distances within this semantic space. Similar work using semantic features based on corpus statistics is described in Turney [2006, 2008], Veale [2004].

We also feel there are interesting theoretical challenges related to the zero-shot learning model, from proving performance guarantees under different modeling assumptions, to choosing optimal coding of semantic information. The choice of optimal semantic code is closely related to the problem of active learning, which seeks to choose the most useful training examples for the learning algorithm. This is particularly useful when training examples are difficult to acquire, which is certainly the case when collecting neural data from fMRI.

Given a particular semantic code, an active learning algorithm could minimize the number of training examples necessary to predict a particular set of classes. It could also be used to automatically explore different semantic codes by choosing examples to determine predictability of certain semantic features.

2.5 Conclusion

In this chapter, we presented a formalism for a zero-shot learning algorithm known as the semantic output code classifier. This classifier can predict novel classes that were omitted from a training set by leveraging a semantic knowledge base that encodes features common to both the novel classes and the training set. We studied this model in a PAC framework and proved the first formal guarantee that shows conditions under which this classifier will predict novel classes.

We applied this framework to the task of *concrete noun recognition* and demonstrated that we can discriminate concrete nouns that people are thinking about far above the chance level without any training examples of those nouns. This is an important result for the neural imaging community as it shows that using a small amount of training data, it is possible to discriminate a much larger set of cognitive states, thereby saving significant time and expense when collecting data from human subjects.

When applied to fMRI data, we showed that the model can discriminate certain nouns that are within the same category with significantly higher accuracy than previously possible. These results advance the state-of-the-art in neural decoding and are a promising step towards a large

vocabulary brain-computer interface.

2.6 Additional Chapter Remarks

Sample Code

We provide the fMRI datasets used in this chapter as well as sample code for the described methodologies. The code and data can be downloaded from: `http://www.thoughtrec.com`

Additional Acknowledgements

Many thanks to Marcel Just and Vladimir Cherkassky for acquiring and preprocessing the fMRI data used in this chapter. Gustavo Sudre, Dean Pomerleau, Geoff Hinton, Tom Mitchell were co-authors on a paper [Palatucci et al., 2011] that included the majority of these results. They contributed greatly to the model design and experimental results. Andy Carlson and Dean Pomerleau collected the semantic feature sets. We would also like to thank Intel Labs for their support, especially Richard Gass and Mike Ryan for their help using the Open Cirrus computing cluster.

Chapter 3

Semantic Feature Selection using Support Vector Ranking

In the previous chapter we found that performance of the zero-shot learning model varied greatly depending on the semantic features used. While we considered several different semantic feature sets independently, our models used all the available semantic features during training, without considering that several of the features may be irrelevant, meaning they are unpredictable from the input data and potentially harmful to task performance.

In this chapter, we present a method based on *support vector ranking* [Herbrich et al., 1999] and structured prediction [Ratliff et al., 2007] that can select the most useful semantic features during training time. This allows the model to automatically disregard the least useful features, easing the burden of creating a semantic feature set by eliminating the need to identify the most useful features *a priori*. The interpreted model can yield insights as to what semantic features are not only predictable, but also useful for discriminating the classes from one another.

While we found the support vector ranking method can identify useful subsets of semantic features, the performance benefit was most substantial when the final semantic feature sets used in the model were small. Surprisingly, the benefit of selecting subsets of semantic features decreased as the number of features used in the model increased. Consequently, the zero-shot learning model appears quite robust to irrelevant features, meaning *it is far more important to make sure useful features are included in the model, rather than prevent irrelevant features from being excluded.* Further, we report initial theoretical results that appear to predict this phenomenon.

Chapter Notes for Neural Imaging Researchers

For the neural imaging researcher, we report the semantic features that were most useful when applying zero-shot learning to a neural imaging task. These include several related to *density and internal structure* as well as *shelter* qualities of the object. It is important to note that these semantic features are not only predictable, but also useful for separating the classes. Some features that are commonly known to be predictable above chance, such as whether an object is manmade, were not found useful for discriminating individual classes very precisely, and were given low weight by the support vector ranking method.

Chapter Notes for Machine Learning Researchers
For the machine learning researcher, the primary result of this chapter is the robustness of the zero-shot learning model to irrelevant features given sufficient size of the semantic feature set. Also interesting, is the theoretical analysis that may explain this behavior, as well as a conjecture as to the number of semantic features to use. Specifically, our theoretical results suggest the number should be greater than fifty, but it also shows there might be little benefit to using more than a few hundred.

3.1 Background

In the previous chapter, we presented a zero-shot learning algorithm called the *semantic output code classifier* that uses a collection of intermediate semantic features in a 2-stage process to make class predictions. Recall the formulation:

Semantic Output Code Classifier
A semantic output code classifier $\mathcal{H} : X^d \rightarrow Y$ maps points from some d dimensional raw-input space X^d to a label from a set Y such that $\mathcal{H}$ is the composition of two other functions, $\mathcal{S}$ and $\mathcal{L}$, such that:

$$\begin{aligned} \mathcal{H} &= \mathcal{L}(\mathcal{S}(\cdot)) \\ \mathcal{S} &: X^d \rightarrow F^p \\ \mathcal{L} &: F^p \rightarrow Y \end{aligned}$$

One obvious question regarding this model is how to choose the best set of intermediate semantic features F^p for the given task. While it is possible to test many different sets manually, we would like to generate an algorithm that can automatically choose a useful subset of semantic features from a potentially much larger set, ignoring the features that are irrelevant to the current task.

Recall from the previous chapter that the implementation of this model uses a simple Euclidean distance metric (L_2) in its second stage $\mathcal{L}(\cdot)$, to compute the nearest class in the semantic knowledge base. This metric weights each feature equally when computing the distance between the predicted features and the classes in the knowledge base. Ideally, we could find a metric that would weight the features differently, according to their usefulness. For example, a feature like *"does it have feathers?"* is so specific that it may not be accurately predictable, and therefore would provide little information to discriminate words. We would like such less relevant features to have a lower weight when we compute the distance metric.

One way to compute such a metric is to use *support vector ranking*, a technique similar in spirit to traditional distance metric learning, but computationally much simpler to implement. Intuitively, the algorithm looks for features that are predicted well that also help discriminate the classes from each other, and weights these features higher in the metric. The algorithm achieves this by finding the smallest weight vector w, such that the weighted sum of squared-errors (a.k.a the distance) between the predicted features and true semantic encoding for a given word is less than that for any other word in the semantic knowledge base.

This is not simply choosing the features that are best predicted, however, as the feature weights are computed relative to the semantic encodings of the other classes. Even if a certain feature is predicted well, it may not provide much information to help discriminate between classes in the knowledge base, and so will be given a low weight in the new metric. Thus, this method tries to balance the selection of features to choose those that are predicted well, but also those that create the best separating margin between the different classes.

3.2 Support Vector Ranking

3.2.1 Model Formalism

We now present the formalism for this method based on the structured prediction work of Ratliff et al. [2007]. Suppose we have a knowledge base of semantic features for N words $\mathcal{K} = \{s_i^*\}_{1:N}$ where s_i^* is the "true" semantic vector for word i. Suppose we have trained our zero-shot learning model and get a prediction $\widehat{s}_i$ for word i from our first stage $\mathcal{S}(\cdot)$.

Let $\widehat{f}_i^* = (\widehat{s}_i - s_i^*)^2$ be the vector whose pth element is the square of the pth element of $(\widehat{s}_i - s_i^*)$, the vector of differences between the prediction and the true encoding in the knowledge base for word i. Similarly, let $\widehat{f}_i^j = (\widehat{s}_i - s_j^*)^2$ be the vector of squared distances when the prediction for i is compared with the true encoding for word j.

We learn a vector of weights w that we'll use to compute a weighted sum of prediction errors. To do this, we perform sub-gradient descent [Shor, 1985, Ratliff et al., 2007] using the following update rule with learning rate η and regularization parameter λ:

$$\begin{aligned} j &= argmin_{k \neq i}[w^T \widehat{f}_i^k] \\ w &\leftarrow w - \\ &\quad \eta\Big[w + \big[(w^T \widehat{f}_i^* - w^T \widehat{f}_i^j < -1)\ ?\ 0 : \lambda(\widehat{f}_i^* - \widehat{f}_i^j)\big]\Big] \end{aligned}$$

where "?" is the ternary operator and the notation *(condition) ? A : B* means *if (condition) then A else B*.

While the derivation is slightly non-trivial (see section A), the update rule is easy to implement. After learning the regression coefficients in Equation (2.6), we apply this update rule to each example in our training set, and continue to cycle through the examples until the change in w is below some threshold. As is typical in gradient descent methods, the learning rate η must be sufficiently small to assure convergence. We initialize w to a vector of 1's, weighting all feature prediction errors equally. We also initialize our learning rate to 1, although we decrease it after each round through the training data as is common in gradient descent [Bishop, 2006]:

$$\eta \leftarrow \frac{\eta}{\text{Current round \#}}$$

We also initialize our regularization parameter λ to 1. We found, however, that performance is very robust to changes in this parameter. When evaluating a test example, we apply our coefficients from Equation (2.6) as usual, but then in the 2nd stage of the classifier, $\mathcal{L}(\cdot)$, we weight our squared errors of the semantic features by the learned vector w, to compute the nearest match in our knowledge base.

Highest Weighted Semantic Features	Least Weighted Semantic Features
IS IT DENSE?	*IS IT A MAMMAL?*
IS IT PART OF SOMETHING LARGER?	*IS IT A BODY PART?*
DOES IT OPEN?	*IS IT A PERSON?*
DOES IT LIVE ABOVE GROUND?	*IS IT A VEGETABLE / PLANT?*
DOES IT HAVE A HARD INSIDE?	*IS IT AN INSECT?*
DOES IT CAST A SHADOW?	*IS IT AN ANIMAL?*
DOES IT HAVE AT LEAST ONE HOLE?	*IS IT MANMADE?*
CAN YOU SIT ON IT?	*DOES IT LAY EGGS?*
DOES IT CONTAIN SOMETHING ELSE?	*IS IT ALIVE?*
IS IT POINTED / SHARP?	*DOES IT HAVE FEATHERS?*
CAN IT BREAK?	*DOES IT HAVE SEEDS?*
DOES IT HAVE INTERNAL STRUCTURE?	*DOES IT LIVE IN WATER?*
WOULD YOU FIND IT IN A RESTAURANT?	*CAN IT BITE OR STING?*
CAN YOU WALK ON IT?	*DOES IT HAVE A HARD NOSE/BEAK?*
IS IT YELLOW?	*CAN IT JUMP?*
CAN YOU BUY IT?	*IS IT TASTY?*
DOES IT COME IN A BUNCH/PACK?	*IS IT MANUFACTURED?*
DO YOU USE IT DAILY?	*DOES IT HAVE ROOTS?*
WOULD YOU FIND IT ON A FARM?	*DOES IT GROW?*
IS IT ALWAYS THE SAME COLOR(S)?	*DOES IT HAVE A BACKBONE?*
DO YOU SEE IT DAILY?	*IS IT A VEHICLE?*
DO YOU LOVE IT?	*IS IT MORE THAN ONE COLOR?*
DOES IT PROVIDE PROTECTION?	*DOES IT HAVE WHEELS?*
CAN IT KEEP YOU DRY?	*DOES IT HAVE SOME SORT OF NOSE?*
DO YOU INTERACT WITH IT?	*IS IT WARM BLOODED?*

Table 3.1: The 25 highest weighted features (highest to lowest) and the 25 lowest weighted features (lowest to highest) as discovered by the Support Vector Ranking algorithm when applied to Subject 1.

3.3 Experimental Results

What are the best semantic features and how do we choose them?

We applied the support vector ranking method to learn a ranking of semantic features for the *leave-one-out* prediction task described in the previous chapter. On each iteration of the cross-validation for our most accurate subject, we computed a new weight vector using the method, and summed the total weight for each feature across all iterations. Using this total we ranked the features of the `human218` feature set. The top 25 and bottom 25 ranked features are shown in Table (3.1).

We then ran an experiment using the standard kernel ridge regression (Equation 2.6) model from the previous chapter using just the top N ranked semantic features and compared this

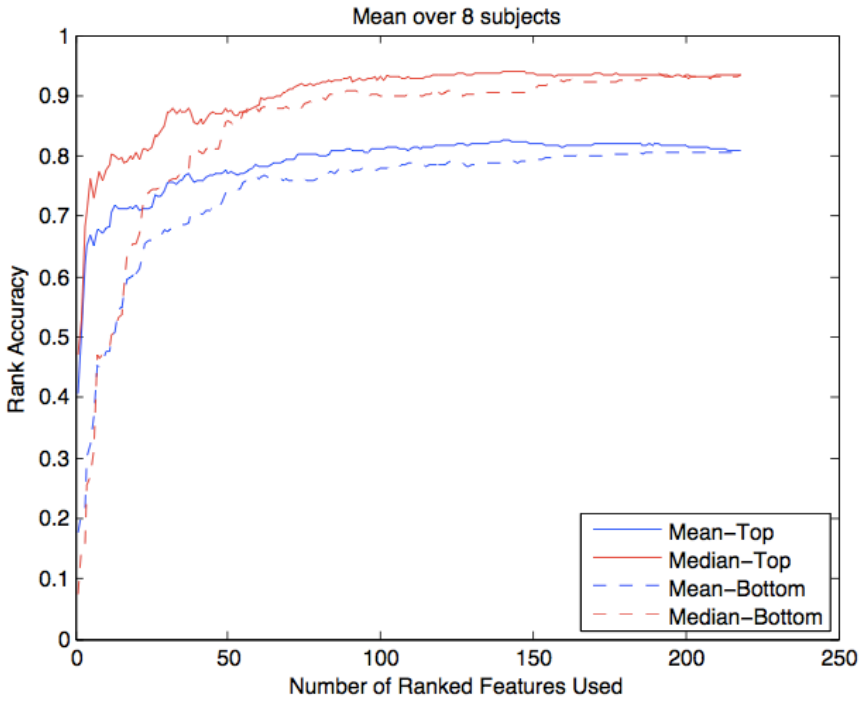


Figure 3.1: Rank accuracies for the top N weighted semantic features (solid line) and bottom N weighted semantic features (dashed line). The red lines are for the median rank accuracy while the blue lines are for the mean rank accuracy. Note that eventually the sets contain the same features which is why the solid and dashed red lines converge, and also why the solid and dashed blue lines converge. The semantic features were learned using support vector ranking on our most accurate fMRI subject. The accuracy results reported are then averaged over the remaining 8 fMRI subjects.

against the same model using the bottom N ranked semantic features. We repeated this experiment for all values of N, from 1 to 218 using the `human218` feature set and averaged the results for the remaining 8 fMRI subjects. The results are shown in Figure (3.1). Note that as N grows larger, the two sets converge to using the same semantic features.

From this graph, we see that *the choice of semantic features can make a dramatic impact on performance, particularly when small numbers of features are used*. However, the importance of choosing the best features decreases as the size of the feature set increases. This is an important consequence for the zero-shot learning model. While the peak accuracy used slightly more than half the features, the performance decrease was minimal as we added additional poorly ranked features. Therefore:

When selecting semantic feature sets for the semantic output code classifier using nearest neighbor, it is far more important to make sure useful features are included in the model, rather than prevent irrelevant features from being excluded.

This result is surprising, but useful from a practical standpoint. Unless semantic feature

analysis is desired, or absolute best performance is required, using a larger feature set without paying particular attention to importance or ranking of features may yield sufficient results.

We also ran an experiment in which we learned a vector of semantic feature weights during each cross-validation and applied these weights in the second stage of the classifier. When all semantic features were used, we found little improvement of the weighted method over the results in Figure (2.9). Consistent with our previous result, weighting semantic features seems to make an important impact only when the size of the semantic feature set is small.

As a result, the primary usefulness of this method is when there is a non-trivial cost to obtaining semantic features, such as when human labeling is required. A large number of semantic features could be collected for a small number of words then evaluated using this method. Only the most useful features selected from this set should then be collected for a much larger set of words.

3.4 Relationship to Other Feature Selection Methods

While the method we just described is useful in certain circumstances, the methodology is fairly different than other classical feature selection techniques such as mutual information or explained variance. For an excellent overview of these and other feature selection methods, see Guyon and Elisseeff [2003].

In certain scenarios, mutual information might be applied in a zero-shot learning framework, but this relies on reliably computing conditional probabilities of the semantic features given the class values. In our datasets, each class label had only one example in semantic feature space. As a result, it is not obvious how to obtain conditional probability distributions for a given class. If more examples of semantic features were collected for a given class, then it would be possible to estimate this distribution.

Another challenge of using mutual information is that our data is real-valued, and during our preprocessing step described in Chapter 2, the vector of semantic features for a given class is normalized to unit length. This normalization step is likely to lead to very non-Gaussian distributions for both the marginal and conditional probabilities of the semantic features. In our case it is non-obvious how to obtain these distributions. Note that this is a limitation of our particular datasets and normalization process, and is not a limitation of general zero-shot learning. In other datasets and learning scenarios, mutual information may select useful semantic features.

Other techniques for dealing with real-valued data can be used such as explained variance Guyon and Elisseeff [2003]. Selecting accurately predicted features, however, happens entirely in the first stage of our zero-shot learner and ignores the relative distribution of the semantic features. There is no guarantee that semantic features selected using explained variance will be useful for discriminating a particular set of classes. This is in contrast to the support vector ranking [Herbrich et al., 1999] method we described that relies on structured prediction [Ratliff et al., 2007] to select features for the task we really care about, which is predicting the class label. We suspect that any advantage of support vector ranking over explained variance would be most useful when feature sets are small, however, more experiments are required to confirm this conjecture.

3.5 Theoretical Considerations

We were surprised by the robustness of the model to irrelevant and less useful features, especially as the size of the semantic feature set increased. We asked ourselves if there might be a theoretical explanation for this behavior, and what insights theory might offer as to the number of semantic features to use in a semantic feature set.

In the implementation of our zero-shot learning algorithm, we used *1-nearest neighbor* as the second stage, $\mathcal{L}(\cdot)$, to look up the nearest class in the semantic feature set given a prediction of the features from the first stage $\mathcal{S}(\cdot)$. While other methods could be used for the second stage $\mathcal{L}(\cdot)$, we felt nearest neighbor was one of the simplest to implement, and therefore useful for an initial implementation of a zero-shot learner.

We now conjecture that the robustness we see to irrelevant features is a consequence of using the nearest neighbor algorithm for this second stage. As supporting evidence for this claim, we will once again consider similar analysis to that used in approximate nearest neighbor search from Ciaccia and Patella [2000].

Intuitively, points (classes) that are farther away from others in the semantic feature space are more tolerant to error. If points are densely packed together, then the nearest neighbor to a particular point is less likely to be the predicted point if there is any error in the prediction of the semantic features. See Figure 3.2. Thus, the tolerance to error depends on a specific point, and where that point lies relative to all the other points in the semantic space.

Recall in Section 2.2.4 we assumed the points in the semantic space were distributed according to a distribution $\mathcal{F}$. We we able to obtain samples from this distribution for a knowledge base $\mathcal{K}$, by sampling a class from an arbitrary distribution $\mathcal{Y}$ over classes and observing the output of semantic labeling function $\mathcal{L}^{-1}(\cdot)$. See Figure 2.2. Using these samples, we were able to compute the relative distribution with respect to a particular point q and knowledge base $\mathcal{K}$.

If we let $d(q, q')$ be the distance between the point of interest q and some other sample q' representing another class in our sampled knowledge base, we can define the *relative distribution* $R_{q,\mathcal{K}}$ for a point q and sampled knowledge base $\mathcal{K}$ as the probability that the distance from q to another point q' is less than some distance z:

$$R_{q,\mathcal{K}}(z) = \mathbb{P}\left(d(q, q') \leq z\right) \qquad \forall q' \in \mathcal{K}, q' \neq q$$

This empirical distribution is just the fraction of points in $\mathcal{K}$ that are less than some distance z away from q:

$$R_{q,\mathcal{K}}(z) = \mathbb{P}\left(d(q, q') \leq z\right) = \frac{\#\text{points } q' \neq q \text{ in } \mathcal{K} \text{ with distance } d(q, q') <= z}{M}$$

Without knowing the distribution of points $\mathcal{F}$ precisely we need to sample to obtain estimates for the relative distribution $R_{q,\mathcal{K}}$. However, if the distribution $\mathcal{F}$ is known, then under certain circumstances it may be possible to derive the relative distribution with respect to a point q for any arbitrary additional sample q' from $\mathcal{F}$:

$$R_q(z) = \mathbb{P}\left(d(q, q') \leq z\right)$$

We can also define a distribution on the distance to the nearest neighbor of q, defined as η_q:

$$G_q(z) = \mathbb{P}\left(\eta_q \leq z\right)$$

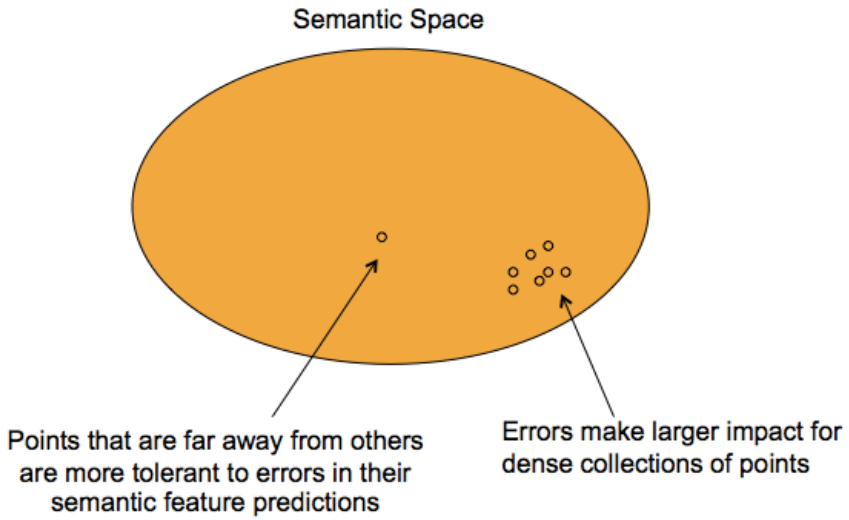


Figure 3.2: A visualization of a simple semantic space. Points (representing classes) that are densely packed together require greater accuracy when predicting individual semantic features, particularly when the nearest neighbor algorithm is used to recover the true class from the predicted semantic features.

We can estimate this distribution empirically using a sample of M points from $\mathcal{F}$ to form a knowledge base $\mathcal{K}$. This estimate is given in Ciaccia and Patella [2000] as:

$$\gamma = G_{q,\mathcal{K}}(z) = 1 - (1 - R_q(z))^M \tag{3.1}$$

This equation basically gives the probability γ that at least one point from the M samples, the nearest neighbor, falls within distance z of the point q. Using the inverse,

$$z = G^{-1}_{q,\mathcal{K}}(\gamma)$$

we can guarantee that with probability γ, the nearest neighbor from $\mathcal{K}$ will be no farther than z distance away from the point q. Recall that the point q represents the true class, and $\widehat{q}$ is the prediction of the semantic features for that class. Intuitively, z represents our error tolerance for q, since a large z will allow the prediction $\widehat{q}$ to be closer than the nearest neighbor to q in $\mathcal{K}$. Since $G_{q,\mathcal{K}}(\cdot)$ is a distribution function it monotonically increases from 0 to 1. However, there is no guarantee that a unique closed form inverse $G^{-1}_{q,\mathcal{K}}(\cdot)$ exists. However, under certain assumptions it is straightforward to compute such an inverse, and we will analyze one scenario as a means to study how the error tolerance changes depending on the number of semantic features used.

To simplify our analysis, we will assume that points lie in a semantic space that consists of the unit hypercube of p dimensions, with each point $q \in [0,1]^p$. The points are distributed in the semantic space according to a uniform distribution $\mathcal{F}$ and the distance metric $d(q,q')$ is ℓ_∞:

$$\ell_\infty(q,q') = \max_{k=1\ldots p} \left(|q_k - q'_k|\right)$$

which returns the max absolute distance for any particular dimension. Besides simplifying our analysis, this metric is useful when dimensions are not easily numerically comparable (e.g. different scales). Further, this metric is very general, as theoretical results show many nearest

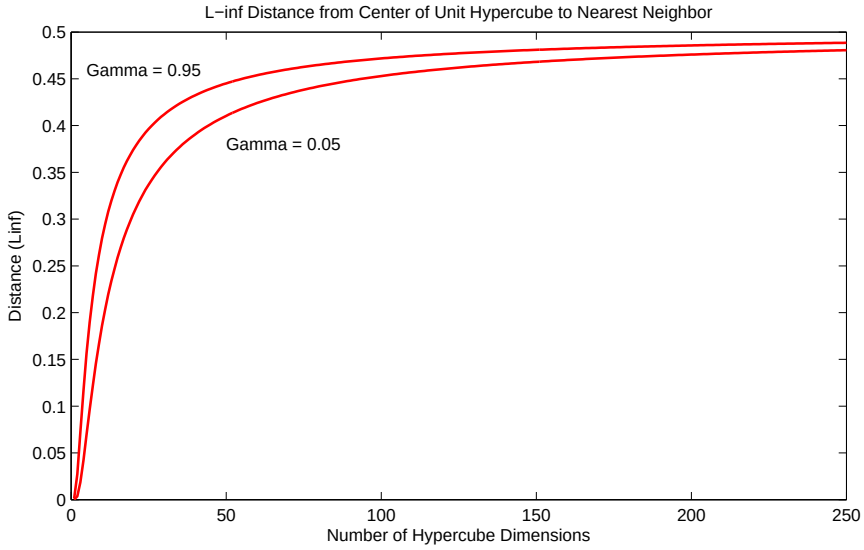


Figure 3.3: A visualization of Equation 3.2 that shows the effect on error tolerance as a function of the number of dimensions used in nearest neighbor classifier. The gamma bounds guarantees with the specified probability that the nearest neighbor is no farther than the indicated distance. As the number of dimensions increases, the error tolerance increases quickly and then the rate of increase declines quickly as more than 50 dimensions are used. In this equation, we use $M = 1000$, and show two probability bounds, $\gamma = 0.95$ and also $\gamma = 0.05$. We also assume the semantic feature points lie in a unit-hypercube with uniform distribution.

neighbor problems for various metrics can be transformed into a nearest neighbor problem with a ℓ_∞ metric [Andoni et al., 2008].

We also assume the point q is at the center of the hypercube: $q_{cen} \in (0.5, 0.5, \ldots, 0.5)$. For this point, the relative distribution can be computed directly:

$$R_{q_{cen}}(z) = (2z)^p$$

This is straightforward to see because the relative distribution is just the fraction of points (from a very large sample) that are no more than z distance from the center. Because of the ℓ_∞ metric, this means that all points no more than z distance away lie in the hypercube centered on q with each side of the hypercube being length $2z$. The volume of this smaller hypercube is just $(2z)^p$ since there are p dimensions. Given that points are distributed according to the uniform distribution, the relative distribution can be computed as the ratio of this volume to that of the overall hypercube volume which is just 1 because the we have assumed a unit hypercube:

$$R_{q_{cen}}(z) = \frac{(2z)^p}{1}$$

Substituting into Equation 3.1, we obtain the nearest neighbor distribution $G_{q_{cen},\mathcal{K}}$ for a sample of M points:

$$\gamma = G_{q_{cen},\mathcal{K}}(z) = 1 - (1 - (2z)^p)^M$$

which we will rearrange to obtain the inverse $G^{-1}_{q_{cen},\mathcal{K}}(\gamma)$:

$$\begin{aligned}
\gamma &= 1-(1-(2z)^p)^M \\
(1-(2z)^p)^M &= 1-\gamma \\
(1-(2z)^p) &= (1-\gamma)^{1/M} \\
1-(1-\gamma)^{1/M} &= (2z)^p \\
(1-(1-\gamma)^{1/M})^{1/p} &= (2z) \\
\frac{1}{2}(1-(1-\gamma)^{1/M})^{1/p} &= z = G^{-1}_{q_{cen},\mathcal{K}}(\gamma) \qquad (3.2)
\end{aligned}$$

The key insight from Equation 3.2 is that the inner term of the left-hand-side $(1-(1-\gamma)^{1/M})$ will always be less than one for $\gamma > 0$. As a result, *as the dimension p increases, the $1/p$ exponent causes the distance/error tolerance z to increase*. To visualize this effect, in Figure 3.3 we plot Equation 3.2 as a function of dimension for $p = 1 \ldots 250$, where $M = 1000$, $\gamma = 0.05$ and also $\gamma = 0.95$.

We see that as the number of dimensions increases, the distance from the nearest neighbor to the point q increases for the same probability bound. Intuitively, this means that the nearest neighbor can be farther away, while still guaranteeing with high probability that it is within this distance. Since this distance can be thought of as our error tolerance, we see that our tolerance increases very quickly as p approaches 50 dimensions, and then the rate of increase declines as more dimensions are added. Note that this increase in tolerance does not depend on the predictability of the individual semantic features (i.e. dimensions), but is merely a consequence of using additional dimensions (i.e. semantic features) in the zero-shot framework with nearest neighbor as the second stage $\mathcal{L}(\cdot)$ of the model.

In summary, if there are few semantic features, then the first stage $\mathcal{S}(\cdot)$ of our zero-shot learner must be very accurate in order to recover the true class since our error tolerance is very low. As the number of features (dimensions) increases, the tolerance to error increases as well, with the largest gains occurring when there are few dimensions used. Of course, adding a dimension may also increase the total error if the first stage of the zero-shot learner $\mathcal{S}(\cdot)$ is imperfect. Depending on the accuracy of $\mathcal{S}(\cdot)$, the error may grow faster or slower than the error tolerance allowed given the dimensionality. When a small number of dimensions are considered, the increase of error tolerance is largest. As a result, if a few very accurately predictable features are added, it is more likely that the tolerance to error will increase faster than the total error by the first stage $\mathcal{S}(\cdot)$, thereby resulting in an overall performance gain of the zero-shot leaner.

Even if an added feature is not predicted perfectly and additional error is accumulated, the error tolerance still increases as well. *This increase in error tolerance mitigates the additional accumulated error, resulting in robustness to poorly predicted features*. Eventually, the benefit of adding additional dimensions decreases, suggesting that the accumulated error will eventually grow faster than the error tolerance, resulting in an overall decrease in performance.

This suggests that for some error rate in the semantic feature predictions, adding features will initially lead to an increase in performance since error tolerance would increase faster than total error. This is then followed by little to no change in performance as the increase in total error is offset by the increase in error tolerance. Eventually we would expect to see a decrease in performance as error accumulates faster than the increase in error tolerance.

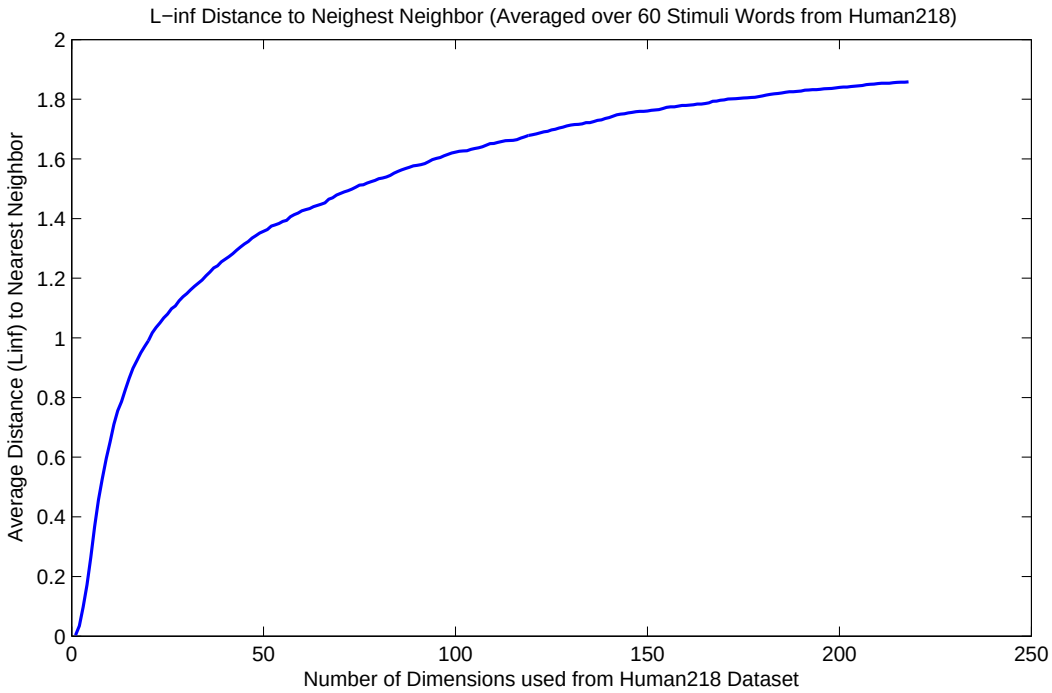


Figure 3.4: This figure shows the average ℓ_∞ distance between the 60 stimuli words and their nearest neighbor as a function of the number of dimensions used in the the semantic feature set. We computed this for sets ranging from 1 to 218 dimensions using the `human218` dataset. The plot shows the results averaged by repeating the experiment 25 times using random permutations of the features to eliminate any artifacts due to a specific feature ordering. Even using real data, we see the plot follows the same progression that is predicted by the theory in Figure 3.3. Specifically, we see the average distance increases quickly as dimensions are added, and then tapers off as more than 50 dimensions are used.

Interestingly, this expected behavior is very consistent with the observed performance of the zero-shot learner plotted in Figure 3.1). To test this conjecture further, we plotted the average error tolerance, i.e. the average distance to the nearest neighbor for the 60 stimuli words from the `human218` semantic feature dataset. We plotted this distance as a function of the number of dimensions used in the the semantic feature set, ranging from 1 to 218 dimensions. The plot shows the results averaged by repeating the experiment 25 times using random permutations of the features to eliminate any artifacts due to a specific feature ordering. Even using real data, we see the plot follows the same progression that is predicted by the theory in Figure 3.3. Specifically, we see the average distance increases quickly as dimensions are added, and then tapers off as more than 50 dimensions are used. This suggests that the number of semantic features used should not be small (e.g. less than 50), but it also suggests the number does not need to be very large either (e.g. more than a few hundred).

Although this theoretical model uses a distribution assumption and metric that are different from our empirical tests, the observed behavior is fairly consistent with what the theoretical analysis would expect. Although these distribution and metric differences prevent the analysis

from being entirely conclusive, we believe this analysis provides a likely explanation for the observed behavior. It may also serve as a useful framework for additional work in understanding the behavior of the zero-shot learning model that uses nearest neighbor.

3.6 Future Work

While we presented one method for selecting semantic features based on support vector ranking, there are potentially many other metrics that may be useful for the zero-shot learning framework. The algorithm we presented chooses features that are both predictable, but also useful for discriminating between the various classes. While certainly useful for increasing performance when using small semantic feature sets, one negative aspect of this method is that it makes model interpretation more difficult because semantic features that may be predictable might not be useful for discriminating between classes.

As a result, when features are left out of the model, it is not immediately clear whether it is because they are not predictable, or whether they are just not useful for discriminating between the classes. It would be interesting to know whether other methods that rely solely on choosing the most predictable semantic features would perform as well. We suspect that unless the algorithm uses a small number of semantic features (e.g. less than 50), including all features that are predicted well may lead to sufficient performance.

In addition, we suspect our current algorithm may not work as well if semantic features are very highly correlated. This is because the learned weights may be spread across highly correlated features, and simply ranking the features by learned weight is likely not appropriate. We performed additional experiments to address this concern by including sparsity constraints during learning, but we measured no improvement over the method we described. While this was not problematic for our particular dataset, if may be necessary to adjust the algorithm for datasets with highly correlated features.

Further, we showed that our zero-shot learning model based on nearest neighbor demonstrated significant robustness to irrelevant features. We presented a theoretical argument that claimed that this robustness was a consequence of nearest neighbor being used in the second stage of the zero-shot learning algorithm. While our analysis only considered uniform distributions of semantic features under the ℓ_∞ metric, it would also be useful to study this robustness property based on additional metrics and distributions. It would be also useful to know if other implementations of a zero-shot learner that do not use nearest neighbor also have this robustness property.

3.7 Conclusion

In this chapter, we presented a method based on support vector ranking that is useful for selecting semantic features for a zero-shot learning algorithm. This lowers the burden of determining the most useful semantic features *a priori*, and allows the algorithm to learn the most useful features from a much larger set. When applied to our neural imaging dataset, we found features related to *density and internal structure* as well as *shelter* qualities of the objects we considered.

Surprisingly, the importance of selecting semantic features decreased as the size of the original semantic set increased. Further, our empirical results suggest that a zero-shot learning algorithm that utilizes nearest neighbor is very robust to potentially irrelevant features. Thus, *we found it is far more important to make sure useful features are included in the model, rather than prevent irrelevant features from being excluded.*

We presented an initial theoretical analysis of using nearest neighbor in the zero-shot learning algorithm. This analysis showed under certain distribution and metric assumptions, the tolerance for error in the semantic features predictions can grow with the number of features. Our theory suggests that there is a benefit to using a semantic feature set that is sufficiently large (i.e. more than fifty semantic features), however we also found there may not be much benefit to using a semantic feature set that consists of more than few hundred semantic features.

3.8 Additional Chapter Remarks

Sample Code

An implementation of the SVR method described in this chapter can be downloaded from:
`http://www.thoughtrec.com`

Additional Acknowledgements

Many thanks to Drew Bagnell for discussions relating to the Support Vector Ranking method, and to Avrim Blum for discussions on the theoretical considerations of the zero-shot learning model. Also to Gustavo Sudre, Dean Pomerleau, Geoff Hinton, and Tom Mitchell, who were co-authors on a paper [Palatucci et al., 2011] that included several of the results presented in this chapter.

Chapter 4

Predicting Neural Activity using Multi-Task Learning

In the previous chapters, we described the zero-shot learning model as a means to classify images of neural activity. Given a novel neural image, this model predicts an intermediate set of semantic features, which are then used to classify the most likely concrete noun for that image.

In this chapter, we consider the inverse problem: given the semantic features for a novel concrete noun, we want a model that can make a prediction of the neural image. This model is *generative* in the sense that it produces the image we would expect to see if a person thought about a particular concrete noun. This problem was originally described in Mitchell et al. [2008] who use as semantic features a set of co-occurrence statistics from twenty-five sensory action verbs.

In a similar fashion to the previous chapter, an open question remains as to the best set of semantic features to use in the model. While the previous work of Mitchell et al. [2008] used twenty-five hand-selected features based on cognitive neuroscience assumptions, we would like an algorithm that can automatically choose the best semantic features from a much larger set. Although similar to the problem we addressed in the previous chapter using support vector ranking, we cannot use the same approach in this model because the semantic features are used as *inputs to the model* rather than *outputs from the model*.

However, we show in this chapter that regularized models that induce sparsity can be used to automatically select semantic features most useful to the model. In particular, we show that the problem of predicting a neural image from a large possible set of semantic features can be formulated as a multi-task Lasso problem that uses $\ell_{1,\infty}$ regularization constraints. This allows the algorithm to choose the set of semantic features that are most useful for predicting a neural image which represents the observed neural activity at hundreds of locations.

In collaboration with statisticians working to develop the first scalable algorithm to solve this objective, we presented the first case study of a large scale multi-task Lasso problem, and showed how we can automatically learn a useful set of semantic features that perform as well or better than the handcrafted set of features reported in Mitchell et al. [2008]. This allows selection of useful semantic features without requiring specific domain knowledge of cognitive neuroscience.

Chapter Notes for Neural Imaging Researchers

For the neural imaging researcher, the primary result is that multi-task learning models make useful tools for evaluating theories of neural representation. These tools have feature selecting qualities that assist in model interpretation and allow researchers to test models with fewer assumptions.

Whereas previous work suspected that semantic features based on co-occurrences statistics with verbs yielded the best semantic features [Mitchell et al., 2008], we formulated the problem using multi-task learning and allowed the model to select semantic features from co-occurrence statistics from thousands of words with different parts of speech. We found a set of semantic features that did not include any verbs, yet performed as well and often better than the verbs used in previous work. Other recent work has also confirmed than non-verbs (i.e. nouns) are useful as semantic features as well [Rustandi et al., 2009]. Further, we evaluated the model on specific *regions of interest* (ROIs), and show which semantic features are most useful for predicting neural activity in different brain regions.

Chapter Notes for Machine Learning Researchers

For the machine learning researcher, the algorithm for solving the multi-task Lasso will be of most interest. We refer the reader to an ICML paper that we co-authored with our statistician collaborators [Liu et al., 2009a] who derived the descent technique. The main technical result of this ICML paper (that we utilized in this chapter) is that it is possible to develop a cyclical blockwise coordinate descent algorithm for the multi-task Lasso ($\ell_{1,\infty}$ regularization norm) that efficiently solves problems with thousands of features and tasks. Specifically, the paper showed that a closed-form Winsorization operator can be obtained for the sup-norm penalized least squares regression. This allows the algorithm to find solutions to very large-scale multi-task feature learning problems. This result complements the pioneering work of Friedman et al. [2007b] for the single-task Lasso as well as other recently developed techniques for multi-task feature learning that use online learning [Bradley, 2010] as well as a greedy approach [Tropp et al., 2006, Kolar and Xing, 2010]. These recent techniques do not solve the multi-task Lasso objective $\ell_{1,\infty}$, but do provide alternative approaches to feature selection that are also highly scalable.

4.1 Introduction to the Neural Activity Prediction Problem

In this chapter, we present a case study of the multi-task Lasso by applying it to a problem in cognitive neuroscience. Specifically, we consider the task of predicting a person's neural activity in response to an arbitrary concrete noun in English as described in Mitchell et al. [2008]. Their approach is to predict the neural image that would be recorded using *functional magnetic resonance imaging* (fMRI) when a person thinks about a given word. In their work, the semantic meaning of a word is encoded by co-occurrence statistics with other words in a very large text corpus. They use a small number of training words to learn a linear model that maps these co-occurrence statistics to images of neural activity recorded while a person is thinking about those words. Their model can then predict images for new words that were not included in the training set and shows that the predicted images are similar to the observed images for those words. See

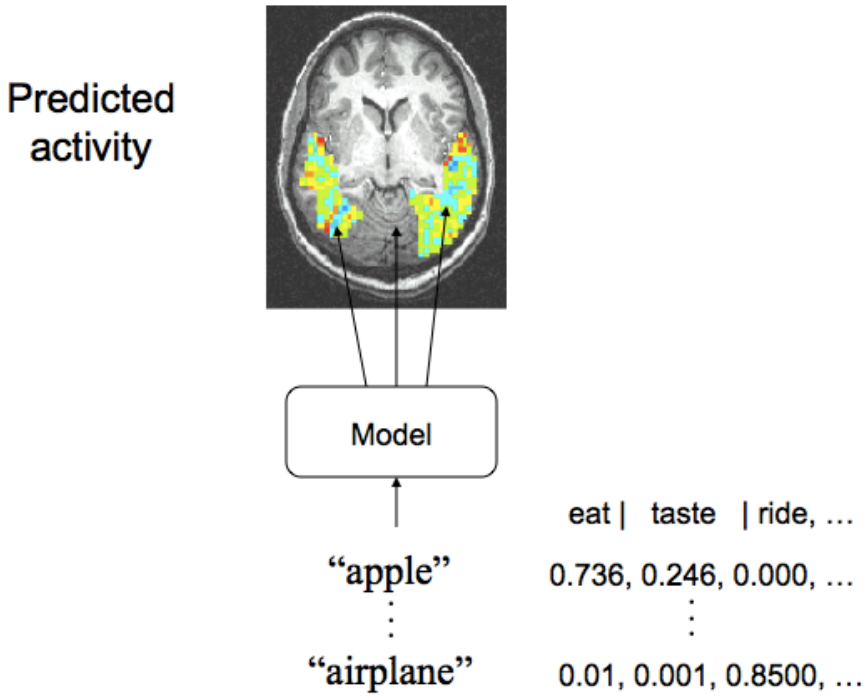


Figure 4.1: This image describes the neural prediction problem from Mitchell et al. [2008]. Words are represented as a vector of statistical co-occurrences with 25 sensory-action verbs such as `eat` and `ride` in a large text corpora. A model is trained to predict a neural image given the co-occurrence statistics for a given word. The image is the expected neural activity while a human is thinking about the particular word.

Figure 4.1.

In their initial model each word is encoded by a vector of co-occurrences with 25 sensory-action verbs (e.g. eat, ride, wear). For example, words related to foods such as "apples" and "oranges" would have frequent co-occurrences with the word "eat" but few co-occurrences with the word "wear". Conversely, words related to clothes such as "shirt" or "dress" would co-occur frequently with the word "wear" but not the word "eat". Thus "eat" and "wear" are example *basis words* used to encode relationships of a broad set of other words. These 25 sensory-motor verbs (shown in Table 4.1) were hand-crafted based on domain knowledge from the cognitive neuroscience literature and are considered a *semantic feature basis* of latent word meaning. A natural question is:

What is the optimal basis of words to represent semantic meaning across many concepts?

Rather than relying on models that require manual selection of a set of words, this work tries to build models that will perform *automatic variable selection* to learn a semantic basis of word meaning. In this way, we not only want to predict neural activity well, but also give insights into how the brain represents the meaning of different concepts. The hope is that learning directly from data will allow the researcher to test theories of cognition with fewer initial assumptions.

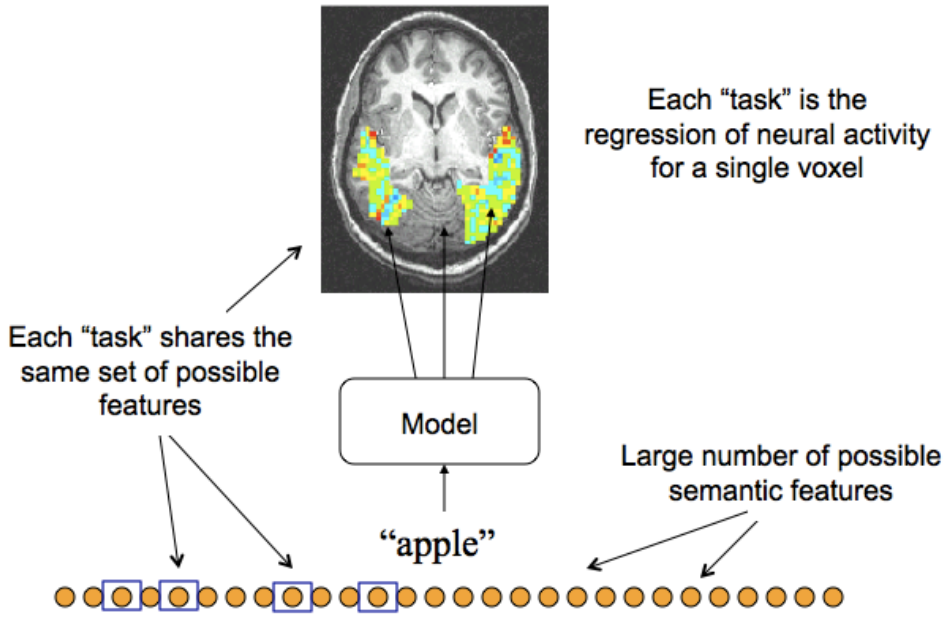


Figure 4.2: We formulate the neural image prediction problem as a multi-task learning problem. In this image, each voxel of neural activity is regressed onto a large set of possible semantic features for the word `apple`. The multi-task learning model learns all tasks simultaneously, and selects a common subset of features that are useful *as a basis* for all the tasks. Note that each task will weight each feature as necessary to predict its output, and tasks may have different weights on the same feature.

One popular approach to variable selection is to use models such as the Lasso that rely on ℓ_1 regularization constraints [Tibshirani, 1996]. Given a large number of inputs, the Lasso is able to select only the most relevant variables useful for predicting the output. We could apply such a model to the problem of neural activity prediction, but neural images contain hundreds and often thousands of voxels. Therefore, we must train many separate models, one for each specific voxel of neural activity in the image, leading to potentially very different sets of selected variables. This makes model interpretation more difficult, and may make the selected features highly variable to the number of training examples in the model [Turlach et al., 2005].

One approach to this problem is to use multi-task learning, which attempts to predict many outputs (tasks) simultaneously [Ando and Zhang, 2005, Tropp et al., 2006, Zhang, 2006, Obozinski et al., 2006, Argyriou et al., 2008, Bradley, 2010]. For the neural prediction problem, we want an algorithm that can select a single set of semantic features that are useful for predicting the neural activity for hundreds and potentially thousands of voxels simultaneously. See Figure 4.2.

One such model is the multi-task Lasso [Zhang, 2006] which generalizes the single-task ℓ_1 regularization constraint to the multi-task constraint based on the $\ell_{1,\infty}$ matrix norm. Like the single-task Lasso, the multi-task Lasso constraint encourages sparsity in the model. However, it attempts to learn a solution that is row-sparse, meaning that it picks a small number of input variables that are useful for predicting all the tasks. See Figure 4.3.

See	Eat	Run	Say	Enter
Hear	Touch	Push	Fear	Drive
Listen	Rub	Fill	Open	Wear
Taste	Approach	Move	Lift	Break
Smell	Manipulate	Ride	Near	Clean

Table 4.1: The semantic basis used in Mitchell et al. (2008)

Prior to the work of our statistician collaborators [Liu et al., 2009a] in 2008, there was no scalable method that could solve the multi-task Lasso $\ell_{1,\infty}$ for hundreds or thousands of variables and tasks. However, recent work on coordinate descent methods for the single-task Lasso have proved useful for the multi-task case as well. Our work complements other highly scalable methods for multi-task feature learning that do not utilize the multi-task Lasso $\ell_{1,\infty}$ norm [Kolar and Xing, 2010, Bradley, 2010]. The work of Kolar and Xing [2010] uses a greedy approach based on SOMP [Tropp et al., 2006], while Bradley [2010] uses an online gradient descent technique that can be used for many different matrix norms. However, their technique cannot solve the multi-task Lasso directly using the $\ell_{1,\infty}$ norm, but only an approximate version, (e.g $\ell_{1.2,5}$). Other recent work from Quattoni et al. [2009] addresses the $\ell_{1,\infty}$ norm using a projected gradient method.

To solve the multi-task Lasso which utilizes the $\ell_{1,\infty}$ norm, we'll use a blockwise coordinate descent method. The next three sections provide the technical background on coordinate descent and its application to the multi-task Lasso solution.

4.2 Background on Coordinate Descent Optimization

The cyclical coordinate descent algorithm was proposed to solve the ℓ_1-regularized least squares regression (i.e. the Lasso) almost ten years ago [Fu, 1998], but not until recently was this technique's power fully appreciated [Friedman et al., 2007b, Wu and Lange, 2008]. In particular, Friedman et al. [2007b] show that the coordinate descent method, if implemented with certain computation techniques, can be used to evaluate the entire regularization path of the Lasso remarkably faster than almost all of the existing state-of-the-art methods. The main reasons for such a surprising performance of the coordinate descent algorithm can be summarized as: (i) During each iteration, the coordinate-wise update can be written as a closed-form soft-thresholding operator, thus an inner loop is avoided; (ii) If the underlying feature vector is very sparse, the soft-thresholding operator can very efficiently detect the zeros by a simple check, thus only a small number of updates are needed. (iii) Many computational techniques, like the *covariance update* or *warm start*, can be easily incorporated into the coordinate descent procedure for a significant speedup [Friedman et al., 2008].

In this chapter, we describe an algorithm for solving of the multi-task Lasso [Zhang, 2006], which generalizes the Lasso to the multi-task setting by replacing the ℓ_1-norm regularization with the sum of sup-norm $\ell_{1,\infty}$ regularization. The approach to solving the multi-task Lasso parallels

the ideas in Friedman et al. [2007b] and results in a scalable, cyclical blockwise coordinate descent algorithm that can evaluate the entire regularization path efficiently. The key to solving the multi-task Lasso using coordinate descent is that the main sub-problem within each descent iteration can be very efficiently solved by a *Winsorization* operator[1], i.e. a proportion of the extreme values of the given vector are truncated while the others remain unchanged. This extends the results of Friedman et al. [2007b] to the multi-task setting. This extension as described in [Liu et al., 2009a] was non-trivial, since the results in Friedman et al. [2007b] utilize the fact that the ℓ_1-norm decouples into a sum of individual absolute values which are easy to handle. In contrast, the sup-norm couples the whole vector together and the non-differentiability conditions become more complicated.

4.3 The Multi-task Lasso

In this section, we introduce the multi-task Lasso and some related work. We start by describing some notation. Consider a K-task linear regression, for $k = 1, \ldots, K$

$$Y^{(k)} = \sum_{j=1}^{p} \beta_j^{(k)} X_j^{(k)} + \epsilon^{(k)} \text{ where } Y^{(k)}, X_j^{(k)}, \epsilon^{(k)} \in \mathbf{R}^{n_k}$$

where the superscript k indexes the tasks, p is the number of features, and n_k is the number of instances within the k-th task. We assume the data is standardized so the constant terms can be dropped, i.e. $X_j^{(k)}$ and $Y^{(k)}$ have mean 0 and $\|X_j^{(k)}\|_2 = 1$ where $\|\cdot\|_2$ is the ℓ_2-Euclidean norm. Let

$$\beta_j \equiv (\beta_j^{(1)}, \ldots, \beta_j^{(K)})^T \tag{4.1}$$

be the vector of all coefficients for the j^{th} feature across different tasks, the multi-task Lasso estimate is formulated as the solution to the optimization problem

$$\min_{\beta} \left\{ \frac{1}{2} \sum_{k=1}^{K} \|Y^{(k)} - \sum_{j=1}^{p} \beta_j^{(k)} X_j^{(k)}\|_2^2 + \lambda \sum_{j=1}^{p} \|\beta_j\|_\infty \right\}, \tag{4.2}$$

where $\|\beta_j\|_\infty \equiv \max_k |\beta_j^{(k)}|$ is the sup-norm in the Euclidean space. This sum of sup-norm regularization has the effect of "grouping" the elements in β_j such that they can achieve zeros simultaneously. If all the tasks share a common design matrix, the multi-task regression reduces to a multivariate-response regression. In this case, Turlach et al. [2005] proposes the same sum of sup-norm regularization and names the resulting objective in (4.2) as the simultaneous Lasso. It is obvious that any solver for the multi-task Lasso can also solve the simultaneous Lasso. Existing methods to solve (4.2) from the machine learning and statistics communities include the double coordinate descent method from Zhang [2006], the interior-point method from Turlach et al. [2005], and the geometric solution path method from Zhao et al. [2009]. These methods, however, have difficulty scaling to thousands of features and tasks. It is worth noting the online

[1] After Charles P. Winsor, [Mallows, 1990]

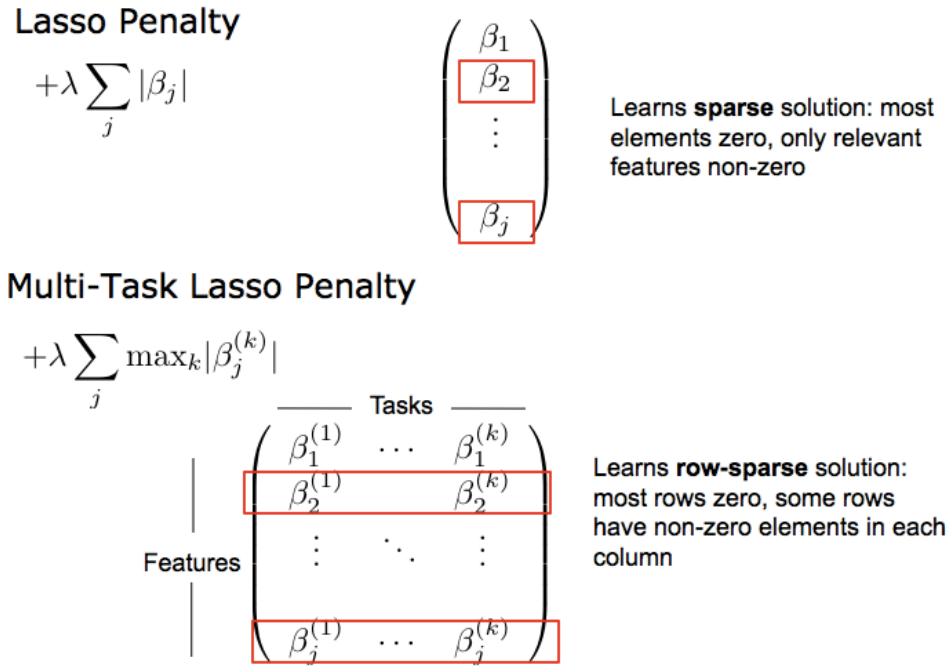


Figure 4.3: (Top) The regularization constraint penalty for the single-task Lasso and its effect on the solution vector. Only the most relevant variables have non-zero coefficients in the solution vector. (Bottom) The regularization constraint penalty for the multi-task Lasso and its effect on the solution matrix. Like the single-task Lasso, the multi-task Lasso encourages sparsity in the model. However, it attempts to learn a solution that is row-sparse, meaning that it picks a small number of input variables that are useful for predicting all the tasks.

gradient descent technique from Bradley [2010] can efficiently an approximation to the $\ell_{1,\infty}$ norm such as $\ell_{1.2,5}$ although it cannot solve the exact $\ell_{1,\infty}$ norm because of a differentiability requirement of their current algorithm. They suggest that a sub-gradient approach [Shor, 1985] might address this problem.

4.4 Blockwise Coordinate Descent

For a fixed regularization parameter λ, the blockwise coordinate descent algorithm for the multi-task Lasso problem in Equation (4.2) is given in Figure 4.4 where $\langle \cdot, \cdot \rangle$ denotes the inner product operator of two vectors. Recall that β_j in (4.1) represents the coefficient vector of the j-th feature across all the K tasks. We call β_j a feature block. The algorithm consists of simultaneously updating the coefficients within each feature block while holding all the others fixed, then cycling through this process. Therefore, if the current estimates are $\widehat{\beta}_\ell$, $\ell = 1, \ldots, p$, then β_j is updated by the following sub-problem:

$$\widehat{\beta}_j = \arg\min_{\beta_j} \left\{ \frac{1}{2} \sum_{k=1}^{K} \left\| R_j^{(k)} - \beta_j^{(k)} X_j^{(k)} \right\|_2^2 + \lambda \|\beta_j\|_\infty \right\} \tag{4.3}$$

BLOCKWISE COORDINATE DESCENT ALGORITHM

Input: Data $(X_1^{(k)}, \ldots, X_p^{(k)}, Y^{(k)}), k = 1, \ldots, K$ and the regularization parameter λ;

Iterate over $k \in \{1, \ldots, K\}$ and $j \in \{1, \ldots, p\}$

(1) $c_j^{(k)} \leftarrow \langle Y^{(k)}, X_j^{(k)} \rangle$;

(2) $\beta_j^{(k)} \leftarrow$ initial values (either 0 or $\widehat{\beta}_j^{(k)}$ for the previous λ if doing a warm-start);

(3) For each $i \in \{1, \ldots, p\}$: $d_{ij}^{(k)} \leftarrow \langle X_i^{(k)}, X_j^{(k)} \rangle$;

End;

Iterate until convergence (Main Loop):

For each feature $j \in \{1, \ldots, p\}$:

(1) $\forall$ tasks $k \in \{1, \ldots, K\}$, $\alpha_j^{(k)} \leftarrow c_j^{(k)} - \sum_{i \neq j} \beta_i^{(k)} d_{ij}^{(k)}$;

(2) If $\sum_{k=1}^{K} |\alpha_j^{(k)}| \leq \lambda$ Then $\beta_j \leftarrow \mathbf{0}$ Else

(a) Sort the indices according to $|\alpha_j^{(k_1)}| \geq |\alpha_j^{(k_2)}| \geq \ldots \geq |\alpha_j^{(k_K)}|$;

(b) $m^* \leftarrow \arg\max_{1 \leq m \leq K} \left(\sum_{i=1}^{m} |\alpha_j^{(k_i)}| - \lambda \right) / m$;

(c) For each $i \in \{1, \ldots, K\}$

If $i > m^*$ Then $\beta_j^{(k_i)} \leftarrow \alpha_j^{(k_i)}$ Else

$$\beta_j^{(k_i)} \leftarrow \frac{\operatorname{sign}(\alpha_j^{(k_i)})}{m^*} \left[\sum_{\ell=1}^{m^*} |\alpha_j^{(k_\ell)}| - \lambda \right] ;$$

End;

Output: $\widehat{\beta}_j^{(k)} \leftarrow \beta_j^{(k)}$ for $j = 1, \ldots, p$ and $k = 1, \ldots, K$;

Figure 4.4: The algorithm for the multi-task Lasso.

where $R_j^{(k)} \equiv Y^{(k)} - \sum_{\ell \neq j} \widehat{\beta}_\ell^{(k)} X_\ell^{(k)}$ denotes the partial residual vector. If the regularization parameter $\lambda = 0$, the problem in (4.3) decouples and the tasks become independent of each other, allowing the updates for the coefficients of a given task to be solved independently of the other tasks. We denote $\alpha_j^{(k)}$ as the unpenalized least squares coefficient for feature j and task k after the update:

$$\alpha_j^{(k)} = \langle R_j^{(k)}, X_j^{(k)} \rangle, \quad \text{for } \forall j, k. \tag{4.4}$$

In fact, since

$$\langle R_j^{(k)}, X_j^{(k)} \rangle = \langle Y^{(k)}, X_j^{(k)} \rangle - \sum_{\ell \neq j} \beta_\ell^{(k)} \langle X_\ell^{(k)}, X_j^{(k)} \rangle,$$

we can pre-calculate the quantities $c_j^{(k)} = \langle Y^{(k)}, X_j^{(k)} \rangle$ and $d_{ij}^{(k)} = \langle X_i^{(k)}, X_j^{(k)} \rangle$. This is the same *covariance update* idea as in Friedman et al. [2008] and corresponds to the first double loop in the algorithm described in Figure 4.4. If we have a decreasing sequence of the regularization parameters λ's, the initial values of $\beta_j^{(k)}$ for each fixed λ comes from the solutions $\widehat{\beta}_j^{(k)}$ calculated from the previous λ value. This is the same *warm start* technique as in Friedman et al. [2008] and can significantly speedup the algorithm's performance when evaluating the entire solution path. Since the quantities $c_j^{(k)}$ and $d_{ij}^{(k)}$ do not depend on the regularization parameter λ, they only need to be calculated once and serve for the whole pathwise evaluation.

For $\lambda > 0$, (4.3) becomes a sup-norm penalized least squares regression. If we use Newton's method or a coordinate descent procedure to solve it [Zhang, 2006], an inner loop will be needed. This is not scalable if the number of tasks K is very large. Fortunately, Theorem 1 below shows that the solution to (4.3) is equivalent to a closed-form Winsorization operation applied on the previously calculated unpenalized least squares results $\alpha_j^{(k)}$'s. This corresponds to the main loop of the algorithm in Figure 4.4.

Theorem 1. *Let $\alpha_j^{(k)}$ be defined as in (4.4) and order the indices according to $|\alpha_j^{(k_1)}| \geq |\alpha_j^{(k_2)}| \geq \ldots \geq |\alpha_j^{(k_K)}|$. Then the solution to (4.3) is*

$$\widehat{\beta}_j^{(k_i)} = \frac{\operatorname{sign}(\alpha_j^{(k_i)})}{m^*} \left[\sum_{i'=1}^{m^*} |\alpha_j^{(k_{i'})}| - \lambda \right]_+ \cdot \mathbf{1}_{\{i \leq m^*\}} + \alpha_j^{(k_i)} \cdot \mathbf{1}_{\{i > m^*\}}$$

where $m^* = \arg\max_m \frac{1}{m} \left(\sum_{i'=1}^{m} |\alpha_j^{(k_{i'})}| - \lambda \right)$, $\mathbf{1}_{\{\cdot\}}$ is the indicator function, and $[\cdot]_+$ denotes the positive part.

In Appendix B, we provide Han Liu's proof of this theorem, along with convergence guarantees and complexity analysis of the algorithm.

4.5 Case Study: Neural Activity Prediction

4.5.1 Datasets

For our empirical study of neural activity prediction, we utilize the two datasets described in Mitchell et al. [2008]. The first dataset was collected using fMRI. Nine participants were pre-

sented with 60 different words and were asked to think about each word for several seconds while their neural activities were recorded. The 60 words are composed of nouns from 12 categories with 5 exemplars per category. For example, a *bodypart* category includes `Arm, Eye, Foot, Hand, Leg`, a *tools* category includes the words `Chisel, Hammer, Pliers, Saw, Screwdriver`, and a *furniture* category includes `Bed, Chair, Dresser, Desk, Table`, etc.

The second dataset is a symmetric matrix of text co-occurrences between the 50,000 most frequent words in English. These co-occurrences are derived from the Google Trillion Word Corpus[2]. The goal is to use these co-occurrences to construct a feature vector that represents word meaning. The hope is that two words that cause similar neural activites would also have high inner product between their co-occurrence vectors. One simple representation of each word is to use the raw 50,000 dimensional feature vector of co-occurrences (normalized to unit length row norm). Typically a much smaller representation is desired such as the hand-crafted 25-word basis described above.

For each participant, there are altogether $n = 60$ fMRI images taken[3], each one corresponds to one stimulus word. A typical fMRI image contains over $K = 20,000$ different neural responses. Each neural response is the activity in a *voxel* (volume-element) in the brain. Each voxel represents a 3x3x6 mm^3 volume in the brain and is the basic spatial unit of measurement in this fMRI data. Here we show the problem of learning a semantic basis can be formulated into a multi-task Lasso as in (4.2). Since the goal of learning a semantic basis is to find a common set of predictor variables that will predict the neural response well across multiple voxels, where each predictor variable is the text co-occurrences with a particular word from the Google Trillion Word Corpus. Therefore, for each participant, we have roughly $K = 20,000$ tasks, all these tasks share the common design columns $\{X_j\}_{j=1}^{p} \in \mathbf{R}^n$, representing the co-occurrences of $n = 60$ training words with $p = 50,000$ other English words in the Google Corpus. The response vector $Y^{(k)}$ for each task contains the neural activations for a single voxel (volume-element) across the $n = 60$ fMRI images. Therefore, this is a multi-task learning problem with a very large number of tasks $K = 20,000$ and a very large number of features $p = 50,000$. While the algorithm we develop can solve a problem of this scale in only a few minutes, our primary results use a smaller dataset where $p = 5,000$ and $K = 500$, so that our experiments are directly comparable with other published results. We also provide additional results where $K = 4500$.

4.5.2 Experimental Results

To evaluate our methods and compare them to existing results, we use exactly the same experimental protocols described in Mitchell et al. [2008]. As a small additional step we use the multi-task Lasso to perform a variable selection. Note that the multi-task Lasso is used in this context only to learn *which* variables should be input into the final model. Specifically, the *leave-two-out-cross-validation* procedure is as follows:

We repeat this experiment for each of the nine different participants in the fMRI study and use the same method in Mitchell et al. [2008] to ensure consistency while testing various semantic

[2]http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html

[3] Each image is actually the average of 6 different recordings of each word.

a Create a $60 \times 5,000$ design matrix of semantic features using co-occurrences of the 60 experimental stimuli with the 5,000 most frequent words in English (minus 100 stop words). A stop word is a high frequency common word like "the".

b Select 2 stimuli out of 60 for testing and use the other 58 stimuli for training. Using (4.2), learn the coefficients β by setting each X_j to be the 58×1 vector of co-occurences for each of the 5000 basis words and each $Y^{(k)}$ to be the 58×1 column vector for each of the top $K = 500$ voxel responses selected using the stability criterion score described in Mitchell et al. [2008] . In the language of multi-task Lasso, this problem corresponds to the scale $n = 58, p = 5000, K = 500$. The regularization parameter here can be set to choose the desired number of non-zero coefficients. We set this parameter to yield basis sets with 25, 140, and 350 elements.

c Create a new matrix of semantic features that is $58 \times d$, where d is the number of selected feature blocks in β. In our case d will be either 25, 140, or 350. Each non-zero block should correspond to a word selected from the original set of 5,000. This word shall now become a semantic feature in the new basis.

d Train a linear model using ridge regularization to predict each of the 500 voxels from the semantic feature basis. The solution can be obtained using the standard normal equations for ridge regression. Note that we only use the multi-task Lasso to select features as we want to compare the automatically selected features directly against the handcrafted features from Mitchell et al. [2008]. We want to know the performance of the Mitchell et al. [2008] model using these automatically semantic features.

e For each of the two test examples, predict the neural response of the 500 selected voxels. Compute the cosine similarity of each prediction with each of the held out images. Based on the combined similarity scores, choose which prediction goes with each held out image. Test if the joint labeling was correct. This leads to an output of 0 or 1. For more details, see Mitchell et al. [2008].

f Repeat steps b-e for all $\binom{60}{2}$ possible pairs of words (1,770 total). Count the number of incorrect labelings in step e to determine the accuracy of the basis set.

Figure 4.5: The leave-two-out-cross-validation protocols

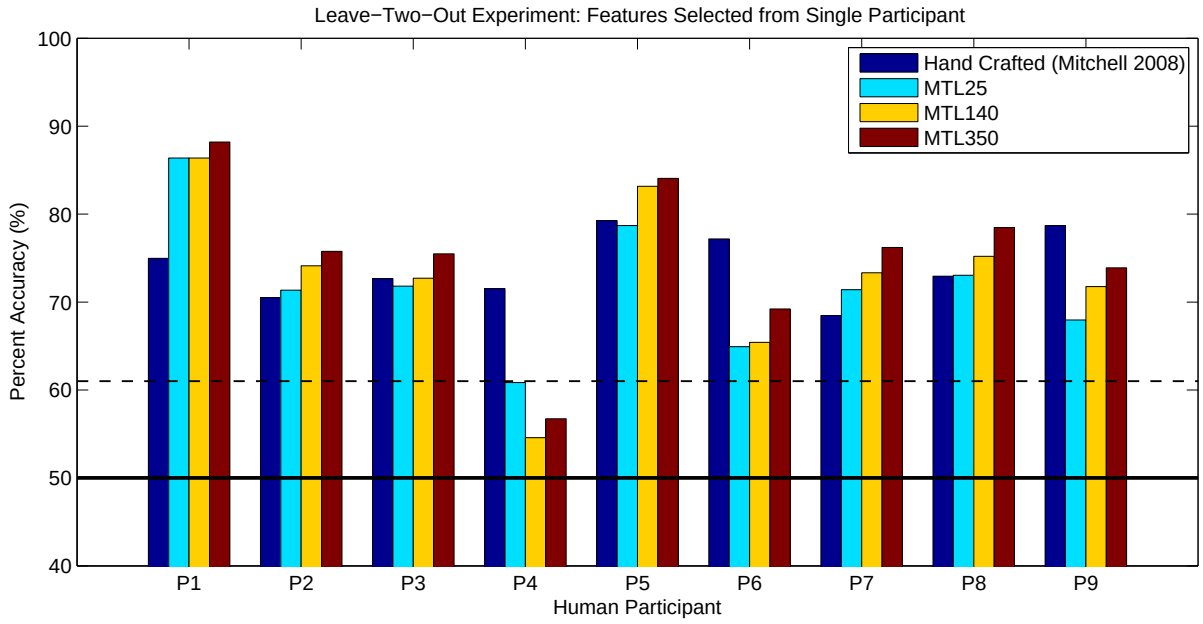


Figure 4.6: Accuracies for 9 fMRI Participants. Features were selected *per-subject*. The chance accuracy level is shown as a solid black line and the significant threshold (at the 5% level) is shown as the dotted black line. This significance threshold is computed in Mitchell et al. [2008] by computing the empirical distribution of accuracies by randomly choosing semantic features from the 500-5000 most frequent words and applying the same leave-two-out train and test procedure.

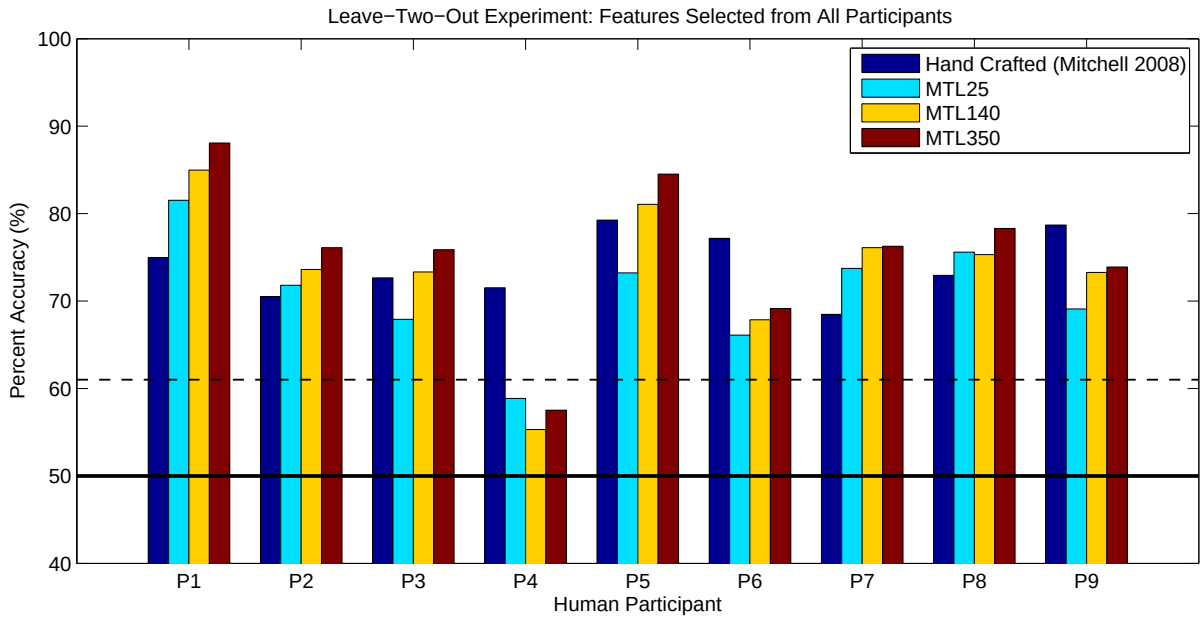


Figure 4.7: Accuracies for 9 fMRI Participants. Features were selected from *combined participant data*. The chance accuracy level is shown as a solid black line and the significant threshold (at the 5% level) is shown as the dotted black line. This significance threshold is computed in Mitchell et al. [2008] by computing the empirical distribution of accuracies by randomly choosing semantic features from the 500-5000 most frequent words and applying the same leave-two-out train and test procedure.

features. The regularization parameter for the ridge regression in step d is chosen automatically using the cross-validation error score described in Hastie et al. [2001, Page 216].

We performed several experiments using the above protocols to compare the performance of the semantic feature sets learned using the multi-task Lasso with the hand-crafted features described earlier. Using these results we now pose and answer several questions:

1. Can we automatically learn a semantic basis?

As in Figure 4.5, we use the multi-task Lasso to learn a semantic basis from the 5,000 most frequent words in English and adjust the regularization parameter to obtain different basis sizes. We denote MTL25, MTL140, MTL350 as models that have 25,140, and 350 words respectively in their basis sets each time they were trained. Note that we train the models on each iteration of the cross-validation and keep the regularization parameter the same between iterations. As a result, there can be a slight variation, typically 1-3 features, in the actual number of features selected on each iteration.

We see in Figure 4.6 the results of the 4 models. The result for the 25 features hand-crafted by cognitive neuroscientists is also reported. Chance accuracy for this prediction task is 0.5 and statistically significant accuracy at the 5% level is 0.621. This is computed in Mitchell et al. [2008] by computing the empirical distribution of accuracies by randomly choosing semantic features from the 500-5000 most frequent words and applying the same leave-two-out train and test procedure. We see that in 8 of 9 subjects, all three multi-task Lasso models learn a semantic basis that leads to statistically significant accuracies. This suggests that we can indeed learn a semantic basis directly from data.

The MTL140 and MTL350 models achieve higher accuracies than the hand-crafted features in 6 of 9 participants. It is exciting that the model can often meet or exceed the performance of the hand-crafted features *using far fewer assumptions about neuroscience*. It is also useful that our learned basis is different than the handcrafted features, which suggests potential benefits from a model averaging approach. For the MTL25 model, we report accuracies higher than the hand-crafted features in 4 of 9 participants. We also see that two larger basis sets outperform the MTL25 set, suggesting that more than 25 features are necessary to capture the variance of the data.

An interesting observation comes from participant 4. On this participant all three learned models performed worse than the hand-crafted model. The very abnormal behavior of the learned models on this participant versus the other participants suggests that this particular participant might be an outlier or a very noisy observation (as is common in fMRI studies). However, the hand-crafted feature does not suffer from this case. More investigation is suggested to study this.

2. Is there any advantage to learning the basis across multiple subjects simultaneously?

An interesting question is whether we could ameliorate this problem by learning the basis simultaneously across all subjects at once. Figure 4.7 shows the results of learning the basis by combining the fMRI data for all participants (thereby yielding a 58 x 4500) target matrix. This corresponds to a multi-task Lasso where $K = 4500$. On average, we found a slight advantage on the MTL140 and MTL350 models of about 1%, and slight disadvantage for the MT25 model

(also around 1%). Although encouraging, this is hardly conclusive evidence, and we feel this is an interesting direction for future work. In particular, it would be worth studying the impact of noisy data by removing an outlier such as participant 4.

3. What is the learned semantic basis?

Table 4.2 shows one example of 25 basis words learned using the MTL25. It is easy to see relationships between many of the words in the basis set and the 60 stimulus words described before. For example, the model learned `Tools` as a basis word, which is interesting because 5 different instances of tools were shown (e.g. `Screwdriver, Hammer`, etc.). The basis word `Bedroom` clearly refers to words in the furniture category (`Bed, Dresser,` etc.) and `Arms` is related to body parts (`Leg, Hand,` etc.).

Tools	Car	Dog	Wine	Pottery
Model	Station	Bedroom	Breakfast	Cup
Mad	Rentals	Fishing	Cake	Tip
Arms	Walk	Cleaning	Cheese	Gay
Right	White	Front	Contents	Result

Table 4.2: An example of 25 learned semantic basis words.

For a given basis word, we can train a simple linear model to predict neural activations across all 20,000 voxels in the brain from this single basis word. Note that this is a multiple output regression and each learned coefficient corresponds to a physical location in the brain. By plotting the coefficients, we can discover how different basis words activate different regions of the brain. Figure 4.8 shows a 3D map of these coefficients for the basis word `Tools`. We see strong activation (red) in the superior temporal sulcus which is believed to be associated with the perception of biological motion [Vaina et al., 2001]. We also see strong activation in the postcentral gyrus which is believed to be associated with premotor planning [Pelphrey et al., 2005].

4.6 Visualizing Semantic Features for Individual Regions of Interest (ROIs)

In the previous section, we studied the ability of the multi-task Lasso to select semantic features from a large text corpus. We compared our results to those in Mitchell et al. [2008], who only considered semantic features that were sensory-action *verbs*. It was not previously known if non-verbs could also yield useful semantic features, but our work showed that the multi-task Lasso could automatically select *non-verb* semantic features that often performed as well or better.

Regardless, we believe that semantic features based on co-occurrence statistics collected from web corpora are difficult to interpret, primarily because of the overloaded meaning of individual words. For example, the word *bee* (the insect) most frequency occurs with the word *gees*, obviously because of the rock band *The Bee Gees*. As another example, we found the word *gay*

appeared in our semantic feature set MTL25 described in the previous section. We found this surprising, until we realized that the word *gay* has extremely high co-occurrence with the word *bear*, which was one of our fMRI stimulus words. Clearly, context matters, and there are many unintended consequences of using co-occurrence statistics based on world-wide-web data.

This limitation of co-occurrence data was the primary motivation for collecting the crowd-sourced dataset `human218` described in Chapter 2 from Mechanical Turk. When collecting this dataset, we explicitly specified the context of words with multiple meanings, and selected attributes that were more easily interpretable such as *is it made of metal?* and *is it manmade?*.

Using this cleaner dataset of semantic features, we trained the multi-task Lasso model to predict neural activities across individual *regions of interest* (ROIs). These ROIs represent geographic locations in the brain often tied to a particular function. Often, observed neural activity is very similar within an individual region. This suggests that predicting neural activity for a particular region involves modeling several related tasks, where each task is the prediction of the activity for a particular voxel.

This allowed us to train the multi-task Lasso model and ask what semantic features, selected from the `human218` feature set, were most useful for predicting the neural activity in a particular region. To do this, we first isolated the voxels in a particular region, and then trained the model using all available examples. We then ranked the semantic features based on the magnitude of their learned weights.

To visualize the results, we created a graphical interface tool (MTL-GUI) which plots the ROI voxels in 3D, along with the ten highest weighted features for a model trained on that region. Figures 4.9-4.14 show resultant plots for 6 regions of interest that include front, sides, and rear portions of the brain. The MTL-GUI tool, along with plots for all twenty-nine regions of interest are available at: `http://www.thoughtrec.com`.

In several regions, we see semantic features that appear consistent with commonly known neural theories. For example, we found several of the selected features near the motor cortex Left-Post-Central (Figure 4.10) are related to manipulation: *can you hold it? can you pick it up? is it a tool?*. In the region Right-Medial-Frontal (Figure 4.9), we found features relating to danger assessment, *can it cause you pain? is it pointed/sharp? is it dangerous?*.

However, in both these and other regions, it is often unclear why certain semantic features were selected. It is not known if this is the result of unknown confounding variables, a limitation of the model, or perhaps it provides evidence towards a new neural theory. As a result, we doubt the primary use of such a tool would be to generate novel theories of neural function. However, we feel this model would be useful for evaluating competing theories, using its feature selecting qualities of to make predictions with fewer initial assumptions, thereby limiting the potential for implicit overfitting.

4.7 Discussion and Future Work

From a cognitive neuroscience perspective, we believe that learning methods that include sparsity constraints provide useful tools for neuroscientists to test and evaluate theories of cognition. By using fewer assumptions *a priori* and allowing the models to choose the most useful features automatically, sparse learning methods reduce the risk of implicit overfitting, and may lead to

more believable results. For certain problems, we believe the joint task constraints will lead to more interpretable models than those that do not combine tasks, but more work needs to be done to compare models learned from single task data to those that combine data from multiple tasks. The recent work of Kolar and Xing [2010] is one step in this direction.

From a machine learning perspective, more multi-task learning methods should be evaluated. Since the original publication of our results in 2009, several other procedures have been published relating to multi-task learning using multi-task Lasso $\ell_{1,\infty}$ regularization norms [Quattoni et al., 2009] as well as $\ell_{1,2}$ norms [Vogt and Roth, 2010]. The work from [Bradley, 2010] provides a general technique for efficient optimization of different matrix norms in an online setting, however their approach can only approximately solve problems using the $\ell_{1,\infty}$ norm because of differentiability requirements of their current technique.

Little explicit comparison has been done between the different norms, except for Vogt and Roth [2010] and Bradley [2010] who empirically found that norms close to $\ell_{1,2}$ led to better predictors on several datasets. The work from Bradley [2010] goes further, and also includes a theoretical regret analysis for the online setting and claims that it is undesirable to use $\ell_{1,p}$ norms with $p > 2$ because of high regret, particularly when tasks are not highly related and feature weights vary across tasks. The benefit and risk of $\ell_{1,\infty}$ has also been analyzed recently by Negahban and Wainwright [2011], which confirms the benefit of this norm requires high relatedness of tasks.

Despite these works, it is not entirely clear from the literature under what conditions each norm should be used. In the single-task case, there are subtleties that depend on correlation of features, as well as whether true sparsity is required [Wainwright, 2009]. For example, many of these theoretical results guaranteeing recovery of the support set (i.e. the true model features) are possible using the Lasso ℓ_1 norm, but not the ridge regression ℓ_2 norm.

We also expect similar subtleties in the multi-task case, and expect that correlation of features, the number of examples, sparsity rates, as well as task relatedness will affect the appropriate choice of multi-task norm. From a computational standpoint, the $\ell_{1,2}$ regularization norm appears easier to solve at large scale given the current literature, and existing empirical and theoretical work supports using the $\ell_{1,2}$ as the first approach method as well, especially if all the tasks are not highly related as is likely in an empirical context. Consistent with the results of Bradley [2010], Negahban and Wainwright [2011], we believe that the $\ell_{1,2}$ norm will be more robust to violations of the task-relatedness assumption. In fact, in our own work we noticed that the $\ell_{1,\infty}$ norm tended to squash the learned coefficients between tasks to similar values. This might be a benefit if the tasks are truly similar. However, we suspect that the $\ell_{1,2}$ norm would allow more variation of a particular feature's coefficients across tasks, a useful benefit when the tasks are not very similar. Alternatively, $\ell_{1,\infty}$ norm might be used only for feature selection, while another model with less constraints could use the selected features to make the final prediction. In fact, this is the procedure we used in our experimental section where we used the multi-task Lasso to select the features, and a ridge regression to learn the final predictive weights. We suspect this approach would lead to better performance than a strict $\ell_{1,\infty}$ model. However, additional work, both theoretical and empirical, would be very useful to confirm this conjecture and also to fully understand the subtle effects of multi-task norms.

4.8 Conclusion

In this chapter, we studied the inverse of the zero-shot learning model. Specifically, we built a model to predict brain activity from the semantic features, instead of predicting semantic features from brain activity. This model is generative in the sense that given the semantic features for a novel word, it can often predict the brain activity that would be observed in fMRI images collected from a human subject thinking about the word.

Similar to the zero-shot learning model, we had the problem of choosing the best semantic features to input into the model. Although the previous Support Vector Ranking approach described in Chapter 3 does not apply to the generative model, we demonstrated an alternative method that can be used when predicting brain activity from semantic features. Specifically, we showed how to formulate this semantic feature selection problem using the multi-task Lasso objective function. In collaboration with statisticians working to develop the first scalable algorithm to solve this objective, we presented the first case study of a large scale multi-task Lasso problem, and showed how we can automatically learn a useful set of semantic features that perform as well or better than the handcrafted set of features reported in Mitchell et al. [2008]. Further, we applied the multi-task Lasso to different regions of interest (ROIs) of the brain, and reported the set of semantic features that are most useful for predicting activity in that region.

We believe that learning methods that include sparsity constraints provide useful tools for neuroscientists to test and evaluate theories of cognition. By using fewer assumptions *a priori* and allowing the models to choose the most useful features automatically, sparse learning methods reduce the risk of implicit overfitting, and may lead to more believable results.

4.9 Additional Chapter Remarks

Sample Code

We provide a MATLAB implementation of the blockwise descent procedure for solving the multi-task Lasso, as well the graphical user interface tool (MTL-GUI) for studying regions of interest at:
`http://www.thoughtrec.com`

Additional Acknowledgements

Thanks to Geoff Gordon and Larry Wasserman for discussions relating to the convex formulation of the multi-task learning problem. Also to Han Liu and Jian Zhang who were co-authors on a ICML paper [Liu et al., 2009a] that included many of the results presented in this chapter. Finally, many thanks to Andy Carlson for collecting the co-occurrence semantic features and to Dean Pomerleau for collecting the human labeled features using Mechanical Turk.

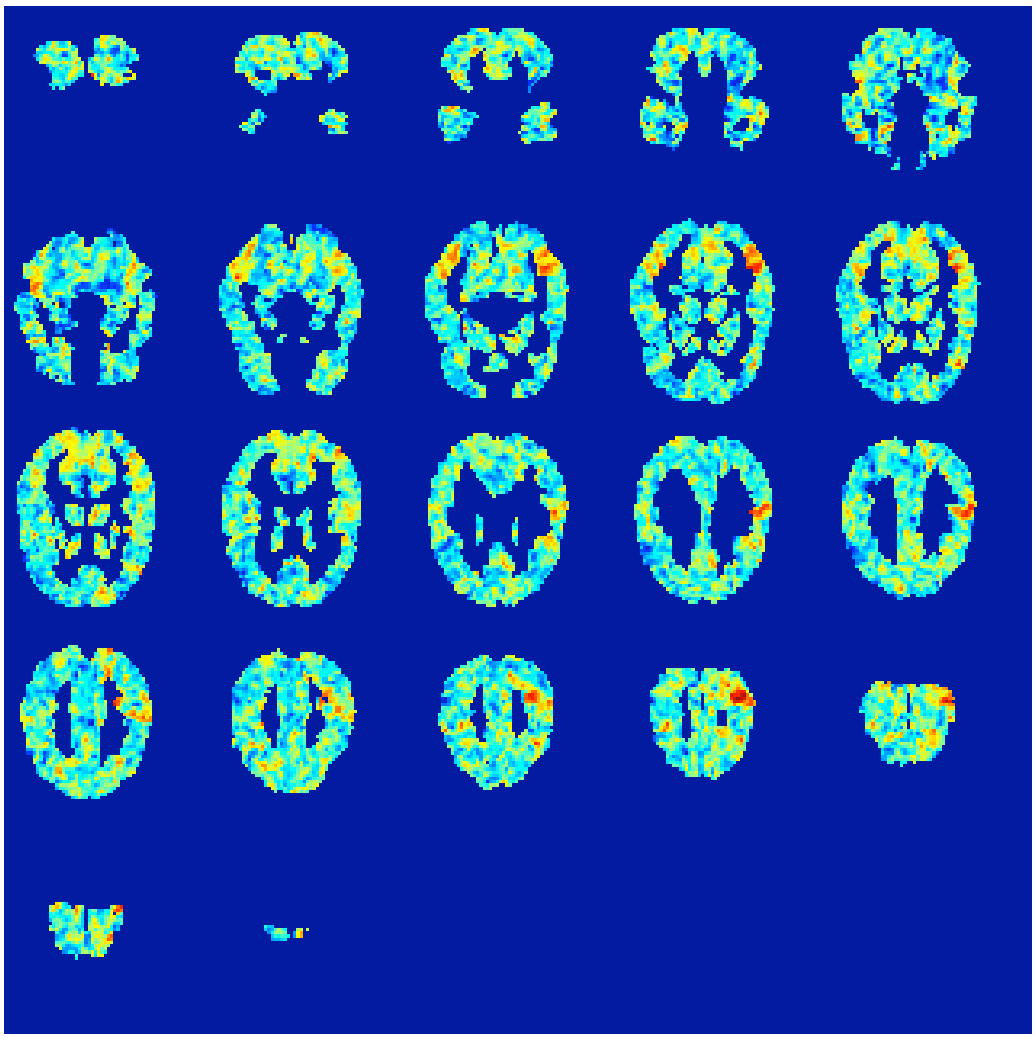


Figure 4.8: Regression weights to each voxel for word Tools. We see strong activation (red) in the superior temporal sulcus which is believed to be associated with the perception of biological motion [Vaina et al., 2001]. We also see strong activation in the postcentral gyrus which is believed to be associated with premotor planning [Pelphrey et al., 2005].

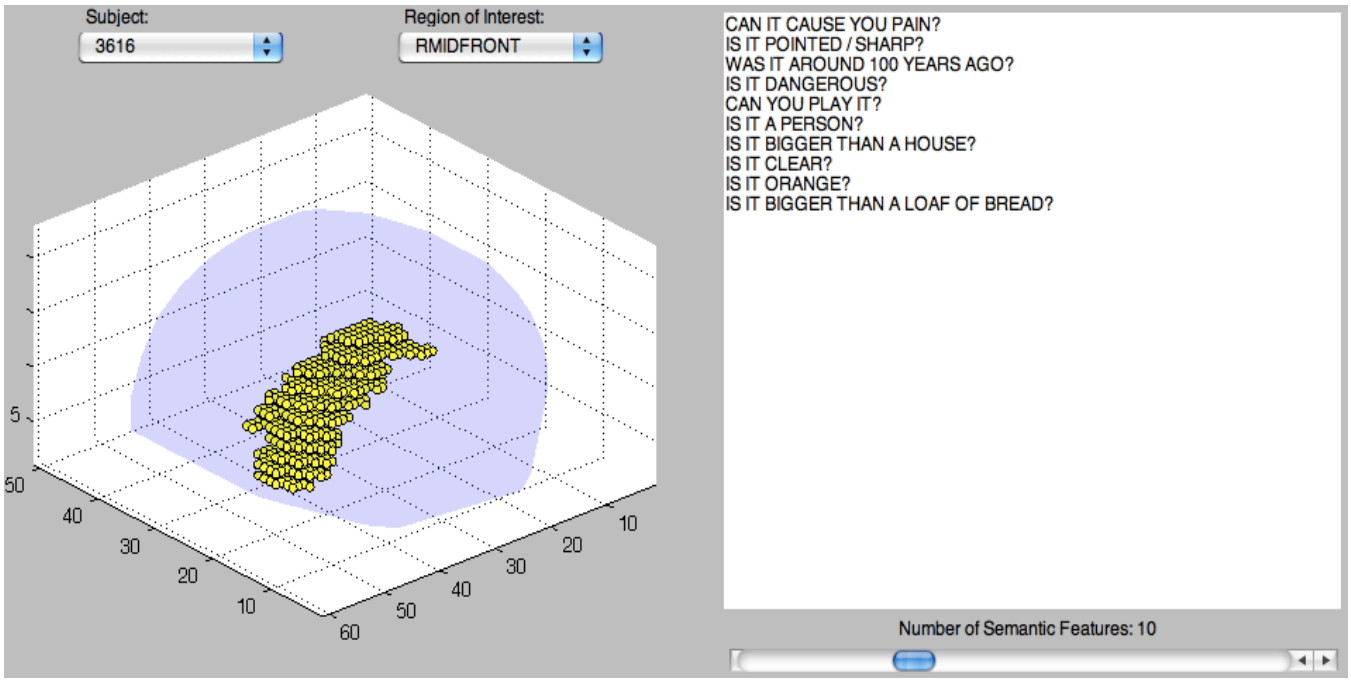


Figure 4.9: The 10 most highly weighted semantic features for the brain region Right-Medial-Frontal.

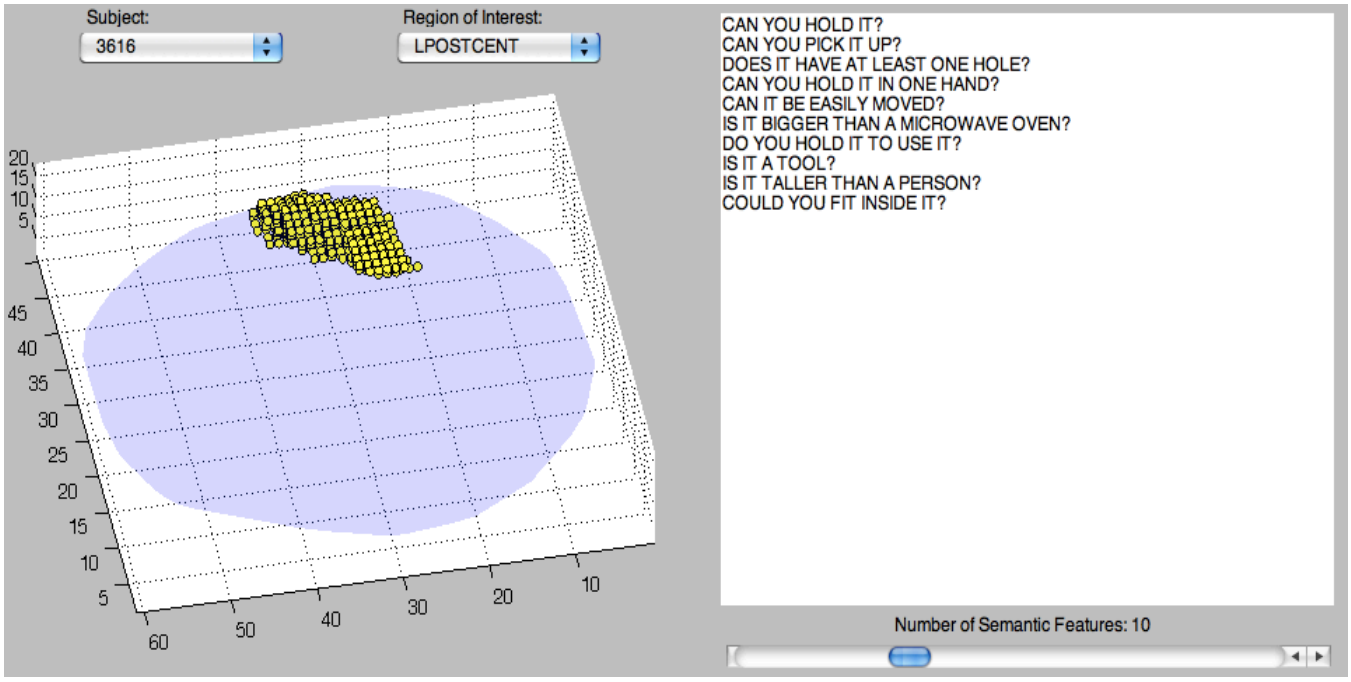


Figure 4.10: The 10 most highly weighted semantic features for the brain region Left-Post-Central.

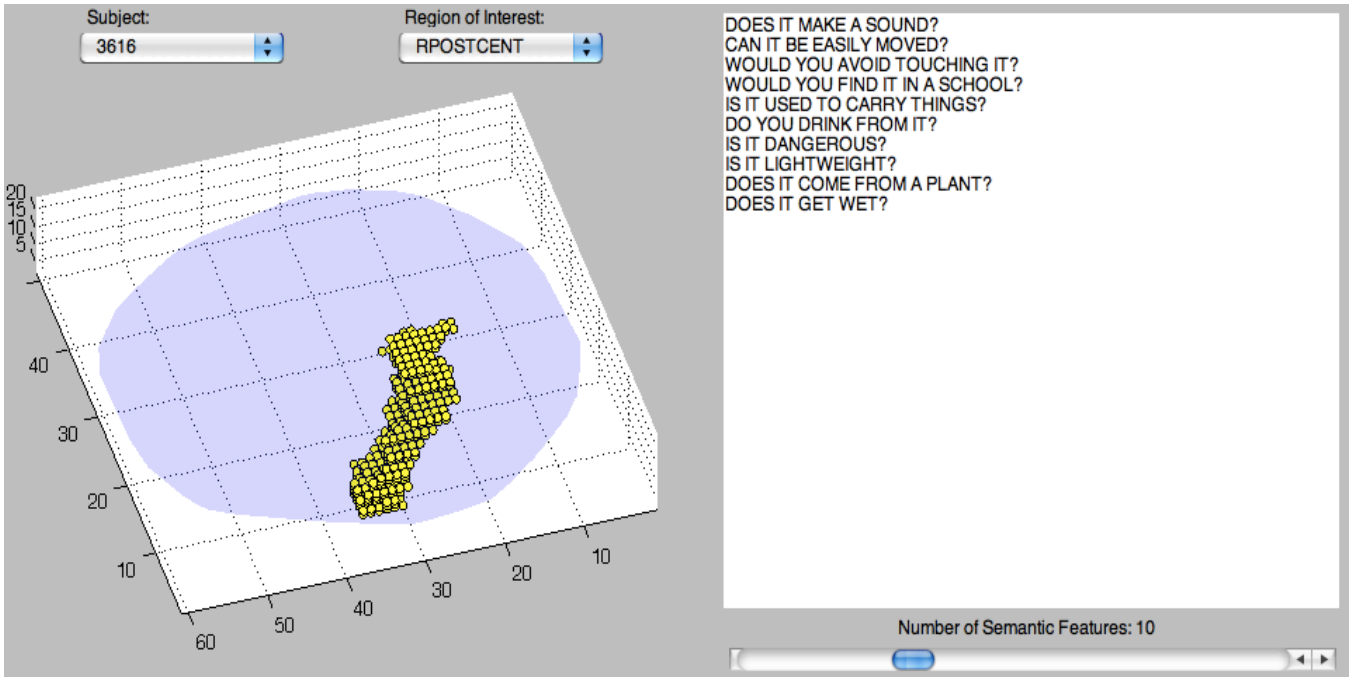


Figure 4.11: The 10 most highly weighted semantic features for the brain region Right-Post-Central.

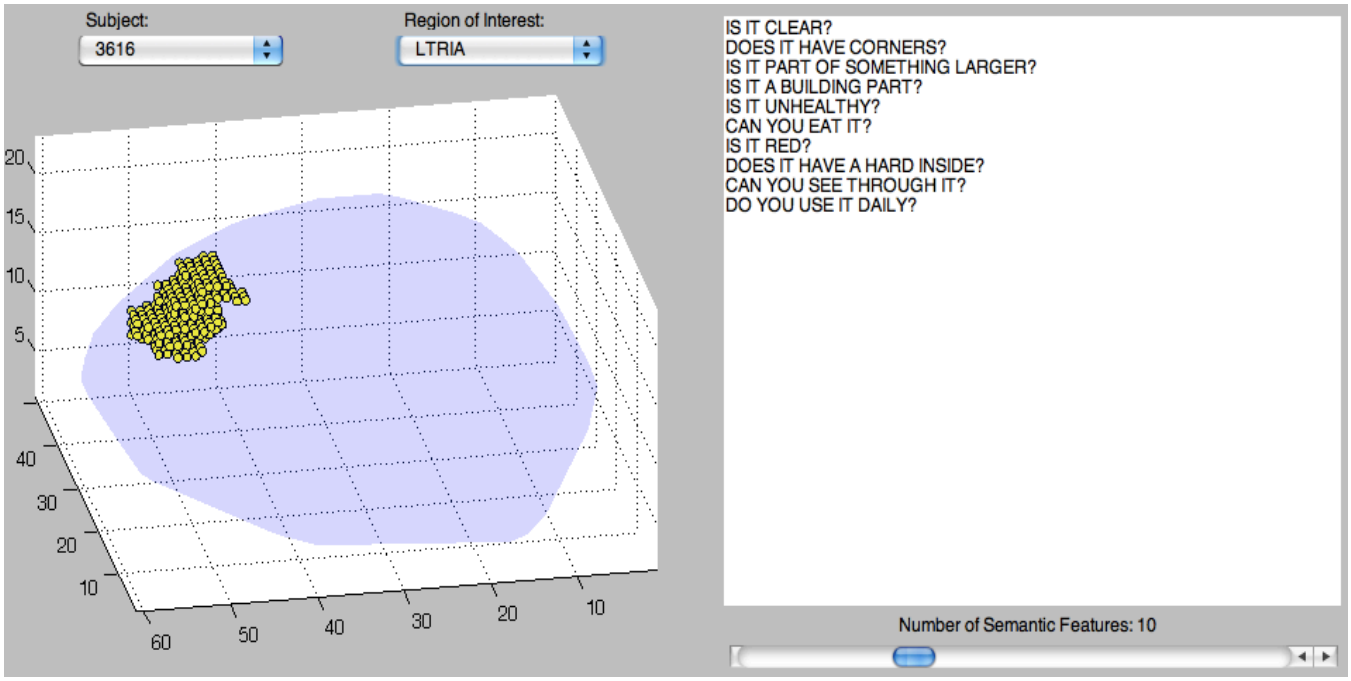


Figure 4.12: The 10 most highly weighted semantic features for the brain region Left-Triangularis.

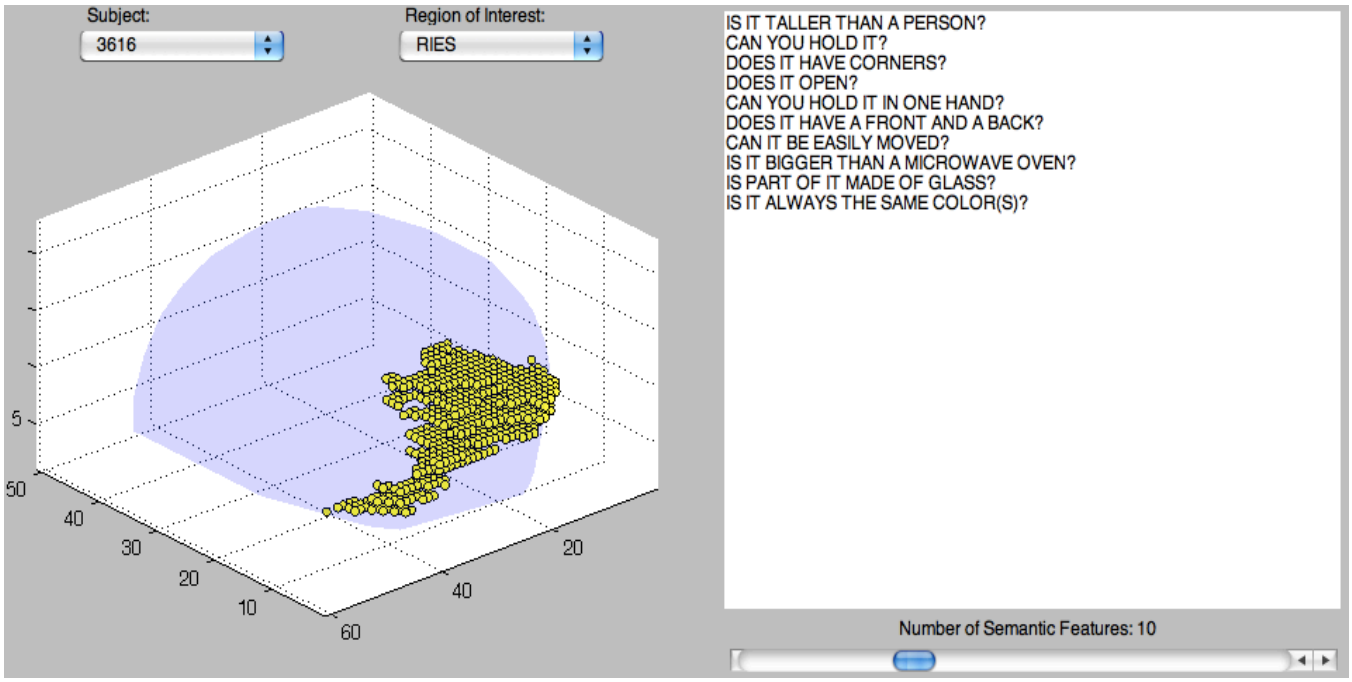


Figure 4.13: The 10 most highly weighted semantic features for the brain region Right-Inferior-Extrastriate.

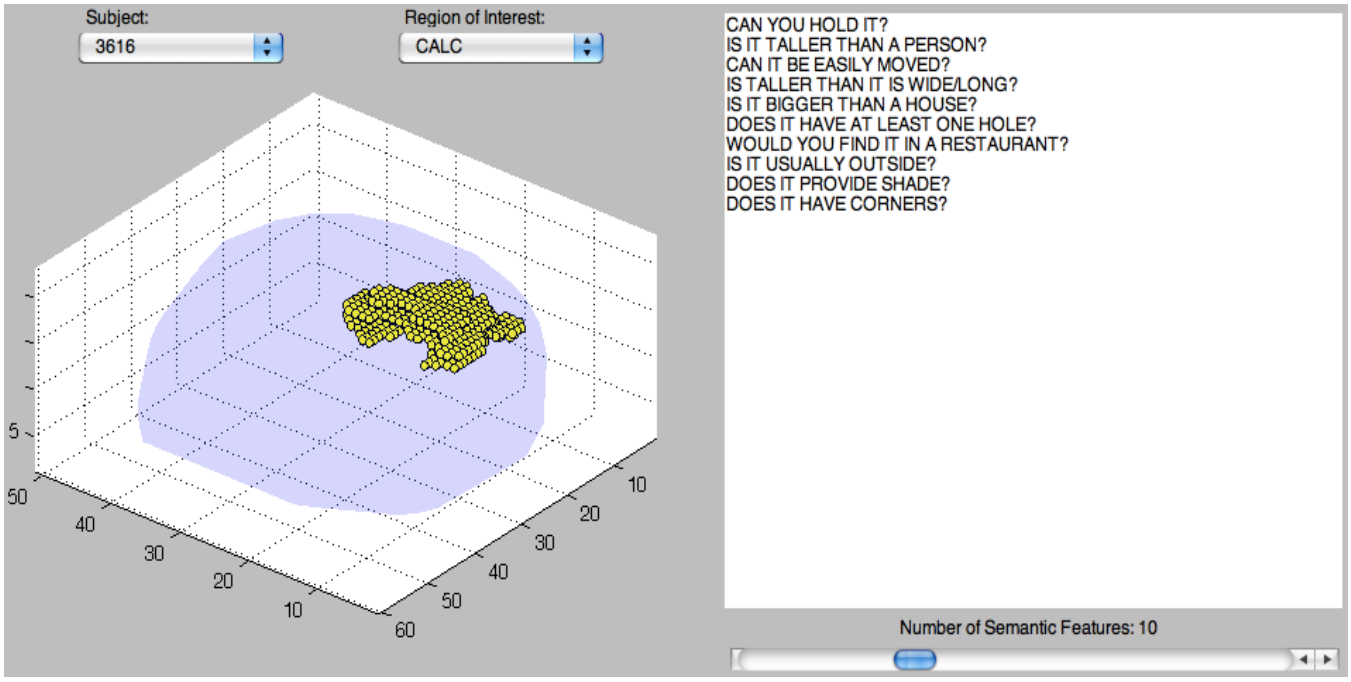


Figure 4.14: The 10 most highly weighted semantic features for the brain region Calcarine-Sulcus.

Chapter 5

Combining Data from Multiple Scanning Modalities

In previous chapters, we applied the zero-shot learning model to data collected from functional magnetic resonance imaging (fMRI). In this chapter, we apply the zero-shot learning model to data collected from lower resolution but more practical neural scanners including magnetoencephalography (MEG) and electroencephalography (EEG). While we found it difficult to predict words with the same granularity and accuracy as with fMRI, we found it is possible to discriminate pairs of words far above the chance level in these imaging modalities as well, even when examples of those words were omitted from the training set. We also found that it is possible to improve the prediction accuracy of the MEG/EEG data by learning latent neural factors common to fMRI and MEG/EEG using kernel canonical correlation analysis (kCCA), and then using these factors as a filter to improve signal quality of the lower spatial resolution MEG/EEG data. This result shows it is possible to leverage data from other human subjects *at training time* to improve prediction accuracy for an individual subject *at test time*, *even when data from other subjects were recorded using a different scanning modality*. This is an important development for neural engineering and brain-computer-interfaces, as it suggests that the lower quality signal of a more practical (i.e. lower cost, better form factor) device can be improved by learning an *a priori* filter using a less practical but higher resolution device.

Chapter Notes for Neural Imaging Researchers

For the cognitive neuroscientist or brain-computer-interface researcher, the most interesting result is how data from multiple scanning modalities can be combined to improve classification performance in a neural imaging task. It is worth noting that the method we propose is similar in spirit to the multimodal techniques that utilize fMRI to augment the timeseries produced by MEG or EEG [Dale et al., 2000, Liu et al., 2009b] for the purposes of studying neural function. By comparison, the method we propose focuses on classification and does not require the same human subjects, structural scans, or explicit registration between modalities.

Chapter Notes for Machine Learning Researchers

For the machine learning researcher, the most interesting result is how performance of one task

can be improved by leveraging data from a related task, even when the related task has significantly different dimensionality and coordinate system. Explicit registration between variables of the two tasks is not required, as a linear transformation is automatically learned as part of the canonical correlation analysis (CCA) model. This is an important consequence, as CCA learns latent variables that are invariant to linear transformations, allowing easier transfer of knowledge between datasets from seemingly different domains.

5.1 Introduction

The zero-shot learning methodology described in earlier chapters was designed for application to an individual subject's data. A common question in neural classification studies, however, is whether performance on a particular classification task can be improved by combining or pooling data from multiple human subjects. Since most imaging studies are unable to collect large numbers of samples, the hope is that data from multiple subjects could be combined to improve classification performance for an individual subject.

Although neural activity patterns are often correlated across subjects for a particular experimental paradigm, structural and functional registration remain difficult challenges [Rustandi, 2010]. No clear consensus has emerged in the literature as to the best methodologies to use.

There has been recent progress from Rustandi et al. [2009], however, that shows it is possible to improve performance in a fMRI classification task by combining data from multiple human subjects using a latent variable model called *canonical correlation analysis*. In their work, they showed that prediction accuracy for a single *fMRI* subject can be improved by utilizing *fMRI* data from the other subjects.

The ability to combine data from multiple subjects would be particularly useful for neural studies performed with magnetoencephalography (MEG) and electroencephalography (EEG), which generally have much lower spatial resolution signals than fMRI, leading to generally lower performance on neural classification tasks. This is because for many classification tasks, the spatial dimension is far more important for robust performance as the discriminating neural activity can be highly localized.

Despite this lower resolution, both the cost and form of these scanners are significantly better than fMRI. This is especially true for EEG, which is the most commonly used neural activity recording device in most state-of-the-art (non-invasive) brain computer interfaces [Guger et al., 2011].

Although the work of [Rustandi et al., 2009] focused on fMRI, we feel similar methods would apply to MEG and EEG as well. While it might be useful to combine lower resolution data from multiple subjects together, in this chapter we consider a slightly different question:

Can training with data from higher resolution fMRI subjects be useful for improving test classification performance of data collected from lower resolution MEG and EEG subjects?

In other words, we investigate the question of combining data from multiple subjects, but we propose a method that can improve performance in a classification task for a subject collected from one scanning modality, by combining with another subject's data from *a different,*

higher resolution scanning modality. Specifically, we'll show how this method can improve performance of MEG and EEG classification tasks by leveraging other subjects' data collected for the *same experimental paradigm*, but using fMRI data during the training phase. No additional fMRI data is needed at testing time.

The goal is to leverage the higher resolution fMRI data as a constraint during the training phase to transform the original MEG or EEG features from a single subject into a new feature representation that performs better with respect to the model than the original features. As shown in Figure 5.1, both MEG/EEG and fMRI data are fed into the CCA algorithm to learn a filter for the MEG/EEG data. This filter is later used during the testing phase to transform the MEG/EEG features into the new feature representation. These new features are then input into the zero-shot learning model described in Chapter 2. Similar to Rustandi et al. [2009], we use *canonical correlation analysis (CCA)* to learn latent variables that can transform the original features into this new representation. We will now describe CCA and then demonstrate how we use it to learn better features for our model.

5.2 Canonical Correlation Analysis (CCA)

Suppose we have two datasets $\mathbf{U} \in \Re^{N*d_u}$ and $\mathbf{V} \in \Re^{N*d_v}$ that represent two different "views" of some underlying object. Both views have N training examples, and d_u and d_v are the respective dimensionalities of the views. For example, a row of $\mathbf{U}$ might represent the neural activity image recorded with MEG in response to a particular stimulus (i.e. the MEG view), while the same row of $\mathbf{V}$ might be the neural activity image recorded with fMRI for the same stimulus (i.e. the fMRI view).

Formally, CCA finds two vectors $\mathbf{z}_u \in \Re^{d_u*1}$ and $\mathbf{z}_v \in \Re^{d_v*1}$, such that the vectors produced by $\mathbf{a}_u = \mathbf{U}\mathbf{z}_u$ and $\mathbf{a}_v = \mathbf{V}\mathbf{z}_v$ are maximally correlated:

$$\max_{\mathbf{z}_u, \mathbf{z}_v} corr(\mathbf{U}\mathbf{z}_u, \mathbf{V}\mathbf{z}_v) \tag{5.1}$$

The vectors $\mathbf{a}_u$ and $\mathbf{a}_v$ are called the first canonical variates, while the solution $\mathbf{z}_u$ and $\mathbf{z}_v$ are called the loadings for the first canonical component. The data matrices can be deflated using the canonical variates, and the process can be repeated to find additional canonical variates. The solution to Equation (5.1) can be written as a generalized eigenvalue problem, the solution of which provides all canonical variates and loadings in an efficient manner. See Hardoon et al. [2004], Hotelling [1936] for details.

Intuitively, CCA can be thought of as a dimensionality reduction method similar to PCA, but CCA uses a different objective function when learning its lower dimensional representation. Whereas PCA finds directions that maximize variance of the data, CCA finds correlation between two variables, $\mathbf{U}$ and $\mathbf{V}$, after projecting the variables onto a common coordinate system. The advantage is that linear relationships can be discovered even if the two variables use different coordinate systems. Without such projection, two variables may appear uncorrelated even if one variable is a direct linear map of the other, assuming the original coordinate systems are sufficiently different. This is an important point, as it allows large flexibility when learning common factors of the two views.

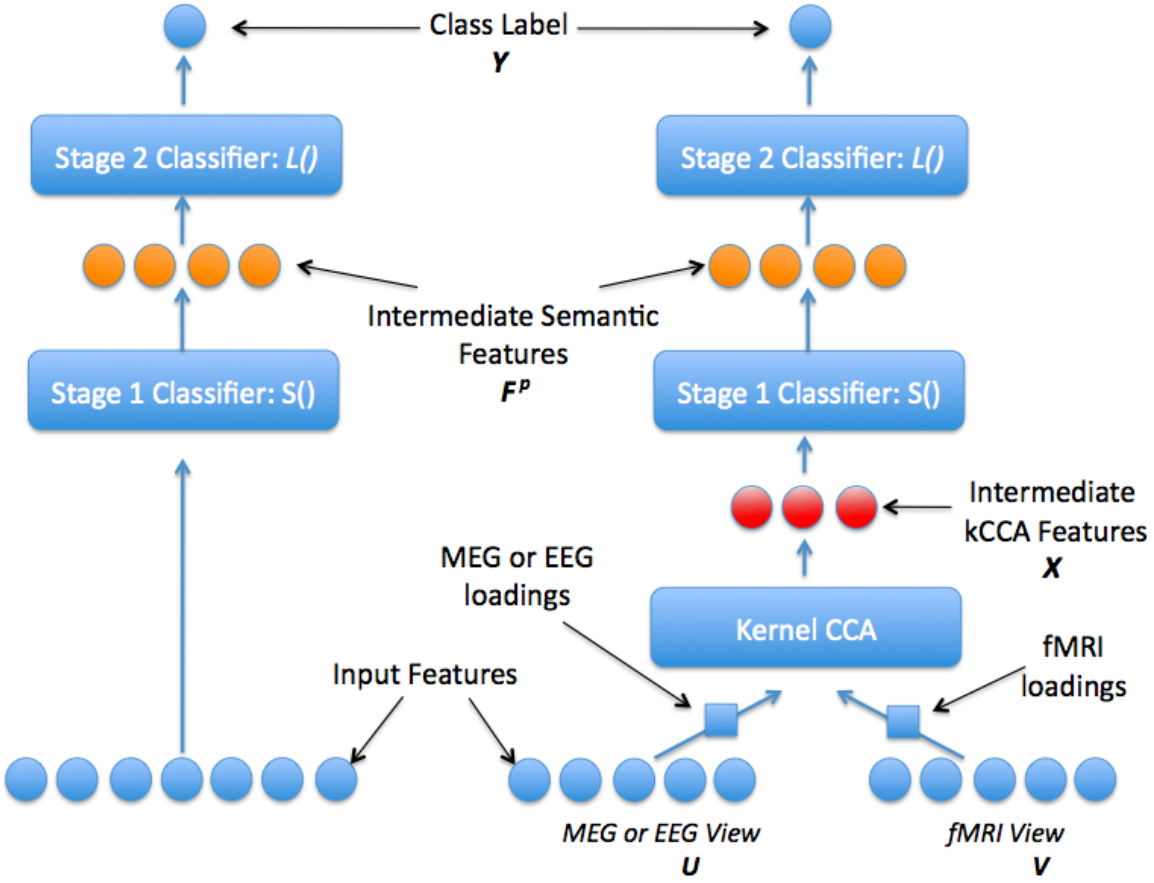


Figure 5.1: The standard zero-shot learning classifier shown on left, compared to the model augmented with intermediate kCCA features. The features were generated by finding the kCCA loadings that maximize the correlation between projections of the MEG view and fMRI view of the data. After training, the MEG or EEG data can be transformed in the kCCA feature space using its corresponding loading. The features can then be used as normal in the zero-shot learning classifier. Note that fMRI data is not needed when applying the classifier at test time.

For the purpose of neural imaging, we conjecture that there are common factors that represent the underlying pattern of neural firing in response to a stimulus, although this pattern is observed differently across scanning modalities and may even be distributed across different regions of the brain. The flexibility of CCA can help us find these common neural factors without requiring us to align the coordinate systems between the different scanners.

The basic CCA formulation above can be extended into kernel form and can include regularization constraints on the loadings to prevent overfitting. This allows the model to train quickly with many thousands of features, even if the number of training examples is small. In our work, we use the kernel canonical correlation analysis (kCCA) derived by Hardoon et al. [2004] and implemented by Rustandi et al. [2009].

We extend our regression model described earlier according to Figure (5.1). Formally, let the matrices $\mathbf{U} \in \Re^{N*d_u}$ be a collection of N examples from MEG (or EEG) and $\mathbf{V} \in \Re^{N*d_v}$ be

a collection of examples from fMRI. Each row of the matrix corresponds to the same stimulus example.

During our training phase, we apply kCCA on $\mathbf{U}$ and $\mathbf{V}$ to learn a matrix of loadings $\mathbf{Z}_u \in \Re^{N*k}$ in kernel space for MEG (or EEG), and $\mathbf{Z}_v \in \Re^{N*k}$ for fMRI, where k is the number of components used. We transform these loadings into their primal form, $\mathbf{L}_u$, which we can use to transform our training and test examples into the new feature space:

$$\mathbf{L}_u = \mathbf{U}^T \mathbf{Z}_u$$

Applied to the training data, we obtain the MEG/EEG examples $\mathbf{X}$ in the new feature space that we created using the combined MEG and fMRI training data:

$$\mathbf{X} = \mathbf{U}\mathbf{L}_u = \mathbf{U}\mathbf{U}^T \mathbf{Z}_u$$

The matrix $\mathbf{X}$ can then be used directly in our regression model in Equation 2.6. Similarly, a new test example $\mathbf{t}$ can be converted into the new feature space by applying the primal loading:

$$\mathbf{x} = \mathbf{t}\mathbf{L}_u$$

which can then be used to obtain a prediction of semantic features as in Equation 2.7. Note that after training, we discard the fMRI loadings $\mathbf{Z}_v$ as they are only used as a constraint to learn the MEG/EEG loadings and are not necessary for testing. We could in theory use them during testing if we had fMRI test examples, but our goal is to test whether adding fMRI data during *training* time can improve classification test performance of MEG/EEG. The goal is to show that classification performance is improved by adding a small amount of high-quality fMRI data to the training data.

The implementation from Rustandi et al. [2009] requires the choice of both a regularization parameter and number of components. For our MEG+fMRI experiments we use the same parameters as recommended by Rustandi, namely 10 components and a regularization parameter of 0.5. Note that this regularization parameter ranges from 0 to 1. The choice of 0.5 balances the magnitude of the projection weights against the correlation of the projected vectors. Since EEG has only 19 raw features (vs. 306 for MEG), we used 5 components for EEG. At the risk of overfitting, we explored various regularization parameters and found the results to be highly robust to this choice. However, the results reported below used the recommended value of 0.5. The performance was much more affected by the choice of number of components, and we found the best performance around 5-15 components, and decreased smoothly as additional components were added. We believe the appropriate choice of components is likely related to number of input dimensions used in the model.

Further, as a requirement for the kCCA experiments, we further normalize each training example (row) of the data matrices input to kCCA to unit length and recompute zero-mean columns per training iteration as recommended by Rustandi et al. [2009]. For fair comparisons, we report the MEG/EEG baseline results using these normalizations as well. We did find these additional normalizations may occasionally decrease performance when compared to the same model that does not use them.

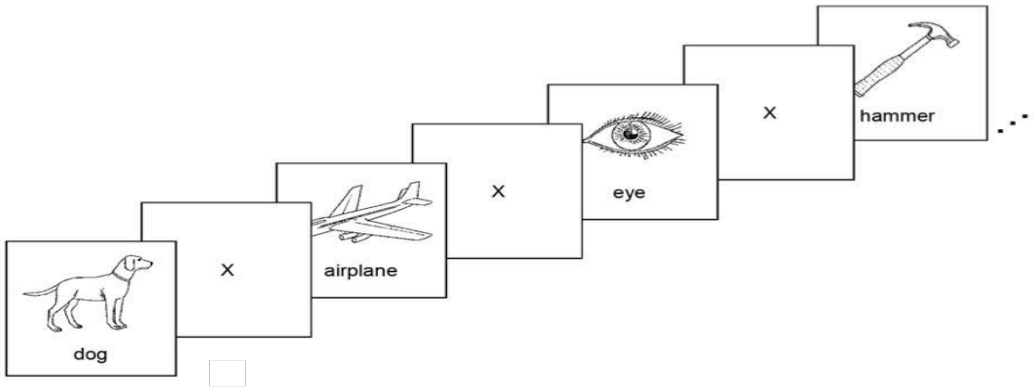


Figure 5.2: The event related paradigm. Line drawings were presented as white lines on black background (shown reversed here for clarity).

5.3 Case Study: Combining Data from fMRI and MEG/EEG

We now apply the above kCCA model to the fMRI data described in Chapter 2, along with new data collected from MEG and EEG.

5.3.1 Paradigm and Task

We used the same experimental paradigm as described in Chapter 2. Recall that participants were presented with 60 stimuli, each being a picture drawing and corresponding word labeling of a concrete noun. There were five exemplars from twelve different semantic categories (Table 2.1). The line drawings were taken from or modeled after those in Snodgrass and Vanderwart [1980]. White lines and white text were shown over a black background.

The following event-related paradigm was used in fMRI: the stimulus was presented for 3s, followed by 7s of a fixation condition where an X was shown in the center of the screen (Figure 5.2). The set of 60 stimuli were shown six times for fMRI with a random permutation of the set used for each presentation.

A similar sequence was used in our MEG and EEG experiments, but different timings were used since MEG and EEG are not subject to the hemodynamic delay: 500ms fixation, 2s stimulus, and 1s blank screen. There were at least six (and up to ten) presentations of each stimuli for each MEG/EEG participant.

The participants were asked to think about properties of the stimulus words during presentation. To encourage a repeatable set of imagined properties for each word, the participants were asked before entering the scanner to create a list of properties for each of the 60 words. Each participant was allowed to create his/her own set of properties and were not limited to a particular set. Further, each participant had no knowledge of the properties selected by the other participants.

5.3.2 MEG/EEG Data Collection Procedures

Nine right-handed human participants were scanned using an Elekta Neuromag® scanner. All subjects gave their written informed consent approved by the University of Pittsburgh (protocol PRO09030355) and Carnegie Mellon (protocol HS09-343) Institutional Review Boards. The MEG scanner has a total of 306 channels, of which 102 are magnetometers and the other 204 are planar gradiometers. The channels are distributed in 102 sensor triplets, each of them containing one magnetometer and two gradiometers that measure the differential magnetic field in orthogonal directions. The data were acquired at 1Khz, high-pass filtered at 0.1Hz and low-pass filtered at 330Hz. Eye movements (EOG) were also monitored by recording differential activity of muscles above, below, and lateral to the eyes. These signals captured horizontal and vertical eye movements, as well as eye blinks. Additionally, four head position indicator (HPI) coils were placed on the subject's scalp to record the position of the head with relation to the MEG helmet at the beginning of each session. These coils, along with three cardinal points (nasal, left and right pre-auricular), were digitized into the system and were later used for source localization. For *three* subjects, EEG data were acquired simultaneously with the MEG data acquisition. Nineteen EEG electrodes were placed on the subject's scalp using the traditional 10-20 system [Niedermeyer and da Silva, 2004]. The Signal Space Separation method [Taulu et al., 2004] was used to remove noise in the MEG data that originated outside of a sphere containing the subject's head. The data were then band-pass filtered between 1-8Hz and down-sampled to 40Hz. Finally, Freesurfer software (`http://surfer.nmr.mgh.harvard.edu/`) was used to construct the 3D model of the brain from the structural MRIs, and the Minimum Norm Estimates method [Hämäläinen and Ilmoniemi, 1994] was used to obtain source localized estimates of brain activity from raw MEG data (MNE software, `http://www.nmr.mgh.harvard.edu/martinos/userInfo/data/sofMNE.php`). Both sensor and source space data were used in the experiments.

5.3.3 Data Norm and Preprocessing

For MEG/EEG, we used the same process that was described in Section 2.3.5 for fMRI data, but first removed the time dimension by averaging the samples taken from the window between 350-450ms after stimulus onset. We chose this time window based on previous MEG work that studied neural activity timings for language processing tasks [Salmelin, 2007]. While we primarily consider this time window, one of the experiments described in our results section explicitly tests model performance for different time windows.

In the experiments below that combine data between fMRI and MEG, we use the same preprocessed fMRI that was used in Chapter 2.

5.3.4 Model

We used the same kernel ridge regression model as described in Section 2.3.6. For our baseline experiments described below, the MEG/EEG features are input directly into the model. In the kCCA experiments, the kCCA features are input into the model.

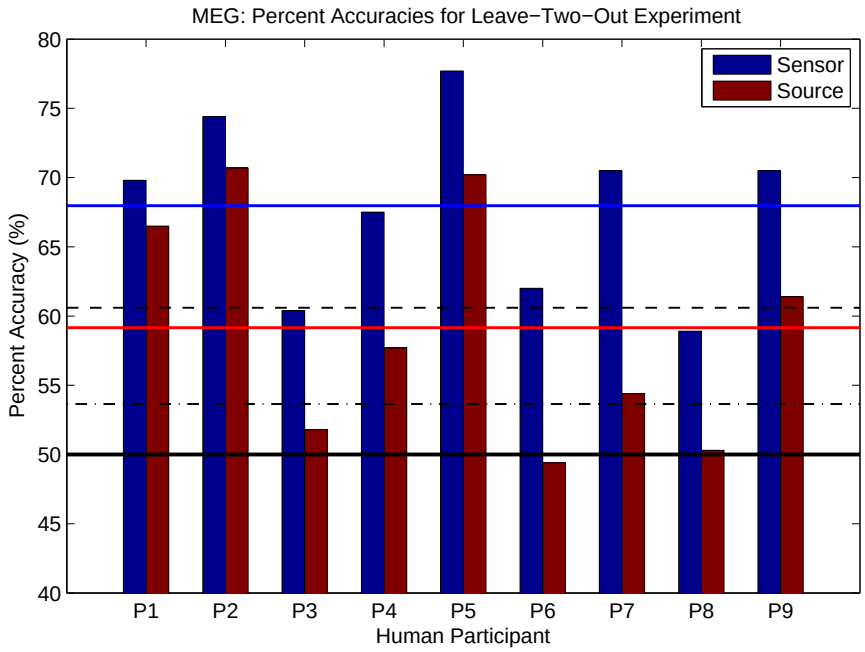


Figure 5.3: Percent accuracies of the 9 MEG participants for the leave-two-out task using both source and sensor data. The chance accuracy is shown as a solid black line, the mean accuracy for all participants (sensor) is shown as a solid blue line, and the mean accuracy for all participants (source) is shown as a solid red line. We also show the 5% significance level (i.e. the threshold for which the p-value would be found significant if $p = 0.05$) for individual participant accuracy as a dashed "-" line, and the 5% significance level for the mean accuracy shown as a dash-dot "- ." line. We found accuracies of sensor data to be significantly better than source data, with eight of nine participants significant at the 5% level. Both sensor and source means were found significant at the 5% level as well. We computed these significance values by computing the empirical distribution of accuracies using random data in place of the real MEG/EEG data and repeating the experiment over a thousand times.

5.4 Experimental Results

5.4.1 MEG and EEG Baseline Experiments

We now pose and answer several questions regarding the performance of MEG/EEG in same *leave-one-out* and *leave-two-out* experiments described in Section 2.3.7. We perform these experiments to obtain baseline accuracies for MEG/EEG before applying the kCCA model described above.

We repeated these experiments using data from nine MEG participants and three EEG participants separately. Given the significantly better performance of the `human218` semantic feature set on the fMRI data, we only used this set for the MEG and EEG experiments described below. The nine participants were different from those used for fMRI. The EEG data was collected simultaneously with the MEG data for only three of the participants.

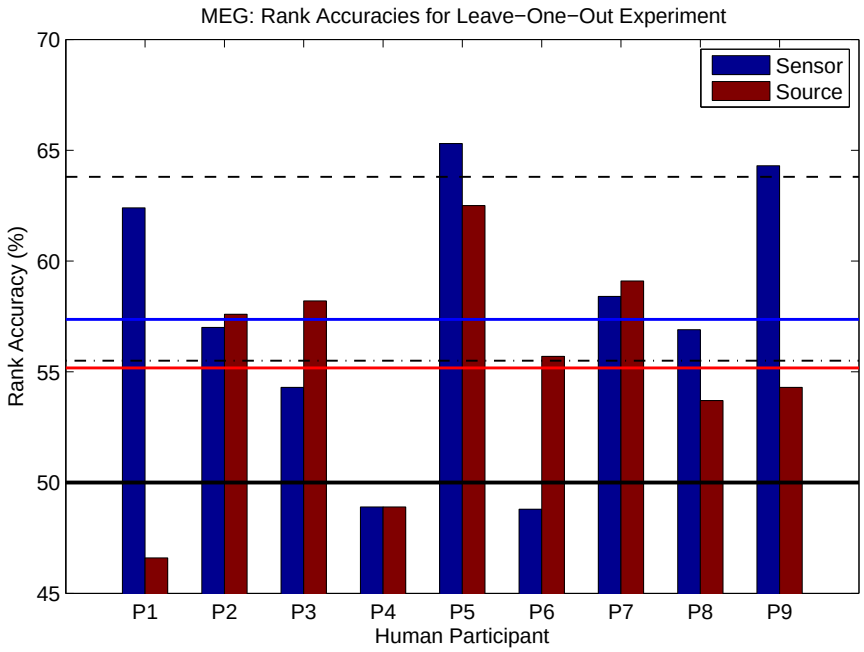


Figure 5.4: Percent rank accuracies of the 9 MEG participants for the harder leave-one-out task using both source and sensor data. The chance accuracy is shown as a solid black line, the mean accuracy for all participants (sensor) is shown as a solid blue line, and the mean accuracy for all participants (source) is shown as a solid red line. We also show the 5% significance level for individual participant accuracy as a dashed "-" line, and the 5% significance level for the mean accuracy shown as a dash-dot "- ." line. We found accuracies of sensor data to be significantly better than source data in this experiment as well. In this harder task, few participants were found significant at the 5% level, although the mean of sensor data was found significant at the 5% level, and the source mean just under the 5% level.

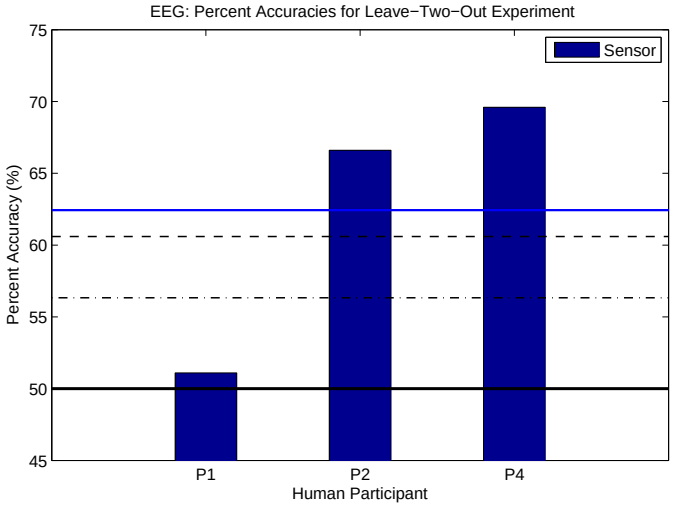


Figure 5.5: Percent accuracies of the 3 available EEG participants (P1-P2-P4) for the leave-two-out task using sensor data. The chance accuracy is shown as a solid black line, the mean accuracy for the three participants is shown as a solid blue line. We also show the 5% significance level for individual subject accuracy as a dashed "-" line, and the 5% significance level for the mean accuracy shown as a dash-dot "- ." line. We found two of three participants significant at the 5% level, and mean was found significant at the 5% level as well.

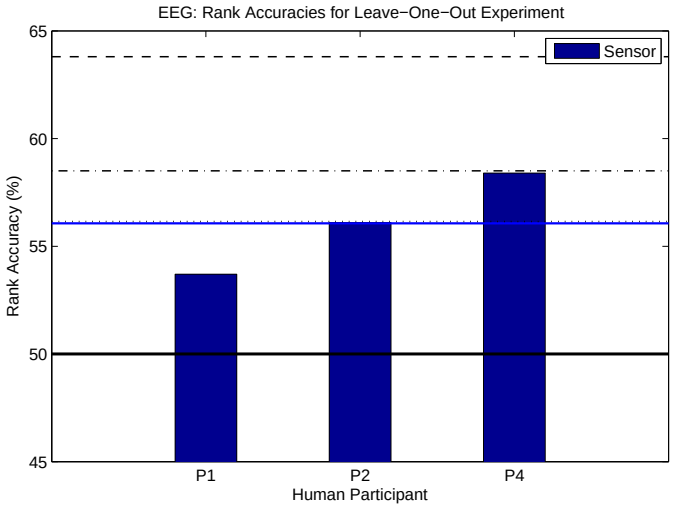


Figure 5.6: Percent rank accuracies of the 3 available EEG participants (P1-P2-P4) for the harder leave-one-out task using sensor data. The chance accuracy is shown as a solid black line, the mean accuracy for the three participants is shown as a solid blue line. We also show the 5% significance level for individual subject accuracy as a dashed "-" line, and the 5% significance level for the mean accuracy shown as a dash-dot "- ." line. This is the hardest task, with the noisiest data, and neither the individual subject accuracies nor mean were found significant at the 5% level. The mean accuracy was found significant at the 10% level (shown just under the dotted "..." line).

How well can we predict words using MEG and EEG?

We found that both MEG and EEG could classify above the chance level for the easier *leave-two-out* experiment, with eight of nine MEG participants and two of three EEG participants significant at the 5% level. The mean accuracies across all participants was found to be significant at the 5% level as well with 68% for MEG sensor, 59.2% for MEG source, and 62.4% for EEG. See Figures 5.3 and 5.5. *This shows that using the zero-shot learning model, both MEG and EEG can perform well even when the test classes were omitted from the training set*.

For the harder *leave-one-out* experiment, where the task is to discriminate from a set of nearly 1,000 words, both MEG and EEG performed far below the performance of fMRI, and while slightly above the chance level, only two participants were significant at the 5% level, although the mean accuracy across participants was significant at the 5% level. See Figures 5.4 and 5.6. It is possible these results could be improved by using different input features, rather than the temporally averaged features used here. It is also possible that different semantic feature sets may be better predicted than those used in the `human218` set. However, we conjecture that for the `human218` semantic features, spatial resolution is far more important than temporal resolution for predicting the semantic properties of the stimuli. Noise characteristics of each scanner and the differences in what is sensed may also explain the performance gap.

Is MEG performance higher using sensor data or source localized data?

The raw data collected using the MEG scanner has 306 different sensors. For many MEG experiments described in the literature, the MEG data are *source localized* using structural MRI scans to register the neural activity to specific regions of the brain. While it is not possible to find a definite solution to the inverse problem in MEG source localization [Baillet et al., 2001], different methods can be used to estimate the sources of magnetic field in the cortex. This work used the popular *minimum norm estimate* (MNE) method [Hämäläinen and Ilmoniemi, 1994].

Although useful for interpreting the neural scans, it is unclear the effect of source localization on neural classification tasks. The impact seems to be task specific, and no clear answer has emerged in the literature as to whether better performance should be expected in classification studies using source localized vs. the minimally processed sensor data.

We compared our results using both sensor and source data[1] in Figures 5.3 and 5.4. *We found that in both experiments, the sensor data performed better than the source localized data,* particularly for the leave-two-out experiment. While this might be attributable to parameter settings for the MNE localization, we found no setting that yielded better results than the sensor data. Given the number of steps required in the preprocessing of the data, is it difficult to make any conclusive claim as whether source vs. sensor data are better for classification tasks, and more work needs to be done in the community to establish a standard preprocessing pipeline. Further, it would be useful to know what neural classification tasks would likely perform better using source data, and what tasks would likely perform better using sensor data.

What is the best time window for prediction?

[1]Low-pass filtering and down sampling were applied to the raw data.

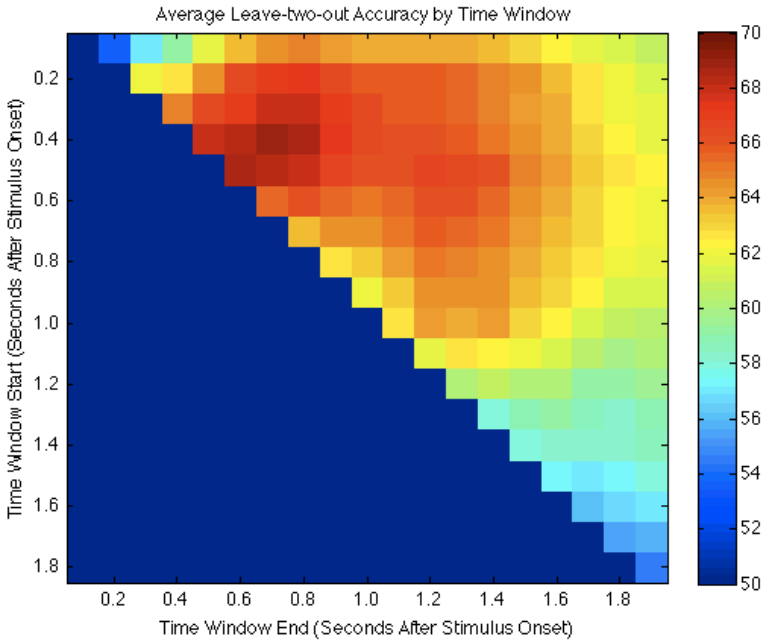


Figure 5.7: MEG accuracies using sensor data for the leave-two-out task averaged over all nine human subjects. Results are reported for each possible time window between 100-1900ms using 100ms increments. The Y-axis indicates the start time of the window in seconds after stimulus onset. The X-axis indicates the end time of the window in seconds after stimulus onset.

The MEG and EEG results reported earlier were based on features computed by averaging the time series data from 350-450ms after stimulus offset. This time window was chosen based on prior results reported in the literature [Salmelin, 2007] regarding semantic processing. An interesting question is whether this is the optimal time window to use, or if other windows might yield superior results[2].

We performed the *leave-two-out* experiment on the MEG sensor data for 171 different time windows. These include all the time windows using 100ms increments up to 1900ms after stimulus offset (e.g. 100-200ms, 100-300ms, 100-400ms, ..., 200-300ms, 200-400ms, ..., 1800-1900ms). The results are shown in Figure (5.7). *We see the peak accuracies occur in time windows that begin between 300-500ms and lasting until 600-800ms*. Although the time window we chose based on prior literature was not the absolute peak accuracy, the performance was very similar. This provides additional evidence that the best time window for predicting semantic features occurs around 400ms.

[2]There are many other ways to compute features from the raw time series data, including frequency-based features. In this work, however, we only consider features based on averaging signal amplitude over different time windows.

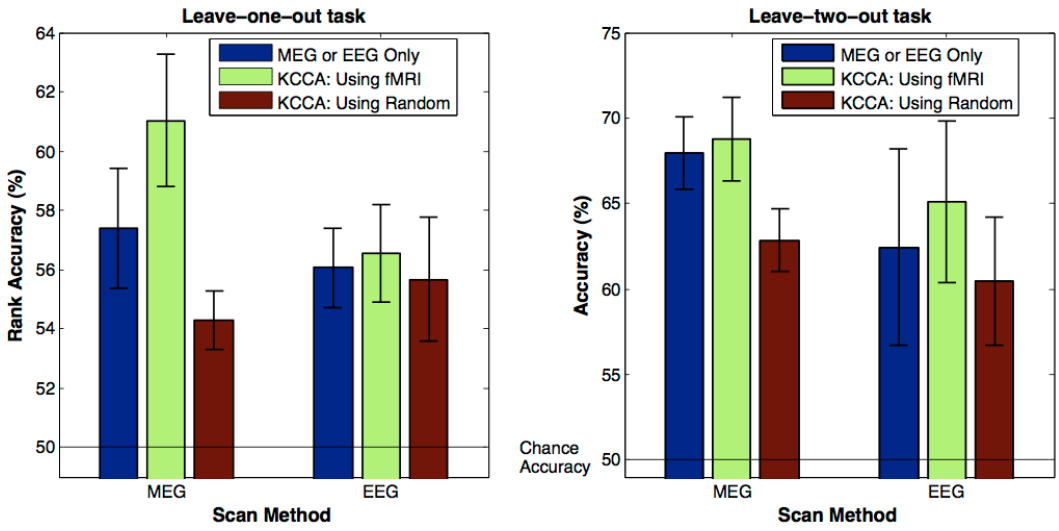


Figure 5.8: Average effect over nine human subjects on MEG and EEG test performance by training the kCCA model using fMRI data. Both leave-one-out and leave-two-out experiments were performed. Results are also reported when the model was trained using random Gaussian data instead of real fMRI data. Jackknife 95% confidence intervals are reported for the mean estimator.

5.4.2 kCCA Experiments with MEG/EEG and fMRI

Can we improve performance by leveraging data from another subject in a different imaging modality?

We trained the *kernel canonical correlation analysis* model to test whether we can improve MEG and EEG classification performance by leveraging higher spatial resolution fMRI data *from another human subject* during classifier training. Figure (5.8) shows the results of the leave-one-out and leave-two-out tasks for both MEG and EEG. In each experiment, we see that training with data from the subject increased the performance of the classifier.

We also tested the same kCCA model using random Gaussian data as replacement for real fMRI data. The goal was to test whether the improvement was really due to the fMRI data, or some other consequence of the model. We see in Figure (5.8) that using random data rather than the real fMRI data substantially decreased performance in each of the four experiments.

Figure (5.9) shows the MEG performance for each subject in the leave-one-out task for both the kCCA and baseline models. We see that kCCA helped performance in almost all subjects, and only one subject had a notable decrease. In some subjects, the performance increase was dramatic, and Subject 2 had more than 15 percentage point improvement.

These results provide evidence that classifier performance using lower spatial resolution MEG or EEG scanners can be improved by training with another human subject's data collected from the high resolution fMRI scanner.

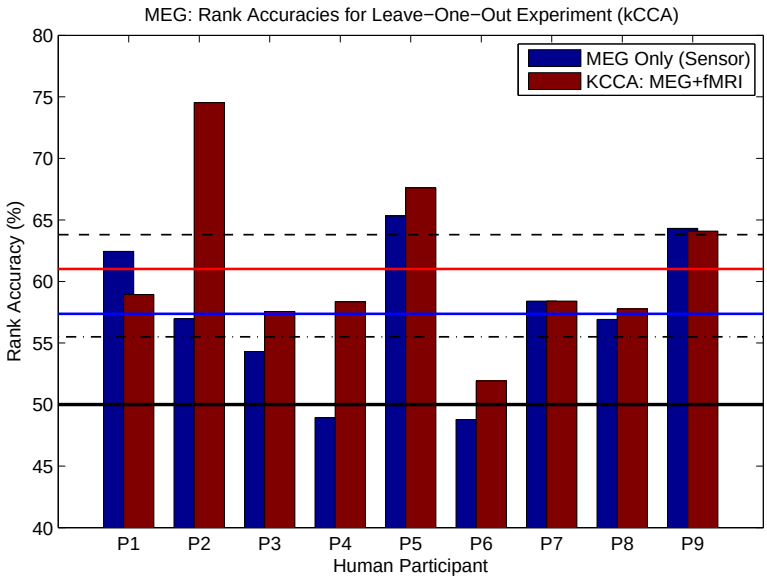


Figure 5.9: Comparison of the MEG leave-one-out task using the standard regression model, and the kCCA model that includes fMRI data during training. Shown are percent rank accuracies of the 9 MEG participants for the harder leave-one-out task. The chance accuracy is shown as a solid black line, the mean accuracy for all participants (standard regression model) is shown as a solid blue line, and the mean accuracy for the kCCA model that includes fMRI data is shown as a solid red line. We also show the 5% significance level for individual participant accuracy as a dashed "-" line, and the 5% significance level for the mean accuracy shown as a dash-dot "- ." line. We found that kCCA only decreased accuracy in one participant, while occasionally causing an increase of 8-15%. While most subjects were not individually significant at the 5% level, both means were found significant at the 5% level.

To our knowledge this is the first result that makes this claim. We believe this could be an important consequence for brain-computer-interface problems, as it shows the lower quality signal of a more practical (i.e. lower cost, better form factor) device like EEG can be improved by learning an *a priori* filter using a less practical but higher resolution device such as fMRI.

As a practical advantage, it should be noted that no explicit registration of the time series or structural data is required. Although this is an exciting result, it is worth noting that the choice of semantic features, time windows, and preprocessing can have a larger effect on performance than kCCA. If maximum performance is desired, these other experimental factors should be optimized first before attempting to leverage data from fMRI to increase performance.

The results described here leveraged data from our best performing fMRI subject. Using a generalized version of kCCA that extends to more than two views of the data, we tested a similar model that combined the MEG/EEG data with data from all available fMRI subjects simultaneously. We found, however, the accuracy of this model to be quite similar to one that utilized only the best fMRI subject's data, despite being significantly computationally slower. Although more experiments should be performed, we have not found any advantage of using data from multiple subjects instead of just the best performing one.

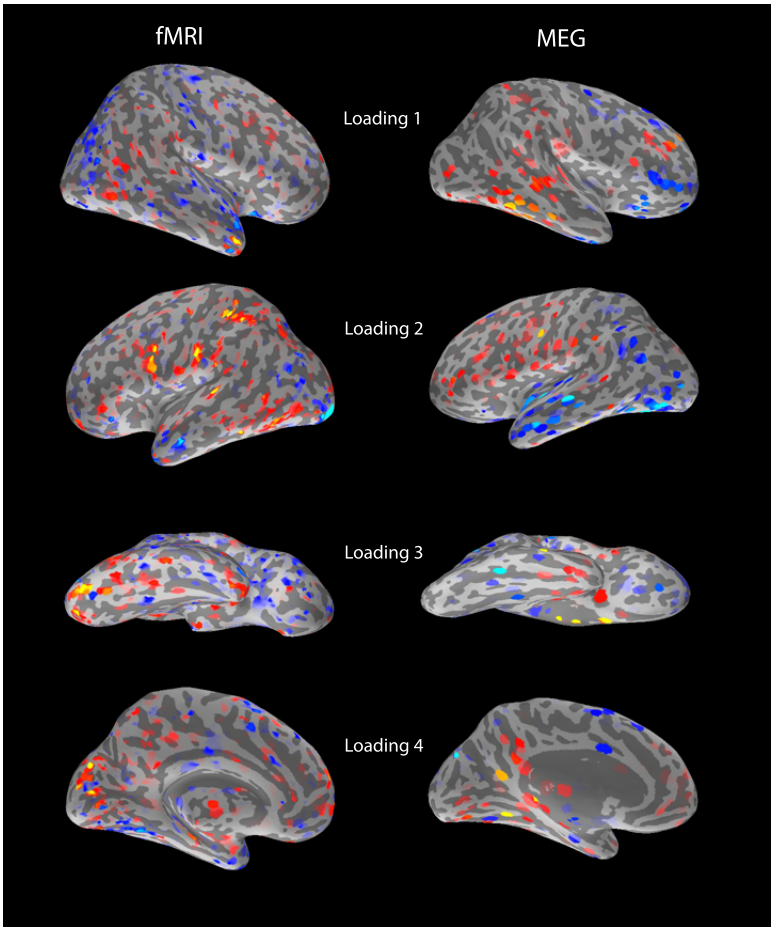


Figure 5.10: Plots of the four most dominant kCCA loadings when plotted onto anatomical brains using MNE software. These plots show neural activities measured in fMRI and MEG for similar neural concepts. Although kCCA does not consider geometry in its learning algorithm, several spatial correlations can be seen. Several activities, however, only appear in one scanning modality.

How are similar neural concepts represented in fMRI and MEG?

The kCCA algorithm learns loadings that project the fMRI and MEG data to a lower dimensional representation such that the two projections are maximally correlated. These learned loadings have a physical interpretation and can be mapped onto the geometry of the brain. Visualizing this mapping can yield insights into how neural activity appears in fMRI and MEG for similar neural concepts.

In order to visualize the loadings, we ran kCCA using the full brain fMRI data and the source localized MEG data. We utilized all available examples since no cross-validation is required and plotted the loading weights onto a 3D brain model using the MNE software. The four most dominant loadings were considered in Figure (5.10).

We found that several of the first four learned loadings exhibit spatial correlations in the temporal and frontal lobes, especially in the first and second loadings. This is an interesting result because kCCA does not consider the geometry of the brain or encourage spatial smoothness when learning the loadings. Although fMRI and MEG measure very different physical processes, we do see that both can observe similar activity.

There are also several differences in observed activities between the two modalities. In the third and fourth fMRI loadings, we see activities in the occipital lobe that do not appear in MEG. This is likely due to the time window used in the MEG data. We considered the window of 350-450ms post stimulus onset which is typically associated with semantic processing and not sensory input. In the temporal lobe, we also see activities in the second and third fMRI loadings that do not appear in the MEG loadings, as well as activities in first and fourth MEG loadings that do not appear in the fMRI loadings.

5.5 Future Work

From a neural imaging and brain-computer-interface perspective, we believe that augmenting data from lower resolution scanners with data from high resolution scanners is an interesting direction for future work. Most of the work in this chapter only focused on features created by averaging a small interval of the time series data from 350-450ms for MEG/EEG, but other intervals should be considered as well. The kernelized ridge regression model presented in this chapter should easily accommodate significantly larger feature sets and it may be possible to test all time intervals simultaneously, allowing model regularization to automatically choose the most useful time points.

Although we showed CCA as a potentially useful latent variable model, there may be other latent variable models that are more effective at transferring knowledge between the different scanning modalities. One particular idea would be a CCA-like model that includes full time series data in one view (rather than just temporally averaged features), and non-time series data in another view. This would create a latent variable model that could potentially transfer information more effectively between two datasets, even when they have little temporal relationship.

From a machine learning perspective, we feel that CCA is an important model for transferring knowledge between learning tasks. Unlike other dimensionality reduction techniques like PCA, the CCA objective function learns latent variables that are invariant to linear transformations, allowing a great deal of flexibility when transferring knowledge between tasks. Although linear transformations are only one particular function class, we believe there may exist other, similar models to CCA that learn latent variables that are invariant to other transformations. Research in this direction may lead to a powerful class of models that can learn different types invariant features, leading to greater abstraction and transfer of knowledge between tasks.

5.6 Conclusion

We experimented with different neuro-imaging datasets and found that it is possible to apply the zero-shot learning paradigm to lower resolution MEG and EEG data, allowing classification of

novel concrete words that were omitted from a training set. Although performance was significantly less accurate than fMRI, it was possible to discriminate between pairs of novel concrete nouns far above chance, while the harder task of classifying from a set of a thousand words was only slightly above chance. We also explicitly tested different time windows of the data, and found that the best performance occurred when the window 300-600 milliseconds (post stimulus onset) was used. This appears consistent with previous work in semantic priming that observed activation at 400 milliseconds (post stimulus onset) [Salmelin, 2007]. This activation is often referred to as the N400 [Kutas and Federmeier, 2011].

While performance using MEG and EEG data was less accurate than fMRI, we found that it is possible to utilize higher resolution fMRI data from another human subject in order to increase classification performance of the MEG and EEG data *at test time, without additional fMRI examples*. To our knowledge, this is the first time that data from a human subject in one scanner modality was used to increase classification performance on another human's data taken from another scanning modality. This has an important consequence for brain-computer interfaces, as it shows that a high-resolution, but less practical scanning modality like fMRI can be used to improve performance of a low-resolution but better form factor scanner like EEG.

5.7 Additional Chapter Remarks

Sample Code

We provide a MATLAB implementation of the kCCA procedure at:
`http://www.thoughtrec.com`

Additional Acknowledgements

Thanks to Gustavo Sudre who collected and preprocessed the MEG/EEG data and generated the 3D MEG plots. Gustavo, along with Dean Pomerleau, Tom Mitchell, and Geoff Hinton, were all co-authors on a paper that included many of these results [Palatucci et al., 2011]. Also, to Indra Rustandi who provided a fast kCCA implementation and guidance on data normalization [Rustandi, 2010], and to David Hardoon for publishing an excellent tutorial about kCCA [Hardoon et al., 2004].

Chapter 6

Conclusion

In this thesis we showed that it is possible for a machine learning classifier to discriminate classes that were not explicitly included in a training set by leveraging semantic knowledge of the classes such as their physical or functional attributes. Further, we found it is possible for the classifier to automatically select the semantic attributes or features that are most useful for a particular task, minimizing the effort required by a human to precisely define the optimal set of semantic attributes in advance.

Using this technique, we showed that the semantic properties of concrete nouns are predictable from the brain activity observed while a person is thinking about those nouns. This makes it possible to train a machine learning classifier to discriminate concrete nouns that people are thinking about, even without explicitly collecting examples of those nouns for a training set. Further, this allows discrimination of certain nouns that are within the same category with significantly higher accuracies than previously possible.

While this thesis focused primarily on classification of neural imaging data, many of the methodologies developed apply more broadly to the field of machine learning and we believe are very relevant to other application areas. Below we summarize the main conclusions, discuss their broader impacts, and suggest avenues for future work.

6.1 Summary of Contributions

6.1.1 Thought Recognition using Zero-Shot Learning

We first presented a formalism for a zero-shot learning algorithm known as the semantic output code classifier. This classifier can predict novel classes that were omitted from a training set by leveraging a semantic knowledge base that encodes features common to both the novel classes and the training set. We studied this model in a PAC framework and proved the first formal guarantee that shows conditions under which this classifier will predict novel classes.

We applied this framework to the task of *concrete noun recognition* and demonstrated that we can discriminate concrete nouns that people are thinking about far above the chance level without any training examples of those nouns. This is an important result for the neural imaging

community as it shows that using a small amount of training data, it is possible to discriminate a much larger set of cognitive states, thereby saving significant time and expense when collecting data from human subjects.

When applied to fMRI data, we showed that the model can discriminate certain nouns that are within the same category with significantly higher accuracy than previously possible. These results advance the state-of-the-art in neural decoding and are a promising step towards a large vocabulary brain-computer interface.

6.1.2 Semantic Feature Selection using Support Vector Ranking

To improve on the usefulness of the zero-shot learning algorithm, we presented a method based on support vector ranking that is useful for selecting semantic features for a zero-shot learning algorithm. This lowers the burden of determining the most useful semantic features *a priori*, and allows the algorithm to select the most useful features from a much larger set. When applied to our neural imaging dataset, we selected features related to *density and internal structure* as well as *shelter* qualities of the objects we considered.

Surprisingly, the importance of selecting semantic features decreased as the size of the original semantic set increased. Further, our empirical results suggest that a zero-shot learning algorithm that utilizes nearest neighbor is very robust to potentially irrelevant features. Thus, we found it is far more important to make sure useful features are included in the model, rather than prevent irrelevant features from being excluded.

We presented an initial theoretical analysis of using nearest neighbor in the zero-shot learning algorithm. This analysis showed under certain distribution and metric assumptions, the tolerance for error in the semantic features predictions can grow with the number of features. Our theory suggests that there is a benefit to using a semantic feature set that is sufficiently large (i.e. more than fifty semantic features), however we also found there may not be much benefit to using a semantic feature set that consists of more than few hundred semantic features.

6.1.3 Predicting Neural Activity using Multi-Task Learning

We also studied the inverse of the zero-shot learning model. Specifically, we built a model to predict brain activity from the semantic features, instead of predicting semantic features from brain activity. This model is generative in the sense that given the semantic features for a novel word, it can often predict the brain activity that would be observed in fMRI images collected from a human subject thinking about the word.

Similar to the zero-shot learning model, we had the problem of choosing the best semantic features to input into the model. Although the previous Support Vector Ranking approach does not apply to the generative model, we demonstrated an alternative method that can be used when predicting brain activity from semantic features. Specifically, we showed how to formulate this semantic feature selection problem using the multi-task Lasso objective function. In collaboration with statisticians working to develop the first scalable algorithm to solve this objective, we presented the first case study of a large scale multi-task Lasso problem with thousands of features and tasks, and showed how we can automatically learn a useful set of semantic features that perform as well or better than the handcrafted set of features reported in Mitchell et al. [2008].

Further, we applied the multi-task Lasso to different regions of interest (ROIs) of the brain, and reported the set of semantic features that are most useful for predicting activity in that region.

We believe that learning methods that include sparsity constraints provide useful tools for neuroscientists to test and evaluate theories of cognition. By using fewer assumptions *a priori* and allowing the models to choose the most useful features automatically, sparse learning methods reduce the risk of implicit overfitting, and may lead to more believable results.

6.1.4 Combining Data from Multiple Scanning Modalities

We experimented with different neuro-imaging datasets and found that it is possible to apply the zero-shot learning paradigm to lower resolution MEG and EEG data, allowing classification of novel concrete words that were omitted from a training set. Although performance was significantly less accurate than fMRI, it was possible to discriminate between pairs of novel concrete nouns far above chance, while the harder task of classifying from a set of a thousand words was only slightly above chance. We also explicitly tested different time windows of the data, and found that the best performance occurred when the window 300-600 milliseconds (post stimulus onset) was used. This appears consistent with previous work in semantic priming that observed activation at 400 milliseconds (post stimulus onset) [Salmelin, 2007]. This activation is often referred to as the N400 [Kutas and Federmeier, 2011].

While performance using MEG and EEG data was less accurate than fMRI, we found that it is possible to utilize higher resolution fMRI data from another human subject in order to increase classification performance of the MEG and EEG data *at test time, without additional fMRI examples*. To our knowledge, this is the first time that data from a human subject in one scanner modality was used to increase classification performance on another human's data taken from another scanning modality. This has an important consequence for brain-computer interfaces, as it shows that a high-resolution, but less practical scanning modality like fMRI can be used to improve performance of a low-resolution but better form factor scanner like EEG.

6.2 Software Contributions

We provide sample code for all the methodologies described in this thesis including kernel ridge regression with efficient cross-validation, support vector ranking, multi-task Lasso, MTL-GUI, and kernel canonical correlation analysis.
The code can be downloaded from: `http://www.thoughtrec.com`

6.3 Future Work

We now summarize avenues for future work in both machine learning and also neuro-imaging. In machine learning, we suggest additional ways to extend the zero-shot learning model and also describe other domains where the model is useful. In neuro-imaging, we suggest additional experiments to classify different parts of speech, and ways to classify data across modalities.

6.3.1 Thought Recognition using Zero-Shot Learning

We found in our zero-shot learning experiments in Chapter 2 that the largest improvement in classifier accuracy resulted from using the human labeled semantic features (`human218`) rather than those generated from corpus statistics (`corpus5000`). While the `human218` semantic feature set performed reasonably well, additional work should explore what other semantic features could be classified accurately. For example, features related more to function or purpose should be explored in greater depth.

Future work should also explore classifying words that are not concrete nouns, such as adjectives, verbs, and also more abstract nouns like *democracy*. If different parts of speech can be classified, an obvious next goal would be to experiment with classifying phrases or short sentences. This challenging goal will require models that fully utilize the temporal dimension of neural activity, rather than the simpler single time point features described in this thesis. It is likely that scanners such as MEG and EEG that better measure the temporal dimension of neural activity will be required.

Another interesting direction would be to combine the methodologies described here with other brain-computer-interface techniques like P300 [Donchin et al., 2000]. P300 is a very consistent pattern of neural activation that occurs 300 milliseconds after a thought or perception and is commonly used as a triggering signal in brain-computer-interfaces [Guger et al., 2011]. To combine with P300, the zero-shot learning model could be used to narrow word selections down to a small set of semantically similar choices, while the P300 signal could be used to accurately make a selection from the small set. We believe the combination of these two techniques may eventually lead to non-invasive *vocal prosthetics*. While it may not be possible to decode full sentences, recognition of even small vocabularies combined with speech synthesis capabilities may lead to useful applications for the disabled.

Other researchers have explored vocal prosthetics using invasive techniques that measure firings of small numbers of neurons directly [Brumberg et al., 2010]. Rather than considering word semantics, these researchers focus on predicting specific phonemes of the desired words. To our knowledge, exploring *invasive* techniques for classification of word semantics have not been explored, and we believe this would be an interesting area for further research. However, in our own work and also that of Mitchell et al. [2008], we've found the neural activity patterns of semantic processing to be highly distributed, making it difficult to study invasively using the tiny microarrays commonly used in invasive neural prosthetics [Schwartz, 2004].

For more machine learning oriented researchers, the zero-shot learning paradigm is a rich area for further study. We believe there are many applications of this model including computer vision and analogy solving. For example, in computer vision, it has recently been shown that it is possible to classify certain attributes of objects from images [Lampert et al., 2009, Farhadi et al., 2009]. Similar to our own work, these papers show that it is possible to classify objects with far fewer training examples, with Lampert et al. [2009] demonstrating the same zero-shot capability described in this thesis but in the vision domain.

With regards to analogy solving, we believe one way to solve analogies, such as those used in standardized tests like the SAT, is to model the relationships between words using crowdsourcing tools like Mechanical Turk to generate a semantic feature space that contains both physical and functional attributes of words. Solving an analogy might be reduced to computing

distances within this semantic space. Similar work using semantic features based on corpus statistics is described in Turney [2006, 2008], Veale [2004].

We also feel there are interesting theoretical challenges related to the zero-shot learning model, from proving performance guarantees under different modeling assumptions, to choosing optimal coding of semantic information. The choice of optimal semantic code is closely related to the problem of active learning, which seeks to choose the most useful training examples for the learning algorithm. This is particularly useful when training examples are difficult to acquire, which is certainly the case when collecting neural data from fMRI.

Given a particular semantic code, an active learning algorithm could minimize the number of training examples necessary to predict a particular set of classes. It could also be used to automatically explore different semantic codes by choosing examples to determine predictability of certain semantic features.

6.3.2 Semantic Feature Selection using Support Vector Ranking

While we presented one method for selecting semantic features based on support vector ranking, there are potentially many other metrics that may be useful for the zero-shot learning framework. The algorithm we presented chooses features that are both predictable, but also useful for discriminating between the various classes. While certainly useful for increasing performance when using small semantic feature sets, one negative aspect of this method is that it makes model interpretation more difficult because semantic features that may be predictable might not be useful for discriminating between classes.

As a result, when features are left out of the model, it is not immediately clear whether it is because they are not predictable, or whether they are just not useful for discriminating between the classes. It would be interesting to know whether other methods that rely solely on choosing the most predictable semantic features would perform as well. We suspect that unless the algorithm uses a small number of semantic features (e.g. less than 50), including all features that are predicted well may lead to sufficient performance.

In addition, we suspect our current algorithm may not work as well if semantic features are very highly correlated. This is because the learned weights may be spread across highly correlated features, and simply ranking the features by learned weight is likely not appropriate. We performed additional experiments to address this concern by including sparsity constraints during learning, but we measured no improvement over the method we described. While this was not problematic for our particular dataset, if may be necessary to adjust the algorithm for datasets with highly correlated features.

Further, we showed that our zero-shot learning model based on nearest neighbor demonstrated significant robustness to irrelevant features. We presented a theoretical argument that claimed that this robustness was a consequence of nearest neighbor being used in the second stage of the zero-shot learning algorithm. While our analysis only considered uniform distributions of semantic features under the ℓ_∞ metric, it would also be useful to study this robustness property based on additional metrics and distributions. It would be also useful to know if other implementations of a zero-shot learner that do not use nearest neighbor also have this robustness property.

6.3.3 Predicting Neural Activity using Multi-Task Learning

From a cognitive neuroscience perspective, we believe that learning methods that include sparsity constraints provide useful tools for neuroscientists to test and evaluate theories of cognition. By using fewer assumptions *a priori* and allowing the models to choose the most useful features automatically, sparse learning methods reduce the risk of implicit overfitting, and may lead to more believable results. For certain problems, we believe the joint task constraints will lead to more interpretable models than those that do not combine tasks, but more work needs to be done to compare models learned from single task data to those that combine data from multiple tasks. The recent work of Kolar and Xing [2010] is one step in this direction.

From a machine learning perspective, more multi-task learning methods should be evaluated. Since the original publication of our results in 2009, several other procedures have been published relating to multi-task learning using multi-task Lasso $\ell_{1,\infty}$ regularization norms [Quattoni et al., 2009] as well as $\ell_{1,2}$ norms [Vogt and Roth, 2010]. The work from [Bradley, 2010] provides a general technique for efficient optimization of different matrix norms in an online setting, however their approach can only approximately solve problems using the $\ell_{1,\infty}$ norm because of differentiability requirements of their current technique.

Little explicit comparison has been done between the different norms, except for Vogt and Roth [2010] and Bradley [2010] who empirically found that norms close to $\ell_{1,2}$ led to better predictors on several datasets. The work from Bradley [2010] goes further, and also includes a theoretical regret analysis for the online setting and claims that it is undesirable to use $\ell_{1,p}$ norms with $p > 2$ because of high regret, particularly when tasks are not highly related and feature weights vary across tasks. The benefit and risk of $\ell_{1,\infty}$ has also been analyzed recently by Negahban and Wainwright [2011], which confirms the benefit of this norm requires high relatedness of tasks.

Despite these works, it is not entirely clear from the literature under what conditions each norm should be used. In the single-task case, there are subtleties that depend on correlation of features, as well as whether true sparsity is required [Wainwright, 2009]. For example, many of these theoretical results guaranteeing recovery of the support set (i.e. the true model features) are possible using the Lasso ℓ_1 norm, but not the ridge regression ℓ_2 norm.

We also expect similar subtleties in the multi-task case, and expect that correlation of features, the number of examples, sparsity rates, as well as task relatedness will affect the appropriate choice of multi-task norm. From a computational standpoint, the $\ell_{1,2}$ regularization norm appears easier to solve at large scale given the current literature, and existing empirical and theoretical work supports using the $\ell_{1,2}$ as the first approach method as well, especially if all the tasks are not highly related as is likely in an empirical context. Consistent with the results of Bradley [2010], Negahban and Wainwright [2011], we believe that the $\ell_{1,2}$ norm will be more robust to violations of the task-relatedness assumption. In fact, in our own work we noticed that the $\ell_{1,\infty}$ norm tended to squash the learned coefficients between tasks to similar values. This might be a benefit if the tasks are truly similar. However, we suspect that the $\ell_{1,2}$ norm would allow more variation of a particular feature's coefficients across tasks, a useful benefit when the tasks are not very similar. Alternatively, $\ell_{1,\infty}$ norm might be used only for feature selection, while another model with less constraints could use the selected features to make the final prediction. In fact, this is the procedure we used in our experimental section where we used the multi-task Lasso to

select the features, and a ridge regression to learn the final predictive weights. We suspect this approach would lead to better performance than a strict $\ell_{1,\infty}$ model. However, additional work, both theoretical and empirical, would be very useful to confirm this conjecture and also to fully understand the subtle effects of multi-task norms.

6.3.4 Combining Data from Multiple Scanning Modalities

From a neural imaging and brain-computer-interface perspective, we believe that augmenting data from lower resolution scanners with data from high resolution scanners is an interesting direction for future work. Most of the work in this chapter only focused on features created by averaging a small interval of the time series data from 350-450ms for MEG/EEG, but other intervals should be considered as well. The kernelized ridge regression model presented in this chapter should easily accommodate significantly larger feature sets and it may be possible to test all time intervals simultaneously, allowing model regularization to automatically choose the most useful time points.

Although we showed CCA as a potentially useful latent variable model, there may be other latent variable models that are more effective at transferring knowledge between the different scanning modalities. One particular idea would be a CCA-like model that includes full time series data in one view (rather than just temporally averaged features), and non-time series data in another view. This would create a latent variable model that could potentially transfer information more effectively between two datasets, even when they have little temporal relationship.

From a machine learning perspective, we feel that CCA is an important model for transferring knowledge between learning tasks. Unlike other dimensionality reduction techniques like PCA, the CCA objective function learns latent variables that are invariant to linear transformations, allowing a great deal of flexibility when transferring knowledge between tasks. Although linear transformations are only one particular function class, we believe there may exist other, similar models to CCA that learn latent variables that are invariant to other transformations. Research in this direction may lead to a powerful class of models that can learn different types invariant features, leading to greater abstraction and transfer of knowledge between tasks.

Appendix A

Derivation of Update Rule for Support Vector Ranking

We now present the derivation of the support vector ranking update rule presented in Chapter 3. Our technique mirrors the procedures described in Ratliff et al. [2007] for structured prediction problems.

Suppose we have a knowledge base of semantic features for N words $\mathcal{K} = \{s_i^*\}_{1:N}$ where s_i^* is the "true" semantic vector for word i. Suppose we've trained our zero-shot learning model and get a prediction $\widehat{s}_i$ for word i from our first stage $\mathcal{S}(\cdot)$.

Let $\widehat{f}_i^* = (\widehat{s}_i - s_i^*)^2$ be the vector whose pth element is the square of the pth element of $(\widehat{s}_i - s_i^*)$, the vector of differences between the prediction and the true encoding in the knowledge base for word i. Similarly, let $\widehat{f}_i^j = (\widehat{s}_i - s_j^*)^2$ be the vector of squared distances when the prediction for i is compared with the true encoding for word j.

We can formulate our learning algorithm as a constrained optimization problem. We learn a vector of weights w that we'll use to compute a weighted sum of prediction errors:

Objective:

$$\arg\min_w \frac{1}{2}||w||^2$$

Subject to:

$$\begin{aligned} w^T \widehat{f}_i^* &\leq min_{j \neq i}[w^T \widehat{f}_i^j] - 1 \\ &\vdots \quad \text{repeat for each word } i \text{ in training set} \end{aligned}$$

Intuitively, the idea behind this formulation is that we want the smallest weight vector w, such that the weighted sum of squared-errors (a.k.a the distance) between the predicted features and true semantic encoding for a given word, $w^T \widehat{f}_i^*$, is less than that for any other word in the knowledge base of N examples, $min_{j \neq i}[w^T \widehat{f}_i^j]$. However, we want there to be some margin as well, so we subtract an arbitrary number (the number 1 is commonly chosen in the support vector machine literature as it only affects the relative magnitude of the weight vector). We repeat this set of constraints for every word in the training set. Further, we do not require this to be a hard constraint, so we add a slackness term ξ_i that must be greater than or equal to zero, to make it

easier to find a solution:

Objective:

$$\arg\min_{w} \frac{1}{2}||w||^2 + \lambda \sum_{i=1}^{N} \xi_i$$

Subject to:

$$\begin{aligned} w^T \widehat{f}_i^* &\leq min_{j \neq i}[w^T \widehat{f}_i^j] - 1 + \xi_i \\ &\vdots \quad \text{repeat for each word } i \text{ in training set} \\ \xi_i &\geq 0 \qquad \forall i, 1..N \end{aligned}$$

where $\lambda \geq 0$ controls our tolerance to margin violations through the slackness variables. Note that for each constraint:

$$w^T \widehat{f}_i^* - min_{j \neq i}[w^T \widehat{f}_i^j] + 1 \leq \xi_i$$

Since $\xi_i \geq 0$, the constraints are tight, meaning that if $w^T \widehat{f}_i^* - min_{j \neq i}[w^T \widehat{f}_i^j] + 1 \geq 0$, then at the optimal weight w:

$$w^T \widehat{f}_i^* - min_{j \neq i}[w^T \widehat{f}_i^j] + 1 = \xi_i$$

This is true because increasing ξ_i beyond $w^T \widehat{f}_i^* - min_{j \neq i}[w^T \widehat{f}_i^j] + 1$ would only increase the objective. Similarly, if $w^T \widehat{f}_i^* - min_{j \neq i}[w^T \widehat{f}_i^j] + 1 < 0$ then $\xi_i = 0$ as no slackness is needed because the margin is not violated. As a result, minimizing the constrained problem above is equivalent to minimizing the unconstrained problem:

$$\arg\min_{w} \frac{1}{2}||w||^2 + \lambda \sum_{i=1}^{N} \max\left[0, w^T \widehat{f}_i^* - min_{j \neq i}[w^T \widehat{f}_i^j] + 1\right] \tag{A.1}$$

To see this, note that if the margin is not violated, we can't make the objective smaller through ξ_i since it is already 0. In this case, we can only make it smaller by shrinking the weight vector w. If the margin is violated, then we have a tight constraint for the reasons stated above, and minimizing ξ_i with respect to w is the same as minimizing $\left[w^T \widehat{f}_i^* - min_{j \neq i}[w^T \widehat{f}_i^j] + 1\right]$ with respect to w.

We account for these scenarios using the *max* operator. Similar to the structure prediction problem described in Ratliff et al. [2007], we'll use stochastic sub-gradient descent to optimize (A.1) for w. The sub-gradient method deals with the discontinuity of the *max* operator, while the stochastic form uses one example at a time, rather than summing all examples at once. See Shor [1985], Ratliff et al. [2007] for a discussion of this optimization technique.

We will now compute the sub-gradient of (A.1) to determine the online update rule for w. We use the *ternary operator* "?" where *[(condition) ? A : B]* means *[if (condition) then A else B]*, to represent the conditions of the sub-gradient:

$$\begin{aligned} & \nabla\left[\frac{1}{2}||w||^2 + \lambda \cdot \max\left[0, w^T \widehat{f}_i^* - min_{j \neq i}[w^T \widehat{f}_i^j] + 1\right]\right] \\ = \; & w + \\ & \left[(w^T \widehat{f}_i^* - min_{j \neq i}[w^T \widehat{f}_i^j] < -1) \; ? \; 0 : \lambda(\widehat{f}_i^* - min_{j \neq i}[\widehat{f}_i^j])\right] \end{aligned}$$

Which then leads to the update rule with learning rate η and regularization parameter λ:

$$\begin{aligned} j &= argmin_{k \neq i}[w^T \widehat{f}_i^k] \\ w &\leftarrow w - \\ & \quad \eta\Big[w + \big[(w^T \widehat{f}_i^* - w^T \widehat{f}_i^j < -1) \; ? \; 0 : \lambda(\widehat{f}_i^* - \widehat{f}_i^j)\big]\Big] \end{aligned}$$

Appendix B

Correctness Proof of the Blockwise Coordinate Descent Algorithm for Solving the Multi-task Lasso

The correctness of the blockwise descent algorithm presented in Chapter 4 (Figure 4.4) was proved by Han Liu in our ICML paper [Liu et al., 2009a]. Although the proof is highly technical, we include it here for completeness.

Theorem 2. *Let $\alpha_j^{(k)}$ as defined in (4.4) and order the indices according to $|\alpha_j^{(k_1)}| \geq |\alpha_j^{(k_2)}| \geq \ldots \geq |\alpha_j^{(k_K)}|$. Then the solution to (4.3) is*

$$\widehat{\beta}_j^{(k_i)} = \frac{\operatorname{sign}(\alpha_j^{(k_i)})}{m^*}\left[\sum_{i'=1}^{m^*}|\alpha_j^{(k_{i'})}| - \lambda\right]_+ \cdot \mathbf{1}_{\{i \leq m^*\}} + \alpha_j^{(k_i)} \cdot \mathbf{1}_{\{i > m^*\}}$$

where $m^* = \arg\max_m \frac{1}{m}\left(\sum_{i'=1}^{m}|\alpha_j^{(k_{i'})}| - \lambda\right)$, $\mathbf{1}_{\{\cdot\}}$ is the indicator function, and $[\cdot]_+$ denotes the positive part.

Proof: The proof proceeds by discussing several cases separately: (i) All the elements in the sup-norm are zeros; (ii) One unique element in the sup-norm achieves the maximum; (iii) At least two elements in the sup-norm achieve the maximum. These cases correspond to Propositions 7, 8, and 10 respectively. Given these key points, the technical details can be safely ignored without affecting the flow. We put the whole technical details here to ease the review.

Since the given objective function in (4.3) is convex, its solution can be characterized by the Karush-Kuhn-Tucker conditions as the following

$$\left(R_j^{(k)} - \widehat{\beta}_j^{(k)} X_j^{(k)}\right)^T X_j^{(k)} = \lambda \eta_k \;\; \forall k \in \{1, \ldots, K\}, \tag{B.1}$$

where $\{\eta_k\}_{k=1}^K$ satisfy $\eta \equiv (\eta_1, \ldots, \eta_K)^T \in \partial\|\cdot\|_\infty\big|_{\beta_j}$. Here, $\partial\|\cdot\|_\infty\big|_{\beta_j}$ denotes the subdifferential of the convex functional $\|\cdot\|_\infty$ evaluated at β_j, it lies in a K-dimensional Euclidean space. Next, the following proposition from Rockafellar and Wets [1998] can be used to characterize the subdifferential of sup-norms.

Lemma 5. *The subdifferential of* $\|\cdot\|_\infty$ *in* $\mathbf{R}^K$ *is*

$$\partial\|\cdot\|_\infty\big|_x = \begin{cases} \{\eta : \|\eta\|_1 \le 1\} & x = \mathbf{0} \\ \mathrm{conv}\{\mathrm{sign}(x_i)e_i : |x_i| = \|x\|_\infty\} & \text{o.w.} \end{cases} \tag{B.2}$$

where $\mathrm{conv}(A)$ *denotes the convex hull, and* e_i *is the* i*-th canonical unit vector in* $\mathbf{R}^K$.

Proposition 6. *Consider the solutions* $\widehat{\beta}_j^{(k)}$ *to (4.3) and the corresponding* $\alpha_j^{(k)}$ *as defined in Theorem 2, if* $\widehat{\beta}_j^{(k)} \neq 0$*, then* $\mathrm{sign}(\widehat{\beta}_j^{(k)}) = \mathrm{sign}(\alpha_j^{(k)})$.

Proof to Proposition 6: Since $\widehat{\beta}_j^{(k)} \neq 0$, the result trivially follows from the convexity and continuity of the objective function in (4.3). □

Firstly, we consider the case that $\sum_{k=1}^K |\alpha_j^{(k)}| \le \lambda$ and show that $\mathbf{0}$ must be a solution.

Proposition 7. $\widehat{\beta}_j = \mathbf{0}$ *if and only if* $\sum_{k=1}^K |\alpha_j^{(k)}| \le \lambda$.

Proof to Proposition 7: From (B.1), we know that $\widehat{\beta}_j = \mathbf{0}$ if and only if $\exists \eta_1, \ldots, \eta_K$ such that $\sum_{k=1}^K |\eta_k| \le 1$ and

$$\eta_k = \frac{R_j^{(k)^T} X_j^{(k)}}{\lambda} = \frac{\alpha_j^{(k)}}{\lambda}. \tag{B.3}$$

If $\sum_{k=1}^K |\alpha_j^{(k)}| \le \lambda$, choosing η_k as in (B.3) would guarantee that $\sum_{k=1}^K |\eta_k| \le 1$, therefore $\widehat{\beta}_j = \mathbf{0}$.

On the other hand, If $\widehat{\beta}_j = \mathbf{0}$, from (B.3), we know that $\lambda\eta_k = \alpha_j^{(k)}$, $k = 1, \ldots, K$ and $\sum_{k=1}^K |\eta_k| \le 1$. This implies that $\sum_{k=1}^K |\alpha_j^{(k)}| \le \lambda$. □

Next, we consider the case that $\sum_{k=1}^K |\alpha_j^{(k)}| > \lambda$ and $|\alpha_j^{(k_1)}| - \lambda > |\alpha_j^{(k_2)}|$. Here we show that $\widehat{\beta}_j^{(k)} = \alpha_j^{(k)}$ for $\forall k \neq k_1$, while $\widehat{\beta}_j^{(k_1)} = \mathrm{sign}\left(\alpha_j^{(k_1)}\right)\left[|\alpha_j^{(k_1)}| - \lambda\right]$.

Proposition 8. $|\widehat{\beta}_j^{(k_1)}| > |\widehat{\beta}_j^{(k)}|$ *for* $\forall k \neq k_1$ *if and only if* $|\alpha_j^{(k_1)}| - \lambda > |\alpha_j^{(k_2)}|$.

Proof to Proposition 8: If $|\widehat{\beta}_j^{(k_1)}| > |\widehat{\beta}_j^{(k)}|$ for $\forall k \neq k_1$, this implies that $\partial\|\cdot\|_\infty\big|_{\beta_j} = e_{k_1}$, where e_{k_1} is the k_1-th canonical vector. Therefore, from (B.1),

$$\left(R_j^{(k)} - \widehat{\beta}_j^{(k)} X_j^{(k)}\right)^T X_j^{(k)} = \begin{cases} \lambda \mathrm{sign}(\widehat{\beta}_j^{(k_1)}) & \text{if } k = k_1 \\ 0 & \text{o.w.} \end{cases}$$

From the above we know $\widehat{\beta}_j^{(k_1)} = \alpha_j^{(k_1)} - \lambda\mathrm{sign}(\widehat{\beta}_j^{(k_1)})$ and $\widehat{\beta}_j^{(k)} = \alpha_j^{(k)}$ for $\forall k \neq k_1$. Combined with the fact $|\widehat{\beta}_j^{(k_1)}| > |\widehat{\beta}_j^{(k)}|$ for $\forall k \neq k_1$, we get

$$|\alpha_j^{(k_1)} - \lambda\mathrm{sign}(\widehat{\beta}_j^{(k_1)})| > |\alpha_j^{(k)}| \text{ for } \forall k \neq k_1.$$

From Proposition 6, we have $\mathrm{sign}(\alpha_j^{(k_1)}) = \mathrm{sign}(\widehat{\beta}_j^{(k_1)})$. Further, if $\widehat{\beta}_j^{(k_1)} > 0$, then $|\alpha_j^{(k_1)}| > \lambda$, we have $|\alpha_j^{(k_1)} - \lambda\mathrm{sign}(\alpha_j^{(k_1)})| = |\alpha_j^{(k_1)}| - \lambda$. Therefore, $|\alpha_j^{(k_1)}| - \lambda > |\alpha_j^{(k_2)}|$. This is also true for $\widehat{\beta}_j^{(k_1)} < 0$.

On the other hand, assuming $|\alpha_j^{(k_1)}| - \lambda > |\alpha_j^{(k_2)}|$ but for some $n > 1$, there exist

$$|\widehat{\beta}_j^{(k_1)}| = \ldots = |\widehat{\beta}_j^{(k_n)}| = \|\widehat{\beta}_j\|_\infty. \tag{B.4}$$

Then, by (B.2), there must exist two nonnegative numbers $a_1, a_2 \in [0, 1]$ and $a_1 + a_2 \leq 1$. From (B.1), we have

$$\begin{aligned}\left(R_j^{(k_1)} - \widehat{\beta}_j^{(k_1)} X_j^{(k_1)}\right)^T X_j^{(k_1)} &= \lambda a_1 \text{sign}(\widehat{\beta}_j^{(k_1)}) \\ \left(R_j^{(k_2)} - \widehat{\beta}_j^{(k_2)} X_j^{(k_2)}\right)^T X_j^{(k_2)} &= \lambda a_2 \text{sign}(\widehat{\beta}_j^{(k_2)}).\end{aligned}$$

From these two equations and (B.4), we get

$$|\alpha_j^{(k_1)} - \lambda a_1 \text{sign}(\widehat{\beta}_j^{(k_1)})| = |\alpha_j^{(k_2)} - \lambda a_2 \text{sign}(\widehat{\beta}_j^{(k_2)})|$$

By Proposition 6 and $|\alpha_j^{(k_1)}| > \lambda a_1$, we have $|\alpha_j^{(k_1)} - \lambda a_1 \text{sign}(\widehat{\beta}_j^{(k_1)})| = |\alpha_j^{(k_1)}| - \lambda a_1$. If $|\alpha_j^{(k_2)}| > \lambda a_2$, we get $|\alpha_j^{(k_1)}| - \lambda \left(\text{sign}(a_1 \widehat{\beta}_j^{(k_1)}) + a_2 \text{sign}(\widehat{\beta}_j^{(k_2)})\right) = |\alpha_j^{(k_2)}|$. Since $a_1 + a_2 \leq 1$, this obviously contradicts with the assumption that $|\alpha_j^{(k_1)}| - \lambda > |\alpha_j^{(k_2)}|$. The same result also hold for the case $|\alpha_j^{(k_2)}| \leq \lambda a_2$. □

Lastly, for the case $\sum_{k=1}^K |\alpha_j^{(k)}| > \lambda$ and $|\alpha_j^{(k_1)}| - \lambda \leq |\alpha_j^{(k_2)}|$. We start with an auxiliary proposition.

Proposition 9. *For $m > 1$, if there are precisely m entries $|\widehat{\beta}_j^{(k_1)}|, \ldots, |\widehat{\beta}_j^{(k_m)}|$ achieve $\|\widehat{\beta}_j\|_\infty > 0$, then*

$$\widehat{\beta}_j^{(k_i)} = \frac{\text{sign}(\alpha_j^{(k_i)})}{m} \left[\sum_{\ell=1}^m |\alpha_j^{(k_\ell)}| - \lambda\right] \quad \forall i = 1, \ldots, m.$$

Proof to Proposition 9: Since exactly m entries $|\widehat{\beta}_j^{(k_1)}|, \ldots, |\widehat{\beta}_j^{(k_m)}|$ achieve $\|\widehat{\beta}_j\|_\infty > 0$, by (B.2), there must exist m nonnegative numbers $a_1, \ldots, a_m$, such that $\sum_{\ell=1}^m a_\ell = 1$ and for each $\ell \in \{1, \ldots, m\}$

$$\left(R_j^{(k_\ell)} - \widehat{\beta}_j^{(k_\ell)} X_j^{(k_\ell)}\right)^T X_j^{(k_\ell)} = \lambda a_\ell \text{sign}(\widehat{\beta}_j^{(k_\ell)}).$$

Which can be re-written as

$$\alpha_j^{(k_\ell)} = \lambda a_\ell \text{sign}(\widehat{\beta}_j^{(k_\ell)}) + \widehat{\beta}_j^{(k_\ell)} \quad \forall \ell \in \{1, \ldots, m\}. \tag{B.5}$$

Using the fact that $|\widehat{\beta}_j^{(k_1)}| = \ldots = |\widehat{\beta}_j^{(k_m)}|$, summing over the absolute value of both sides of all the equations in (B.5), we obtain $\sum_{\ell=1}^m |\alpha_j^{(k_\ell)}| = \sum_{\ell=1}^m |\lambda a_\ell \text{sign}(\widehat{\beta}_j^{(k_\ell)}) + \widehat{\beta}_j^{(k_\ell)}|$. Since $|\lambda a_\ell \text{sign}(\widehat{\beta}_j^{(k_\ell)}) + \widehat{\beta}_j^{(k_\ell)}| = \lambda a_\ell + |\widehat{\beta}_j^{(k_\ell)}|$ and $\sum_{\ell=1}^m a_\ell = 1$, we have

$$|\widehat{\beta}_j^{(k_i)}| = \frac{1}{m} \left[\sum_{\ell=1}^m |\alpha_j^{(k_i)}| - \lambda\right]. \ \forall i \in \{1, \ldots, m\}. \tag{B.6}$$

Finally, by Proposition 6, we know that $\text{sign}(\alpha_j^{(k_i)}) = \text{sign}(\widehat{\beta}_j^{(k_i)})$ for $i = 1, \ldots, m$, therefore

$$\widehat{\beta}_j^{(k_i)} = \frac{\text{sign}(\alpha_j^{(k_i)})}{m} \left[\sum_{\ell=1}^{m} |\alpha_j^{(k_\ell)}| - \lambda \right] \quad i = 1, \ldots, m. \quad \square$$

To finish the proof of Theorem 2, we still need to describe the exact conditions that there are exactly $m > 1$ elements that achieve the sup-norm. This is given in the following Proposition 10. A similar result was also given in Fornasier and Rauhut [2008] under a more general linear inverse problem framework.

Proposition 10. *For $m > 1$, there exist precisely m entries $|\widehat{\beta}_j^{(k_1)}|, \ldots, |\widehat{\beta}_j^{(k_m)}|$ that achieve $\|\widehat{\beta}_j\|_\infty > 0$ if and only if $|\alpha_j^{(k_1)}| - \lambda \leq |\alpha_j^{(k_2)}|$ and $|\alpha_j^{(k_m)}| \geq \frac{1}{m-1} \left(\sum_{\ell=1}^{m-1} |\alpha_j^{(k_\ell)}| - \lambda \right)$ and $|\alpha_j^{(k_{m+1})}| < \frac{1}{m} \left(\sum_{\ell=1}^{m} |\alpha_j^{(k_\ell)}| - \lambda \right)$.*

Proof to Proposition 10: Assume exactly $m > 1$ entries $|\widehat{\beta}_j^{(k_1)}|, \ldots, |\widehat{\beta}_j^{(k_m)}|$ achieve $\|\widehat{\beta}_j\|_\infty > 0$, from Proposition 9, we know that for $i = 1, \ldots, m$,

$$\widehat{\beta}_j^{(k_i)} = \frac{\text{sign}(\alpha_j^{(k_i)})}{m} \left[\sum_{\ell=1}^{m} |\alpha_j^{(k_\ell)}| - \lambda \right]. \tag{B.7}$$

By (B.5) and Proposition 6, we have

$$a_\ell = \frac{\alpha_j^{(k_\ell)} - \widehat{\beta}_j^{(k_\ell)}}{\lambda \text{sign}(\widehat{\beta}_j^{(k_\ell)})} = \frac{|\alpha_j^{(k_\ell)}| - |\widehat{\beta}_j^{(k_\ell)}|}{\lambda} \quad \ell \in \{1, \ldots, m\}.$$

Plugging (B.7) into a_m, since $a_m \geq 0$, we get

$$|\alpha_j^{(k_m)}| \geq \frac{1}{m-1} \left(\sum_{\ell=1}^{m-1} |\alpha_j^{(k_\ell)}| - \lambda \right). \tag{B.8}$$

Further, since $|\widehat{\beta}_j^{(k_m)}| > |\widehat{\beta}_j^{(k_{m+1})}|$, the necessity follows from $|\alpha_j^{(k_{m+1})}| = |\widehat{\beta}_j^{(k_{m+1})}| < |\widehat{\beta}_j^{(k_m)}| = \frac{1}{m} \left[\sum_{\ell=1}^{m} |\alpha_j^{(k_\ell)}| - \lambda \right]$.

For the sufficiency, we assume that precisely $n \neq m$ entries $|\widehat{\beta}_j^{(k_1)}|, \ldots, |\widehat{\beta}_j^{(k_n)}|$ achieve $\|\widehat{\beta}_j\|_\infty > 0$, then follow exactly the same argument as the necessity part to obtain a contradiction. $\square$

To prove Theorem 2, we know from Proposition 10 there are precisely m^* entries in $\widehat{\beta}_j$ that achieve $\|\widehat{\beta}_j\|_\infty > 0$ if and only if $m^* = \arg\max_m \frac{1}{m} \left(\sum_{\ell=1}^{m} |\alpha_j^{(k_\ell)}| - \lambda \right)$. The result follows by combining Propositions 7 and 8.

Remark 11. We conducted experiments to quantitatively compare the performance of the blockwise coordinate descent algorithm with the Log-barrier interior-point method in a similar setting as in Friedman et al. [2007b]. Although we do not report the simulation details here, we found the blockwise coordinate descent algorithm to be significantly faster.

The complexity analysis of the algorithm is straightforward. Within the main loop, the most expensive step is sorting the K elements, which can be done in $O(K \log K)$. This makes the algorithm scalable to very large number of tasks. From the Winsorization operator, we do not need to update a block if $\sum_{k=1}^{K} |\alpha_j^{(k)}| \leq \lambda$, which happens frequently if the problem is sparse. This makes the algorithm scalable to very large number of features. Furthermore, since many quantities can be pre-calculated, the algorithm can be also applied to large sample datasets. The numerical convergence of this algorithm is summarized in the following theorem.

Theorem 3. *The solution sequence generated by the blockwise coordinate descent algorithm in Figure 4.4 is bounded and every cluster point is a solution of the multi-task Lasso defined in* (4.2).

Proof The optimization objective function in (4.2) is continuous on a compact level set, and is convex (but not strictly convex) and nondifferentiable. Furthermore, notice that the nondifferentiable part $\lambda \sum_{j=1}^{p} \|\beta_j\|_\infty$ is separable, i.e. it can be decomposed into a sum of individual functions, one for each block of variables. By Theorem 4.1 in Tseng [2001] we obtain the claimed results. □

Appendix C

The `Human218` Semantic Features

IS IT AN ANIMAL?	IS IT A BODY PART?
IS IT A BUILDING?	IS IT A BUILDING PART?
IS IT CLOTHING?	IS IT FURNITURE?
IS IT AN INSECT?	IS IT A KITCHEN ITEM?
IS IT MANMADE?	IS ONE MORE THAN ONE COLORED?
CAN YOU EAT IT?	IS IT A VEHICLE?
IS IT A PERSON?	IS IT A VEGETABLE / PLANT?
IS IT MADE OF METAL?	IS IT A FRUIT?
IS IT MADE OF PLASTIC?	IS PART OF IT MADE OF GLASS?
IS IT MADE OF WOOD?	IS IT SHINY?
CAN YOU SEE THROUGH IT?	IS IT COLORFUL?
DOES IT CHANGE COLOR?	IS IT A TOOL?
IS IT ALWAYS THE SAME COLOR(S)?	IS IT WHITE?
IS IT RED?	IS IT ORANGE?
IS IT FLESH-COLORED?	IS IT YELLOW?
IS IT GREEN?	IS IT BLUE?
IS IT SILVER?	IS IT BROWN?
IS IT BLACK?	IS IT CURVED?
IS IT STRAIGHT?	DOES IT HAVE A FRONT AND A BACK?
IS IT FLAT?	DOES IT HAVE A FLAT / STRAIGHT TOP?
IS IT LONG?	DOES IT HAVE FLAT / STRAIGHT SIDES?
IS TALLER THAN IT IS WIDE/LONG?	IS IT TAPERED?
IS IT POINTED / SHARP?	IS IT ROUND?
DOES IT HAVE CORNERS?	IS IT SYMMETRICAL?
IS IT HAIRY?	IS IT FUZZY?
IS IT CLEAR?	IS IT SMOOTH?
⋮	⋮

IS IT SLIPPERY?	CAN IT CHANGE SHAPE?
IS IT DENSE?	CAN IT STRETCH?
DOES IT HAVE MOVING PARTS?	IS IT FRAGILE?
DOES IT HAVE PARTS?	CAN IT BREAK?
DOES IT COME IN PAIRS?	DOES IT OPEN?
DOES IT COME IN A BUNCH/PACK?	IS IT ALIVE?
DOES IT LIVE IN GROUPS?	IS IT HOLLOW?
IS IT PART OF SOMETHING LARGER?	IS IT SOFT?
DOES IT CONTAIN SOMETHING ELSE?	IS IT HEAVY?
DOES IT HAVE INTERNAL STRUCTURE?	CAN IT BEND?
DOES IT HAVE A HARD INSIDE?	IS IT LIGHTWEIGHT?
DOES IT HAVE A HARD OUTER SHELL?	DOES IT GROW?
DOES IT HAVE AT LEAST ONE HOLE?	WAS IT EVER ALIVE?
IS IT A SPECIFIC GENDER?	IS IT MANUFACTURED?
IS IT SMALLER THAN A GOLFBALL?	WAS IT AROUND 100 YEARS AGO?
ARE THERE MANY VARIETIES OF IT?	DOES IT COME IN DIFFERENT SIZES?
WAS IT INVENTED?	IS IT BIGGER THAN A LOAF OF BREAD?
IS IT BIGGER THAN A MICROWAVE OVEN?	IS IT BIGGER THAN A BED?
IS IT BIGGER THAN A CAR?	IS IT BIGGER THAN A HOUSE?
IS IT TALLER THAN A PERSON?	DOES IT HAVE A TAIL?
DOES IT HAVE LEGS?	DOES IT HAVE FOUR LEGS?
DOES IT HAVE FEET?	DOES IT HAVE PAWS?
DOES IT HAVE CLAWS?	DOES IT HAVE HORNS / SPIKES?
DOES IT HAVE HOOVES?	DOES IT HAVE A FACE?
DOES IT HAVE A BACKBONE?	DOES IT HAVE WINGS?
DOES IT HAVE EARS?	DOES IT HAVE ROOTS?
DOES IT HAVE SEEDS?	DOES IT HAVE LEAVES?
DOES IT COME FROM A PLANT?	DOES IT HAVE FEATHERS?
DOES IT HAVE SOME SORT OF NOSE?	DOES IT HAVE A HARD NOSE/BEAK?
DOES IT CONTAIN LIQUID?	DOES IT HAVE WIRES OR A CORD?
DOES IT HAVE WRITING ON IT?	DOES IT HAVE WHEELS?
DOES IT MAKE A SOUND?	DOES IT MAKE A NICE SOUND?
DOES IT MAKE SOUND CONTINUOUSLY?	CAN IT RUN?
DOES IT ROLL?	IS ITS JOB TO MAKE SOUNDS?
IS IT FAST?	CAN IT FLY?
CAN IT JUMP?	CAN IT FLOAT?
CAN IT SWIM?	CAN IT DIG?
CAN IT CLIMB TREES?	CAN IT CAUSE YOU PAIN?
CAN IT BITE OR STING?	DOES IT STAND ON TWO LEGS?
IS IT WILD?	IS IT A HERBIVORE?
IS IT A PREDATOR?	IS IT WARM BLOODED?
⋮	⋮

IS IT A MAMMAL?	IS IT NOCTURNAL?
DOES IT LAY EGGS?	IS IT CONSCIOUS?
DOES IT HAVE FEELINGS?	IS IT SMART?
IS IT MECHANICAL?	IS IT ELECTRONIC?
DOES IT USE ELECTRICITY?	CAN IT KEEP YOU DRY?
DOES IT PROVIDE PROTECTION?	DOES IT PROVIDE SHADE?
DOES IT CAST A SHADOW?	DO YOU SEE IT DAILY?
IS IT HELPFUL?	DO YOU INTERACT WITH IT?
CAN YOU TOUCH IT?	WOULD YOU AVOID TOUCHING IT?
CAN YOU HOLD IT?	CAN YOU HOLD IT IN ONE HAND?
DO YOU HOLD IT TO USE IT?	CAN YOU PLAY IT?
CAN YOU PLAY WITH IT?	CAN YOU PET IT?
CAN YOU USE IT?	DO YOU USE IT DAILY?
CAN YOU USE IT UP?	DO YOU USE IT WHEN COOKING?
IS IT USED TO CARRY THINGS?	CAN YOU PICK IT UP?
CAN YOU CONTROL IT?	CAN YOU SIT ON IT?
CAN YOU RIDE ON/IN IT?	IS IT USED FOR TRANSPORTATION?
COULD YOU FIT INSIDE IT?	IS IT USED IN SPORTS?
DO YOU WEAR IT?	CAN IT BE WASHED?
IS IT COLD?	IS IT COOL?
IS IT WARM?	IS IT HOT?
IS IT UNHEALTHY?	IS IT HARD TO CATCH?
CAN YOU PEEL IT?	CAN YOU WALK ON IT?
CAN YOU SWITCH IT ON AND OFF?	CAN IT BE EASILY MOVED?
DO YOU DRINK FROM IT?	DOES IT GO IN YOUR MOUTH?
IS IT TASTY?	IS IT USED DURING MEALS?
DOES IT HAVE A STRONG SMELL?	DOES IT SMELL GOOD?
WOULD YOU FIND IT IN A LANDFILL?	IS IT RARE?
IS IT USUALLY OUTSIDE?	WOULD YOU FIND IT ON A FARM?
WOULD YOU FIND IT IN A SCHOOL?	WOULD YOU FIND IT IN A ZOO?
WOULD YOU FIND IT IN AN OFFICE?	WOULD YOU FIND IT IN A RESTAURANT?
WOULD YOU FIND IN THE BATHROOM?	WOULD YOU FIND IT IN A HOUSE?
WOULD YOU FIND IT NEAR A ROAD?	DOES IT SMELL BAD?
WOULD YOU FIND IT IN THE FOREST?	WOULD YOU FIND IT IN A GARDEN?
WOULD YOU FIND IT IN THE SKY?	DO YOU FIND IT IN SPACE?
DOES IT LIVE ABOVE GROUND?	DOES IT GET WET?
DOES IT LIVE IN WATER?	CAN IT LIVE OUT OF WATER?
DO YOU TAKE CARE OF IT?	DOES IT MAKE YOU HAPPY?
DO YOU LOVE IT?	WOULD YOU MISS IT IF IT WERE GONE?
IS IT SCARY?	IS IT DANGEROUS?
IS IT FRIENDLY?	IS IT USUALLY INSIDE?
CAN YOU BUY IT?	IS IT VALUABLE?

Bibliography

Alon, U. et al. Broad patterns of gene expression revealed by clustering of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences of the United States of America*, 96:6745–6750, 1999.

Ando, R. K. and Zhang, T. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 2005. 1.5.3, 4.1

Andoni, A., Croitoru, D., and Patrascu, M. Hardness of nearest neighbor under l-infinity. *Proceedings of Symposium on Foundations of Computer Science (FOCS'08)*, 2008. 3.5

Argyriou, A., Evgeniou, T., and Pontil, M. Convex multi-task feature learning. *Machine Learning*, 73:243–272, 2008. 1.5.3, 4.1

Avetisyan, A. I., Campbell, R., Gupta, I., Heath, M. T., Ko, S. Y., Ganger, G. R., Kozuch, M. A., O'Hallaron, D., Kunze, M., Kwan, T. T., Lai, K., Lyons, M., Milojicic, D. S., Lee, H. Y., Soh, Y. C., Ming, N. K., Luke, J.-Y., and Namgoong, H. Open cirrus: A global cloud computing testbed. *IEEE Computer*, April 2010. 2.3.3

Baillet, S., Mosher, J., and Leahy, R. Electromagnetic brain mapping. *IEEE Signal processing magazine*, 18(6):14–30, 2001. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=962275`. 1.4, 5.4.1

Bart, E. and Ullman, S. Cross-generalization: learning novel classes from a single example by feature replacement. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, June 2005. 1.5.2, 2.2.2

Baxter, J. A bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning*, 28:7–39, 1997.

Ben-David, S. and Schuller, R. Exploiting task relatedness for multiple task learning. In *Sixteenth Annual Conference on Learning Theory COLT*, 2003.

Bengio, Y., Lamblin, P., and Popovici, D. Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems 19*, 2007. 1.5.4

Benjamini, Y. and Hochberg, Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society, Series B (Methodological)*, 57(1):289300, 1995.

Birbaumer, N. and Cohen, L. Brain-computer interfaces: communication and restoration of movement in paralysis. *The Journal of Physiology*, 579(3):621, 2007. 1.5.1

Bishop, C. *Pattern Recognition and Machine Learning*. Springer, 2006. 1.5.4, 3.2.1

Blitzer, J., Foster, D., and Kakade, S. Zero-shot domain adaptation: A multi-view approach. *TTI-TR-2009-1*, 2009.

Boyd, S. and Vanderberghe, L. *Convex Optimization*. Cambridge University Press, 2004.

Bradley, D. M. *Learning in Modular Systems*. Carnegie Mellon Thesis: CMU-RI-TR-09-26, Pittsburgh, PA, 2010. 1.5.3, 4, 4.1, 4.3, 4.7, 6.3.3

Brumberg, J., Nieto-Castanon, A., Kennedy, P., and Guenther, F. Brain-computer interfaces for speech communication. *Speech Communication*, 2010. 2.4, 6.3.1

Caprihan, A., Pearlson, G., and Calhoun, V. Application of principal component analysis to distinguish patients with schizophrenia from healthy controls based on fractional anisotropy measurements. *Neuroimage*, Jan 2008. URL `http://linkinghub.elsevier.com/retrieve/pii/S1053811908005934`. 1.1, 1.5.1, 2.1

Caruana, R. Multitask learning. *Machine Learning*, 28(1):41–75, 1997. 1.5.3

Casella, G. and Berger, R. L. *Statistical Inference*. Duxbury, Pacific Grove, CA, 2002.

Chan, A. M., Halgren, E., Marinkovic, K., and Cash, S. S. Decoding word and category-specific spatiotemporal representations from meg and eeg. *NeuroImage*, October 2010. 2.1

Choi, K. and Cichocki, A. Control of a wheelchair by motor imagery in real time. *Intelligent Data Engineering and Automated Learning – IDEAL 2008*, pages 1–8, Sep 2008. 1.1

Ciaccia, P. and Patella, M. PAC nearest neighbor queries: Approximate and controlled search in high-dimensional and metric spaces. *Data Engineering, International Conference on*, page 244, 2000. 2.2.4, 3.5

Cox, D. and Savoy, R. Functional magnetic resonance imaging (fmri)"brain reading": detecting and classifying distributed patterns of fmri activity in human visual cortex. *Neuroimage*, 2003. 1.5.1

Dale, A., Liu, A., Fischl, B., Buckner, R., and Belliveau, J. Dynamic statistical parametric mapping:: Combining fmri and meg for high-resolution imaging of cortical activity. *Neuron*, Jan 2000. URL `http://linkinghub.elsevier.com/retrieve/pii/S0896627300811381`. 5

Davatzikos, C., Ruparel, K., Fan, Y., and Shen, D. Classifying spatial patterns of brain activity with machine learning methods: application to lie detection. *Neuroimage*, Jan 2005. URL `http://linkinghub.elsevier.com/retrieve/pii/S1053811905005914`. 1.5.1, 2.1

David, H. and Nagaraja, H. *Order Statistics*. Wiley, Hoboken, NJ, 2003.

Dietterich, T. G. and Bakiri, G. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 1995. 2.2.3, 2.3.7

Donchin, E., Spencer, K., and Wijesinghe, R. The mental prosthesis: Assessing the speed of a p300-based braincomputer interface. *IEEE Transactions on Rehabilitation Engineering*, 2000. 2.4, 6.3.1

Farhadi, A., Endres, I., Hoiem, D., and Forsyth, D. Describing objects by their attributes. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recogni-*

tion (CVPR), 2009. 1.5.2, 2, 2.2.2, 2.4, 6.3.1

Farhadi, A., Endres, I., and Hoiem, D. Attribute-centric recognition for cross-category generalization. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 1.5.2

Fei-Fei, L., Fergus, R., and Perona, P. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006. 2.2.1

Feller, W. *An Introduction to Probability Theory and Its Applications*. Wiley, New York, NY, 1957.

Fornasier, M. and Rauhut, H. Recovery algorithms for vector valued data with joint sparsity constraints. *SIAM Journal of Numerical Analysis*, 46:577–613, 2008. B

Frank, E. and Witten, I. H. Using a permutation test for attribute selection in decision trees. In *ICML '98: Proceedings of the Fifteenth International Conference on Machine Learning*, pages 152–160. Morgan Kaufmann Publishers Inc., 1998.

Friedman, J., Hastie, T., and Tibshirani, R. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, Dec 2007a. 1.5.3

Friedman, J., Hastie, T., Hüdotofling, H., and Tibshirani, R. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007b. 4, 4.2, 11

Friedman, J. H. On bias, variance, 0/1 loss, and the curse-of-dimensionality. *Data Mining and Knowledge Discovery*, 1(1):55–77, 1997.

Friedman, J. H., Hastie, T., and Tibshirani, R. Regularized paths for generalized linear models via coordinate descent. *Technical report, Stanford University*, 2008. 4.2, 4.4

Fu, W. J. Penalized regressions: The bridge versus the lasso. *Journal of Computational and Graphical Statistics*, 7:397–416, 1998. 4.2

Gelman, A., Carlin, J., Stern, H., and Rubin, D. *Bayesian Data Analysis*. Chapman and Hall/CRC Press, Boca Raton, NY, Second Edition/2003.

Guenther, F. et al. A wireless brain-machine interface for real-time speech synthesis. *PLoS ONE*, 2009.

Guger, C. et al. State-of-the-art in bci research: Bci award 2010. *Recent Advances in Brain-Computer Interface Systems*, 2011. URL `http://www.intechopen.com/articles/show/title/state-of-the-art-in-bci-research-bci-award-2010`. 1.4, 1.5.1, 2.4, 5.1, 6.3.1

Guyon, I. Kernel ridge regression. *Technical Note*, June 2005. 2.3.6

Guyon, I. and Elisseeff, A. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003. 1.5.3, 3.4

Hämäläinen, M., Hari, R., Ilmoniemi, R. J., Knuutila, J., and Lounasmaa, O. V. Magnetoencephalography—theory, instrumentation, and applications to noninvasive studies of the working human brain. *Reviews of Modern Physics*, 65(2):413–497, Apr 1993. 1.4

Hämäläinen, M. and Ilmoniemi, R. Interpreting magnetic fields of the brain: minimum norm estimates. *Medical and Biological Engineering and Computing*, 32(1):35–42, 1994. URL

http://www.springerlink.com/index/4474345W652H8581.pdf. 5.3.2, 5.4.1

Hansen, P. C., Kringelbach, M. L., and Salmelin, R., editors. *MEG: An Introduction to Methods*. Oxford, 2010. 1.4

Hardoon, D. R., Szedmak, S., and Shawe-Taylor, J. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12):2639–2664, 2004. doi: 10.1162/0899766042321814. URL http://www.mitpressjournals.org/doi/abs/10.1162/0899766042321814. 5.2, 5.7

Hastie, T., Tibshirani, R., and Friedman, J. H. *The Elements of Statistical Learning*. Springer, 2001. 2.3.6, 2.3.6, 4.5.2

Hayasaka, S., Du, A., Duarte, A., Kornak, J., Jahng, G., Weiner, M., and Schuff, N. A non-parametric approach for co-analysis of multi-modal brain imaging data: Application to alzheimer's disease. *Neuroimage*, Jan 2006. URL http://linkinghub.elsevier.com/retrieve/pii/S1053811905024134. 1.1, 1.5.1, 2.1

Haynes, J. and Rees, G. Decoding mental states from brain activity in humans. *Nature Reviews Neuroscience*, Jan 2006. URL http://www.nature.com/nrn/journal/v7/n7/abs/nrn1931.html. 1.5.1, 2.1

Herbrich, R., Graepel, T., and Obermayer, K. Support vector learning for ordinal regression. *Artificial Neural Networks*, 1999. 3, 3.4

Heskes, T. Solving a huge number of similar tasks: a combination of multi-task learning and a hierarchical bayesian approach. In *International Conference of Machine Learning ICML*, 1998.

Hinton, G., Osindero, S., and Teh, Y. A fast learning algorithm for deep belief nets. *Neural Computation*, 2006. 1.5.4

Hotelling, H. Relations between two sets of variates. *Biometrika*, 1936. 5.2

Huettel, S. A., Song, A. W., and McCarthy, G. *Functional Magnetic Resonance Imaging*. Sinauer, 2004. 1.3

Hutchinson, R. A. *Hidden Process Models*. Carnegie Mellon Thesis: CMU-CS-09-179, Pittsburgh, PA, 2009.

Hutchinson, R. A., Mitchell, T. M., and Rustandi, I. Hidden process models. In *International Conference of Machine Learning ICML*, 2006.

Imada, T., Zhang, Y., Cheour, M., Taulue, S., Ahonene, A., and Kuhl., P. Neuroreport. *Infant speech perception activates Brocas area: a developmental magnetoencephalography study*, 17(10), July 2006. 1.4

Jensen, D. and Cohen, P. Multiple comparisons in induction algorithms. *Machine Learning*, 38: 309–338, 2000.

Jordan, M. I. and Latham, D. Linear and ridge regression, and kernels. *Lecture Notes: Advanced Topics in Learning and Decision Making*, 2004. 2.3.6

Just, M. A., Cherkassky, V. L., Aryal, S., and Mitchell, T. M. A neurosemantic theory of concrete noun representation based on the underlying brain codes. *PLoS ONE*, Jan 2010. 2.1

Kaplan, J., Freedman, J., and Iacoboni, M. Us versus them: Political attitudes and party affiliation influence neural response to faces of presidential candidates. *Neuropsychologia*, Jan 2007. URL http://linkinghub.elsevier.com/retrieve/pii/S0028393206001606. 1.5.1, 2.1

Kay, K. N., Naselaris, T., Prenger, R. J., and Gallant, J. L. Identifying natural images from human brain activity. *Nature*, 452(7185):352–355, 03 2008. 1.5.1, 2.1

Kolar, M. and Xing, E. P. Ultra-high dimensional multiple output learning with simultaneous orthogonal matching pursuit: A sure screening approach. *AIStats*, 2010. 4, 4.1, 4.7, 6.3.3

Kutas, M. and Federmeier, K. D. Thirty years and counting: Finding meaning in the n400 component of the event-related brain potential (erp). *Annual Review of Psychology*, 2011. 5.6, 6.1.4

Lampert, C. H., Nickisch, H., and Harmeling, S. Learning to detect unseen object classes by between-class attribute transfer. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 1.5.2, 2, 2.2.2, 2.4, 6.3.1

Larochelle, H., Erhan, D., and Bengio, Y. Zero-data learning of new tasks. *AAAI Conference on Artificial Intelligence*, 2008. 1.5.2, 2.2.2

Lee, H., Battle, A., Raina, R., and Ng, A. Efficient sparse coding algorithms. *Advances in neural information processing systems*, 2007.

Lee, H., Ekanadham, C., and Ng, A. Sparse deep belief net model for visual area v2. *Advances in neural information processing systems*, 20, 2008. 1.5.4

Lee, H., Grosse, R., Ranganath, R., and Ng, A. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, 2009. 1.5.4

Lee, P. M. *Bayesian Statistics*. Hodder Arnold, London, UK, Third Edition/2004.

Li, W. and Grosse, I. Gene selection criterion for discriminant microarray data analysis based on extreme value distributions. In *RECOMB '03: Proceedings of the Seventh Annual International Conference on Research in Computational Molecular Biology*, pages 217–223, New York, NY, USA, 2003.

Lin, C., Ko, L., Chiou, J., Duann, J., Huang, R., Liang, S., Chiu, T., and Jung, T. Noninvasive neural prostheses using mobile and wireless eeg. *Proceedings of the IEEE*, 96(7):1167–1183, 2008. 1.1, 1.5.1

Liu, H., Palatucci, M., and Zhang, J. Blockwise coordinate descent procedures for the multi-task lasso, with applications to neural semantic basis discovery. *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009a. 4, 4.1, 4.2, 4.9, B

Liu, Z., Zhang, N., Chen, W., and He, B. Mapping the bilateral visual integration by eeg and fmri. *Neuroimage*, Jan 2009b. URL http://linkinghub.elsevier.com/retrieve/pii/S1053811909002687. 5

Mallows, C. L., editor. *The collected works of John W. Tukey. Volume VI: More mathematical, 1938–1984*. Wadsworth & Brooks/Cole, 1990. 1

Mitchell, T., Hutchinson, R., Just, M., and Niculescu, R. Classifying instantaneous cognitive states from fmri data. *American Medical Informatics Association Symposium*, October 2003. 1.5.1

Mitchell, T. et al. Predicting human brain activity associated with the meanings of nouns. *Science*, 320:1191–1195, 2008. (document), 1.5.1, 1.5.2, 1.6, 2.1, 2.3.1, 2.3.6, 2.3.7, 2.3.7, 2.7, 2.4, 4, 4.1, 4.1, 4.5.1, 4.5.2, 4.5.2, 4.6, 4.7, 4.6, 4.8, 6.1.3, 6.3.1

Mitchell, T. M. *Machine Learning*. McGraw-Hill, New York, 1997. 2.2.4, 2.2.4

Mitchell, T. M., Hutchinson, R., Niculescu, R. S., Pereira, F., Wang, X., Just, M., and Newman, S. Learning to decode cognitive states from brain images. *Machine Learning*, 57(1-2):145–175, 2004. 1.5.1

Negahban, S. and Wainwright, M. J. Simultaneous support recovery in high dimensions: Benefits and perils of block ℓ_1/ℓ_∞-regularization. *Preprint: To appear in IEEE Transactions on Information Theory*, 2011. 4.7, 6.3.3

Niculescu, R. S. *Exploiting Parameter Domain Knowledge for Learning in Bayesian Networks*. Carnegie Mellon Thesis: CMU-CS-05-147, Pittsburgh, PA, 2005.

Niculescu, R. S. and Mitchell, T. M. Bayesian network learning with parameter constraints. *Journal of Machine Learning Research*, 7:1357–1383, 2006.

Niedermeyer, E. and da Silva, F. L. *Electroencephalography: Basic Principles, Clinical Applications, and Related Fields*. Lippincott, Williams, Wilkins, 2004. ISBN 0781751268,9780781751261. 5.3.2

Norman, K., Polyn, S., Detre, G., and Haxby, J. Beyond mind-reading: multi-voxel pattern analysis of fmri data. *Trends in Cognitive Sciences*, Jan 2006. URL `http://linkinghub.elsevier.com/retrieve/pii/S1364661306001847`. 1.5.1, 2.1

Obozinski, G., Taskar, B., and Jordan, M. Multi-task feature selection. *ICML Workshop on Structural Knowledge Transfer for Machine Learning*, 2006. 1.5.3, 4.1

Owen, A. M., Coleman, M. R., Boly, M., Davis, M. H., Laureys, S., and Pickard, J. D. Detecting awareness in the vegetative state. *Science*, 313, 2006. 1.5.1

Palatucci, M. and Carlson, A. On the chance accuracies of large collections of classifiers. *Proceedings of the 25th Annual International Conference on Machine Learning (ICML)*, 2008. 1.5.1

Palatucci, M. and Mitchell, T. M. Classification in very high dimensional problems with handfuls of examples. *Proceedings of ECML/PKDD*, 2007. 1.5.1

Palatucci, M., Mitchell, T., and Liu, H. Discovering a semantic basis of neural activity using simultaneous sparse approximation. *Proceedings of the 25th Annual International Conference on Machine Learning (ICML) - Sparsity and Variable Selection Workshop*, 2008.

Palatucci, M., Pomerleau, D., Hinton, G., and Mitchell, T. M. Zero-shot learning with semantic output codes. *Advances in Neural Information Processing Systems (NIPS)*, 2009. 1.5.2, 2.2.1

Palatucci, M., Sudre, G., Pomerleau, D., Hinton, G., and Mitchell, T. M. Classifying concrete nouns from neural images. *CMU Technical Report*, 2011. 2.6, 3.8, 5.7

Parikh, D. and Grauman, K. Interactively building a discriminative vocabulary of nameable attributes. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 1.5.2

Pelphrey, K. A., Morris, J. P., Michelich, C. R., Allison, T., and McCarthy, G. Functional anatomy of biological motion perception in posterior temporal cortex: An fmri study of eye, mouth and hand movements. *Cerebral Cortex*, 2005. (document), 4.5.2, 4.8

Pereira, F., Mitchell, T., and Botvinick, M. Machine learning classifiers and fmri: a tutorial overview. *Neuroimage*, March 2009. 2.1

Plaut, D. C. Graded modality-specific specialization in semantics: A computational account of optic aphasia. *Cognitive Neuropsychology*, 19:603–639, 2002. 1.5.2, 2.2.2

Quattoni, A., Carreras, X., Collins, M., and Darrell, T. An efficient projection for l 1,infinity regularization. *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, 2009. 4.1, 4.7, 6.3.3

Raina, R., Battle, A., Lee, H., Packer, B., and Ng, A. Self-taught learning: Transfer learning from unlabeled data. *Proceedings of the 24th international conference on Machine learning*, 2007. 1.5.4

Raina, R., Madhavan, A., and Ng, A. Large-scale deep unsupervised learning using graphics processors. *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009.

Ratliff, N., Bagnell, J. A., and Zinkevich, M. A. (online) subgradient methods for structured prediction. *Proceedings of Artificial Intelligence and Statistics (AIStats)*, 2007. 3, 3.2.1, 3.4, A, A

Ravikumar, P., Liu, H., Lafferty, J., and Wasserman, L. Spam: sparse additive models. *Advances in neural information processing systems*, Jan 2008. 1.5.3

Rocha, G. and Yu, B. Greedy and relaxed approximations to model selection: a simulation study. *Tampere International Center for Signal Processing*, 2008.

Rockafellar, R. T. and Wets, R. J.-B. *Variational Analysis*. Springer-Verlag Inc, 1998. B

Rustandi, I., Just, M., and Mitchell, T. Integrating multiple-study multiple-subject fmri datasets using canonical correlation analysis. *Proceedings of the MICCAI 2009 Workshop: Statistical modeling and detection issues in intra- and inter-subject functional MRI data analysis*, September 2009. 1.5.4, 4, 5.1, 5.2

Rustandi, I. Hierarchical gaussian naive bayes classifier for multiple-subject fmri data. In *NIPS Workshop: New Directions on Decoding Mental States from fMRI Data*, 2006.

Rustandi, I. *Predictive fMRI Analysis for Multiple Subjects and Multiple Studies*. Carnegie Mellon Thesis: CMU-CS-10-117, Pittsburgh, PA, 2010. 5.1, 5.7

Sajda, P., Müller, K., and Shenoy, K. Brain-computer interfaces. *IEEE Signal Processing Magazine*, 2008. 1.1, 1.5.1

Salmelin, R. Clinical neurophysiology of language: The MEG approach. *Clinical Neurophysiology*, 2007. URL `http://linkinghub.elsevier.com/retrieve/pii/`

S1388245706013976. 5.3.3, 5.4.1, 5.6, 6.1.4

Schalk, G. Braincomputer symbiosis. *Journal of Neural Engineering*, 2008. 1.1

Schmah, T., Hinton, G., Zemel, R., Small, S., and Strother, S. Generative versus discriminative training of rbms for classification of fmri images. *Neural Information Processing Systems*, 2008. 1.5.1

Schwartz, A. Cortical neural prosthetics. *Annual Review of Neuroscience*, 2004. 2.4, 6.3.1

Settles, B., Craven, M., and Ray, S. Multiple-instance active learning. *Advances in neural information processing systems*, 2007.

Settles, B. Active learning literature survey. *Computer Science Technical Report*, pages 1–46, Jan 2009.

Shinkareva, S., Mason, R., Malave, V., Wang, W., Mitchell, T. M., and Just, M. A. Using fmri brain activation to identify cognitive states associated with perception of tools and dwellings. *PLoS ONE*, Jan 2008. 2.1

Shor, N. *Minimization Methods for Non-Differentiable Functions*. Springer, 1985. 3.2.1, 4.3, A

Skelly, L., Calhoun, V., Meda, S., and Kim, J. Diffusion tensor imaging in schizophrenia: Relationship to symptoms. *Schizophrenia research*, Jan 2007. 1.5.1

Snodgrass, J. and Vanderwart, M. A standardized set of 260 pictures: Norms for name agreement, image agreement, familiarity and visual complexity. *Journal of Experimental Psychology: Human Learning and Memory*, pages 174–215, 1980. 2.3.1, 2.3.7, 5.3.1

Swartz, B. and Goldensohn, E. Timeline of the history of eeg and associated fields. *Electroencephalography and clinical neurophysiology*, 1998. 1.4

Taulu, S., Kajola, M., and Simola, J. The signal space separation method. *Arxiv preprint physics/0401166*, 2004. URL http://arxiv.org/abs/physics/0401166. 5.3.2

Thrun, S. Is learning the n-th thing any easier than learning the first? *Advances in neural information processing systems*, 1996a. 1.5.3

Thrun, S. Learning to learn: Introduction. In *Learning To Learn*, 1996b.

Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58:267–288, 1996. 1.5.3, 4.1

Torralba, A. and Murphy, K. P. Sharing visual features for multiclass and multiview object detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(5), 2007. 1.5.2, 2.2.2

Tropp, J. A., Gilbert, A. C., and Strauss, M. J. Algorithms for simultaneous sparse approximation. part i: Greedy pursuit. *Signal Processing*, 86(3):572–588, 2006. 4, 4.1

Tropp, J. Algorithms for simultaneous sparse approximation. part ii: Convex relaxation. *Signal Processing*, 86(3):589–602, 2006. 1.5.3

Tseng, P. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of optimization theory and applications*, 109:475–494, 2001. B

Turlach, B., Venables, W. N., and Wright, S. J. Simultaneous variable selection. *Technometrics*, 27:349–363, 2005. 1.5.3, 4.1, 4.3

Turney, P. Similarity of semantic relations. *Computational Linguistics*, 2006. 2.4, 6.3.1

Turney, P. A uniform approach to analogies, synonyms, antonyms, and associations. *Proceedings of the 22nd International Conference on Computational Linguistics*, 2008. 2.4, 6.3.1

Uusitalo, M. a. and Ilmoniemi, R. J. Signal-space projection method for separating MEG or EEG into components. *Medical & biological engineering & computing*, 35(2):135–40, March 1997. ISSN 0140-0118. URL `http://www.ncbi.nlm.nih.gov/pubmed/9136207`.

Vaadia, E. and Birbaumer, N. Grand challenges of brain computer interfaces in the years to come. *Frontiers in Neuroscience*, 2009. 1.1, 1.5.1

Vaina, L. M., Solomon, J., Chowdhury, S., Sinha, P., and Belliveau, J. Proceedings of the national academy of sciences. *Cerebral Cortex*, 2001. (document), 4.5.2, 4.8

van der Maaten, L. and Hinton, G. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008. 2.3.7

Veale, T. Wordnet sits the sat: A knowledge-based approach to lexical analogy. *Proceedings of the 16th European Conference on Artificial Intelligence*, 2004. 2.4, 6.3.1

Vogt, J. E. and Roth, V. The group-lasso: L 1,infinity regularization versus l 1,2 regularization. *Pattern Recognition*, 2010. 4.7, 6.3.3

Waibel, A. Modular construction of time-delay neural networks for speech recognition. *Neural Computation*, pages 39–46, 1989. 1.5.2, 2.2.2

Wainwright, M. Information-theoretic limits on sparsity recovery in the high-dimensional and noisy setting. *IEEE Transactions on Information Theory*, 2009. 4.7, 6.3.3

Wang, G., Forsyth, D., and Hoiem, D. Comparative object similarity for improved recognition with few or no examples. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 1.5.2

Wasserman, L. *All of Statistics*. Springer, New York, NY, 2005.

Weiskopf, N., Scharnowski, F., Veit, R., Goebel, R., Birbaumer, N., and Mathiak, K. Self-regulation of local brain activity using real-time functional magnetic resonance imaging (fmri). *Journal of Physiology-Paris*, 98(4-6):357–373, 2004. 1.5.1

Wilson, M. The MRC psycholinguistic database: Machine readable dictionary, version 2. *Behavioral Research Methods*, pages 6–11, 1988. 2.3.7

Wong, W.-K., Moore, A., Cooper, G., and Wagner, M. Rule-based anomaly pattern detection for detecting disease outbreaks. In *Eighteenth national conference on Artificial intelligence*, pages 217–223. American Association for Artificial Intelligence, 2002.

Wu, T. T. and Lange, K. Coordinate descent algorithms for lasso penalized regression. *The Annals of Applied Statistics*, 2(1):224–244, 2008. 4.2

Zhang, J. A probabilistic framework for multitask learning. Technical Report CMU-LTI-06-006, Ph.D. thesis, Carnegie Mellon University, 2006. 4.1, 4.2, 4.3, 4.4

Zhang, L., Samaras, D., Tomasi, D., and Alia-Klein, N. Exploiting temporal information in functional magnetic resonance imaging brain data. *Lecture Notes in Computer Science*, Jan 2005. URL `http://www.springerlink.com/index/bh8144jhx8uj7048.pdf`.

1.5.1, 2.1

Zhao, P., Rocha, G., and Yu, B. The grouped and hierarchical model selection through composite absolute penalties. *The Annals of Statistics*, 2009. 4.3