Volume 3, Issue 10, October 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Experimental Investigation using Random Forest for Enhancing Classification of EEG Data

Dr.V.Baby Deepa,
Head, Department of Software Engineering,
M.Kumarasamy College of Engineering, Karur, India

Abstract: data mining, classification of data is being a tiresome task for further analysis. The EEG data classification requires more efficient algorithms. In this paper we propose to suggest a technique that classifies EEG data efficiently. In our investigation Naïve Bayes classifier and Random Forest have been used to classify the data. In order to suggest a technique that yields a better result it is required to apply Naïve Bayes and Random Forest individually with a combination of FHT with SMOTE and Resampling. In a bulk data set of EEG signals, the signals are classified into many channels. To enhance the classification these techniques are applied to the EEG dataset individually. After the application, the results suggest that the percentage of correctly classified instances is high when resampling is combined with FHT and further with Random forest.

Keywords: EEG (Electro-encephalogram), BCI (Brain Computer Interface), FHT (Fast Hartley Transform), Random Forest, SMOTE (Synthetic Minority Over-sampling Technique).

I – Introduction

The EEG data set obtained from BCI is used for classification. In the EEG signals there would be a cluster of features. It is vital to extract good features from that cluster. Classification of EEG dataset involves much careful effort. Identifying and extracting good features from the signals is a crucial step in the design of BCI. The features extracted from EEG are not relevant and do not describe well the neurophysiologic signals employed, the classification algorithm which will use such features will have trouble in identifying the class of these features i.e., mental state of the user. Consequently the correct recognition rates of mental states will be very low, which will make use of the interface not convenient or even impossible for the user. It is sometimes possible to use raw signals as the input of the classification algorithm, it is recommended to select and extract good features in order to maximize the performances of the system by making easier the task of subsequent classification algorithm. According to researchers, it seems that the choice of a good preprocessing and feature extraction method have more impact on the final performances than the choice of a good classification algorithm. In the section II the classifiers have been described. This section gives a clear picture of how the classifiers work along with classification filter FHT, SMOTE, and Resampling. The filters are applied to the data during the experimentation and the results are given in the section III. Finally it has been concluded that a combination of FHT, Chebyshev and FT tree have the potential to enhance the classification of EEG data.

II – Classification

A. **Naïve Bayes Classifier**

The Naive Bayes classifier is a probabilistic classifier simplifying Bayes' theorem by *naively* assuming class conditional independence[1]. It is known for its high efficiency and generalization ability and has the advantage of good classification accuracy. It is widely used in several domains. The classifier has a conditional model over a dependent variable C over a dependent class variable *C* with a small number of outcomes or *classes*, conditional on several feature variables F_1 through F_n. Using Bayes theorem the classifier model is formulated as

$$p(C \mid F_1,\ldots, Fn) = \frac{p(C)\, p(F_1,\ldots, Fn \mid C)}{p(F_1,\ldots, Fn)}.$$

The Naïve Bayes classifier estimation includes parameter estimation of class and which is otherwise called prior probability estimation and conditional probability or density estimation [2]. Since the classifier has high efficiency the EEG data have been classified using this technique. After the classification FHT is combined with Resamplng and SMOTE separately and each combination if further applied with this classifier to yield high performance in classification. The performance of classification is observed every time when the combination is applied with Naïve Bayes.

B. The Fast Hartley Transform (FHT)

EEG data are inherently real valued, yet most general Fourier transform algorithms accept complex valued input and return complex valued output. The generality of these algorithms is also their weakness, for in the process of transforming real data they perform twice as many operations (arithmetic, address and transfer) as is necessary. Since the

Fourier transform is commonly used in the analysis of real signals, special versions of almost every transform algorithm have been developed to deal more efficiently with real data. Unfortunately, when it comes to inverse transformation, *another* special version of the algorithm is required to efficiently transform the complex output back into the real sequence.

The Hartley transform[4] distinguishes itself from its close cousin, the Fourier transform , by being real valued; it produces real output from real input. Even so, it provides the same phase and amplitude information about the data as the Fourier transform. The Hartley transform may also be computed using a 'fast' algorithm which requires O (Nlog2N) operations. Finally, the fast Hartley transform (FHT) [3] is twice as fast as a complex valued FFT, requiring virtually the same number of operations as the real valued FFT algorithms.

Formally, the discrete Hartley transform is a linear, invertible function H: $\mathbf{R}^n$ -> $\mathbf{R}^n$ (where **R** denotes the set of real numbers). The N real numbers x_0, ..., x_{N-1} are transformed into the N real numbers H_0, ..., H_{N-1} according to the formula

$$H_k = \sum_{n=0}^{N-1} x_n \left[\cos\left(\frac{2\pi}{N}nk\right) + \sin\left(\frac{2\pi}{N}nk\right)\right] \qquad k = 0, \ldots, N-1$$

where π is Pi.

The combination cos (z) + sin(z) = √ 2 cos (z - π / 4) is sometimes denoted cos(z), and should be contrasted with the e^{-iz} = cos(z) – i sin(z) that appears in the DFT definition (where *i* is the imaginary unit).

C. Synthetic Minority Over-sampling Technique (SMOTE)

The SMOTE [5] is an over-sampling method which combines the informed over-sampling of minority class with random under-sampling of the majority class. The number of synthetic samples generated by SMOTE is fixed in advance, thus not allowing any flexibility in the re-balancing rate.

(i) SMOTE Algorithm

```
Algorithm SMOTE(T, N, k)
Input: Number of minority class samples T; Amount of SMOTE N%; Number of nearest neighbors k
Output: (N/100) * T synthetic minority class samples
(If N is less than 100%, randomize the minority class samples as only a random percent of them will be SMOTEd)
  If N < 100 then randomize the T minority class samples
      T = (N/100) T
      N = 100
   end if
  N = (int) (N/100) (The amount of SMOTE is assumed to be in integral multiples of 100.)
  k = Number of nearest neighbours
      nattr = Number of attributes
      Sample[ ][ ]: array for original minority class samples
       newindex: keeps a count of number of synthetic samples
       generated, initialized to 0
       Synthetic[ ][ ]: array for synthetic samples
(Compute k nearest neighbours for each minority class sample only.)
       for i ← 1 to T
           Compute k nearest neighbours for i, and save the indices in the nnarray
           EEG(N, i, nnarray)
         endfor
EEG(N, i, nnarray) ( Function to generate the  synthetic samples.)
      while N 6= 0
 Choose a random number between 1 and k, call it nn. This step chooses one of the k nearest neighbours of i.
        for attr ← 1 to nattr
       Compute: dif = Sample[nnarray[nn]][attr] −
                        Sample[i][attr]
           Compute: gap = random number between 0 and 1
             Synthetic[newindex][attr] = Sample[i][attr] + gap  dif
        endfor
         newindex++
          N = N − 1
      endwhile
        return (End of EEG)
End of Pseudo-Code.
```

D. Resampling

Resampling is a process to produce a random subsample of a dataset using either sampling [6] with replacement or without replacement. The number of instances in the generated dataset is usually specified. Resampling algorithms can be made to maintain the class distribution in the subsample, or to bias the class distribution toward a uniform distribution.

E. Random Forest

Random forest (or random forests) [7] is an ensemble classifier that consists of many decision trees and outputs the class that is the mode of the class's output by individual trees. The method combines bagging idea and the random selection of features in order to construct a collection of decision trees with controlled variation.

The selection of a random subset of features is an example of the random subspace method, which is a way to implement stochastic discrimination proposed by Eugene Kleinberg.

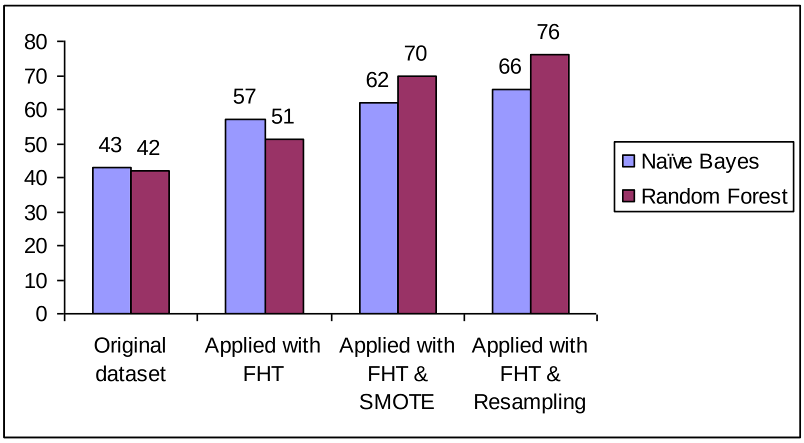
Learning algorithm

Each tree is constructed using the following algorithm:

1. Let the number of training cases be *N*, and the number of variables in the classifier be *M*.
2. We are told the number *m* of input variables to be used to determine the decision at a node of the tree; *m* should be much less than *M*.
3. Choose a training set for this tree by choosing *N* times with replacement from all *N* available training cases (i.e. take a bootstrap sample). Use the rest of the cases to estimate the error of the tree, by predicting their classes.
4. For each node of the tree, randomly choose *m* variables on which to base the decision at that node. Calculate the best split based on these m variables in the training set.
5. Each tree is fully grown and not pruned (as may be done in constructing a normal tree classifier).

III – Experimental Results

	Naïve Bayes	Random Forest
Original dataset	43	42
Applied with FHT	57	51
Applied with FHT & SMOTE	62	70
Applied with FHT & Resampling	66	76



IV – Conclusion

In our investigation the classification is done on the EEG dataset using classifiers Naïve Bayes and Random forest. When classifier Random forest is applied with Fast Hartley Transform filter and Resampling, the percent of correctly classified instances is remarkable and high. Thus, based on the experimental results it is suggested that Random forest with FHT and Resampling would help in enhancing classification of EEG dataset.

References

[1] Rish, Irina. (2001). "An empirical study of the naive Bayes classifier". IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence.

[2] Rennie J, Shih L, Teevan J, and Karger D. Tackling The Poor Assumptions of Naive Bayes Classifiers. In Proceedings of the Twentieth International Conference on Machine Learning (ICML). 2003.

[3] R. N. Bracewell, "The fast Hartley transform," *Proc. IEEE* 72 (8), 1010–1018 (1984).

[4] R. N. Bracewell, "Computing with the Hartley Transform," *Computers in Physics* 9 (4), 373–379 (1995).

[5] Chawla, N., Bowyer, K., Hall, L., & Kegelmeyer, P. (2000). SMOTE: Synthetic Minority Over-sampling TEchnique. In *International Conference of Knowledge Based ComputerSystems*, pp. 46–57. National Center for Software Technology, Mumbai, India, Allied Press.

[6] Metha, C. R., Patel, N. R. and Senchaudhuri, P. (1988). "Importance sampling for estimating exact probabilities in permutational inference", *J. Am. Statist. Assoc.*,

[7] Breiman, Leo (2001). "Random Forests". *Machine Learning* 45 (1): 5–32. doi:10.1023/A:1010933404324

