Classification of brain imaging data

Charles Jackson
Department of Biomedical Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
charlesjackson@cmu.edu

Abstract

This report presents a technique to enhance existing methods of decoding cognitive states from brain imaging data. The concept is to take the set of voxels in the image and reduce the resolution in each dimension by an optimal factor as determined in the training process. Additionally, the set of voxels are split into smaller subsets of voxels that are localized in time and space. By training classifiers separately on each subset of voxels, and merging the outputs of these classifiers to form a final decision, I have demonstrated a significant improvement over the case where the entire set of voxels is used in a single classifier and at its orginal resolution.

1 Introduction

The goal of this project is to use fMRI images of brain activation to determine whether a subject is reading a sentence or viewing a picture. These images are taken in 3D and at multiple time intervals during which the subject is performing the task. This problem is highly relevant to help provide insight into the nature of cognitive processes in the brain.

This project attempts to enhance existing work by considering the effects of localization and resolution in both time and space. Specifically, I divide the images into distinct regions in space and time and look at each region separately. I also look for the optimal resolution in time and in each of the spatial dimensions. Hence, the main aspect of this project is finding the best regions and the best resolutions to maximize classification accuracy.

The outline of the report is as follows. Section 2 provides more details on the dataset and the specifics of the problem. Section 3 covers some related work on the topic. Section 4 describes the proposed method, the reasons behind it, and the algorithms that are used. The experimental results are then presented in Section 5.

2 Problem definition

The data set contains six different subjects. For each subject we have 40 sets of fMRI images that were taken when he or she was reading a sentence, and another 40 sets of

fMRI images taken when he or she is viewing a picture. The goal is to design a classifier to correctly identify which sets of images were taken when the subject was reading a sentence, and which were taken when the subject was viewing a picture.

Each set of fMRI images contains 16 3D images, which are spaced every 500msec in time. Each 3D image contains up to 64 x 64 x 8 voxels. Each image also has regions of interests (ROIs) defined on it, which correspond to specific anatomic areas of the brain.

The classifier is trained using ten-fold cross-validation, that is, it is trained on 36 sets of images and tested on the remaining 4. The test images are then rotated. Note that *a separate classifier is trained for each subject.*

3 Related work

Mitchell et al have employed several classifiers for this task, including Gaussian Naïve Bayes (GNB), Support Vector Machine (SVM), and Nearest-Neighbor classifiers[1]. In the absence of feature selection (ie. with all voxels being used as features), GNB and SVM were the most successful classifiers, achieving an accuracy of 66%. The introduction of feature selection improved accuracy to 82% (GNB) and 89% (SVM). The feature selection was based on the activity of the voxels, by comparing the activation of voxels while a task was being performed with their activation while the subject was at rest. This comparison was done with a t-test, and the voxels with the greatest t-statistic are chosen by the feature selection process. Note that the results from my project and cannot be directly compared with the one mentioned here, because I only had data for 6 of the 13 subjects that were used.

A detailed report that specifies the classification accuracy on individual subjects can be found in [2]. This allows a more direct comparison of my classifier with previous work. This report also recommends focussing on specific ROIs. The recommended set of 7 ROIs is {'CALC', 'LDLPFC', 'LIPL', 'LIPS', 'LOPER', 'LT', 'LTRIA'}. Another recommended subset of only 4 ROIs is {'CALC', 'LIPL', 'LIPS', 'LOPER'}. Some of my experiments use all of the ROIs, some use the subset of 7 ROIs, and some use the subset of 4 ROIs.

4 Proposed method

4.1 Overview

The proposed method is to use resolution and localization to improve the accuracy of the GNB. The first aspect involves determining the optimal resolution for each dimension in space and time under which to train the GNB. The voxels are reduced to this optimal resolution by averaging them together along the relevant dimension. The second aspect of the method is to divide the set of voxels into several smaller localized subsets and to train a separate GNB on each of these localized subsets of voxels. The resulting decisions from each classifier are then combined through a weighting procedure. We denote the term *classification bank* for this set of individual classifiers, each designed for a separate

subset of voxels, and each with a weighting that allows their outputs to be combined to yield an overall decision. Thus, the process of designing a classification bank involves 1) choosing the resolution under which it will operate, 2) choosing the subsets of voxels to assign to each classifier within the bank, 3) choosing the weightings for each classifier within the bank, and 4) training these (GNB) classifiers.

4.2 Intuition

Presently the GNB uses the highest available resolution, that is, it uses all of the voxels. There are two possible advantages to lowering the resolution in any given dimension. First, voxels at lower resolution will have less noise. The benefits of reduced noise may outweigh the advantages of a high resolution, particularly given that we do not know if the high frequency information is useful anyway. Second, the GNB assumes conditional independence between all voxels. This assumption is unlikely to be true. Reducing the resolution the number of conditional independence assumptions that we need to make. For example, a voxel is highly correlated in time. By removing all time resolution (ie. by taking the average value of a given voxel in all time frames), we remove any requirement of conditional independence between voxels in successive time frames. The effect of averaging all voxels together over time was found to improve classification accuracy by over 10%.

There are also two reasons behind dividing the overall set of voxels into smaller localized subsets of voxels. The first relates to conditional independence again. Voxels that are placed into different subsets do not have to be conditionally independent. Hence, if we do not think that a voxels are conditionally independent in the time dimension, we can simply place the voxels in each time frame into a different subset. This was found to give a 6% improvement in classificaiton accuracy. The second reason for this localization is that it provides an alternative form of feature selection. It is probable that voxels in certain regions in space (and possibly time) do not provide useful information, and rather serve only to add noise to the system. A common feature selection technique is to remove any voxels that do not seem to be assisting with classification on the training set. However, with localized subsets, we can remove the influence of an entire group of voxels. Such a method of feature selection may be less vulnerable to noise.

4.3 Algorithm Overview

As mentioned earlier, to create the best classification bank we need to choose 1) the optimal resolution and localized subsets of voxels on which to train the individual classifers, 2) the weightings for each individual classifier in the bank, and 3) we need to train these individual classifiers. The training and testing process requires a nested cross validation scheme using three loops as explained below:

4.3.1 Outer loop

In the outer loop we split the full dataset into a training and testing set. The training set is used to choose the ideal classification bank (ie. choose the optimal resolution and

subsets of voxels) and also to train this classification bank. The result is then tested using the testing set. With all of these loops, we rotate the testing set (ie. cross validation) to maximize the use of our limited data.

4.3.2 Middle loop

The middle loop is used to choose the best classification bank. The training dataset is further split into a mini-training set and a mini-testing set. On the training set we train several different classification banks, ie. banks with different combinations of resolutions and localized subsets. The process for choosing these combinations is outlined in Section 5.2. We must now determine which of these classification banks is the best. To do this, we take our trained classification banks and test them on the mini-testing set. The bank with the highest accuracy is chosen, and this bank is returned to the outer loop. As a possible extension to this method, we could actually create a "super-bank" that was itself a weighted combination of individual classification banks. In other words, instead of choosing the best classification bank, we would use all of them but with different weightings assigned to each of them. However, this would raise computational complexity considerably.

4.3.3 Inner loop

The inner loop is used to actually train the classification bank. The data set is split yet again into a training and testing set. In the training set, we train individual classifiers for each of the localized subsets of voxels. These classifiers are then tested on the testing set. The challenge is now to choose a weighting for each classifier so that the final output of the classification bank is a weighted combination of the outputs of each individual classifier. A natural choice might be to choose weights that optimize accuracy on the training set using an iterative method. However, instead I chose to use a least-squares method which is explained in more detail in the next section. This method gave a similar accuracy on the testing set to that of an iterative method, and had the advantage that it is closed-form and very fast. This computational efficiency is an important advantage because the weighting procedure takes place inside the inner loop, and is thus performed many many times.

4.4 Algorithm details and extensions

4.4.1 Process for training the weights

The algorithm for computing the weights is given a set of training examples to use for this task. For each training example we divide the voxels into their approriate subsets. We then use a portion of the training examples to train a separate GNB for each subset of voxels, and test each GNB on the remaining portion of the examples. When we test a GNB, two numbers are returned for each test example, representing the probabilities that the subject was reading a sentence or viewing a picture. We define the output of the GNB as the latter number subtracted from the former. Hence a positive output signifies that the subject was reading a sentence, and a negative output signifies that the subject

was viewing a picture. By performing cross-validation, we end up with GNB outputs for every example in the set. We have a different output for each of the localized voxel subsets in the set. We arrange these outputs into a data matrix A where the number of rows equals the number of training examples, and the number of columns equals the number of voxel subsets. We now create a target column vector t which has a length equal to the number of training examples and entries of 1 if the subject was reading a sentence and -1if the subject was viewing a picture. By solving the equation $Aw = t$ with a least-squares algorithm, we obtain a column vector w which contains the optimal weights for each localized classifier.

We make one extension to this method. The least-squares solution will often contain weights that are negative. This is somewhat nonsensical because it implies that the GNB classifier for the relevant voxel subset was actually doing worse than the random guess. If this occurs, we assume that the classifier is overfitting the data, and to solve this problem we remove the relevant voxel subsets from the classification bank. We then retrain the weights using the equation $Aw = t$, but this time without the outputs relating to the aforementioned voxel subsets.

This latter extension made a significant improvement to results, and there may be room for further improvement here by also removing voxel subsets with a positive but very small weighting. In effect, we are performing a type of feature selection here, but on a more global level than that mentioned in the related work.

4.4.2 Rapid mode

The algorithms for testing this classifier took a long time to run. This is a result of the three nested loops: the outer being used to ensure a true separation of training and testing data, the middle being used to choose the best classification bank (ie. the best resolution/localization scheme), and the inner loop being used to train the weights. These last two loops can actually be combined if we use the same set of images to train the weights as to choose the best classification bank. Such a procedure is not entirely valid because we will be training the weights on data that is later used to choose the classification bank. However the speed-up is very significant. Note that this procedure does not in any way compromise the validity of the final result. The outer loop, which creates the initial split between training and testing data, is still left intact.

The effect of using rapid mode was a reduction in final accuracy of around 1%, as reported in Section 5.3, but a tenfold reduction in runtime.

5 Experiments

5.1 Conditions

- All classification banks are trained for a *single subject*. In cases where an average accuracy is given, this accuracy will be averaged across all six subjects, however each of the six subjects would have had its own classification bank.

- The results cited in the related work generally used feature selection, choosing voxels with high activity compared to when at rest. *None of my experiments use feature selection.*
- All experiments are performed with ten-fold cross validation.

5.2 Procedure

The first stage in the training process is to choose an optimal voxel resolution in each dimension, and to choose how to split the voxels into localized subsets. The difficulty with this stage is that there are a large number of possible combinations. It is not viable to do an exhaustive search through all combinations. Therefore, I used a guided search, as outlined in the case study that follows.

First we measure the accuracy of the initial GNB using all voxels at the highest resolution. This step is necessary to ensure that our resolution and localization operations are actually helping. The average accuracy over the six subjects was **74.2%** when 7 ROIs are used. All results quoted in this section are using 7 ROIs.

We now refer to Table 1 to explain the iterative progression of the algorithm. The first iteration simply tries reducing the resolution by a factor of 2 in time, in z, and in xy. These are done individually, so first we just reduce the resolution in time but not in any other dimension, then we just reduce the resolution in z but not in any other dimension, and so forth. Note that the x and y dimensions are treated as one, so whenever we reduce the resolution in x, we also reduce it in y. The resulting accuracies for these experiments are shown in the first 3 columns in Table 1 on the top row (iteration 1). We also perform localization by a factor of 2 on each dimension in turn. There are 16 timeframes for each voxel, so a localization by 2 in time means that the voxels from the first 8 timeframes are put into one subset, and the voxels from the last 8 timeframes are put in another subset. As with resolution, the x and y axes are considered as one. Hence, if we perform localization by 2 on the xy axes, we end up with 4 subsets, with each subset of size 32x32 representing the 4 quadrants of the xy plane (which is of size 64x64). The results from these 3 experiments are also shown in Table 1 on the row corresponding to iteration 1.

We now look at the 6 results in iteration 1 and choose the highest accuracy. The highest accuracy occurred when we localized by 2 in the xy dimension. Therefore, the decision from iteration 1 is to localize by 2 in xy.

The second iteration tries to improve the accuracy by localizing by 2 in xy as concluded above, but also performing one other operation. Hence the first column of iteration 2 corresponds to localizing by 2 in xy *and* reducing the time resolution by 2. The second column corresponds to localizing by 2 in xy *and* reducing the z resolution by 2, and so forth. The final accuracy column in iteration 2 is when we localize by 2 in xy twice, which is equivalent to localizing by 4. This would mean that the original 64x64 voxels in the xy plane would be split into 16 subsets of 16x16 voxels.

The highest accuracy in iteration 2 corresponded to the case where the time resolution was reduced by 2. Hence at this point, the optimal scheme is to localize the voxels by 2 in the xy dimension, and to reduce the time resolution by a factor of 2. The next iteration tries to improve accuracy by adding one more operation, and the process continues until

Table 1: Iterations in training procedure

	Resolution reduction			Localization			
Iteration	Time	z	xy	Time	z	xy	Optimal scheme
1	78.3%	76.7%	73.5%	81.9%	85.8%	**86.7%**	Split into 2 subsets along the x axis and 2 subsets along the y axis
2	**89.6%**	87.5%	87.3%	89.0%	87.7%	85.8%	Do the above, and also reduce time resolution by 2
3	90.3%	90.1%	89.6%	**90.8%**	90.2%	90.6%	Do the above, and also split into 2 subsets along the time axis
4	**92.5%**	91.3%	89.8%	90.2%	90.6%	92.0%	Do the above, but now reduce time resolution by 4
5	**92.9%**	91.3%	91.5%	91.3%	92.7%	91.9%	Do the above, but now reduce time resolution by 8
6	N/A	92.5%	92.5%	N/A	**93.3%**	92.7%	Do the above, and also split into 2 subsets along the z axis
7	N/A	92.6%	92.5%	N/A	92.9%	92.7%	No further improvement possible

there is no further increase in accuracy. Note that iterations 5 and 6 have N/A under the time localization and time resolution columns. The reason for this is that at this point in our algorithm we are reducing time resolution by 8, hence changing from 16 timeframes to only 2 timeframes. We are also localizing in time, thereby splitting these 2 timeframes into separate subsets, with each subset containing the voxels from one timeframe. Clearly it is not possible to further reduce the resolution in time, or to further localize in time, hence these operations are unavailable at this point in the algorithm.

Therefore, the final result from this process was an accuracy of 93.3% (on the training set). This accuracy was achieved by reducing the time resolution of the voxels by 8 (ie. averaging together the first 8 timeframes and the last 8 timeframes), splitting voxels from each of the resultant timeframes into separate subsets, and also further dividing the voxels in half along the xy and z dimensions. The final classification bank will contain 16 subsets of voxels with each subset of voxels containing 32x32x4 voxels at full spatial resolution, and with the time resolution reduced by 8.

5.3 Results

The previous section describes the procedure for choosing the resolution and localization scheme. However the 93.3% quoted cannot be taken as a valid accuracy because the resolution/localization scheme was tuned on the full dataset, and is thus tantamount to training on the testing data. To obtain a true accuracy we need to use the full nested cross validation scheme as described in Section 4.3. Table 2 shows the accuracies we obtain with this algorithm for the case of all ROIs, 7 ROIs, and 4 ROIs as outlined in Section 3. The baseline accuracy represents the accuracy if we just train a GNB on the original data. Rapid mode represents the accuracy that we get if we use the algorithm described in Section 4.4.2.

Note that these accuracies may increase slightly if 40-fold cross validation is used instead

Table 2: Final results

	All ROIs	7 ROIs	4 ROIs
Baseline accuracy	70.8%	74.2%	71.6%
Rapid mode	88.8%	91.0%	89.5%
Full mode	89.8%	92.0%	90.2%

of 10-fold cross validation. The training procedure also has several layers of nested cross validation, so we could also use more folds of cross-validation in the inner loops.

6 Conclusions

The technique of altering resolution and dividing the voxels into localized subsets has been shown to increase accuracy from 74.2% to 92.0% for the case of 7 ROIs. An important point to note is that this method did not perform feature selection as done in the related work. Consequently, it does not require access to voxel activity when the subject was at rest. This technique also outperforms the quoted value of 82% which is given for the GNB with feature selection, and also outperforms the 89% quoted for the SVM with feature selection. An interesting addition would be to add the feature selection method to my algorithm. This may cause further improvements, or we may find that the advantages of feature selection have somehow been implicity incorporated into my algorithm already, and thus no further gains are possible.

There are several natural extensions to the technique. The main restrictions with the algorithm in its present form is that 1) every localized subset of voxels must have the same resolution, 2) the localized subsets of voxels must be of the same size, 3) no voxel can appear in more than one subset, or at more than one resolution. The only reason for these restrictions is to reduce the volume of data and the number of possible combinations of resolution and localizations. With a large training set, we could easily remove these restrictions and still reliably search through the data and all possible combinations to find the optimal classifier. However, even with the size of the existing training set, there may be methods that allow us to remove some of the aforementioned restrictions without completely swamping the training process with an overload of data.

Therefore, in conclusion the concept of using multiple localized GNB classifiers and searching for the optimal resolution of the data has shown to be an effective enhancement, and it has the potential to yield even further improvements yet.

References

[1] T.M. Mitchell, R. Hutchinson, R.S. Niculescu, F.Pereira, X. Wang, M. Just, and S. Newman, Learning to Decode Cognitive States from Brian Images", *Machine Learning*, Vol. 57, Issue 1-2, pp. 145-175. October 2004.

[2] Xuerui Wang, Tom Mitchell, "Detecting Cognitive States using Machine Learning", October 9, 2002