Classification in Very High Dimensional Problems with Handfuls of Examples

Mark Palatucci and Tom M. Mitchell

School of Computer Science
Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, USA
[mpalatuc,tom.mitchell]@cs.cmu.edu

Abstract. Modern classification techniques perform well when the number of training examples exceed the number of features. If, however, the number of features greatly exceed the number of training examples, then these same techniques can fail. To address this problem, we present a hierarchical Bayesian framework that shares information between features by modeling similarities between their parameters. We believe this approach is applicable to many sparse, high dimensional problems and especially relevant to those with both spatial and temporal components. One such problem is fMRI time series, and we present a case study that shows how we can successfully classify in this domain with 80,000 original features and only 2 training examples per class.

1 Introduction

There are many interesting domains that have high dimensionality. Some examples include the stream of images produced from a video camera, the output of a sensor network with many nodes, or the time series of functional magnetic resonance images (fMRI) of the brain. Often we want use this high dimensional data as part of a classification task. For instance, we may want our sensor network to classify intruders from authorized personnel, or we may want to analyze a series of fMR images to determine the cognitive state of a human subject.

Unfortunately, for many of these high dimensional classification tasks, the number of available training examples is far fewer than the number of dimensions. Using regularization can certainly help, and classifiers like logistic regression with $\mathcal{L}_1$ penalized weights have been shown to scale to many thousands of dimensions. There are other techniques like PCA, ICA, and manifold learning that explicitly try to reduce the data dimension. These methods, however, are unlikely to help when the amount of training data is only a few examples per class.

For many of these sparse, high dimensional problems the features are not truly independent. This is easy to imagine for time series data as features may not change much from one time point to the next. If we assumed that our data were temporally continuous, we could imagine smoothing each feature by other features nearby in time. This smoothing could remove noise and improve our estimate of the feature.

Any assumption that we make *a priori* introduces inductive bias into our learning task. If the assumption is accurate then the bias will help the learning task when the number of training examples is very limited. Thus, to build a classifier that will perform well with small numbers of examples, we desire a way to incorporate any inductive biases (i.e. domain knowledge) we might have about the relationships between features.

We present such a classifier based on a hierarchical Bayesian model. Our model is both parametric and generative, and allows us to encode assumptions about the features *a priori.* We demonstrate this classifier on fMRI time series data and show that it scales tractably (even with 80,000 features). The classifier is robust to noise and extraneous features, and can classify with only 2 examples per class as compared to a standard Gaussian Naive Bayes classifier that fails completely on the same data.

1.1 Case Study: Cognitive State Classification using Functional Magnetic Resonance Images

Recent work has shown that it is possible to classify cognitive states from fMRI data. For example, researchers have been able to determine the category of words that a person is reading (e.g. fruits, buildings, tools, etc.) [10] by analyzing fMR images of their neural activity. Others have shown that is is possible to classify between drug addicted persons and non-drug using controls [15]. One study even used fMRI data to classify whether participants were lying or telling the truth [4].

Classification in this domain is tricky. The data are very high dimensional and noisy, and training examples are sparse. A typical experiment takes a 3D volumetric image of the brain every second. Each image has roughly 5,000 voxels[1], each of which measures the neural activity at a specific location in the brain[2].

The experiments considered here are often divided into trials, with each lasting approximately 60 seconds. A trial is repeated several times within an experiment to collect multiple samples of the subject's neural response to some stimulus. A classifier may treat each voxel-timepoint as a feature, and each trial would be one example of that voxel-timepoint. Thus, an experiment with V voxels, T images per trial, and N trials would have $V * T$ features, with only N examples per feature. A typical experiment may have $V = 5,000$, $T = 60$, and $N = 20$, yielding 300,000 features with only 20 training examples per feature (per class). With this much data and such few examples, it is amazing that classification is even possible.

Why reducing the number of examples for fMRI experiments is important
Although others have shown classification methods that work for this domain,

[1] The total number of voxels depends on the fMR scanner and particular subject.

[2] fMRI technically measures blood oxygenation level which is believed to be correlated with underlying neural activity.

even these methods fail as we further reduce the number of training examples (to say 2-3 examples per class). Human subjects can get fatigued after long periods in the fMR scanner, and any movements they make reduce the usability of their data. Reducing the number of trials needed for an experiment would improve participant comfort, and would allow the testing of more varied stimuli within a given allotment of time.

1.2 Related Work

Hierarchical Bayesian methods have been used for quite some time within the statistics community for combining data from similar experiments. A good introduction is given in [6]. In general, these methods fall under “shrinkage” estimators. If we want to estimate several quantities that we believe are related, then in the absence of large sample sizes for the individual quantities, these methods shrink (smooth) the estimate toward some statistic of the combined quantities. For example, if we want to compute the mean for each of several random variables, we could shrink the sample mean for each variable toward the sample mean over all the variables. If the samples sizes are small and the variables related, this can provide a better estimate of the individual means.

Shrinkage estimators are very similar in spirit to multi-task learning algorithms within the machine learning community. With multi-task[3] or “lifelong” learning, the goal is to leverage “related” information from similar tasks to help the current learning task [14]. The overall goal in both these communities is to learn concepts with fewer data. A good example of using hierarchical Bayes in a multi-task learning application is [7]. There has also been some interesting theoretical work to explain why these methods are beneficial [2, 1].

Hierarchical Bayesian methods have been applied successfully within the fMRI domain to the task of multiple subject classification. [13] demonstrates a hierarchical model that improves classification of a single human subject by combining data from multiple subjects within the same study. Our model, by contrast, focuses on sharing information between features of a single subject.

The most similar work to ours within the fMRI domain is [11, 12]. This work demonstrates that sharing parameters between voxels can lead to more accurate models of the fMRI data. Our work by comparison, does not directly couple the parameters of shared features, but rather shares information through hyperparameters.

2 Models

2.1 Gaussian Naive Bayes

The hierarchical model we describe below is based on the Gaussian Naive Bayes (GNB) classifier. The classifier is popular for fMRI classification tasks because it scales to thousands of features, and is robust to noise and irrelevant features. The model is based on Bayes rule:

$$\mathbb{P}(Y|X) \propto \mathbb{P}(X|Y)\mathbb{P}(Y)$$

where $X \in \Re^J$ represents the example and $Y \in \{0,1\}$ is the class label. We treat each component X_j of the vector X as a feature in the classifier where $1 \leq j \leq J$. The classifier makes the assumption that the X_j are independent given the class variable Y. We can then model the likelihood of the ith example for a feature j using a normal Gaussian:

$$X_{ij}|Y = c \sim N(\theta_j^{(c)}, \sigma_j^{2(c)}) \quad i = 1 \dots N$$

where N is the total number of examples. The joint likelihood then becomes the product over all the features:

$$\mathbb{P}\left(X|Y = c\right) = \prod_{j=1}^{J} \mathbb{P}\left(X_j|Y = c\right)$$

The classification rule is therefore:

$$\begin{aligned}\text{Predicted Class} &= \operatorname*{argmax}_c \mathbb{P}\left(Y = c\right) \prod_{j=1}^{J} \mathbb{P}\left(X_j|Y = c\right) \\ &= \operatorname*{argmax}_c \mathbb{P}\left(Y = c\right) \prod_{j=1}^{J} N(\widehat{\theta}_j^{(c)}, \widehat{\sigma}_j^{2(c)}) \qquad (1)\end{aligned}$$

Here, $\widehat{\theta}_j^{(c)}$ and $\widehat{\sigma}_j^{2(c)}$ are just the sample mean and variance for each feature j and class c, and the prior $\mathbb{P}\left(Y = c\right)$ is given simply by the relative class frequencies in the training data. Here $\delta(\cdot)$ is the indicator function:

$$\begin{aligned}\widehat{\theta}_j^{(c)} &= \frac{1}{\sum_{i=1}^{N} \delta(Y_i = c)} \sum_{i=1}^{N} \delta(Y_i = c) X_{ij} \\ \hat{\sigma}_j^{2(c)} &= \frac{1}{\sum_{i=1}^{N} \delta(Y_i = c)} \sum_{i=1}^{N} \delta(Y_i = c)(X_{ij} - \widehat{\theta}_j^{(c)})^2 \\ \mathbb{P}\left(Y = c\right) &= \frac{1}{N} \sum_{i=1}^{N} \delta(Y_i = c)\end{aligned}$$

2.2 Standard Hierarchical Bayesian Model

If we believe that the individual θ_j are related by some distribution, then we can incorporate that belief using a hierarchical model. For example, if we thought that the θ_j were all drawn from a common normal distribution, then we could model that as:

$$\begin{aligned}X_{ij}|\theta_j &\sim N(\theta_j, \sigma^2) \\ \theta_j &\sim N(\mu, \tau^2)\end{aligned}$$

Here μ and τ^2 are called *hyperparameters* for the model. (Note that for notational simplicity, we'll leave out mention of the class c until we return to the subject of classification.) Now in this hierarchical model, we want to know the posterior distribution of $\theta_j|\mu, \tau^2, X$. Intuitively, we want to know our best estimate of θ_j given not only the data, but also our prior belief about its distribution. If we assume for the moment the sampling variance is common across features, that is $\forall j, \sigma_j^2 = \sigma^2$, then we can obtain the MAP estimate of θ_j as:

$$\widehat{\theta}_j = \frac{\frac{N}{\sigma^2}\bar{X}_{\bullet j} + \frac{1}{\tau^2}\mu}{\frac{N}{\sigma^2} + \frac{1}{\tau^2}} \tag{2}$$

Equation (2) is surprisingly intuitive. It is just the weighted average of the sample mean $\bar{X}_{\bullet j} = \frac{1}{N}\sum_{i=1}^{N} X_{ij}$ and the hypermean μ, where N is the sample size. The weights are given by the inverse of the respective variances since σ^2/N is the variance of the sample mean. If the number of samples N is large, then we see more weight being placed on the sample mean. Similarly, if the number of samples is small, the variance of the sample mean may be larger than that of the hypermean. More weight would be placed on the hypermean. The beauty of this estimator is that it automatically balances the estimate with the number of available samples. As N grows large, the weight on the hypermean grows smaller.

Of course there are a few difficulties that we must address. One problem is that usually we do not know the variance σ^2. This quantity must somehow be estimated from the data. Another problem is how to choose the hyperparameters μ and τ^2. We could perform a fully Bayesian approach and apply a non-informative hyperprior distribution to μ and τ^2. This would then require simulation to calculate the posterior for θ_j. Another, more tractable approach is to estimate the hyperparameters directly from the data. This technique is often called *empirical Bayes*[9] and uses point estimates for the hyperparameters:

$$\widehat{\mu} = \frac{1}{J}\sum_{j=1}^{J} \bar{X}_{\bullet j} \qquad \widehat{\tau}^2 = \frac{1}{J}\sum_{j=1}^{J} (\bar{X}_{\bullet j} - \widehat{\mu})^2$$

Here we are just taking the sample mean and variance for all the individual sample means. We use a similar empirical approach in the method we now describe.

2.3 Feature Sharing Empirical Bayesian Model

One problem with the standard hierarchical model is the assumption that all the parameters θ_j are drawn from the same distribution. To demonstrate this, consider two variables that are perfectly correlated while the parameters that characterize their distributions are wildly different. Assuming the parameters for these two variables are drawn from a common normal distribution would lead to poor estimates of the hyperparameters μ and τ^2 and subsequently the smoothed parameter θ_j. Nonetheless, the variables certainly contain information about each other that we want to leverage when making an estimate about either one.

We address this problem by allowing each θ_j to be drawn from a different distribution. We propose an approach that uses the parameters of other related variables, say θ_k and θ_i, to estimate the hyperparameters of the distribution for θ_j. We define this formally as follows: assume we have two random variables, X and Y, parameterized by θ_X and θ_Y. Let $m_{X\to Y}(\theta_X)$ be a *parameter transformation* function that maps parameters of variable X to those of variable Y. Let $\mathcal{G}_j$ be the index set of all other variables that we believe contain information about variable j. Let $G_j = |\mathcal{G}_j|$ be the number of variables in that set.

We define a new smoothing estimator based on the normal model in Equation (2). Rather than assume all θ_j come from a common distribution, we assume that each θ_j has its own variance and hyperparameters μ_j and τ_j.

$$\widehat{\theta}_j = \frac{\frac{N}{\widehat{\sigma}_j^2}\bar{X}_{\bullet j} + \frac{1}{\widehat{\tau}_j^2}\widehat{\mu}_j}{\frac{N}{\widehat{\sigma}_j^2} + \frac{1}{\widehat{\tau}_j^2}} \tag{3}$$

These hyperparameters are calculated by point estimates of the transformed parameters of the variables in $\mathcal{G}_j$:

$$\widehat{\mu}_j = \frac{1}{G_j}\sum_{g=1}^{G_j} m_{g\to j}(\bar{X}_{\bullet g}) \tag{4}$$

$$\widehat{\tau}_j^2 = \frac{1}{G_j}\sum_{g=1}^{G_j} \Big(m_{g\to j}(\bar{X}_{\bullet g}) - \widehat{\mu}_j\Big)^2 \tag{5}$$

Intuitively, we first compute estimates of the variable j's parameters from the other variables, and use these to estimate the hyperparameter μ_j. We then smooth the sample mean using this hypermean as before.

Note that we still need estimates for the variances σ_j^2. Let $m'_{g\to i}(\cdot)$ be the parameter transformation function for the variance parameters. We could take the mean of these transformed parameters as before:

$$\widehat{\sigma}_j^2 = \frac{1}{G_j}\sum_{g=1}^{G_j} m'_{g\to j}(S_g^2) \tag{6}$$

where S_g^2 is sample variance for feature g. Empirically, we have found that pooling the variance parameters together $m'_{g\to i}(\sigma_g^2) = \sigma_g^2$ and taking the median (vs. mean) gives a estimator that is robust to extremely noisy variables:

$$\forall j, \widehat{\sigma}_j^2 = \text{median}\{S_1^2, S_2^2, \ldots, S_{G_j}^2\} \tag{7}$$

Given sets of sharing groups and parameter transformation functions, we can define a feature sharing classifier using the new estimators defined in Equations (3),(4),(5), and (7). The classifier is still based on the Gaussian Naive Bayes rule defined in Equation (1). Only now, for each class c we replace the estimate for $\widehat{\theta}_j$ with that from Equation (3) and $\widehat{\sigma}_j^2$ with either Equation (6) or Equation (7).

3 Case Study of Feature Sharing with fMRI Data

We now demonstrate this feature sharing model on a real fMRI classification task. We first show how to formulate the problem into the feature sharing framework described above, and then compare the feature sharing classifier against a standard Gaussian Naive Bayes classifier for the same task.

3.1 Notation

Since fMRI data are a time series we consider each voxel-timepoint as a feature. We index a particular example for a feature as X_{ivt} where i is the trial(example), v is the voxel, and t is the timepoint. The sample mean for a particular feature would then be $\bar{X}_{\bullet vt} = \frac{1}{N}\sum_{i=1}^{N} X_{ivt}$ (where N is the number of trials) and the sample variance would be $S_{vt}^2 = \frac{1}{N-1}\sum_{i=1}^{N} (X_{ivt} - \bar{X}_{\bullet vt})^2$.

3.2 Feature Sharing Empirical Bayesian Model for fMRI

There are two important questions we need to answer to formulate this problem into the feature sharing framework:

1. **Which of the features are related?** Specifically, for a feature j, what is the index set $\mathcal{G}_j$ of features that share information (parameters)?
2. **How are the features related?** Specifically, what are the *parameter transformation* functions $m_{k\to j}(\cdot)$ that map the parameters from feature k to feature j?

To answer these questions for the fMRI domain we consider a key observation made in [11]: *the time courses for neighboring voxels are often similar up to a scaling factor*. We can see this effect by observing several correlated neighborhoods (4-5 voxels) in Figure 1. We use this domain knowledge to define a feature sharing scheme for fMRI:

1. For feature j, let the index set of shared features $\mathcal{G}_j$ be the immediate spatial neighbors of a voxel. Since a voxel is indexed by integer {x,y,z} locations, there can be a maximum of 26 neighbors per voxel.
2. We define $m_{k\to j}(\cdot)$ to be the mean *parameter transformation* function from feature k to feature j. We define the function as a linear scaling factor $m_{k\to j}(\bar{X}_{\bullet k}) = \beta_{k\to j}\bar{X}_{\bullet k}$. We must remember, however, that each voxel-timepoint is a feature. To simplify, we'll assume that the parameter transformation function is the same for each pair of voxels, regardless of the timepoint. Therefore, for voxels j and k at any time t we have $m_{kt\to jt}(\bar{X}_{\bullet kt}) = \beta_{k\to j}\bar{X}_{\bullet kt}$. We also define the variance parameter transformation $m'_{kt\to jt}(\cdot)$ to be the median pooling estimator described in Equation (7)[3].

[3] We have found empirically that for the variance parameter it is advantageous to share over all the voxel-timepoints rather than just the immediate neighbors

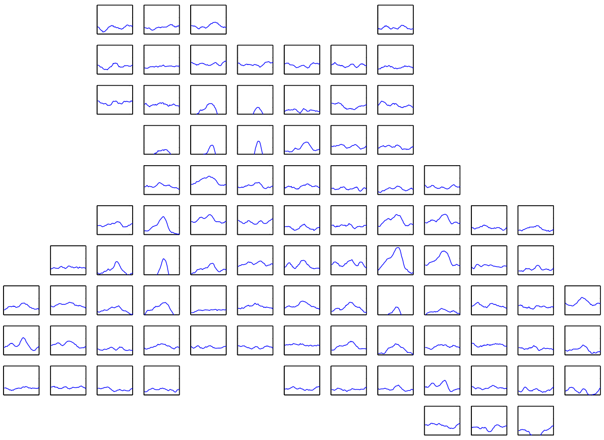

Fig. 1. Time series of the voxels in the visual cortex averaged over all trials. We see that several local neighborhoods (4-5 voxels) are similar but have different amplitudes.

We can solve for the $\beta_{k\to j}$ constants by assuming a linear regression model:

$$\begin{aligned}\bar{X}_{\bullet jt} &= \beta_{k\to j}\bar{X}_{\bullet kt} + \epsilon \\ \widehat{\bar{X}}_{\bullet jt} &= \widehat{\beta}_{k\to j}\bar{X}_{\bullet kt}\end{aligned}$$

This allows us to find estimates $\widehat{\beta}_{k\to j}$ using the usual method of least squares:

$$\widehat{\beta}_{k\to j} = \min_{\beta}\sum_{t=1}^{T}(\bar{X}_{\bullet jt} - \beta\bar{X}_{\bullet kt})^2$$

which is given by:

$$\widehat{\beta}_{k\to j} = \frac{\sum_{t=1}^{T}\bar{X}_{\bullet jt}\bar{X}_{\bullet kt}}{\sum_{t=1}^{T}\bar{X}_{\bullet kt}^2} \tag{8}$$

Now that we have our sharing groups and parameter transformation functions we can define a hierarchical model for fMRI:

$$\begin{aligned}X_{ivt}|\theta_{vt} &\sim N(\theta_{vt}, \sigma^2) \\ \theta_{vt} &\sim N(\mu_{vt}, \tau_{vt}^2)\end{aligned}$$

Combining all these equations together, we can now define a feature sharing classifier for fMRI:

A Feature Sharing Classifier for fMRI:

For each class c, compute:

1. $\widehat{\sigma}^{2(c)} = \text{median}(S_{11}^{2(c)}, \ldots, S_{1T}^{2(c)}, S_{21}^{2(c)}, \ldots, S_{2T}^{2(c)}, \ldots, S_{VT}^{2(c)})$
2. $\widehat{\beta}_{k \to j}^{(c)} = \frac{\sum_{t=1}^{T} \bar{X}_{\bullet jt}^{(c)} \bar{X}_{\bullet kt}^{(c)}}{\sum_{t=1}^{T} \bar{X}_{\bullet kt}^{2(c)}}$ For any pairs of voxels j, k that share features
3. $\widehat{\mu}_{vt}^{(c)} = \frac{1}{G_{vt}} \sum_{j=1}^{G_{vt}} \widehat{\beta}_{j \to v}^{(c)} \bar{X}_{\bullet jt}^{(c)} \qquad \widehat{\tau}_{vt}^{2(c)} = \frac{1}{G_{vt}} \sum_{j=1}^{G_{vt}} (\widehat{\beta}_{j \to v}^{(c)} \bar{X}_{\bullet jt}^{(c)} - \widehat{\mu}_{vt}^{(c)})^2$
4. $\widehat{\theta}_{vt}^{(c)} = \frac{\frac{N^{(c)}}{\widehat{\sigma}^{2(c)}} \bar{X}_{\bullet vt}^{(c)} + \frac{1}{\widehat{\tau}_{vt}^{2(c)}} \widehat{\mu}_{vt}^{(c)}}{\frac{N^{(c)}}{\widehat{\sigma}^{2(c)}} + \frac{1}{\widehat{\tau}_{vt}^{2(c)}}}$

The predicted class is then:

$$\underset{c}{\text{argmax}}\, \mathbb{P}\,(Y = c) \prod_{v=1}^{V} \prod_{t=1}^{T} N(\widehat{\theta}_{vt}^{(c)}, \widehat{\sigma}^{2(c)})$$

3.3 Experimental Results

Classification Task

We consider the task of classifying whether a subject in an fMRI experiment is "viewing a picture" or "reading a sentence". In this fMRI dataset[4], functional images of the brain were taken every 500ms (for 8 seconds). Each image recorded the neural activity at approximately 5,000 different locations (voxels) in the brain. We consider each *voxel-timepoint* as a feature, thus there were approximately $5,000 * 16 = 80,000$ features per trial. There were 20 "viewing a picture" trials and 20 "reading a sentence" trials. This experiment was repeated for 13 different human subjects.

Test Method

We performed the following testing method to estimate the error of the classifiers:

1. Split the dataset randomly in half. One half is used for training and one half is used for testing. We enforce an equal number of examples per class. Therefore, our training and test sets each have 20 examples total (10 per class).
2. Sample, at random, 2 examples per class from the training set. These are the training examples for this round.
3. Train on the sampled training examples in (2) and test on all examples in the test set.
4. Repeat 1-3 ten times and report the average error.

[4] The dataset used is available at: http://www.cs.cmu.edu/afs/cs/project/theo-73/www/index.html

Discussion

In Figure 2, we show the results of the Feature Sharing classifier compared to a standard Gaussian Naive Bayes classifier for the 13 human subjects available in this study. In this experiment we used all available voxels in the brain ($\approx 5,000$ per subject) yielding $\approx 80,000$ features. Notice that there were *only two training examples per class*. The standard Gaussian Naive Bayes (GNB) classifier performed with near random accuracy for all subjects. The Feature Sharing classifier we described above shows considerable improvement, demonstrating that it is possible to classify even with an extremely small number of training examples.

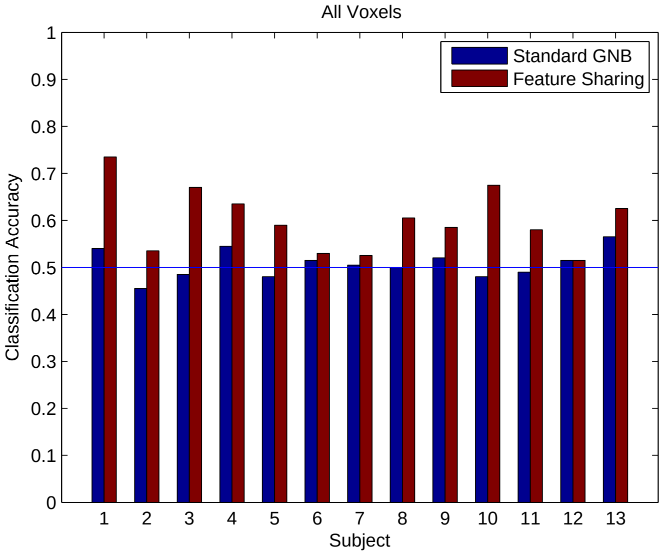

Fig. 2. Accuracies of the standard Gaussian Naive Bayes classifier and the Feature Sharing classifier for 13 human subjects with two training examples per class. The classifier uses all voxels in the brain. Since there are two classes, random accuracy is 0.5.

In Figure 3, we show the results of the same experiment, except now we use only the voxels located in the visual cortex of the brain ($\approx$300 per subject). These voxels are known to contain highly discriminating signal for this particular classification task. The interesting thing to note here is that the standard GNB classifier still fails with random accuracy on all subjects. The Feature Sharing classifier, however, is able to capitalize on the extra signal in these voxels, showing dramatic improvements for many subjects.

In the Feature Sharing classifier, we achieved the best results by sharing both the mean and variance parameters between features. We have found empirically,

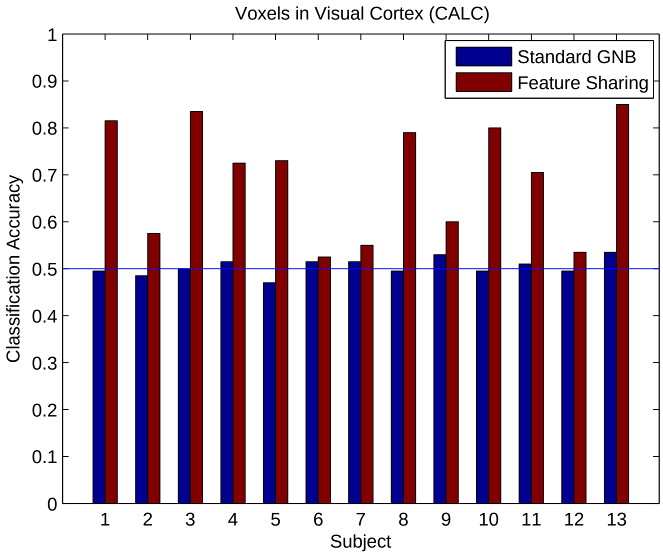

Fig. 3. Accuracies of the standard Gaussian Naive Bayes classifier and the Feature Sharing classifier for 13 human subjects with two training examples per class. The classifier uses only voxels in the Visual Cortex (CALC).

however, that sharing the variance parameter plays the larger role in improving overall classification accuracy. While this might seem surprising at first, some interesting theoretical work [5] shows that in the bias/variance decomposition under 0/1 loss, the variance dominates the bias. This may suggest why sharing the variance parameters caused the larger increase in performance.

4 Conclusion and Future Work

We have shown a feature sharing framework for classifying in very high dimensional problems with only a small number of training examples. This classifier is based on empirical Bayes and allows us to model relationships between features by assuming each class conditional parameter has its own hyperdistribution. The parameters for these hyperdistributions are estimated by sharing information between related groups of features.

We demonstrated this model on a fMRI classification task and showed how we can successfully classify in a problem with 80,000 spatially and temporally related features and only two training examples per class. We used domain knowledge of fMRI to specify feature sharing over local neighborhoods with a linear scaling factor.

An interesting future direction would be to automatically determine groups of features that share information rather than defining each group by the set of immediate neighbors. We could imagine learning a metric between features directly from the data, and then using that metric to define the parameter transformation functions.

5 Acknowledgements

We would like to thank Indra Rustandi and Francisco Pereira for useful discussions.

Mark Palatucci is supported by a NSF Graduate Research Fellowship and by a grant from the W.M. Keck Foundation.

References

1. J. Baxter. A bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning*, 28:7–39, 1997.
2. S. Ben-David and R. Schuller. Exploiting task relatedness for multiple task learning. In *Sixteenth Annual Conference on Learning Theory COLT*, 2003.
3. R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.
4. C. Davatzikos and et al. Classifying spatial patterns of brain activity with machine learning methods: application to lie detection. *Neuroimage*, 28(1):663–668, 2005.
5. J. H. Friedman. On bias, variance, 0/1 loss, and the curse-of-dimensionality. *Data Mining and Knowledge Discovery*, 1(1):55–77, 1997.
6. A. Gelman, J. Carlin, H. Stern, and D. Rubin. *Bayesian Data Analysis.* Chapman and Hall/CRC Press, Boca Raton, NY, Second Edition/2003.
7. T. Heskes. 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.
8. R. A. Hutchinson, T. M. Mitchell, and I. Rustandi. Hidden process models. In *International Conference of Machine Learning ICML*, 2006.
9. P. M. Lee. *Bayesian Statistics.* Hodder Arnold, London, UK, Third Edition/2004.
10. T. M. Mitchell, R. Hutchinson, R. S. Niculescu, F. Pereira, X. Wang, M. Just, and S. Newman. Learning to decode cognitive states from brain images. *Machine Learning*, 57(1-2):145–175, 2004.
11. R. S. Niculescu. *Exploiting Parameter Domain Knowledge for Learning in Bayesian Networks.* Carnegie Mellon Thesis: CMU-CS-05-147, Pittsburgh, PA, 2005.
12. R. S. Niculescu and T. M. Mitchell. Bayesian network learning with parameter constraints. *Journal of Machine Learning Research*, 7:1357–1383, 2006.
13. I. Rustandi. Hierarchical gaussian naive bayes classifier for multiple-subject fmri data. In *NIPS Workshop: New Directions on Decoding Mental States from fMRI Data*, 2006.
14. S. Thrun. Learning to learn: Introduction. In *Learning To Learn*, 1996.
15. L. Zhang, D. Samaras, D. Tomasi, N. Alia-Klein, L. C. A. Leskovjan, N. Volkow, and R. Goldstein. Exploiting temporal information in functional magnetic resonance imaging brain data. In *MICCAI Conference Proceedings*, pages 679–687, 2005.